CRAY RESEARCH, INC.
SOFTWARE TRAINING

TR- Overlays

Course Name: IOS Internals

Type of Material: Selected Overlay Listings

Release Level (include BF): V116 BF1

Date Printed: July 7, 1987

Security Class: CRAY PRIVATE

AVAILABLE FROM
OTIC

1.16
OVERLAYS

AMAP
ADEM
AMSG
BEGIN
CALL
CDEM
CLOCK
CRAY
CRTDEM
INDD29
INDD39

INDD49
LISTO
LISTP
OVLNUM
PATCH
SCRUB
SYSS
ACOM
D4DEM
DD49
DISK

ICOM
BCOM
TDEM
COMMØ1
COMMØ2
ISPDRV
TAPELOAD
DISKLOAD

MHVM1 STN218 7/06/87 16:56:28 G.M.T. WAS THE ORIGIN

DEST: MN01 FILE: 7409 NAME: LIST LISTING DIST: TNGSTAC RECS: 00033002

```
0     <macro>                    OVERLAY   AMAP,TYPE=DATA                                    AMAP.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                    ******************************************                AMAP.5
                    *                                        *                AMAP.6
                    * 'AMAP'                                 *                AMAP.7
                    *  CONFIGURATION MAP FOR 'A' PROCESSOR   *                AMAP.8
                    *  CONTAINS CHANNEL INFO, SIZES OF       *                AMAP.9
                    *  MEMORY CHAINS, ETC.   IT IS           *                AMAP.10
                    *  INPUT TO OVERLAY LOADER, AND IS       *                AMAP.11
                    *  FIRST OVERLAY LOADED BY THE KERNEL    *                AMAP.12
                    *                        8.03.79         *                AMAP.13
                    *                                        *                AMAP.14
                    ******************************************                AMAP.15
                                                                              AMAP.16
                                                                              AMAP.17
                    ENTRY     AMSG                                            AMAP.18
            *       Start of common deck REV                                   REV.2
4           $REV    MICRO     '1.16'                                           REV.3
            *       End of common deck REV                                     REV.4
```

```
*                    = CN if front-end concentrator channel.                AMAP.102
*                    = VA if VAX type A concentrator channel.               AMAP.103
*                    = VB if VAX type B concentrator channel.               AMAP.104
*                    = NS if NSC channel.                                   AMAP.105
*                    = UC if User channel.                                  AMAP.106
*                    = FP if FEI Logical Path connection                    AMAP.107
*                                                                           AMAP.108
*                    If TYPE is not specified, then it is the first         AMAP.109
*                    entry of the table and NUM is the highest channel.     AMAP.110
*                                                                           AMAP.111
*                 ST = Status:                                              AMAP.112
*                    = UP if channel to be used (default).                  AMAP.113
*                    = DOWN if channel not to be used.                      AMAP.114
*                                                                           AMAP.115
*                 ORD = Channel ordinal for use in FEI table                AMAP.116
*                       valid for NSC and front end type channels           AMAP.117
*                                                                           AMAP.118
*                 DEST = Target memory for high speed channel:              AMAP.119
*                      = CMEM if Central Memory (default)                   AMAP.120
*                      = SSD if SSD Memory                                  AMAP.121
*                 UN = Highest unit number.  This parameter is requiredR13739JA.1
*                      for DD40 disks if a shadow drive is configured. R13739JA.2
*                      If this parameter is not specified, unit 0 is   R13739JA.3
*                      assumed to be the highest unit number.          R13739JA.4
*                                                                           AMAP.122
*        Shape    VWD  1/1,1/ST,5/DEST,9/TY for high speed channnels        AMAP.123
*                 VWD  1/1,1/ST,5/ORD,9/TY for front-end channnels          AMAP.124
*                 VWD  1/1,1/ST,5/UN,9/TY for disk channels              R13739JA.5
*                 VWD  1/1,1/ST,5/0,9/TY for all other channnels            AMAP.125
*                                                                           AMAP.126
*                 ST  = Status : 0 if Channel up.                           AMAP.127
*                             : 1 if Channel down.                          AMAP.128
*                 TY  = Device type : CH$ty in APTEXT defines values.       AMAP.129
*                 DEST= Target memory: TM$dest in APTEXT defines values     AMAP.130
*                 ORD = Concentrator ordinal number                         AMAP.131
                                                                            AMAP.132
                                                                            AMAP.133
         MACRO                                                              AMAP.135
<prototype>   L  CHANNEL  NUM,TY=,ST=UP,ORD=,DEST=CMEM,UN=              R13739JA.6
         LOCAL   $S,$D                                                      AMAP.137
<definition>     BASE     0                                                  AMAP.138
<definition>  L  *                                                          AMAP.139
```

```
<definition>                                                                    AMAP.140
<definition>          IFC       #TY#,EQ,##,3                                    AMAP.141
<definition>  ACCESS  LIST      MAC                                             AMAP.142
<definition>          NUM                                                       AMAP.143
<definition>  $END    SKIP                                                      AMAP.144
<definition>                                                                    AMAP.145
<definition>          $PAR      ((UP,0),(DOWN,1)),ST,(ILLEGAL STATUS),1         AMAP.146
<definition>  $S      SET       $OUT                                            AMAP.147
<definition>                                                                    AMAP.148
<definition>          $PAR      ((TY)),EM                                       AMAP.149
<definition>          IFE       $OUT,LT,0,1                                     AMAP.150
<definition>  $EMP    SKIP                                                      AMAP.151
<definition>          IFE       $OUT,EQ,0,1                                     AMAP.152
<definition>  $OUT    SET       1                                               AMAP.153
<definition>  ACCESS  LIST      MAC                                             AMAP.154
<definition>          <<$OUT>>                                                  AMAP.155
<definition>  $END    SKIP                                                      AMAP.156
<definition>  $EMP    ENDIF                                                     AMAP.157
<definition>                                                                    AMAP.158
<definition>  $DEST   IFC       !_TY_!,NE,!CH!                                  AMAP.159
<definition>  $D      SET       UN ORD 0                                  R13739JA.7
<definition>  $DEST   ELSE                                                      AMAP.161
<definition>  $D      SET       TM$_DEST                                        AMAP.162
<definition>  $DEST   ENDIF                                                     AMAP.163
<definition>                                                                    AMAP.164
<definition>                                                                    AMAP.165
<definition>  ACCESS  LIST      MAC                                             AMAP.166
<definition>          VWD       1/1,1/$S,5/$D,D'9/CH$_TY                        AMAP.168
<definition>                                                                    AMAP.172
<definition>  $END    ENDIF                                                     AMAP.173
<definition>  ACCESS  LIST      *                                               AMAP.174
<definition>          BASE      *                                               AMAP.175
<definition>  CHANNEL ENDM                                                      AMAP.176
                                                                                AMAP.178
                                                                                AMAP.179
              ***********************************************************       AMAP.180
              *  MACRO  LDEV                                            *       AMAP.181
              *                                                         *       AMAP.182
              *         DEFINE A LOGICAL DEVICE FOR MIOP                *       AMAP.183
              *                                                         *       AMAP.184
              *         TYPE  BMR = BUFFER MEMORY RESIDENT DATA AREA    *       AMAP.185
              *                     BLKS - NUMBER OF 65K MODULES        *       AMAP.186
```

```
                    *              DSK = STRIPED DISK GROUP                                *    AMAP.187
                    *                    HDS - NUMBER OF HEADS PER CYLINDER (PHYSICAL)     *    AMAP.188
                    *                    SCS - NUMBER OF SECTORS PER TRACK  (PHYSICAL)     *    AMAP.189
                    *                    UNS - LIST OF UNITS IN GROUP                      *    AMAP.190
                    *                          FIRST DIGIT IS THE IOP NUMBER               *    AMAP.191
                    *                          2ND AND 3RD DIGITS IS THE CHANNEL NUMBER    *    AMAP.192
                    *                          I.E. 122 IS IOP-1, CHANNEL 22               *    AMAP.193
                    *                END = END OF LOGICAL DEVICE DESCRIPTORS               *    AMAP.194
                    *         CHN   LOGICAL CHANNEL NUMBER CONFIGURED IN EQT               *    AMAP.195
                    *                                                                      *    AMAP.196
                    ************************************************************************    AMAP.197
                                                                                                AMAP.198
                              MACRO                                                             AMAP.199
<prototype>                   LDEV TYPE=,CHN=,BLKS=,HDS=,SCS=,UNS=                              AMAP.200
                              LOCAL $END,$BMR,$DSK,$NEXT                                        AMAP.201
<definition>        $NEXT     IFC   'TYPE',EQ,'END'                                             AMAP.202
<definition>                  O'177777                                                          AMAP.203
<definition>        $END      SKIP                                                              AMAP.204
<definition>        $NEXT     ENDIF                                                             AMAP.205
<definition>                  IFE  CHN,LT,O'20,1                                                AMAP.206
<definition>                  ERROR   'ILLEGAL CHANNEL NUMBER'                                  AMAP.207
<definition>                  IFE  CHN,GT,O'37,1                                                AMAP.208
<definition>                  ERROR   'ILLEGAL CHANNEL NUMBER'                                  AMAP.209
<definition>                  BASE O                                                            AMAP.210
<definition>        $BMR      IFC  'TYPE',EQ,'BMR'                                              AMAP.211
<definition>                  LDV$BMR                                                           AMAP.212
<definition>                  BASE O                                                            AMAP.213
<definition>                  CHN                                                               AMAP.214
<definition>                  BASE D                                                            AMAP.215
<definition>                  BLKS                                                              AMAP.216
<definition>                  BASE *                                                            AMAP.217
<definition>                  0                                                                 AMAP.218
<definition>        $END      SKIP                                                              AMAP.219
<definition>        $BMR      ENDIF                                                             AMAP.220
<definition>        $DSK      IFC  'TYPE',EQ,'DSK'                                              AMAP.221
<definition>                  LDV$DSK                                                           AMAP.222
<definition>                  BASE O                                                            AMAP.223
<definition>                  CHN                                                               AMAP.224
<definition>                  BASE D                                                            AMAP.225
<definition>                  HDS                                                               AMAP.226
<definition>                  SCS                                                               AMAP.227
<definition>                  $LUNIT (UNS)                                                      AMAP.228
```

```
<definition>        $END     SKIP                                                    AMAP.229
<definition>        $DSK     ENDIF                                                   AMAP.230
<definition>                 ERROR  'UNDEFINED T_YPE SPECIFIED'                      AMAP.231
<definition>        $END     ENDIF                                                   AMAP.232
<definition>                 BASE *                                                  AMAP.233
                    LDEV     ENDM                                                    AMAP.234
                                                                                     AMAP.235
                    ************************************************************     AMAP.236
                    *    MACRO   $LUNIT                                         *    AMAP.237
                    *                                                           *    AMAP.238
                    *    CALLED BY LDV TO INSERT LOGICAL DEVICE UNITS INTO TABLE *    AMAP.239
                    ************************************************************     AMAP.240
                                                                                     AMAP.241
                             MACRO                                                   AMAP.242
<prototype>                  $LUNIT U0,U1,U2,U3,U4,U5,U6                             AMAP.243
                             LOCAL $UNIT,$END                                        AMAP.244
<definition>                 BASE 0                                                  AMAP.245
<definition>        $UNIT    ECHO  A=(U0,U1,U2,U3,U4,U5,U6)                          AMAP.246
<definition>        $END     IFC   'A',NE,,                                          AMAP.247
<definition>                 IFE   A,EQ,0,1                                          AMAP.248
<definition>                 ERROR 'ILLEGAL UNIT NUMBER'                             AMAP.249
<definition>                 A                                                       AMAP.250
<definition>        $END     ENDIF                                                   AMAP.251
<definition>        $UNIT    ENDDUP                                                  AMAP.252
<definition>                 0                                                       AMAP.253
<definition>                 BASE *                                                  AMAP.254
                    $LUNIT   ENDM                                                    AMAP.255
                                                                                     AMAP.256
```

```
                 ***********************************************************************   AMAP.258
                 *                                                                     *   AMAP.259
                 *        Parameters                                                   *   AMAP.260
                 *                                                                     *   AMAP.261
                 ***********************************************************************   AMAP.262
                                                                                             AMAP.263
                          QUAL                                                               AMAP.264
                                                                                             AMAP.265
                          ENTRY     I@MSTDT                                                  AMAP.266
                          ENTRY     I@OPSYS                                                  AMAP.267
                          ENTRY     I@MSTUNT                                                 AMAP.268
                          ENTRY     I@TENTH                                                  AMAP.269
                                                                                             AMAP.270
       2         I@MSTDT  EQUALS    DSK$DD29            .Master device type                  AMAP.271
      12         I@OPSYS  EQUALS    SYS$COS             .Cray operating system               AMAP.272
       0         I@MSTUNT EQUALS    0                   .Master device unit                  AMAP.273
     144         I@TENTH  EQUALS    D'100               .RTC interrupts in 1/10 second       AMAP.274
                 * I@TENTH EQUALS   D'95                .For Model-A IO Subsystem            AMAP.275
                                                                                             AMAP.276
                          QUAL      *                                                        AMAP.277
```

```
 6   000002                    2                    .NUMBER OF 'A' S                    AMAP.279
                               QUAL                                                     AMAP.280
 7                     AMSG    *                                                        AMAP.281
                               QUAL      *                                              AMAP.282
 7   001000                    1000                      .Message area for IOPO         AMAP.283
10   004000                    4000                 .         A1                        AMAP.284
11   000000                    0                    .         A2                        AMAP.285
12   000000                    0                    .         A3                        AMAP.286
13   000021                    AAPT                 .POINTER TO A0 INFO                 AMAP.287
14   000077                    ABPT                 .A1                                 AMAP.288
15   000000                    0                    .A2                                 AMAP.289
16   000000                    0                    .A3                                 AMAP.290
17   000311                    LDEVS                     .POINTER TO LOGICAL DEVICES    AMAP.291
20   000030                    IOS@SIZ                   .MOS for IOS use (65K segments) AMAP.292
                                                                                        AMAP.293
                               *                                                        AMAP.294
                               *  INFO FOR A0 PROCESSOR                                 AMAP.295
                               *                                                        AMAP.296
21   000100                AAPT 100                  .0, 100K WORDS OF MOS FOR KERNEL    AMAP.297
22   000006                    6                    .1, number of local 512 buffers     AMAP.298
23   000040                    D'32                      .Number of MOS software stacks AMAP.299
24   030000                    30000                     .Size of overlay space         AMAP.300
25   000040                    40                   .4, DISK I-O DESCRIPTORS            AMAP.301
26   000001                    1                    .5, LOCAL MEMORY SIZE (X65K)        AMAP.302
27   000034                    AACH                 .6, POINTER TO CHAN CONFIG TABLE    AMAP.303
30   000312                    TTLO                 .7, Deadstart title for CRT         AMAP.304
31   000000                    0                    .10, IOP options                    AMAP.305
32   000362                    CBFO                 .11, D.S. Operator commands start   AMAP.306
33   000373                    CBFOL                .11, D.S. Operator commands end     AMAP.307
                                                                                        AMAP.308
                       *       MIOP Channel Configuration.                              AMAP.309
                                                                                        AMAP.310
34                     AACH    *                                                        AMAP.311
34   <macro>                   CHANNEL   47                                            AMAP.312
34   000047                    47                                                           CHANNEL  1
                                                                                        AMAP.313
35   <macro>                   CHANNEL   (6),TY=A1            .A1 communication (input) AMAP.314
 7w        400144               VWD       1/1,1/%%000006,5/%%000007,D'9/CH$A1                CHANNEL  1
                       ,                                                                    CHANNEL  1
                                                                                            CHANNEL  1
36   <macro>                   CHANNEL   (7),TY=A1            .A1 communication (output) AMAP.315
 7w            200062            VWD       1/1,1/%%000010,5/%%000011,D'9/CH$A1                CHANNEL  1
```

```
                                                                                                                CHANNEL  1
                                                                                                                CHANNEL  1
37    <macro>                    CHANNEL   (10),TY=EM                                                     AMAP.316
37                               <<$OUT>>                                                                      CHANNEL  1
40    <macro>                    CHANNEL   (11),TY=EM                                                     AMAP.317
40                               <<$OUT>>                                                                      CHANNEL  1
41    <macro>                    CHANNEL   (12),TY=EM                                                     AMAP.318
41                               <<$OUT>>                                                                      CHANNEL  1
42    <macro>                    CHANNEL   (13),TY=EM                                                     AMAP.319
42                               <<$OUT>>                                                                      CHANNEL  1
                                                                                                         AMAP.320
43    <macro>                    CHANNEL    (14,15),TY=(EM,2)  .EMPTY                                     AMAP.321
43                               <<$OUT>>                                                                      CHANNEL  1
45    <macro>                    CHANNEL    (16),TY=ER         .ERROR LOG CHANNEL                         AMAP.322
11w        400044                VWD       1/1,1/%%000024,5/%%000025,D'9/CH$ER                                CHANNEL  1
                                                                                                                CHANNEL  1
                                                                                                                CHANNEL  1
46    <macro>                    CHANNEL   (17),TY=EX         .Expander                                  AMAP.323
11w             200020           VWD       1/1,1/%%000026,5/%%000027,D'9/CH$EX                                CHANNEL  1
                                                                                                                CHANNEL  1
                                                                                                                CHANNEL  1
                                                                                                         AMAP.324
47    <macro>                    CHANNEL   (20),TY=CL         .CRAY Low Speed (input)                    AMAP.325
11w                  100034      VWD       1/1,1/%%000030,5/%%000031,D'9/CH$CL                                CHANNEL  1
                                                                                                                CHANNEL  1
                                                                                                                CHANNEL  1
50    <macro>                    CHANNEL   (21),TY=CL         .CRAY Low Speed (output)                   AMAP.326
12w   100034                     VWD       1/1,1/%%000032,5/%%000033,D'9/CH$CL                                CHANNEL  1
                                                                                                                CHANNEL  1
                                                                                                                CHANNEL  1
51    <macro>                    CHANNEL   (22,23),TY=(EM,2)   .Empty                                     AMAP.327
51                               <<$OUT>>                                                                      CHANNEL  1
53    <macro>                    CHANNEL   (24,25),TY=(EM,2)                                              AMAP.328
53                               <<$OUT>>                                                                      CHANNEL  1
55    <macro>                    CHANNEL   (26,27),TY=(EM,2)   .Empty                                     AMAP.329
55                               <<$OUT>>                                                                      CHANNEL  1
                                                                                                         AMAP.330
57    <macro>                    CHANNEL   (30,37),TY=(EM,10)                                             AMAP.331
57                               <<$OUT>>                                                                      CHANNEL  1
                                                                                                         AMAP.332
67    <macro>                    CHANNEL   (40),TY=AMPEX      .AMPEX display input                       AMAP.333
15w                  100023      VWD       1/1,1/%%000044,5/%%000045,D'9/CH$AMPEX                             CHANNEL  1
```

```
                                                                                                  CHAN      1
                                                                                                  CHANNEL   1
 70     <macro>                    CHANNEL   (41),TY=AMPEX        .AMPEX display output        AMAP.334
 16w    100023                     VWD       1/1,1/%%000046,5/%%000047,D'9/CH$AMPEX                CHANNEL   1
                                                                                                  CHANNEL   1
                                                                                                  CHANNEL   1
 71     <macro>                    CHANNEL   (42,43),TY=(EM,2)                                   AMAP.335
 71                                <<$OUT>>                                                        CHANNEL   1
 73     <macro>                    CHANNEL   (44,45),TY=(EM,2)                                   AMAP.336
 73                                <<$OUT>>                                                        CHANNEL   1
 75     <macro>                    CHANNEL   (46),TY=AMPEX        .AMPEX display input         AMAP.337
 17w          400114               VWD       1/1,1/%%000054,5/%%000055,D'9/CH$AMPEX                CHANNEL   1
                                                                                                  CHANNEL   1
                                                                                                  CHANNEL   1
 76     <macro>                    CHANNEL   (47),TY=AMPEX        .AMPEX display output        AMAP.338
 17w                200046          VWD       1/1,1/%%000056,5/%%000057,D'9/CH$AMPEX                CHANNEL   1
                                                                                                  CHANNEL   1
                                                                                                  CHANNEL   1


                                   *  BIOP channel configuration                                 AMAP.340
                                                                                                  AMAP.341
 77                         ABPT   *                                                              AMAP.342
 77     000300                      300                          .Buffer memory for KERNEL        AMAP.343
100     000026                      26                       .Disk buffers - BIOP                 AMAP.344
101     000040                      D'32                         .Number of MOS software stacks   AMAP.345
102     010000                      10000                        .Size of overlay space           AMAP.346
103     000060                      60                       .DAL's - Disk activity links         AMAP.347
104     000001                      1                        .LOCAL MEMORY IS 65K                 AMAP.348
105     000112                      ABCH                     .PTR. TO CHAN CONFIG                 AMAP.349
106     000312                      TTIO                     .Deadstart title for CRT             AMAP.350
107     000000                      0                        .IOP options                         AMAP.351
110     000373                      CBF1                     .D.S. Operator commands start        AMAP.352
111     000376                      CBF1L                    .D.S. Operator commands end          AMAP.353
                                                                                                  AMAP.354
                            *      BIOP Channel Configuration.                                    AMAP.355
                                                                                                  AMAP.356
112                         ABCH   *                                                              AMAP.357
112     <macro>                    CHANNEL   47                                                  AMAP.358
112     000047                     47                                                              CHANNEL   1
```

```
                                                                                                                AMAP.359
113     <macro>                         CHANNEL   (6),TY=A0            .A0 communication (input)                AMAP.360
 22w                    100030          VWD       1/1,1/%%000062,5/%%000063,D'9/CH$A0                              CHANNEL  1
                                ,                                                                                   CHANNEL  1
                                                                                                                    CHANNEL  1
114     <macro>                         CHANNEL   (7),TY=A0            .A0 communication (output)               AMAP.361
 23w    100030                          VWD       1/1,1/%%000064,5/%%000065,D'9/CH$A0                              CHANNEL  1
                                ,                                                                                   CHANNEL  1
                                                                                                                    CHANNEL  1
115     <macro>                         CHANNEL   (10),TY=EM                                                       AMAP.362
115                                     <<$OUT>>                                                                     CHANNEL  1
116     <macro>                         CHANNEL   (11),TY=EM                                                       AMAP.363
116                                     <<$OUT>>                                                                     CHANNEL  1
117     <macro>                         CHANNEL   (12),TY=EM                                                       AMAP.364
117                                     <<$OUT>>                                                                     CHANNEL  1
120     <macro>                         CHANNEL   (13),TY=EM                                                       AMAP.365
120                                     <<$OUT>>                                                                     CHANNEL  1
                                                                                                                   AMAP.366
121     <macro>                         CHANNEL   (14),TY=CH           .CRAY high speed                         AMAP.367
 24w          400164                    VWD       1/1,1/%%000076,5/%%000077,D'9/CH$CH                              CHANNEL  1
                                ,                                                                                   CHANNEL  1
                                                                                                                    CHANNEL  1
122     <macro>                         CHANNEL   (15),TY=CH           .CRAY high speed                         AMAP.368
 24w                200072               VWD       1/1,1/%%000100,5/%%000101,D'9/CH$CH                              CHANNEL  1
                                ,                                                                                   CHANNEL  1
                                                                                                                    CHANNEL  1
123     <macro>                         CHANNEL   (16,17),TY=(EM,2)                                                AMAP.369
123                                     <<$OUT>>                                                                     CHANNEL  1
                                                                                                                   AMAP.370
125     <macro>                         CHANNEL   (20),TY=D2           .DD29 disk drive                         AMAP.371
 25w          400010                    VWD       1/1,1/%%000104,5/%%000105,D'9/CH$D2                              CHANNEL  1
                                ,                                                                                   CHANNEL  1
                                                                                                                    CHANNEL  1
126     <macro>                         CHANNEL   (21),TY=EM                                                       AMAP.372
126                                     <<$OUT>>                                                                     CHANNEL  1
127     <macro>                         CHANNEL   (22),TY=EM                                                       AMAP.373
127                                     <<$OUT>>                                                                     CHANNEL  1
130     <macro>                         CHANNEL   (23),TY=EM                                                       AMAP.374
130                                     <<$OUT>>                                                                     CHANNEL  1
131     <macro>                         CHANNEL   (24),TY=EM                                                       AMAP.375
131                                     <<$OUT>>                                                                     CHANNEL  1
132     <macro>                         CHANNEL   (25),TY=EM                                                       AMAP.376
```

```
132                                 <<$OUT>>                                                    CHAN    1
133     <macro>                     CHANNEL   (26),TY=EM                                      AMAP.377
133                                 <<$OUT>>                                                    CHANNEL  1
134     <macro>                     CHANNEL   (27),TY=EM                                      AMAP.378
134                                 <<$OUT>>                                                    CHANNEL  1
                                                                                              AMAP.379
135     <macro>                     CHANNEL   (30,37),TY=(EM,10)   .Empty                     AMAP.380
135                                 <<$OUT>>                                                    CHANNEL  1
                                                                                              AMAP.381
145     <macro>                     CHANNEL   (40,41),TY=(EM,2)                               AMAP.382
145                                 <<$OUT>>                                                    CHANNEL  1
147     <macro>                     CHANNEL   (42),TY=AMPEX        .AMPEX display input       AMAP.383
 31w                    100023      VWD       1/1,1/%%000130,5/%%000131,D'9/CH$AMPEX              CHANNEL  1
                                  ,                                                             CHANNEL  1
                                                                                                CHANNEL  1
150     <macro>                     CHANNEL   (43),TY=AMPEX        .AMPEX display output      AMAP.384
 32w    100023                      VWD       1/1,1/%%000132,5/%%000133,D'9/CH$AMPEX              CHANNEL  1
                                  ,                                                             CHANNEL  1
                                                                                                CHANNEL  1
151     <macro>                     CHANNEL   (44,45),TY=(EM,2)                               AMAP.385
151                                 <<$OUT>>                                                    CHANNEL  1
153     <macro>                     CHANNEL   (46,47),TY=(EM,2)                               AMAP.386
153                                 <<$OUT>>                                                    CHANNEL  1
                                                                                              AMAP.387
                                  *         A2 configuration.                                   AMAP.388
                                                                                              AMAP.389
155                               ACPT      *                                                   AMAP.390
155     000040                              40                        .40K Words of MOS for KERNEL  AMAP.391
156     000010                              10                        .Local memory data buffers    AMAP.392
157     000040                              D'32                           .Number of MOS software stacks  AMAP.393
160     010000                              10000                          .Size of overlay space  AMAP.394
161     000140                              D'96                           .Number of DAL's        AMAP.395
162     000001                              1                         .Local memory size            AMAP.396
163     000170                              ACCH                      .Channel table                AMAP.397
164     000312                              TTLO                      .Deadstart title for CRT      AMAP.398
165     000000                              0                         .IOP options                  AMAP.399
166     000000                              0                         .D.S. Operator commands start AMAP.400
167     000000                              0                         .D.S. Operator commands end   AMAP.401
                                                                                              AMAP.402
                                  *         DIOP (IOP 2) Configuration.                         AMAP.403
                                                                                              AMAP.404
170                               ACCH      *                                                   AMAP.405
```

```
170     <macro>                         CHANNEL   47                                                          AMAP.406
170    000047                           47                                                                          CHANNEL  1
                                                                                                                  AMAP.407
171     <macro>                         CHANNEL   (6),TY=A0            .A0 communication (input)                  AMAP.408
 36w          400140                    VWD       1/1,1/%%000142,5/%%000143,D'9/CH$A0                                CHANNEL  1
                                 ,                                                                                   CHANNEL  1
                                                                                                                     CHANNEL  1
172     <macro>                         CHANNEL   (7),TY=A0            .A0 communication (output)                 AMAP.409
 36w                200060               VWD       1/1,1/%%000144,5/%%000145,D'9/CH$A0                                CHANNEL  1
                                 ,                                                                                   CHANNEL  1
                                                                                                                     CHANNEL  1
173     <macro>                         CHANNEL   (10),TY=A1           .A1 communication (input)                  AMAP.410
 36w                  .     100031       VWD       1/1,1/%%000146,5/%%000147,D'9/CH$A1                                CHANNEL  1
                                 ,                                                                                   CHANNEL  1
                                                                                                                     CHANNEL  1
174     <macro>                         CHANNEL   (11),TY=A1           .A1 communication (output)                 AMAP.411
 37w    100031                          VWD       1/1,1/%%000150,5/%%000151,D'9/CH$A1                                CHANNEL  1
                                 ,                                                                                   CHANNEL  1
                                                                                                                     CHANNEL  1
175     <macro>                         CHANNEL   (12),TY=EM                                                        AMAP.412
175                                     <<$OUT>>                                                                         CHANNEL  1
176     <macro>                         CHANNEL   (13),TY=EM                                                        AMAP.413
176                                     <<$OUT>>                                                                         CHANNEL  1
                                                                                                                  AMAP.414
177     <macro>                         CHANNEL   (14,17),TY=(EM,4)   .Empty                                        AMAP.415
177                                     <<$OUT>>                                                                         CHANNEL  1
                                                                                                                  AMAP.416
203     <macro>                         CHANNEL   (20,27),TY=(EM,10)  .Empty                                        AMAP.417
203                                     <<$OUT>>                                                                         CHANNEL  1
                                                                                                                  AMAP.418
213     <macro>                         CHANNEL   (30,37),TY=(EM,10) .Empty                                         AMAP.419
213                                     <<$OUT>>                                                                         CHANNEL  1
                                                                                                                  AMAP.420
223     <macro>                         CHANNEL   (40,41),TY=(EM,2)   .unused                                       AMAP.421
223                                     <<$OUT>>                                                                         CHANNEL  1
225     <macro>                         CHANNEL   (42),TY=AMPEX        .AMPEX display input                       AMAP.422
 45w          400114                    VWD       1/1,1/%%000166,5/%%000167,D'9/CH$AMPEX                             CHANNEL  1
                                 ,                                                                                   CHANNEL  1
                                                                                                                     CHANNEL  1
226     <macro>                         CHANNEL   (43),TY=AMPEX        .AMPEX display output                      AMAP.423
 45w                200046               VWD       1/1,1/%%000170,5/%%000171,D'9/CH$AMPEX                             CHANNEL  1
                                 ,                                                                                   CHANNEL  1
```

```
                                                                                                  CHAN     1
227     <macro>                         CHANNEL   (44,47),TY=(EM,4)    .unused                  AMAP.424
227                                     <<$OUT>>                                                     CHANNEL  1
                                                                                                  AMAP.425
                                *       A3 configuration.                                         AMAP.426
                                                                                                  AMAP.427
233                             ADPT    *                                                         AMAP.428
233     000040                          40                        .40K Words of MOS for KERNEL    AMAP.429
234     000010                          10                        .Local memory data buffers      AMAP.430
235     000040                          D'32                           .Number of MOS software stacks AMAP.431
236     024000                          24000                          .SIZE OF OVERLAY AREA      AMAP.432
237     000020                          20                        .DAL's                          AMAP.433
240     000001                          1                         .Local memory size              AMAP.434
241     000246                          ADCH                      .Channel table                  AMAP.435
242     000312                          TTL0                      .Deadstart title for CRT        AMAP.436
243     000000                          0                         .IOP options                    AMAP.437
244     000401                          CBF3                      .D.S. Operator commands start   AMAP.438
245     000412                          CBF3L                     .D.S. Operator commands end     AMAP.439
                                                                                                  AMAP.440
                                *       A3 Channel configuration.                                 AMAP.441
                                                                                                  AMAP.442
246     <macro>                         ADCH      CHANNEL   47                                    AMAP.443
246     000047                          47                                                           CHANNEL  1
                                                                                                  AMAP.444
247     <macro>                         CHANNEL   (6),TY=A0            .A0 communication (input)  AMAP.445
 51w                    100030          VWD       1/1,1/%%000176,5/%%000177,D'9/CH$A0                CHANNEL  1
                                ,                                                                    CHANNEL  1
                                                                                                     CHANNEL  1
250     <macro>                         CHANNEL   (7),TY=A0            .A0 communication (output) AMAP.446
 52w    100030                          VWD       1/1,1/%%000200,5/%%000201,D'9/CH$A0                CHANNEL  1
                                ,                                                                    CHANNEL  1
                                                                                                     CHANNEL  1
251     <macro>                         CHANNEL   (10),TY=A1           .A1 communication (input)  AMAP.447
 52w        400144                      VWD       1/1,1/%%000202,5/%%000203,D'9/CH$A1                CHANNEL  1
                                ,                                                                    CHANNEL  1
                                                                                                     CHANNEL  1
252     <macro>                         CHANNEL   (11),TY=A1           .A1 communication (output) AMAP.448
 52w            200062                  VWD       1/1,1/%%000204,5/%%000205,D'9/CH$A1                CHANNEL  1
                                ,                                                                    CHANNEL  1
                                                                                                     CHANNEL  1
253     <macro>                         CHANNEL   (12),TY=EM                                      AMAP.449
253                                     <<$OUT>>                                                     CHANNEL  1
```

```
254    <macro>                          CHANNEL   (13),TY=EM                                          AMAP.450
254                                     <<$OUT>>                                                          CHANNEL  1
                                                                                                        AMAP.451
255    <macro>                          CHANNEL   (14,17),TY=(EM,4)  .Empty                             AMAP.452
255                                     <<$OUT>>                                                          CHANNEL  1
                                                                                                        AMAP.453
261    <macro>                          CHANNEL   (20,37),TY=(EM,20)  .Empty                            AMAP.454
261                                     <<$OUT>>                                                          CHANNEL  1
                                                                                                        AMAP.455
301    <macro>                          CHANNEL   (40,41),TY=(EM,2)   .unused                           AMAP.456
301                                     <<$OUT>>                                                          CHANNEL  1
303    <macro>                          CHANNEL   (42),TY=AMPEX       .AMPEX display input              AMAP.457
 60w                     100023         VWD       1/1,1/%%000220,5/%%000221,D'9/CH$AMPEX                  CHANNEL  1
                                  ,                                                                       CHANNEL  1
                                                                                                          CHANNEL  1
304    <macro>                          CHANNEL   (43),TY=AMPEX       .AMPEX display output             AMAP.458
 61w   100023                            VWD       1/1,1/%%000222,5/%%000223,D'9/CH$AMPEX                  CHANNEL  1
                                  ,                                                                       CHANNEL  1
                                                                                                          CHANNEL  1
305    <macro>                          CHANNEL   (44,47),TY=(EM,4)   .unused                           AMAP.459
305                                     <<$OUT>>                                                          CHANNEL  1
                                                                                                        AMAP.460
                                                                                                        AMAP.461
                                  *     LOGICAL DEVICE DEFINITIONS                                        AMAP.462
311                               LDEVS *                                                                AMAP.463
                                                                                                        AMAP.464
                                  *.    LDEV TYPE=BMR,CHN=20,BLKS=BMR@SIZ .BMR datasets (65K segments)    AMAP.465
                                  *.    LDEV TYPE=DSK,CHN=21,HDS=10,SCS=18,UNS=(121,122) .STRIPED DISK    AMAP.466
311    <macro>                          LDEV  TYPE=END                                                  AMAP.467
                                                                                                        AMAP.468


                                  **    CRT Deadstart message(s).                                         AMAP.470
                                                                                                        AMAP.471
                                                                                                        AMAP.472
312                               TTL0  *                                                               AMAP.473
312    005015                           O'5015                    .LF/CR                                 AMAP.474
313                                     'IOP-0 'H                 .IOP number                            AMAP.475
316                                     'KERNEL, VERSION 'H                                              AMAP.476
       <edit>                           '"$REV", 'H               .VERSION                               AMAP.477
```

```
326                                 '1.16, 'H                .VERSION                                    *EDI     0
331                                 'SNxxx/yy, 'H              .SERIAL NUMBER CRAY/IOS                 AMAP.478
336                                 'MENDOTA HEIGHTS, 'H       .LOCATION                               AMAP.479
        <edit>                      '"$DATE"  "$TIME"'                                                 AMAP.480
347                                 '07/06/87  11:54:56'                                                   *EDITED  0
                                                                                                           *EDITED  0
361   000000                        0                          .Insure end of line                     AMAP.481
                                                                                                       AMAP.482
                                    NEWPAGE                                                            AMAP.483
                        50    TTLOL EQUALS     *-TTLO          .Length                                 AMAP.484


                              **    Deadstart Operator commands for each IOP.                          AMAP.486
                                                                                                       AMAP.487
                                                                                                       AMAP.488
362                           CBF0  *                          .IOP 0 Commands                         AMAP.489
362   <macro>                       COMMAND   (CONFIG),D                                               AMAP.490
370   <macro>                       COMMAND   ,DISPLAY         .Null line forces Header                AMAP.491
                                                                                                       AMAP.492
373                           CBF0L *                          .End of commands                        AMAP.493
                                                                                                       AMAP.494
373                           CBF1  *                          .IOP 1 Commands                         AMAP.495
373   <macro>                       COMMAND                    .Null line forces Header                AMAP.496
                                                                                                       AMAP.497
376                           CBF1L *                                                                  AMAP.498
                                                                                                       AMAP.499
376                           CBF2  *                          .IOP 2 Commands                         AMAP.500
376   <macro>                       COMMAND                    .Null line forces Header                AMAP.501
                                                                                                       AMAP.502
401                           CBF2L *                                                                  AMAP.503
                                                                                                       AMAP.504
401                           CBF3  *                          .IOP 3 Commands                         AMAP.505
401   <macro>                       COMMAND   (CONFIG)                                                 AMAP.506
407   <macro>                       COMMAND                    .Null line forces Header                AMAP.507
                                                                                                       AMAP.508
412                           CBF3L *                                                                  AMAP.509
```

```
                                                                    AMAP.511
END                                                                 AMAP.512
```

7	AMSG		2:14 E	10: 3 D						
30	CH$A0	$APTEXT	13: 3	13: 7	15: 5	15: 9	16:26	16:30		
31	CH$A1	$APTEXT	10:38	10:42	15:13	15:17	16:34	16:38		
23	CH$AMPEX	$APTEXT	11:42	12: 4	12:12	12:16	14:13	14:17	15:37	15:41
			17:13	17:17						
35	CH$CH	$APTEXT	13:20	13:24						
34	CH$CL	$APTEXT	11:24	11:28						
2	CH$D2	$APTEXT	13:31							
11	CH$ER	$APTEXT	11:15							
10	CH$EX	$APTEXT	11:19							
2	DSK$DD29	$APTEXT	9:14							
2	I@MSTDT		9: 9 E	9:14 D						
0	I@MSTUNT		9:11 E	9:16 D						
12	I@OPSYS		9:10 E	9:15 D						
144	I@TENTH		9:12 E	9:17 D						
30	IOS@SIZ	$APTEXT	10:14							
0X	O$AMAP		1: 4 D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
12	SYS$COS	$APTEXT	9:15							
0	TM$CMEM	$APTEXT	13:19	13:23						

SYMBOL QUALIFIER IS AMAP

0	$OUT	10:37 D	10:37 F	10:37	10:41 D	10:41 F	10:41	11: 3 D	11: 3 F
		11: 3	11: 4	11: 5 D	11: 5 F	11: 5	11: 6	11: 7 D	11: 7 F
		11: 7	11: 8	11: 9 D	11: 9 F	11: 9	11:10	11:12 D	11:12 F
		11:12	11:13	11:14 D	11:14 F	11:14	11:18 D	11:18 F	11:18
		11:23 D	11:23 F	11:23	11:27 D	11:27 F	11:27	11:31 D	11:31 F
		11:31	11:32	11:33 D	11:33 F	11:33	11:34	11:35 D	11:35 F
		11:35	11:36	11:38 D	11:38 F	11:38	11:39	11:41 D	11:41 F
		11:41	12: 3 D	12: 3 F	12: 3	12: 7 D	12: 7 F	12: 7	12: 8
		12: 9 D	12: 9 F	12: 9	12:10	12:11 D	12:11 F	12:11	12:15 D
		12:15 F	12:15	13: 2 D	13: 2 F	13: 2	13: 6 D	13: 6 F	13: 6
		13:10 D	13:10 F	13:10	13:11	13:12 D	13:12 F	13:12	13:13
		13:14 D	13:14 F	13:14	13:15	13:16 D	13:16 F	13:16	13:17
		13:19 D	13:19 F	13:19	13:23 D	13:23 F	13:23	13:27 D	13:27 F
		13:27	13:28	13:30 D	13:30 F	13:30	13:34 D	13:34 F	13:34
		13:35	13:36 D	13:36 F	13:36	13:37	13:38 D	13:38 F	13:38
		13:39	13:40 D	13:40 F	13:40	13:41	13:42 D	13:42 F	13:42

		14: 1	14: 2 D	14: 2 F	14: 2	14: 3	14: 4 D	14: 4 F	14: 4
		14: 5	14: 7 D	14: 7 F	14: 7	14: 8	14:10 D	14:10 F	14:10
		14:11	14:12 D	14:12 F	14:12	14:16 D	14:16 F	14:16	14:20 D
		14:20 F	14:20	14:21	14:22 D	14:22 F	14:22	14:23	15: 4 D
		15: 4 F	15: 4	15: 8 D	15: 8 F	15: 8	15:12 D	15:12 F	15:12
		15:16 D	15:16 F	15:16	15:20 D	15:20 F	15:20	15:21	15:22 D
		15:22 F	15:22	15:23	15:25 D	15:25 F	15:25	15:26	15:28 D
		15:28 F	15:28	15:29	15:31 D	15:31 F	15:31	15:32	15:34 D
		15:34 F	15:34	15:35	15:36 D	15:36 F	15:36	15:40 D	15:40 F
		15:40	16: 2 D	16: 2 F	16: 2	16: 3	16:25 D	16:25 F	16:25
		16:29 D	16:29 F	16:29	16:33 D	16:33 F	16:33	16:37 D	16:37 F
		16:37	16:41 D	16:41 F	16:41	16:42	17: 1 D	17: 1 F	17: 1
		17: 2	17: 4 D	17: 4 F	17: 4	17: 5	17: 7 D	17: 7 F	17: 7
		17: 8	17:10 D	17:10 F	17:10	17:11	17:12 D	17:12 F	17:12
		17:16 D	17:16 F	17:16	17:20 D	17:20 F	17:20	17:21	18:19 D
		18:19 F	18:19	18:20 D	18:20 F	18:20			
34	AACH	10:25	10:33 D						
21	AAPT	10: 9	10:19 D						
112	ABCH	12:32	12:40 D						
77	ABPT	10:10	12:25 D						
170	ACCH	14:34	14:42 D						
155	ACPT	14:27 D							
246	ADCH	16:14	16:22 D						
233	ADPT	16: 7 D							
362	CBF0	10:28	18:18 D						
373	CBF0L	10:29	18:22 D						
373	CBF1	12:35	18:24 D						
376	CBF1L	12:36	18:27 D						
376	CBF2	18:29 D							
401	CBF2L	18:32 D							
401	CBF3	16:17	18:34 D						
412	CBF3L	16:18	18:38 D						
311	LDEVS	10:13	17:25 D						
312	TTL0	10:26	12:33	14:35	16:15	17:38 D	18:10		
50	TTL0L	18:10 D							

```
0     <macro>                    OVERLAY   ADEM                                                    ADEM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ********************************************************************   ADEM.5
                    *                                                                  *   ADEM.6
                    *        ADEM processes the following packet types (destination id)  *   ADEM.7
                    *        received from the cpu:                                    *   ADEM.8
                    *                                                                  *   ADEM.9
                    *        F         Shell or NSC interface request                  *   ADEM.10
                    *        K         Kernel request                                  *   ADEM.11
                    *        T         Terminal i/o request                            *   ADEM.12
                    *                                                                  *   ADEM.13
                    ********************************************************************   ADEM.14
                                                                                              .
6     <macro>                         REGDEFS   ,,(DAL,TTY,FC,CHN,TBL,ORD,TYPE,OVL,POINT,MATCH)   ADEM.16
                410      %S1          EQUALS    $REGORG                                              $REG     3
                411      %S2          EQUALS    $REGORG                                              $REG     3
                412      %S3          EQUALS    $REGORG                                              $REG     3
                413      %S4          EQUALS    $REGORG                                              $REG     3
                414      %S5          EQUALS    $REGORG                                              $REG     3
                415      %T1          EQUALS    $REGORG                                              $REG     3
                416      %T2          EQUALS    $REGORG                                              $REG     3
                417      %T3          EQUALS    $REGORG                                              $REG     3
                420      %T4          EQUALS    $REGORG                                              $REG     3
                421      %T5          EQUALS    $REGORG                                              $REG     3
                422      %T6          EQUALS    $REGORG                                              $REG     3
                423      %W1          EQUALS    $REGORG                                              $REG     3
                424      %W2          EQUALS    $REGORG                                              $REG     3
                425      %W3          EQUALS    $REGORG                                              $REG     3
                426      %W4          EQUALS    $REGORG                                              $REG     3
                427      %W5          EQUALS    $REGORG                                              $REG     3
                430      DAL          EQUALS    $REGORG                                              $REG     3
                431      TTY          EQUALS    $REGORG                                              $REG     3
                432      FC           EQUALS    $REGORG                                              $REG     3
                433      CHN          EQUALS    $REGORG                                              $REG     3
                434      TBL          EQUALS    $REGORG                                              $REG     3
                435      ORD          EQUALS    $REGORG                                              $REG     3
                436      TYPE         EQUALS    $REGORG                                              $REG     3
                437      OVL          EQUALS    $REGORG                                              $REG     3
                440      POINT        EQUALS    $REGORG                                              $REG     3
                441      MATCH        EQUALS    $REGORG                                              $REG     3

                                      EXI       %FEI                                                ADEM.18
```

```
EXT     %LSPI                                   ADEM.19
EXT     %UCT                                    ADEM.20
EXT     DY                                      ADEM.21
EXT     DZ                                      ADEM.22
EXT     EITB                                    ADEM.23
EXT     IUCIO                                   ADEM.24
EXT     NSCPACK                                 ADEM.25
EXT     O$CONC                                  ADEM.26
EXT     O$ENDCON                                ADEM.27
EXT     O$NSC                                   ADEM.28
EXT     O$NSCEND                                ADEM.29
EXT     UNID                                    ADEM.30
EXT     UTTY                                    ADEM.31
```

```
                        *       Get next entry from input queue and branch per the packet      ADEM.33
                        *       type.                                                          ADEM.34
                                                                                               ADEM.35
     6                  ADEM    *                                                              ADEM.36
     6  002000  000000          I = 0                                                          ADEM.37
    10   <macro>                ADDRESS   R!DZ,CPI@AQ,R!%LSPI .ADEM input queue               ADEM.38
    13   <macro>                RKERN     HGET                .Get top entry                   ADEM.39
                                                                                               ADEM.40
    15   <macro>                $IF       (R!DY = 0)          .If queue is empty               ADEM.41
                                                                                .>>>>> 0           $IF      1
    20   <macro>                  PKERN   ESWP                .Wait for a new message          ADEM.42
    22   <macro>                $ENDIF                                                          ADEM.43
                                                                                .<<<<< 0           $ENDIF   1
                                                                                               ADEM.44
    22  020000  024430          R!DAL = R!DY                  .Request packet address          ADEM.45
    24  003000                  I = 1                                                          ADEM.46
    25   <macro>                GET       R!%W1,DA@DIL,R!DAL  .Packet type                     ADEM.47
                                                                                               ADEM.48
    33   <macro>                $IF       (R!%W1 = RQ$TTY)    .If a terminal i/o request       ADEM.49
                                                                                .>>>>> 0           $IF      1
    37  070040                   P = TTYIO                                                     ADEM.50
                                                                                               ADEM.51
    40   <macro>                $ELSEIF   (R!%W1 = RQ$UCHN)   .If a user channel request       ADEM.52
    45  070534                   P = SHELL                                                     ADEM.53
                                                                                               ADEM.54
    46   <macro>                $ELSEIF   (R!%W1 = RQ$NPAK)   .If N-packet interface request   ADEM.55
    53  075000 /001117           P = NSC                                                       ADEM.56
                                                                                               ADEM.57
    55   <macro>                $ELSEIF   (R!%W1 = RQ$KERN)   .If a kernel function           ADEM.58
    62  070230                   P = AKERN                                                     ADEM.59
                                                                                               ADEM.60
    63   <macro>                $ELSE                         .If illegal packet type          ADEM.61
    64   <macro>                  $PUNTIF                                                     ADEM.62
    66   <macro>                $ENDIF                                                          ADEM.63
                                                                                .<<<<< 0           $ENDIF   1
                                                                                               ADEM.64
    66                  NULDAL  *                                                              ADEM.65
    66   <macro>                RELDAL    R!DAL               .Release DAL                     ADEM.66
    76  071070                  P = ADEM                                                       ADEM.67
```

```
                                 ***************************  ****************************************   ADEM.69
                                 *                                                                    *   ADEM.70
                                 *        Terminal i/o request                                        *   ADEM.71
                                 *                                                                    *   ADEM.72
                                 **********************************************************************   ADEM.73
                                                                                                        ADEM.74
  77                             TTYIO    *                                                           ADEM.75
                                                                                                        ADEM.76
                                 *          If the first T-packet, save the destination id.  It       ADEM.77
                                 *          contains Unicos routing information which must be         ADEM.78
                                 *          included in the source id of all T-packets sent to        ADEM.79
                                 *          the Cray.                                                 ADEM.80
                                                                                                        ADEM.81
  77   014000 /000000  012001            R!%W1 = UNID + 1                                              ADEM.82
       024423
                                                                                                        ADEM.83
 103    <macro>                           $IF       ((R!%W1) = 0)        .If first T-packet            ADEM.84
                                                                                        .>>>>> 0          $IF      1
 106   010001  034423                       (R!%W1) = 1                                                ADEM.85
 110    <macro>                             GET      R!%W1,DA@DID,R!DAL   .Get destination id          ADEM.86
 115   014000 /000000  024410               (UNID) = R!%W1               .Save for input messages      ADEM.87
       020423  034410
 122    <macro>                           $ENDIF                                                      ADEM.88
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        ADEM.89
 122    <macro>                           GET       A,DA@TIP,R!DAL       .Get destination              ADEM.90
                                                                                                        ADEM.91
 127    <macro>                           $IF       (A # 0)              .If for a FE terminal         ADEM.92
                                                                                        .>>>>> 0          $IF      1
 131   014000 /000000  024440               R!POINT = NSCPACK                                          ADEM.93
 134    <macro>                             GET      R!POINT,NF@TPK,R!POINT .Read / write tables       ADEM.94
 141   106053                               P = NULDAL, A = 0            .If not active                ADEM.95
                                                                                                        ADEM.96
 142   010014  025440                       R!POINT = R!POINT + NP$WRITE .Output side table            ADEM.97
 144   075000 /001322                       P = QUEUE                    .Queue message for output     ADEM.98
 146    <macro>                           $ENDIF                                                      ADEM.99
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        ADEM.100
 146    <macro>                           GET       R!FC,DA@TFC,R!DAL    .Request function code        ADEM.101
 154    <macro>                           GET       R!%W1,DA@TTY,R!DAL   .Tty number                   ADEM.102
 162   041000  013004  105076             P = NULDAL, A >= UT$NUM        .If invalid tty index         ADEM.103
                                                                                                        ADEM.104
```

```
165   020423  016000 /000000        R!TTY = (R!%W1 + UTTY)        .Table for this tty                ADEM.105
      024410  030410  024431
                                                                                                     ADEM.106
173    <macro>                      $IF       (A = 0)             .If not active                     ADEM.107
                                                                                  .>>>>> 0               $IF      1
175    <macro>                        $IF     (R!FC = TTF$RC)     .If a connect request              ADEM.108
                                                                                  .>>>>> 1               $IF      1
201    <macro>                          CREATE TTY,(O'177777,R!DAL)                                  ADEM.109
224   107136                            P = NULDAL, A # 0                                            ADEM.110
225   071217                            P = ADEM                                                     ADEM.111
226    <macro>                        $ELSE                                                          ADEM.112
227   071141                            P = NULDAL                .Ignore invalid request            ADEM.113
230    <macro>                        $ENDIF                                                         ADEM.114
                                                                                  .<<<<< 1               $ENDIF   1
                                                                                                     ADEM.115
230    <macro>                      $ELSEIF   (R!FC = TTF$DC)     .If a disconnect request           ADEM.116
235    <macro>                        CREATE  TTY,(O'177777,R!DAL)                                   ADEM.117
260   107172                          P = NULDAL, A # 0                                              ADEM.118
261   071253                          P = ADEM                                                       ADEM.119
262    <macro>                      $ENDIF                                                           ADEM.120
                                                                                  .<<<<< 0               $ENDIF   1
                                                                                                     ADEM.121
                            *        Put packet on output queue and start TTYOUT.                    ADEM.122
                                                                                                     ADEM.123
262   002000  000000                I = 0                                                            ADEM.124
264   020430  024000                R!DY = R!DAL                                                     ADEM.125
266    <macro>                      ADDRESS   R!DZ,UT@OUT,R!TTY   .Output queue address             ADEM.126
271    <macro>                      RKERN     HPUT                .Queue the message               ADEM.127
273   003000                        I = 1                                                            ADEM.128
274    <macro>                      ADDRESS   R!%W1,UT@POP,R!TTY                                     ADEM.129
277    <macro>                      POP       R!%W1,0             .Start TTYOUT                    ADEM.130
311   071303                        P = ADEM                                                         ADEM.131
```

```
                                 ***************************  ****************************************   ADEM.133
                                 *                                                                    *   ADEM.134
                                 *        Kernel request                                              *   ADEM.135
                                 *                                                                    *   ADEM.136
                                 ***********************************************************************   ADEM.137
                                                                                                        ADEM.138
 312                             AKERN    *                                                             ADEM.139
 312    <macro>                           GET       R!FC,CXKFC,R!DAL     .Function code                 ADEM.140
                                                                                                        ADEM.141
 320    <macro>                           $IF       (R!FC # KF$CONC),AND,(R!FC # KF$ENDC) .If unknown   ADEM.142
                                                                                        .>>>>> 0          $IF      1
 327    071241                              P = NULDAL                                                   ADEM.143
 330    <macro>                           $ENDIF                                                       ADEM.144
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        ADEM.145
 330    <macro>                           GET       R!ORD,CXCHN,R!DAL    .Channel ordinal               ADEM.146
 336    020000  012002  013007            R!TBL = R!%FEI + FEI@LH - FEI@LE                             ADEM.147
        024434
 342    010013  024423                    R!%W1 = I@NCOR                                               ADEM.148
 344    010000  024424                    R!%W2 = 0                                                    ADEM.149
                                                                                                        ADEM.150
 346    <macro>                           $UNTIL    (R!%W1 = 0),OR,(R!ORD = R!%W2) .Search for FEI entry ADEM.151
 353    010007  025434                      R!TBL = R!TBL + FEI@LE                                     ADEM.152
 355    <macro>                             GET     R!%W2,FEI@CO,R!TBL                                 ADEM.153
 364    027423                              R!%W1 = R!%W1 - 1                                          ADEM.154
 365    <macro>                           $ENDTIL                                                      ADEM.155
                                                                                                        ADEM.156
 366    <macro>                           $IF       (R!ORD = R!%W2)      .If entry found               ADEM.157
                                                                                        .>>>>> 0          $IF      1
                                                                                                        ADEM.158
                                 *          Create the appropriate activity                            ADEM.159
                                                                                                        ADEM.160
 372    010000  024437                      R!OVL = 0                                                  ADEM.161
 374    010000  024424                      R!%W2 = 0                                                  ADEM.162
 376    <macro>                             GET     R!TYPE,FEI@CT,R!TBL .Get the channel type          ADEM.163
                                                                                                        ADEM.164
 401    <macro>                             $IF     (R!TYPE = CH$CN)     .If a FEI channel             ADEM.165
                                                                                        .>>>>> 1          $IF      1
 405    <macro>                               $IF   (R!FC = KF$CONC)                                   ADEM.166
                                                                                        .>>>>> 2          $IF      1
 410    014000 /000000  024437                  R!OVL = O$CONC                                         ADEM.167
 413    <macro>                               $ELSE                                                    ADEM.168
```

```
414   014000 /000000  024437          R!OVL = O$ENDCON                                          ADEM.169
417   010001  024424                  R!%W2 = 1                                                 ADEM.170
421    <macro>                      $ENDIF                                                      ADEM.171
                                                                          .<<<<< 2                $ENDIF   1
421   020435  013003  024435          R!ORD = R!ORD - I@COCO        .Form the CONC number     R16196GA.1
                                                                                                ADEM.173
424    <macro>                    $ELSEIF (R!TYPE = CH$NS)     .If an NSC channel              ADEM.174
431    <macro>                      $IF   (R!FC = KF$CONC)                                      ADEM.175
                                                                          .>>>>> 2                $IF      1
434   014000 /000000  024437          R!OVL = O$NSC                                             ADEM.176
437    <macro>                      $ELSE                                                       ADEM.177
440   014000 /000000  024437          R!OVL = O$NSCEND                                          ADEM.178
443   010001  024424                  R!%W2 = 1                                                 ADEM.179
445    <macro>                      $ENDIF                                                      ADEM.180
                                                                          .<<<<< 2                $ENDIF   1
445    <macro>                    $ENDIF                                                        ADEM.181
                                                                          .<<<<< 1                $ENDIF   1
                                                                                                ADEM.182
445    <macro>                    $IF     (R!OVL # 0)          .If a front-end channel          ADEM.183
                                                                          .>>>>> 1                $IF      1
450    <macro>                      PUT   R!%W2,FEI@OF,R!TBL  .Enable/disable the channel       ADEM.184
461    <macro>                       CREATE R!OVL,(R!ORD),TYPE=NUMBER  .Start Activity        R16196GA.2
                                                                                              R16196GA.3
500    <macro>                    $ENDIF                                                        ADEM.186
                                                                          .<<<<< 1                $ENDIF   1
                                                                                                ADEM.187
500    <macro>                  $ELSE                                                           ADEM.188
                                                                                                ADEM.189
                            *     Look for an active NSC ID                                     ADEM.190
                                                                                                ADEM.191
501    <macro>                    GET     R!TBL,FEI@ON,R!%FEI                                   ADEM.192
506   020434  012000  013002       R!TBL = R!TBL + COR@LH - COR@LE                               ADEM.193
      024434
512   010050  024423               R!%W1 = COR@NE                                                ADEM.194
                                                                                                ADEM.195
514    <macro>                    $UNTIL  (R!%W1 = 0),OR,(R!ORD = R!%W2) .Search for NSC entry  ADEM.196
521   010002  025434                R!TBL = R!TBL + COR@LE                                      ADEM.197
523    <macro>                      GET   R!%W2,COR@OR,R!TBL                                    ADEM.198
526   027423                        R!%W1 = R!%W1 - 1                                           ADEM.199
527    <macro>                    $ENDTIL                                                       ADEM.200
                                                                                                ADEM.201
530    <macro>                    $IF     (R!ORD = R!%W2)      .If found                        ADEM.202
```

```
                                                                .>>>>> 1          $IF      1
534   <macro>                 $IF   (R!FC = KF$CONC)                          ADEM.203
                                                                .>>>>> 2          $IF      1
537   <macro>                   STORE 0,COR@OF,R!TBL    .Enable the ordinal   ADEM.204
543   <macro>                 $ELSE                                           ADEM.205
544   <macro>                   GET R!%W1,COR@AC,R!TBL                        ADEM.206
547   <macro>                   STORE 1,COR@OF,R!TBL    .Shut off the ordinal ADEM.207
555   <macro>                   $IF (R!%W1 # 0)         .If id is active      ADEM.208
                                                                .>>>>> 3          $IF      1
560   <macro>                     CREATE NIDEND,(R!ORD),PRI=5                 ADEM.209
600   <macro>                   $ENDIF                                        ADEM.210
                                                                .<<<<< 3          $ENDIF   1
600   <macro>                 $ENDIF                                          ADEM.211
                                                                .<<<<< 2          $ENDIF   1
600   <macro>               $ENDIF                                            ADEM.212
                                                                .<<<<< 1          $ENDIF   1
600   <macro>             $ENDIF                                              ADEM.213
                                                                .<<<<< 0          $ENDIF   1
                                                                                  ADEM.214
600   071512              P = NULDAL                  .Release the packet       ADEM.215
```

```
                                 ********************************************************************   ADEM.217
                                 *                                                                  *   ADEM.218
                                 *        Shell request                                             *   ADEM.219
                                 *                                                                  *   ADEM.220
                                 ********************************************************************   ADEM.221
                                                                                                        ADEM.222
601                              SHELL    *                                                             ADEM.223
601     <macro>                           GET       R!%W1,CR@IOP,R!DAL  .Target iop                     ADEM.224
                                                                                                        ADEM.225
610     <macro>                           $IF       (R!%W1 = CR$NSC)    .If for NSC interface           ADEM.226
                                                                                          .>>>>> 0          $IF       1
614   075000 /001117                        P = NSC                                                     ADEM.227
616     <macro>                           $ELSEIF    (R!%W1 # MIOP)     .If not for this iop            ADEM.228
622   010004  024423                        R!%W1 = CR$CHN                                              ADEM.229
624   075000 /001023                        P = UCERR                   .Return an error response       ADEM.230
626     <macro>                           $ENDIF                                                        ADEM.231
                                                                                          .<<<<< 0          $ENDIF    1
                                                                                                        ADEM.232
                                 *        Locate user channel table entry.                              ADEM.233
                                                                                                        ADEM.234
626     <macro>                           GET       R!%W1,UC@NAM,R!%UCT .UCT header name                ADEM.235
630     <macro>                           $PUNTIF   (R!%W1 # UC$NAM)    .If table smashed             ADEM.236
                                                                                                        ADEM.237
635     <macro>                           GET       R!CHN,CR@CHN,R!DAL  .Channel number                 ADEM.238
643     <macro>                           GET       R!TBL,UC@FST,R!%UCT .Link to first table            ADEM.239
                                                                                                        ADEM.240
650     <macro>                           $UNTIL    (R!TBL = 0)         .Until all tables scanned       ADEM.241
652     <macro>                             GET     A,UC@CHN,R!TBL      .Channel number                 ADEM.242
654   023433  102102                        P = FOUND, A = R!CHN        .If found                       ADEM.243
656     <macro>                             GET     R!TBL,UC@FOR,R!TBL  .Next entry                     ADEM.244
663     <macro>                           $ENDTIL                                                       ADEM.245
                                                                                                        ADEM.246
                                 *        Entry not found.  Validate the channel number.                ADEM.247
                                                                                                        ADEM.248
664     <macro>                           $IF       (R!CHN < O'50),AND,(IUCIO = (R!CHN + EITB))         ADEM.249
                                                                                          .>>>>> 0          $IF       1
                                 *          Normal user channel definition.                             ADEM.250
701     <macro>                           $ELSEIF   (R!CHN = UN$PRINT)                                  ADEM.251
                                 *          Printer pseudo-channel.                                     ADEM.252
706     <macro>                           $ELSEIF   (R!CHN = UN$TAPE)                                   ADEM.253
                                 *          Tape pseudo-channel.                                        ADEM.254
713     <macro>                           $ELSEIF   (R!CHN = UN$DISK)                                   ADEM.255
```

```
                                *         Disk pseudo-chan                                          ADEM.256
  720   <macro>                           $ELSEIF   (R!CHN = UN$FILE)                               ADEM.257
                                *         Disk file system pseudo-channel.                          ADEM.258
  725   <macro>                           $ELSE                                                     ADEM.259
  726   010004  024423                      R!%W1 = CR$CHN                .Illegal channel number   ADEM.260
  730   075000 /001023                      P = UCERR                                               ADEM.261
  732   <macro>                           $ENDIF                                                    ADEM.262
                                                                                  .<<<<< 0              $ENDIF    1
                                                                                                      ADEM.263
                                *         Create a new shell activity for this channel.             ADEM.264
                                                                                                      ADEM.265
  732   <macro>                           CREATE    UCSHL,(R!DAL),PRI=1                             ADEM.266
  752   106744                            P = ADEM, A = 0               .If successful              ADEM.267
  753   010011  024423                    R!%W1 = CR$OUT                                            ADEM.268
  755   075000 /001023                    P = UCERR                                                 ADEM.269
                                                                                                      ADEM.270
                                *         Queue the message for the shell and activate it.          ADEM.271
                                                                                                      ADEM.272
  757                           FOUND     *                                                         ADEM.273
  757   <macro>                           ADDRESS   R!%W2,UC@RQQ,R!TBL  .Request queue address     ADEM.274
  762   020424  012001  024410            R!%W1 = (R!%W2 + 1)           .Last entry address         ADEM.275
        030410  024423
  767   020423  103003  020424            R!%W1 = R!%W2, R!%W1 = 0      .If queue is empty          ADEM.276
        024423
  773   020430  034423                    (R!%W1) = R!DAL               .Link onto queue            ADEM.277
  775   020424  012001  024410            (R!%W2 + 1) = R!DAL           .New tail entry             ADEM.278
        020430  034410
 1002   010000  034430                    (R!DAL) = 0                                               ADEM.279
 1004   <macro>                           ADDRESS   R!%W1,UC@PSQ,R!TBL  .Wait queue address        ADEM.280
 1007   <macro>                           POP       R!%W1,0             .Activate shell             ADEM.281
 1021   075000 /000006                    P = ADEM                                                  ADEM.282
                                                                                                      ADEM.283
                                *         Return an arror response to the cpu.                      ADEM.284
                                                                                                      ADEM.285
 1023                           UCERR     *                                                         ADEM.286
 1023   <macro>                           KRECORD   TF$UCH1,(R!CHN,0,0,R!DAL,R!%W1)                 ADEM.287
                                *                   Channel                                           ADEM.288
                                *                   0                                                 ADEM.289
                                *                   0                                                 ADEM.290
                                *                   Message address                                   ADEM.291
                                *                   Response code                                     ADEM.292
                                                                                                      ADEM.293
```

```
1042    <macro>          PUT       R!%W1,CR@STS,R!DAL  .Save response code      ADEM.294
1055    <macro>          GET       R!%W1,CR@DID,R!DAL                          ADEM.295
1062    <macro>          GET       R!%W2,CR@SID,R!DAL                          ADEM.296
1067    <macro>          PUT       R!%W2,CR@DID,R!DAL  .Swap sid and did       ADEM.297
1074    <macro>          PUT       R!%W1,CR@SID,R!DAL                          ADEM.298
1101    <macro>          SEND      R!DAL               .Send response to cpu   ADEM.299
1115   075000 /000006    P = ADEM                                              ADEM.300
```

```
                                 **************************   ****************************************   ADEM.302
                                 *                                                                    *   ADEM.303
                                 *        NSC interface request                                       *   ADEM.304
                                 *                                                                    *   ADEM.305
                                 **********************************************************************   ADEM.306
                                                                                                        ADEM.307
1117                             NSC      *                                                             ADEM.308
                                                                                                        ADEM.309
                                 *        Branch per the request type.                                  ADEM.310
                                                                                                        ADEM.311
1117    <macro>                          GET       R!%W1,CR@CFN,R!DAL  .Request type                   ADEM.312
                                                                                                        ADEM.313
1125    <macro>                          $IF       (R!%W1 = CR$RD)     .If read                        ADEM.314
                                                                                        .>>>>> 0            $IF      1
1131    010005  024432                     R!FC = CR$RD                                                 ADEM.315
1133    <macro>                          $ELSEIF   (R!%W1 = CR$RDH)                                    ADEM.316
1140    010005  024432                     R!FC = CR$RD                                                 ADEM.317
1142    <macro>                          $ELSEIF   (R!%W1 = CR$RD2)                                    ADEM.318
1147    010005  024432                     R!FC = CR$RD                                                 ADEM.319
1151    <macro>                          $ELSEIF   (R!%W1 = CR$WRT)    .If write                       ADEM.320
1156    010010  024432                     R!FC = CR$WRT                                                ADEM.321
1160    <macro>                          $ELSEIF   (R!%W1 = CR$WRTH)                                   ADEM.322
1165    010010  024432                     R!FC = CR$WRT                                                ADEM.323
1167    <macro>                          $ELSEIF   (R!%W1 = CR$WRT2)                                   ADEM.324
1174    010010  024432                     R!FC = CR$WRT                                                ADEM.325
1176    <macro>                          $ELSEIF   (R!%W1 = CR$OPN),OR,(R!%W1 = CR$CLS) .If open/close ADEM.326
1206    070160                             P = FUNCTION                                                 ADEM.327
1207    <macro>                          $ELSEIF   (R!%W1 = CR$ALP),OR,(R!%W1 = CR$RLP) .If assign/rel ADEM.328
1217    070147                             P = FUNCTION                                                 ADEM.329
1220    <macro>                          $ELSE                         .If invalid request             ADEM.330
1221    010005  024423                     R!%W1 = CR$CFN              .Response code                   ADEM.331
1223    071200                             P = UCERR                                                    ADEM.332
1224    <macro>                          $ENDIF                                                        ADEM.333
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                        ADEM.334
                                 *        Search for the associated table based on the channel number,  ADEM.335
                                 *        owner, and path number.                                       ADEM.336
                                                                                                        ADEM.337
1224    <macro>                          GET       R!MATCH,CR@CHN,R!DAL .Channel number                ADEM.338
1232    020441  015000 /177776           R!MATCH = R!MATCH & O'177776  .Even numbered side             ADEM.339
        024441
1236    014000 /000000  024440           R!POINT = NSCPACK                                             ADEM.340
```

```
1241   <macro>                GET       R!POINT,NF@CHQ,R!POINT                          ADEM.341
1243   072207                 R = FIND                        .Find channel table       ADEM.342
1244   103117                 P = NOPATH, A # 0                                         ADEM.343
                                                                                         ADEM.344
1245   <macro>                GET       R!MATCH,CR@OWN,R!DAL .Owner identification      ADEM.345
1253   <macro>                GET       R!POINT,NC@OPN,R!POINT                          ADEM.346
1260   072172                 R = FIND                        .Find open channel table  ADEM.347
1261   103102                 P = NOPATH, A # 0                                         ADEM.348
                                                                                         ADEM.349
1262   <macro>                GET       R!MATCH,CR@LP,R!DAL .Logical path number        ADEM.350
1270   <macro>                GET       R!POINT,NO@PTH,R!POINT                          ADEM.351
1275   072155                 R = FIND                        .Find logical path table  ADEM.352
1276   103065                 P = NOPATH, A # 0                                         ADEM.353
                                                                                         ADEM.354
1277   <macro>                GET       A,NP@TRM,R!POINT                                ADEM.355
                                                                                         ADEM.356
1304   <macro>                $IF       (A # 0)               .If path terminated       ADEM.357
                                                                          .>>>>> 0          $IF      1
1306   010054  024423           R!%W1 = NSC$TRM                                         ADEM.358
1310   071265                   P = UCERR                                               ADEM.359
1311   <macro>                $ENDIF                                                    ADEM.360
                                                                          .<<<<< 0          $ENDIF   1
                                                                                         ADEM.361
1311   <macro>                $IF       (R!FC = CR$RD)        .If read                  ADEM.362
                                                                          .>>>>> 0          $IF      1
1315   010000  025440           R!POINT = R!POINT + NP$READ .Read side table            ADEM.363
1317   <macro>                $ELSE                                                     ADEM.364
1320   010014  025440           R!POINT = R!POINT + NP$WRITE .Write side table          ADEM.365
1322   <macro>                $ENDIF                                                    ADEM.366
                                                                          .<<<<< 0          $ENDIF   1
                                                                                         ADEM.367
1322                   QUEUE   *                                                         ADEM.368
1322   002000  000000         I = 0                                                     ADEM.369
1324   020430  024000         R!DY = R!DAL                                              ADEM.370
1326   <macro>                ADDRESS   R!DZ,NP@PAK,R!POINT .Packet queue address     ADEM.371
1331   <macro>                RKERN     HPUT                .Queue packet               ADEM.372
1333   003000                 I = 1                                                     ADEM.373
1334   <macro>                GET       A,NP@BSY,R!POINT                                ADEM.374
                                                                                         ADEM.375
1342   <macro>                $IF       (A = 0)               .If task not busy         ADEM.376
                                                                          .>>>>> 0          $IF      1
1344   <macro>                  ADDRESS R!%W1,NP@PSH,R!POINT .Wait queue address      ADEM.377
```

```
1347    <macro>                          POP     R!%W1,0                                                    ADEM.378
1361    <macro>                        $ENDIF                                                               ADEM.379
                                                                                   .<<<<< 0                     $ENDIF    1
                                                                                                            ADEM.380
1361    075000 /000006                 P = ADEM                                                             ADEM.381
                                                                                                            ADEM.382
                                *      No entry found for this channel, owner, and logical path.            ADEM.383
                                *      Return an error response.                                            ADEM.384
                                                                                                            ADEM.385
1363                            NOPATH *                                                                    ADEM.386
1363    010004  024423                 R!%W1 = CR$CHN                  .Response code                       ADEM.387
1365    071342                         P = UCERR                                                            ADEM.388
                                                                                                            ADEM.389
                                *      For open, close, and assign and release logical path                 ADEM.390
                                *      functions, queue the message for processing by the FNSC              ADEM.391
                                *      overlay.                                                             ADEM.392
                                                                                                            ADEM.393
1366                            FUNCTION *                                                                  ADEM.394
1366    014000 /000000  024440         R!POINT = NSCPACK                                                    ADEM.395
1371    <macro>                        GET     A,NF@QHD,R!POINT        .Message queue                       ADEM.396
                                                                                                            ADEM.397
1375    <macro>                        $IF     (A = 0)                 .If no messages queued               ADEM.398
                                                                                                   .>>>>> 0     $IF       1
1377    <macro>                          CREATE  FNSC,(R!POINT)        .Create NSC function processor       ADEM.399
                                                                                                            ADEM.400
1417    <macro>                          $IF     (A # 0)               .If create failed                    ADEM.401
                                                                                                   .>>>>> 1     $IF       1
1421    010011  024423                     R!%W1 = CR$OUT              .Response code                       ADEM.402
1423    071400                             P = UCERR                                                        ADEM.403
1424    <macro>                          $ENDIF                                                             ADEM.404
                                                                                                   .<<<<< 1     $ENDIF    1
                                                                                                            ADEM.405
1424    <macro>                          PUT     R!DAL,NF@QHD,R!POINT .Pointer to first entry               ADEM.406
                                                                                                            ADEM.407
1431    <macro>                        $ELSE                                                                ADEM.408
1432    <macro>                          GET     R!%W1,NF@QTL,R!POINT .Last entry                           ADEM.409
1437    <macro>                          PUT     R!DAL,DA@LNK,R!%W1  .Link to new entry                     ADEM.410
1441    <macro>                        $ENDIF                                                               ADEM.411
                                                                                   .<<<<< 0                     $ENDIF    1
                                                                                                            ADEM.412
1441    <macro>                        PUT     R!DAL,NF@QTL,R!POINT .Pointer to last entry                 ADEM.413
1446    <macro>                        STORE   0,DA@LNK,R!DAL        .Clear link word                       ADEM.414
```

1450 075000 /000006 P = ADEM ADEM.415

```
                        *       Find table.                                                     ADEM.417
                        *                                                                       ADEM.418
                        *       R!POINT   First table address                                   ADEM.419
                        *       R!MATCH   Match value                                           ADEM.420
                                                                                                ADEM.421
1452                    FIND    *                                                               ADEM.422
1452   <macro>                  $UNTIL    (R!POINT = 0)        .Until all entries scanned       ADEM.423
1454   <macro>                    GET     R!%W1,NC@CHN,R!POINT .Match value                    ADEM.424
                                                                                                ADEM.425
1461   <macro>                    $IF     (R!%W1 = R!MATCH)    .If matching entry               ADEM.426
                                                                                .>>>>> 0            $IF      1
1465   010000                       A = 0                     .Return success                  ADEM.427
1466   001000                       EXIT                                                       ADEM.428
1467   <macro>                    $ENDIF                                                       ADEM.429
                                                                                .<<<<< 0            $ENDIF   1
                                                                                                ADEM.430
1467   <macro>                    GET     R!POINT,NC@LNK,R!POINT .Next entry address           ADEM.431
1471   <macro>                  $ENDTIL                                                        ADEM.432
                                                                                                ADEM.433
1472   010001                   A = 1                          .Return failure                  ADEM.434
1473   001000                   EXIT                                                            ADEM.435
                                                                                                ADEM.436
                                END                                                             ADEM.437
```

0X	%ACTIVE		12: 6 D	12: 6							
0X	%B		1: 4 D	4:27 B	5:35 B	10:12 B	10:15 B	11: 6 B	11:15 B	11:31 B	
			12: 7 B	15: 5 B	16: 1 B						
0X	%EX		4:38 D	4:38	6: 8 D	6: 8	6:17 D	6:17	6:31 D	6:31	
			8:22 D	8:22	9:10 D	9:10	11:12 D	11:12	11:30 D	11:30	
			12: 6 D	12: 6	15: 1 D	15: 1	15:24 D	15:24			
0X	%FEI		2:42 D	7:17	8:31						
10	%GBLNUM	$APTEXT	2:13 F	2:13							
400	%GBLREG	$APTEXT	2:13								
0	%KERNEL	$APTEXT	4: 7 D	4: 7 B	4:11 D	4:11 B	6:28 D	6:28 B	14:36 D	14:36 B	
0X	%LSPI		3: 1 D	4: 6							
0X	%PUNI		4:33 D	4:33	10:22 D	10:22					
0X	%TR1		11:36 D	11:36							
0X	%TR2		11:36 D	11:36							
0X	%TR3		11:36 D	11:36							
0X	%TR4		11:36 D	11:36							
0X	%TR5		11:36 D	11:36							
0X	%TRAC		11:36 D	11:36							
0X	%TVNT		11:36 D	11:36							
0X	%UCT		3: 2 D	10:21	10:25						
177777	AD$NODEM	$APTEXT	6: 8	6:17	8:22	9:10	11:12	15:24			
20	AD@RC@N	$APTEXT	12: 6	12: 6 F							
13	AD@RC@P	$APTEXT	12: 6 F	12: 6							
0	AD@RC@S	$APTEXT	12: 6	12: 6 F							
36	CH$CN	$APTEXT	7:38								
42	CH$NS	$APTEXT	8: 7								
1	COR@AC@N	$APTEXT	9: 6	9: 6 F							
0	COR@AC@P	$APTEXT	9: 6 F								
0	COR@AC@S	$APTEXT	9: 6	9: 6 F							
2	COR@LE	$APTEXT	8:32	8:37							
0	COR@LH	$APTEXT	8:32								
50	COR@NE	$APTEXT	8:34								
1	COR@OF@M	$APTEXT	9: 4 F	9: 7 F							
1	COR@OF@N	$APTEXT	9: 4 F	9: 7 F	9: 7						
0	COR@OF@P	$APTEXT	9: 4 F	9: 7 F							
1	COR@OF@S	$APTEXT	9: 4 F	9: 7 F	9: 7						
137777	COR@OF@X	$APTEXT	9: 4								
377	COR@OR@M	$APTEXT	8:38								
10	COR@OR@N	$APTEXT	8:38								
0	COR@OR@P	$APTEXT	8:38 F								
10	COR@OR@S	$APTEXT	8:38	8:38 F							
14	CPI@AQ@P	$APTEXT	4: 6								

40	CR$ALP	$APTEXT	13:28				
5	CR$CFN	$APTEXT	13:31				
4	CR$CHN	$APTEXT	10:14	11: 5	15:11		
4	CR$CLS	$APTEXT	13:26				
7	CR$NSC	$APTEXT	10:11				
3	CR$OPN	$APTEXT	13:26				
11	CR$OUT	$APTEXT	11:14	15:28			
5	CR$RD	$APTEXT	13:14	13:15	13:17	13:19	14:25
7	CR$RD2	$APTEXT	13:18				
6	CR$RDH	$APTEXT	13:16				
41	CR$RLP	$APTEXT	13:28				
10	CR$WRT	$APTEXT	13:20	13:21	13:23	13:25	
12	CR$WRT2	$APTEXT	13:24				
11	CR$WRTH	$APTEXT	13:22				
10	CR@CFN@N	$APTEXT	13:11	13:11 F			
13	CR@CFN@P	$APTEXT	13:11 F	13:11			
0	CR@CFN@S	$APTEXT	13:11	13:11 F			
77	CR@CHN@M	$APTEXT	10:24	13:39			
6	CR@CHN@N	$APTEXT	10:24	13:39			
12	CR@CHN@P	$APTEXT	10:24 F	10:24	13:39 F	13:39	
12	CR@CHN@S	$APTEXT	10:24	10:24 F	13:39	13:39 F	
20	CR@DID@N	$APTEXT	12: 2	12: 2 F	12: 4 F		
10	CR@DID@P	$APTEXT	12: 2 F	12: 2	12: 4 F	12: 4	
0	CR@DID@S	$APTEXT	12: 2	12: 2 F	12: 4 F		
7	CR@IOP@M	$APTEXT	10: 8				
3	CR@IOP@N	$APTEXT	10: 8	10: 8 F			
12	CR@IOP@P	$APTEXT	10: 8 F	10: 8			
7	CR@IOP@S	$APTEXT	10: 8	10: 8 F			
377	CR@LP@M	$APTEXT	14:10				
10	CR@LP@N	$APTEXT	14:10				
37	CR@LP@P	$APTEXT	14:10 F	14:10			
10	CR@LP@S	$APTEXT	14:10	14:10 F			
7	CR@OWN@N	$APTEXT	14: 5	14: 5 F			
12	CR@OWN@P	$APTEXT	14: 5 F	14: 5			
0	CR@OWN@S	$APTEXT	14: 5	14: 5 F			
20	CR@SID@N	$APTEXT	12: 3	12: 3 F	12: 5 F		
11	CR@SID@P	$APTEXT	12: 3 F	12: 3	12: 5 F	12: 5	
0	CR@SID@S	$APTEXT	12: 3	12: 3 F	12: 5 F		
177	CR@SIS@M	$APTEXT	12: 1				
7	CR@SIS@N	$APTEXT	12: 1 F				
13	CR@SIS@P	$APTEXT	12: 1 F	12: 1			
11	CR@SIS@S	$APTEXT	12: 1 F				

177600	CR@SIS@X	$APTEXT	12: 1						
10	CXCHN@N	$APTEXT	7:16		7:16	F			
12	CXCHN@P	$APTEXT	7:16	F	7:16				
0	CXCHN@S	$APTEXT	7:16		7:16	F			
17	CXKFC@M	$APTEXT	7: 8						
4	CXKFC@N	$APTEXT	7: 8						
13	CXKFC@P	$APTEXT	7: 8	F	7: 8				
14	CXKFC@S	$APTEXT	7: 8		7: 8	F			
20	DA@DID@N	$APTEXT	5:20		5:20	F			
10	DA@DID@P	$APTEXT	5:20	F	5:20				
0	DA@DID@S	$APTEXT	5:20		5:20	F			
377	DA@DIL@M	$APTEXT	4:17						
10	DA@DIL@N	$APTEXT	4:17						
10	DA@DIL@P	$APTEXT	4:17	F	4:17				
10	DA@DIL@S	$APTEXT	4:17		4:17	F			
20	DA@LNK@N	$APTEXT	15:37	F	15:42	F			
0	DA@LNK@P	$APTEXT	15:37	F	15:42	F			
0	DA@LNK@S	$APTEXT	15:37	F	15:42	F			
4	DA@TFC@N	$APTEXT	5:39		5:39	F			
13	DA@TFC@P	$APTEXT	5:39	F	5:39				
0	DA@TFC@S	$APTEXT	5:39		5:39	F			
10	DA@TIP@N	$APTEXT	5:26		5:26	F			
12	DA@TIP@P	$APTEXT	5:26	F	5:26				
0	DA@TIP@S	$APTEXT	5:26		5:26	F			
377	DA@TTY@M	$APTEXT	5:40						
10	DA@TTY@N	$APTEXT	5:40						
12	DA@TTY@P	$APTEXT	5:40	F	5:40				
10	DA@TTY@S	$APTEXT	5:40		5:40	F			
0X	DY		3: 3	D	4:10		4:15	6:26	14:34
0X	DZ		3: 4	D	4: 6		6:27	14:35	
0X	EITB		3: 5	D	10:36				
0X	ESWP		4:11	D	4:11				
55	F$CREATE	$APTEXT	6: 8		6:17		8:22	9:10	11:12	15:24
2	F$POP	$APTEXT	6:31		11:30		15: 1		
27	F$RDAL	$APTEXT	4:38						
34	F$SEND	$APTEXT	12: 6						
77	FEI@CO@M	$APTEXT	7:24						
6	FEI@CO@N	$APTEXT	7:24		7:24	F			
1	FEI@CO@P	$APTEXT	7:24	F	7:24				
1	FEI@CO@S	$APTEXT	7:24		7:24	F			
377	FEI@CT@M	$APTEXT	7:35						
10	FEI@CT@N	$APTEXT	7:35						

0	FEI@CI@P	$APTEXT	7:35	F					
10	FEI@CI@S	$APTEXT	7:35		7:35	F			
7	FEI@LE	$APTEXT	7:17		7:23				
2	FEI@LH	$APTEXT	7:17						
1	FEI@OF@M	$APTEXT	8:21						
1	FEI@OF@N	$APTEXT	8:21	F	8:21				
0	FEI@OF@P	$APTEXT	8:21	F					
3	FEI@OF@S	$APTEXT	8:21	F	8:21				
20	FEI@ON@N	$APTEXT	8:31		8:31	F			
1	FEI@ON@P	$APTEXT	8:31	F	8:31				
0	FEI@ON@S	$APTEXT	8:31		8:31	F			
0*	HGET		4: 7	D	4: 7				
0*	HPUT		6:28	D	6:28		14:36	D	14:36
3	I@COCO	$APTEXT	8: 5						
13	I@NCOR	$APTEXT	7:19						
0*	IUCIO		3: 6	D	10:36				
0	KF$CONC	$APTEXT	7:11		7:40		8: 9		9: 3
1	KF$ENDC	$APTEXT	7:11						
0	MIOP	$APTEXT	10:13						
20	NC@CHN@N	$APTEXT	17: 8		17: 8	F			
1	NC@CHN@P	$APTEXT	17: 8	F	17: 8				
0	NC@CHN@S	$APTEXT	17: 8		17: 8	F			
20	NC@LNK@N	$APTEXT	17:17		17:17	F			
0	NC@LNK@P	$APTEXT	17:17	F					
0	NC@LNK@S	$APTEXT	17:17		17:17	F			
20	NC@OPN@N	$APTEXT	14: 6		14: 6	F			
3	NC@OPN@P	$APTEXT	14: 6	F	14: 6				
0	NC@OPN@S	$APTEXT	14: 6		14: 6	F			
20	NF@CHQ@N	$APTEXT	14: 1		14: 1	F			
0	NF@CHQ@P	$APTEXT	14: 1	F					
0	NF@CHQ@S	$APTEXT	14: 1		14: 1	F			
20	NF@QHD@N	$APTEXT	15:20		15:20	F	15:33	F	
1	NF@QHD@P	$APTEXT	15:20	F	15:20		15:33	F	15:33
0	NF@QHD@S	$APTEXT	15:20		15:20	F	15:33	F	
20	NF@QTL@N	$APTEXT	15:36		15:36	F	15:41	F	
2	NF@QTL@P	$APTEXT	15:36	F	15:36		15:41	F	15:41
0	NF@QTL@S	$APTEXT	15:36		15:36	F	15:41	F	
20	NF@TPK@N	$APTEXT	5:31		5:31	F			
3	NF@TPK@P	$APTEXT	5:31	F	5:31				
0	NF@TPK@S	$APTEXT	5:31		5:31	F			
20	NO@PTH@N	$APTEXT	14:11		14:11	F			
3	NO@PTH@P	$APTEXT	14:11	F	14:11				

0	NO@PTH@S	$APTEXT	14:11		14:11	F			
0	NP$READ	$APTEXT	14:26						
14	NP$WRITE	$APTEXT	5:34		14:28				
1	NP@BSY@M	$APTEXT	14:38						
1	NP@BSY@N	$APTEXT	14:38		14:38	F			
2	NP@BSY@P	$APTEXT	14:38	F	14:38				
1	NP@BSY@S	$APTEXT	14:38		14:38	F			
5	NP@PAK@P	$APTEXT	14:35						
10	NP@PSH@P	$APTEXT	14:42						
1	NP@TRM@N	$APTEXT	14:15		14:15	F			
2	NP@TRM@P	$APTEXT	14:15	F	14:15				
0	NP@TRM@S	$APTEXT	14:15		14:15	F			
54	NSC$TRM	$APTEXT	14:19						
0X	NSCPACK		3: 7	D	5:30		13:42		15:19
0X	O$ADEM		1: 4	D	1: 4				
0X	O$CONC		3: 8	D	7:41				
0X	O$ENDCON		3: 9	D	8: 1				
0X	O$FNSC		15:24	D	15:24				
0X	O$NIDEND		9:10	D	9:10				
0X	O$NSC		3:10	D	8:10				
0X	O$NSCEND		3:11	D	8:12				
0X	O$TTY		6: 8	D	6: 8		6:17	D	6:17
0X	O$UCSHL		11:12	D	11:12				
17	OV@NUM@N	$APTEXT	1: 4						
1	OV@TYP@N	$APTEXT	1: 4						
0	PT$HUH	$APTEXT	4:33		10:22				
113	RQ$KERN	$APTEXT	4:29						
116	RQ$NPAK	$APTEXT	4:26						
124	RQ$TTY	$APTEXT	4:20						
106	RQ$UCHN	$APTEXT	4:23						
11	SM@FST@N	$APTEXT	1: 4						
7	SM@NUM@N	$APTEXT	1: 4						
61	TF$UCH1	$APTEXT	11:36						
3	TTF$DC	$APTEXT	6:16						
4	TTF$RC	$APTEXT	6: 7						
52503	UC$NAM	$APTEXT	10:22						
10	UC@CHN@N	$APTEXT	10:28		10:28	F			
0	UC@CHN@P	$APTEXT	10:28	F					
0	UC@CHN@S	$APTEXT	10:28		10:28	F			
20	UC@FOR@N	$APTEXT	10:30		10:30	F			
2	UC@FOR@P	$APTEXT	10:30	F	10:30				
0	UC@FOR@S	$APTEXT	10:30		10:30	F			

20	UC@FSI@N	$APTEXT	10:25	10:25 F	
2	UC@FST@P	$APTEXT	10:25 F	10:25	
0	UC@FST@S	$APTEXT	10:25	10:25 F	
20	UC@NAM@N	$APTEXT	10:21	10:21 F	
0	UC@NAM@P	$APTEXT	10:21 F		
0	UC@NAM@S	$APTEXT	10:21	10:21 F	
4	UC@PSQ@P	$APTEXT	11:29		
6	UC@RQQ@P	$APTEXT	11:20		
72	UN$DISK	$APTEXT	10:42		
73	UN$FILE	$APTEXT	11: 2		
70	UN$PRINT	$APTEXT	10:38		
71	UN$TAPE	$APTEXT	10:40		
0*	UNID		3:12 D	5:14	5:21
4	UT$NUM	$APTEXT	5:41		
11	UT@OUT@P	$APTEXT	6:27		
14	UT@POP@P	$APTEXT	6:30		
0*	UTTY		3:13 D	6: 1	

SYMBOL QUALIFIER IS ADEM

0	$CONST	9: 4 D	9: 4 F	9: 7 D	9: 7 F	9: 7	15:42 D	15:42 F	
442	$REGORG	2:13 D	2:13 F	2:14	2:14 D	2:15	2:15 D	2:16	2:16 D
		2:17	2:17 D	2:18	2:18 D	2:18 F	2:19	2:19 D	2:20
		2:20 D	2:21	2:21 D	2:22	2:22 D	2:23	2:23 D	2:24
		2:24 D	2:24 F	2:25	2:25 D	2:26	2:26 D	2:27	2:27 D
		2:28	2:28 D	2:29	2:29 D	2:29 F	2:30	2:30 D	2:31
		2:31 D	2:32	2:32 D	2:33	2:33 D	2:34	2:34 D	2:35
		2:35 D	2:36	2:36 D	2:37	2:37 D	2:38	2:38 D	2:39
		2:39 D	2:39 F	4:38	6: 8	6:17	6:31	8:22	9:10
		11:12	11:30	12: 6	15: 1	15:24			
0	%NP	1: 4	2:29 D						
430	%P	1: 4	2:29 D	2:29					
410	%S1	2:13	2:14 D	4:17	5:20	5:21	5:22	5:26	5:31
		5:39	5:40	6: 1	6: 2	7: 8	7:16	7:24	8:31
		10: 8	10:24	10:25	10:30	10:36	11:21	11:26	11:27
		12: 1	12: 2	12: 3	12: 4	12: 5	12: 6	13:11	13:39
		14: 5	14: 6	14:10	14:11	14:15	14:38	15:20	15:33
		15:36	15:41	17: 8					
411	%S2	2:13	2:15 D						
412	%S3	2:13	2:16 D						
413	%S4	2:13	2:17 D						
414	%S5	2:13	2:18 D						

415	%T1	2:19 D							
416	%T2	2:20 D							
417	%T3	2:21 D							
420	%T4	2:22 D							
421	%T5	2:23 D							
422	%T6	2:24 D							
423	%W1	2:25 D	4:17	4:20	4:23	4:26	4:29	5:14	5:18
		5:19	5:20	5:21	5:40	6: 1	6:30	6:31	7:19
		7:22	7:25	8:34	8:36	8:39	9: 6	9: 9	10: 8
		10:11	10:13	10:14	10:21	10:22	11: 5	11:14	11:22
		11:23	11:25	11:29	11:30	11:36	12: 1	12: 2	12: 5
		13:11	13:14	13:16	13:18	13:20	13:22	13:24	13:26
		13:28	13:31	14:19	14:42	15: 1	15:11	15:28	15:36
		15:37	17: 8	17:11					
424	%W2	2:26 D	7:20	7:22	7:24	7:29	7:34	8: 2	8:13
		8:21	8:36	8:38	9: 1	11:20	11:21	11:23	11:26
		12: 3	12: 4						
425	%W3	2:27 D							
426	%W4	2:28 D							
427	%W5	2:29 D							
6	ADEM	4: 4 D	4:39	6:10	6:19	6:32	11:13	11:31	12: 7
		15: 5	16: 1						
312	AKERN	4:30	7: 7 D						
433	CHN	2:33 D	10:24	10:29	10:36	10:38	10:40	10:42	11: 2
		11:36							
430	DAL	2:30 D	4:15	4:17	4:38	5:20	5:26	5:39	5:40
		6: 8	6:17	6:26	7: 8	7:16	10: 8	10:24	11:12
		11:25	11:26	11:28	11:36	12: 1	12: 2	12: 3	12: 4
		12: 5	12: 6	13:11	13:39	14: 5	14:10	14:34	15:33
		15:37	15:41	15:42					
432	FC	2:32 D	5:39	6: 7	6:16	7: 8	7:11	7:40	8: 9
		9: 3	13:15	13:17	13:19	13:21	13:23	13:25	14:25
1452	FIND	14: 2	14: 7	14:12	17: 6 D				
757	FOUND	10:29	11:19 D						
1366	FUNCTION	13:27	13:29	15:18 D					
441	MATCH	2:39 D	13:39	13:40	14: 5	14:10	17:11		
1363	NOPATH	14: 3	14: 8	14:13	15:10 D				
1117	NSC	4:27	10:12	13: 7 D					
66	NULDAT	4:37 D	5:32	5:41	6: 9	6:12	6:18	7:12	9:20
435	ORD	2:35 D	7:16	7:22	7:29	8: 5	8:22	8:36	9: 1
		9:10							
437	OVT	2:37 D	7:33	7:41	8: 1	8:10	8:12	8:20	8:22

440	POINT	2:38 D	5:30		5:34	13:42	14: 1	14: 6	14:11
		14:15	14:26	14:28	14:35	14:38	14:42	15:19	15:20
		15:24	15:33	15:36	15:41	17: 7	17: 8	17:17	
1322	QUEUE	5:35	14:32 D						
601	SHELL	4:24	10: 7 D						
434	TBL	2:34 D	7:17	7:23	7:24	7:35	8:21	8:31	8:32
		8:37	8:38	9: 4	9: 6	9: 7	10:25	10:27	10:28
		10:30	11:20	11:29					
431	TTY	2:31 D	6: 2	6:27	6:30				
77	TTY10	4:21	5: 7 D						
436	TYPE	2:36 D	7:35	7:38	8: 7				
1023	UCERR	10:15	11: 6	11:15	11:35 D	13:32	14:20	15:12	15:29

```
0     <macro>                    OVERLAY   AMSG                                                    AMSG.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                                 *****************  ***************************************   AMSG.5
                                 *                                                         *   AMSG.6
                                 *    AMSG:                                                *   AMSG.7
                                 *      IOP Inter-processor message Demon process          *   AMSG.8
                                 *      Controls messages from other IOP's needing immediate *   AMSG.9
                                 *      care.  ACOM handles the big, MOS resident messages.  *   AMSG.10
                                 *      ACOM and AMSG have exclusive use of Registers MA - MH *   AMSG.11
                                 *                                                         *   AMSG.12
                                 ***********************************************************   AMSG.13
                                                                                               AMSG.14
6     <macro>                    REGDEFS   ,,(DAL,MB,MD,ME,MG,MH,MI,MJ,MK,ML,MM,MN,MO,MP,MDAL,   AMSG.15
                                           DCB,NUM,LN0,LN1)                                    AMSG.16
                                                                                                   $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
                                 EQUALS    $REGORG                                                 $REG    3
```

```
446     MDAL    EQUALS    $REGORG                                    $REG    3
447     DCB     EQUALS    $REGORG                                    $REG    3
450     NUM     EQUALS    $REGORG                                    $REG    3
451     LN0     EQUALS    $REGORG                                    $REG    3
452     LN1     EQUALS    $REGORG                                    $REG    3
                                                                 AMSG.17
                                                                 AMSG.18
430     MF      EQUALS    DAL                                    AMSG.19

                EXT       %ACTIVE                                AMSG.21
                EXT       %LSPI                                  AMSG.22
                EXT     %LSPO                                    AMSG.23
                EXT       %MYID                                  AMSG.24
                EXT       APNTR                                  AMSG.25
                EXT       CA                                     AMSG.26
                EXT       CB                                     AMSG.27
                EXT       CC                                     AMSG.28
                EXT       CD                                     AMSG.29
                EXT       CE                                     AMSG.30
                EXT       DS                                     AMSG.31
                EXT       DY                                     AMSG.32
                EXT       DZ                                     AMSG.33
                EXT       ECPQ                                   AMSG.34
                EXT       ED                                     AMSG.35
                EXT       EH                                     AMSG.36
                EXT       EI                                     AMSG.37
                EXT       HB                                     AMSG.38
                EXT       HC                                     AMSG.39
                EXT       MIMC                                   AMSG.40
                EXT       MIMDALQ                                AMSG.41
                EXT       MIMMQ                                  AMSG.42
                EXT       MPOPCELL                               AMSG.43
                EXT     DCCB                                     AMSG.44
```

```
                                        ******************      ************************************      AMSG.46
  6                             MIMM    *                         .Immediate message to process         AMSG.47
  6   014000 /000000  024431            MB = MIMMQ                .Immediate queue                      AMSG.48
 11   002000  000000                    I = 0                     .                                     AMSG.49
 13   020431  024000                    ED = MB                   .Queue                                AMSG.50
 15    <macro>                          RKERN    ETOQ             .Get top entry                        AMSG.51
 17   020000  024432                    MD = EH                   .Data                                 AMSG.52
 21   102015                            P = MCHKDAL, A = 0        .Any DAL's to process?                AMSG.53
 22   014000 /000000  024410            (MIMC) = (MIMC) + 1       .Count immediate messages             AMSG.54
      036410
 26   020432  004014  024434            MG = MD > 14              .                                     AMSG.55
                                                                                                         AMSG.56
 31   020434  013003  102100            P = CPUDLAY, R!MG = MQ$DLAY   .REQUEST TO END DELAY REQUEST     AMSG.57
 34    <macro>                          $PUNTIF                       .Undefined function code          AMSG.58
                                                                                                         AMSG.59
 36                             MCHKDAL *                                                               AMSG.60
 36   014000 /000000  024000            DZ = MIMDALQ              .Queue DAL's                          AMSG.61
 41   030000  102067                    P = MIDLE, (DZ) = 0       .Any queued?                          AMSG.62
 43    <macro>                          RKERN    HGET             .Get first one                        AMSG.63
 45   020000  024430                    R!DAL = R!DY                                                    AMSG.64
 47    <macro>                          GET       R!ME,DA@IFC,R!DAL   .Get the function code           AMSG.65
                                                                                                         AMSG.66
 54   020433  013042  126000            P = BIGDISK, R!ME = AM$BDRSP  .RESPONSE TO BIGDISK REQUEST    AMSG.67
     /001376
 60   020433  013040  102426            P = BMRIO, R!ME = AM$COSIO    .BMR transfer request           AMSG.68
 63   020433  013041  126000            P = BMRDN, R!ME = AM$IODON    .BMR transfer done              AMSG.69
     /001177
 67   020433  013043  102477            P = XFRIO, R!ME = AM$XFRIO    .Transfer request               AMSG.70
 72   020433  013044  126000            P = XFRDN, R!ME = AM$XFRDN    .Transfer done                  AMSG.71
     /001362
 76   020433  013045  102470            P = HSPIO, R!ME = AM$HSPIO    .High speed transfer request    AMSG.72
101   020433  013046  126000            P = HSPDN, R!ME = AM$HSPDN    .High speed transfer done       AMSG.73
     /001277
105   020433  013020  102041            P = MCREA, ME = AM$ALERT      .CREATE ACTIVITY                AMSG.74
110   020433  013021  102162            P = MCREDN, ME = AM$CRTDN     .ACTIVITY CREATED RESPONSE      AMSG.75
113   020433  013024  102247            P = MAWAKE, ME = AM$AWAKE     .AWAKE AN ACTIVITY              AMSG.76
116   020433  013025  102302            P = MRESPD, ME = AM$RSPND     .RESPOND TO CREATOR             AMSG.77
121   020433  013033  102311            P = OUTCALL, R!ME = AM$OUTC   .Create OUTCALL task            AMSG.78
124   020433  013034  102331            P = OUTRESP, R!ME = AM$OUTDN  .Response to OUTCALL            AMSG.79
127    <macro>                          $PUNTIF                   .                                     AMSG.80
131                             MIDLE   *                                                               AMSG.81
131    <macro>                          PKERN    ESWP             .                                     AMSG.82
```

```
                                                                                AMSG.83
                                                                                AMSG.84
                    ********************************************************    AMSG.85
                    *                                                      *    AMSG.86
                    *       MQ$DLAY    SEND AN MCU INTERRUPT TO THE MAINFRAME .  *    AMSG.87
                    *                  THIS WILL CAUSE EXEC TO INTERRUPT ALL   *    AMSG.88
                    *                  CPUS AND IDLE FOR 50 MICROSECONDS.      *    AMSG.89
                    *                                                      *    AMSG.90
                    ********************************************************    AMSG.91
                                                                                AMSG.92
133                 CPUDLAY  *                                                  AMSG.93
133   <macro>               GET B,CPO@CN,R!%LSPO          .CHANNEL NUMBER       AMSG.94
136   010000                A = 0                         .CLEAR                AMSG.95
137   164000                IOB:4                         .                     AMSG.96
                      4     DUP 4,1                                             AMSG.97
      <definition>          PASS                          .DELAY                AMSG.98
144   014000 /010000        A = O'10000                   .MCU INTERRUPT        AMSG.99
146   164000                IOB:4                         .ISSUE                AMSG.100
147   071141                P = MIMM                      .DONE                 AMSG.101
```

```
                                        ******************  ****************************************  AMSG.103
                                        *  Function 20;  Create activity for joint-IOP functions      AMSG.104
                                        *                activity address and POPCELL returned to     AMSG.105
                                        *                originator.                                  AMSG.106
150                             MCREA   *                                                            AMSG.107
150   010006  024000                    R!HB = PC@LEN@P              .Size of popcell                AMSG.108
152    <macro>                          RKERN     HMA                .Allocate local memory          AMSG.109
154   020000  024434                    R!MG = R!HC                                                  AMSG.110
156   102072                            P = MCFIN, A = 0             .If no memory available         AMSG.111
                                                                                                     AMSG.112
157    <macro>                          CLEAR     START=R!MG,COUNT=PC@LEN@P .Clear the popcell       AMSG.113
172   020430  012006  024410            R!%W1 = (MF + DA@HPO@P)      .Overlay number of new task     R15472BC.1
      030410  024423
177    <macro>                          CREATE    R!%W1,(R!MG),TYPE=NUMBER .Create activity as directed AMSG.115
216    <macro>                          $PUNTIF (A # 0)              .IF ERROR                       AMSG.116
220    <macro>                          GET       R!%W1,AD@P1,R!%ACTIVE .New activity's descriptor   AMSG.117
225    <macro>                          PUT       R!%W1,PC@ACT,R!MG  .Enter in popcell               AMSG.118
232   002000  000000                    I = 0                        .                               AMSG.119
234   014000 /000000  024435            MH = MPOPCELL                .Put this POPCELL at the head of AMSG.120
237   030435  034434                    (MG) = (MH)                  .the POPCELL list               AMSG.121
241   020434  034435                    (MH) = MG                                                    AMSG.122
243   020430  012001  024410            (MF + DA@IFC@P) = AM$CRTDN   .Tell creator that create done  R15472BC.2
      010021  034410
250                             MCFIN   *                                                            AMSG.124
250   020430  012006  024410            (MF + DA@HPO@P) = MG         .Return POPCELL addr to requesterR15472BC.3
      020434  034410
255   020430  024000                    DS = MF                      .                               AMSG.126
257    <macro>                          RKERN     DSUM               .Write to MOS                   AMSG.127
261   020000  004014  013010            EI = EH > 14 - 10            .IOP I.D.                       AMSG.128
      024000
265    <macro>                          RKERN     EMSGIOP            .Send to that IOP               AMSG.129
267   020430  024000                    R!DY = R!MF                  .Local DAL address              AMSG.130
271    <macro>                          RKERN     HONN               .Put it back                    AMSG.131
273   071265                            P = MIMM                     .                               AMSG.132


                                        ************************************************************  AMSG.134
                                        *  Function 21: Reactivate requestor of function 20; supply   AMSG.135
                                        *               entity with POPCELL address in other IOP.     AMSG.136
274                             MCREDN  *                                                            AMSG.137
```

```
274   020430  012004  024410          MH = (MF + DA@ACT@P)          .Activity to dequeue          R15472BC.4
      030410  024435
301   010015  024440                  MK = F$ALERT                .Last call - Alert.                 AMSG.139
303                           MCOMMN  *                                                               AMSG.140
303   020435  012012  024410          MJ = (MH + 12)              .Last service request             AMSG.141
      030410  024437
310    <macro>                        $PUNTIF   (MJ # MK)           . If not F$ALERT or F$AWAKE.    AMSG.142
314   020430  012006  024410          MI = (MF + DA@HPO@P)          .POPCELL address              R15472BC.5
      030410  024436
321    <macro>                        PUT    MI,AD@RC,MH          .Return address to requestor     AMSG.144
326   020435  024000                  EH = MH                     .Activity to queue                AMSG.145
330    <macro>                        RKERN   EQCP                .To CP queue                      AMSG.146
332                           MFREED  *                                                               AMSG.147
332   020430  012005  024410          (MF + 5) = 0                .Clear code cell                  AMSG.148
      010000  034410
337   020430  012002  024441          ML = MF + 2                 .                                 AMSG.149
342   030441  024000                  CB = (ML)                   .MOS upper                        AMSG.150
344   010000  034441                  (ML) = 0                    .                                 AMSG.151
346   026441                          ML = ML + 1                 .                                 AMSG.152
347   030441  024000                  CC = (ML)                   .Lower                            AMSG.153
351   010000  034441                  (ML) = 0                    .                                 AMSG.154
353   010001  024000                  R!CA = 1                      .RELEASE 1 DAL                  AMSG.155
355    <macro>                        RKERN     PUTDAL              .Release the DAL                AMSG.156
357   020430  024000                  DY = MF                     .Local address of DAL             AMSG.157
361    <macro>                        RKERN    HONN               .Return to pool                   AMSG.158
363   071355                          P = MIMM                    .                                 AMSG.159
                                                                                                    AMSG.160
                                      ************************************************************  AMSG.161
                                      *  Function 24: Awake an activity in this IOP.  Queue the    AMSG.162
                                      *               DAL which was supplied by the caller in      AMSG.163
                                      *               another IOP.                                 AMSG.164
364                           MAWAKE  *                                                               AMSG.165
364   020430  012006  024410          MI = (MF + DA@HPO@P)          .POPCELL address              R15472BC.6
      030410  024436
371    <macro>                        ADDRESS   R!MJ,PC@PSH,R!MI    .Activity push queue address     AMSG.167
                                                                                                    AMSG.168
374    <macro>                        $IF       ((R!MJ) # 0)        .If an activity queued           AMSG.169
                                                                                  .>>>>> 0          $IF      1
377    <macro>                          POP     R!MJ,R!MF           .Pop the task                    AMSG.170
411    <macro>                        $ELSE                                                          AMSG.171
412   020430  024000                    R!EH = R!MF                                                  AMSG.172
414    <macro>                          ADDRESS R!ED,PC@DAL,R!MI                                     AMSG.173
```

```
417    <macro>                          RKERN   ENQU          .Queue the DAL on the popcell        AMSG.174
421    <macro>                        $ENDIF                                                       AMSG.175
                                                                        .<<<<< 0                      $ENDIF   1
                                                                                                    AMSG.176
421    071413                         P = MIMM                                                     AMSG.177


                                      ************************************************************  AMSG.179
                                      *  Function 25:  Respond from another processor recognized   AMSG.180
                                      *                Activate the waiting activity, pass parameter AMSG.181
                                      *                to the waiting process through the DAL      AMSG.182
422                          MRESPD   *                                                            AMSG.183
422    010016  024440                 MK = F$AWAKE              .Awakener                          AMSG.184
424    020430  012004  024410         MH = (MF + DA@ACT@P)          .Activity id                R15472BC.7
       030410  024435
431    020435  107127                 P = MCOMMN, MH # 0        .One exists, activate it           AMSG.186
433    071101                         P = MFREED                .None, release MOS, local too      AMSG.187
                                                                                                    AMSG.188
```

```
                                 ********************************************************************   AMSG.190
                                 *                                                                  *   AMSG.191
                                 *        AM$OUTC   Create an activity which begins execution in the  *   AMSG.192
                                 *                  overlay OUTCALL.                                  *   AMSG.193
                                 *                                                                  *   AMSG.194
                                 ********************************************************************   AMSG.195
                                                                                                        AMSG.196
434                              OUTCALL  *                                                             AMSG.197
434     <macro>                          CREATE    OUTCALL,(R!DAL)                                    AMSG.198
454     <macro>                          $PUNTIF (A # 0)              .IF ERROR                       AMSG.199
456    071450                            P = MIMM                                                     AMSG.200


                                 ********************************************************************   AMSG.202
                                 *                                                                  *   AMSG.203
                                 *        AM$OUTDN  Reactivate the task which issued an OUTCALL message *   AMSG.204
                                 *                  to another IOP.                                   *   AMSG.205
                                 *                                                                  *   AMSG.206
                                 ********************************************************************   AMSG.207
                                                                                                        AMSG.208
457                              OUTRESP  *                                                             AMSG.209
457     <macro>                          GET       R!EH,DA@ACT,R!DAL   .Get issuing activity address    AMSG.210
464     <macro>                          GET       A,AD@FU,R!EH                                         AMSG.211
470     <macro>                          $PUNTIF   (A # F$OUTCAL)      .If no OUTCALL issued           AMSG.212
                                                                                                        AMSG.213
473     <macro>                          GET       R!%W1,DA@ARC,R!DAL                                   AMSG.214
500     <macro>                          PUT       R!%W1,AD@RC,R!EH    .Return the response code        AMSG.215
505     <macro>                          RKERN     EQCP                .Reschedule the activity         AMSG.216
507    071155                            P = MFREED                    .Dispose of the DAL              AMSG.217
```

```
                                 ***************************.   ****************************************   AMSG.219
                                 *                                                                    *   AMSG.220
                                 *        AM$COSIO  Transfer data between a target and BMR memorys    *   AMSG.221
                                 *                                                                    *   AMSG.222
                                 *        AM$XFRIO  Transfer data between a target and MOS memorys    *   AMSG.223
                                 *                                                                    *   AMSG.224
                                 *        AM$HSPIO  Transfer data between a target and MOS memorys    *   AMSG.225
                                 *                                                                    *   AMSG.226
                                 ***********************************************************************   AMSG.227
                                                                                                           AMSG.228
510                              BMRIO    *                                                               AMSG.229
                                                                                                           AMSG.230
                                          * Convert function code to transfer direction                   AMSG.231
                                                                                                           AMSG.232
510     <macro>                           GET R!MG,DA@FC,R!DAL          .Function code                   AMSG.233
516     <macro>                           $IF (R!MG = DAF$READ)         .If to target memory             AMSG.234
                                                                                        .>>>>> 0             $IF      1
522     <macro>                             STORE FS$OUT,DA@FC,R!DAL    .Set transfer direction          AMSG.235
532     <macro>                           $ELSE                         .From target memory              AMSG.236
533     <macro>                             STORE FS$IN,DA@FC,R!DAL     .Set transfer direction          AMSG.237
544     <macro>                           $ENDIF                                                         AMSG.238
                                                                                        .<<<<< 0             $ENDIF   1
                                                                                                           AMSG.239
                                          * Save transfer length for response                            AMSG.240
                                                                                                           AMSG.241
544     <macro>                           GET R!MM,DA@LN0,R!DAL         .Length                          AMSG.242
551     <macro>                           GET R!MN,DA@LN1,R!DAL         .                                AMSG.243
556     <macro>                           PUT R!MM,DA@SEQ,R!DAL         .Save                            AMSG.244
563     <macro>                           PUT R!MN,DA@DAT,R!DAL         .                                AMSG.245
                                                                                                           AMSG.246
570                              XFRIO    *                                                               AMSG.247
570                              HSPIO    *                                                               AMSG.248
                                                                                                           AMSG.249
570                              LOOP     *                                                               AMSG.250
                                                                                                           AMSG.251
570     003000                            I = 1                         .Unlock                          AMSG.252
                                                                                                           AMSG.253
                                          * Initialize registers from request                             AMSG.254
                                                                                                           AMSG.255
571     <macro>                           GET R!MG,DA@FC,R!DAL          .Transfer direction              AMSG.256
577     <macro>                           GET R!MH,DA@TM,R!DAL          .Target memory                   AMSG.257
605     <macro>                           GET R!MI,DA@TM0,R!DAL         .Target memory address (upper)   AMSG.258
```

```
 614    <macro>                    GET R!MJ,DA@TM1,R!DAL          .Target memory address (lower)      AMSG.259
 621    <macro>                    GET R!MK,DA@BM0,R!DAL          .MOS address (upper)                AMSG.260
 626    <macro>                    GET R!ML,DA@BM1,R!DAL          .MOS address (lower)                AMSG.261
 633    <macro>                    GET R!MM,DA@LN0,R!DAL          .Length of transfer (upper)         AMSG.262
 640    <macro>                    GET R!MN,DA@LN1,R!DAL          .Length of transfer (lower)         AMSG.263
                                                                                                       AMSG.264
 645   010010  005011  024450      R!NUM = AMSG$LIM < WRD$BUF     .Max transfer request for AMSG      AMSG.265
 650    <macro>                    GET R!MP,AD@PRI,R!%ACTIVE      .Priority of AMSG                   AMSG.266
                                                                                                       AMSG.267
                                   * Transfer data to or from target memory in AMSG$LIM sectors        AMSG.268
                                                                                                       AMSG.269
 656    <macro>                    $UNTIL (R!MM = 0) AND (R!MN = 0) .All transferred                   AMSG.270
                                                                                                       AMSG.271
 664    <macro>                      $IF (R!MM # 0) OR (R!MN > R!NUM) .If more than limit              AMSG.272
                                                                                  .>>>>> 0                $IF      1
 674   020450  024444                 R!MO = R!NUM                 .Set to limit                      AMSG.273
 676    <macro>                      $ELSE                        .                                  AMSG.274
 677   020443  024444                 R!MO = R!MN                  .Set to actual                     AMSG.275
 701    <macro>                      $ENDIF                                                          AMSG.276
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                       AMSG.277
 701    <macro>                      TRANSFER R!MG,R!MH,R!MI,R!MJ,R!MK,R!ML,R!MO                     AMSG.278
                                                                                                       AMSG.279
 730    <macro>                      DADD R!MI,R!MJ,R!MO          .Update target address             AMSG.280
 735    <macro>                      DADD R!MK,R!ML,R!MO          .Update MOS address                AMSG.281
 742    <macro>                      DSUB R!MM,R!MN,R!MO          .Update length                     AMSG.282
                                                                                                       AMSG.283
                                     * Check for other activities ready to run                       AMSG.284
                                                                                                       AMSG.285
 753    <macro>                      $IF (R!MM # 0) OR (R!MN # 0) .If more to do                     AMSG.286
                                                                                  .>>>>> 0                $IF      1
 761   014000 /000000  024410         R!%W1 = (ECPQ)               .Waiting activities                AMSG.287
       030410  024423
 766    <macro>                        $IF (R!%W1 # 0)            .If others waiting                 AMSG.288
                                                                                  .>>>>> 1                $IF      1
 772    <macro>                          GET A,AD@PRI,R!%W1       .Priority                          AMSG.289
 777    <macro>                          $IF (A <= R!MP)          .If waiter is higher               AMSG.290
                                                                                  .>>>>> 2                $IF      1
1006    <macro>                            PUT R!MI,DA@TM0,R!DAL .Save target address                AMSG.291
1022    <macro>                            PUT R!MJ,DA@TM1,R!DAL .                                   AMSG.292
1027    <macro>                            PUT R!MK,DA@BM0,R!DAL .Save MOS address                   AMSG.293
1034    <macro>                            PUT R!ML,DA@BM1,R!DAL .                                   AMSG.294
```

```
1041    <macro>                          PUT R!MM,D[illegible],R!DAL .Save length                    AMSG.295
1046    <macro>                          PUT R!MN,DA@LN1,R!DAL .                                     AMSG.296
                                                                                                     AMSG.297
1053    014000 /000000  024410           (APNTR) = R!DAL        .Save DAL address                    AMSG.298
        020430  034410
1060    <macro>                          GIVEUP A1=0,A2=-1      .Let waiter run                      AMSG.299
1066    014000 /000000  024410           R!DAL = (APNTR)        .Restore DAL address                 AMSG.300
        030410  024430
1073    071303                           P = LOOP               .Continue                            AMSG.301
1074    <macro>                        $ENDIF                                                        AMSG.302
                                                                                .<<<<< 2                $ENDIF   1
1074    <macro>                      $ENDIF                                                          AMSG.303
                                                                                .<<<<< 1                $ENDIF   1
1074    <macro>                    $ENDIF                                                            AMSG.304
                                                                                .<<<<< 0                $ENDIF   1
1074    <macro>                  $ENDTIL                                                             AMSG.305
                                                                                                     AMSG.306
                                 * Send response                                                     AMSG.307
                                                                                                     AMSG.308
1075    <macro>                  GET R!ME,DA@IFC,R!DAL          .Request type                        AMSG.309
1102    <macro>                  $IF (R!ME = AM$COSIO)          .If BMR request                      AMSG.310
                                                                                .>>>>> 0                $IF      1
1106    <macro>                    STORE AM$IODON,DA@IFC,R!DAL .                                     AMSG.311
1113    <macro>                  $ELSEIF (R!ME = AM$XFRIO)      .If transfer I/O                     AMSG.312
1120    <macro>                    STORE AM$XFRDN,DA@IFC,R!DAL                                       AMSG.313
1125    <macro>                  $ELSEIF (R!ME = AM$HSPIO)      .If high speed I/O                   AMSG.314
1132    <macro>                    STORE AM$HSPDN,DA@IFC,R!DAL                                       AMSG.315
1137    <macro>                  $ENDIF                                                              AMSG.316
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                     AMSG.317
1137    <macro>                  STORE 0,DA@RC,R!DAL                                                 AMSG.318
                                                                                                     AMSG.319
1146    002000  000000           I = 0                          .Lock                                AMSG.320
1150    020430  024000           R!DS = R!DAL                   .DAL address                         AMSG.321
1152    <macro>                  RKERN DSUM                     .Write DAL to MOS                    AMSG.322
                                                                                                     AMSG.323
1154    002000  000000           I = 0                          .Lock                                AMSG.324
1156    <macro>                  GET R!EI,DA@IOP,R!DAL          .Requesting IOP                      AMSG.325
1165    <macro>                  RKERN EMSGIOP                  .Send message                        AMSG.326
                                                                                                     AMSG.327
1167    002000  000000           I = 0                          .Lock                                AMSG.328
1171    020430  024000           R!DY = R!DAL                   .DAL address                         AMSG.329
```

```
1173    <macro>                 RKERN HONN                  .Release local DAL          AMSG.330
1175   075000 /000006           P = MIMM                                                AMSG.331
                                                                                         AMSG.332
                                                                                         AMSG.333
                        ********************************************************************AMSG.334
                        *                                                                  *AMSG.335
                        *       AM$IODON  Transfer between a target and BMR memorys done.  *AMSG.336
                        *                 Send response to mainframe.                      *AMSG.337
                        *                                                                  *AMSG.338
                        ********************************************************************AMSG.339
                                                                                         AMSG.340
1177                    BMRDN   *                                                        AMSG.341
                                                                                         AMSG.342
                                * Release the MOS DAL                                     AMSG.343
                                                                                         AMSG.344
1177   010001  024000           R!CA = 1                    .Release single DAL         AMSG.345
1201    <macro>                 GET R!CB,DA@DL0,R!DAL       .MOS address (upper)        AMSG.346
1206    <macro>                 GET R!CC,DA@DL1,R!DAL       .MOS address (lower)        AMSG.347
1213    <macro>                 RKERN PUTDAL                .Release                    AMSG.348
                                                                                         AMSG.349
                                * Swap source and destination id for response            AMSG.350
                                                                                         AMSG.351
1215    <macro>                 GET R!MM,DA@DID,R!DAL       .Get destination            AMSG.352
1222    <macro>                 GET R!MN,DA@SID,R!DAL       .Get source                 AMSG.353
1227    <macro>                 PUT R!MM,DA@SID,R!DAL       .Set source                 AMSG.354
1234    <macro>                 PUT R!MN,DA@DID,R!DAL       .Set destination            AMSG.355
                                                                                         AMSG.356
                                * Restore original transfer length in response           AMSG.357
                                                                                         AMSG.358
1241    <macro>                 GET R!MM,DA@SEQ,R!DAL       .Length                     AMSG.359
1246    <macro>                 GET R!MN,DA@DAT,R!DAL       .                           AMSG.360
1253    <macro>                 PUT R!MM,DA@LN0,R!DAL       .Save                       AMSG.361
1260    <macro>                 PUT R!MN,DA@LN1,R!DAL       .                           AMSG.362
                                                                                         AMSG.363
                                * Send response to mainframe                             AMSG.364
                                                                                         AMSG.365
1265   020430  024000           R!DY = R!DAL                .DAL address                AMSG.366
1267    <macro>                 RKERN IDALSND               .Issue                      AMSG.367
1271    <macro>                 FLDSUB 1,CPI@DP,R!%LSPI     .Dec disk population counter AMSG.368
1275   075000 /000006           P = MIMM                                                AMSG.369
                                                                                         AMSG.370
                        ********************************************************************AMSG.371
```

```
                                 *                                                                  *    AMSG.372
                                 *        AM$XFRDN  Transfer between a target and MOS memorys done.  *    AMSG.373
                                 *                  Reactivate waiting activity.                     *    AMSG.374
                                 *                                                                  *    AMSG.375
                                 *        AM$HSPDN  Transfer between a target and MOS memorys done.  *    AMSG.376
                                 *                  Read data to local memory if from target memory. *    AMSG.377
                                 *                  Reactivate waiting activity.                     *    AMSG.378
                                 *                                                                  *    AMSG.379
                                 ********************************************************************    AMSG.380
                                                                                                         AMSG.381
1277                             HSPDN    *                                                             AMSG.382
                                                                                                         AMSG.383
                                          * Move data from MOS to local memory                          AMSG.384
                                                                                                         AMSG.385
1277    <macro>                           GET R!MG,DA@FC,R!DAL          .Transfer direction             AMSG.386
1305    <macro>                           $IF (R!MG = FS$IN)            .If read                        AMSG.387
                                                                                        .>>>>> 0            $IF      1
1310   002000  000000                       I = 0                       .Lock                           AMSG.388
1312   010001  024000                       R!CA = MIO$BMR              .Read code                      AMSG.389
1314    <macro>                             GET R!CB,DA@BM0,R!DAL       .MOS address (lower)            AMSG.390
1321    <macro>                             GET R!CC,DA@BM1,R!DAL       .MOS address (upper)            AMSG.391
1326    <macro>                             GET R!CD,DA@LOC,R!DAL       .Local address                  AMSG.392
1333    <macro>                             GET R!CE,DA@LN1,R!DAL       .Length of transfer             AMSG.393
1340    <macro>                             RKERN COMIO                 .Issue                          AMSG.394
1342    <macro>                           $ENDIF                                                        AMSG.395
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                         AMSG.396
                                          * Release MOS buffer                                          AMSG.397
                                                                                                         AMSG.398
1342   002000  000000                     I = 0                         .Lock                           AMSG.399
1344   010001  024000                     R!CA = 1                      .Release single buffer          AMSG.400
1346    <macro>                           GET R!CB,DA@BM0,R!DAL         .MOS address (upper)            AMSG.401
1353    <macro>                           GET R!CC,DA@BM1,R!DAL         .MOS address (lower)            AMSG.402
1360    <macro>                           RKERN PUTDISK                                                 AMSG.403
                                                                                                         AMSG.404
                                          * Reactivate waiting activity                                 AMSG.405
                                                                                                         AMSG.406
1362                             XFRDN    *                                                             AMSG.407
                                                                                                         AMSG.408
1362    <macro>                           GET R!EH,DA@ACT,R!DAL         .Activity address               AMSG.409
1367    <macro>                           $IF (R!EH # 0)                .If waiting                     AMSG.410
                                                                                        .>>>>> 0            $IF      1
```

```
1372    <macro>                    RKERN EQCP          .Queue activity                    AMSG.411
1374    <macro>                  $ENDIF                                                   AMSG.412
                                                                      .<<<<< 0                $ENDIF    1
1374   075000 /000332            P = MFREED            .Release DAL                       AMSG.413
```

```
                                 ***************************************************************************   AMSG.415
                                 *                                                                         *   AMSG.416
                                 *              RESPONSE PROCESSING FOR BIG-DISK REQUESTS                  *   AMSG.417
                                 *                        (GENERATED IN CDEM)                              *   AMSG.418
                                 *                                                                         *   AMSG.419
                                 ***************************************************************************   AMSG.420
                                                                                                             AMSG.421
                                          *                                                                  AMSG.422
                                          * FIND DCB FOR DEVICE                                             AMSG.423
                                          *                                                                  AMSG.424
                                                                                                             AMSG.425
1376                             BIGDISK  *                                                                  AMSG.426
1376   003000                            I = 1                          .ENABLE INTERRUPTS                  AMSG.427
1377    <macro>                          GET R!%W1,DA@LCH,R!DAL         .LOGICAL CHANNEL NUMBER             AMSG.428
1405   020423  013020  016000            R!%W1 = R!%W1 - 20 + DCCB      .FIND TABLE                         AMSG.429
      /000000  024423
1412   030423  024447                    R!DCB = (R!%W1)                .                                   AMSG.430
                                                                                                             AMSG.431
                                          *  Determine which IOP owns the MOS DAL.                         AMSG.432
                                                                                                             AMSG.433
1414    <macro>                          GET R!%W1,DA@TYP,R!DAL         .Device type                        AMSG.434
1422    <macro>                          $IF (R!%W1 = DSK$DD19),OR,(R!%W1 = DSK$DD29)                       AMSG.435
                                                                                          .>>>>> 0              $IF      1
1431    <macro>                            GET R!NUM,DA@NUM,R!DAL       .Originating IOP                    AMSG.436
1440    <macro>                          $ELSE                                                              AMSG.437
1441    <macro>                            GET A,DA@MES,R!DAL           .Accumulator msg                    AMSG.438
1445   004012  011003  024450              R!NUM = A > D'10 & 3         .Originating IOP                    AMSG.439
1450    <macro>                          $ENDIF                                                             AMSG.440
                                                                                          .<<<<< 0              $ENDIF   1
                                                                                                             AMSG.441
                                          * FIND MATCHING REQUEST DAL                                        AMSG.442
                                                                                                             AMSG.443
1450    <macro>                          ADDRESS R!MDAL,BD@IQU,R!DCB    .DAL QUEUE                          AMSG.444
1453    <macro>                          GET R!%W2,DA@SQN,R!DAL         .SEQUENCE NUMBER                    AMSG.445
1461   012001  024425                    R!%W3 = A + 1                  .SET UNEQUAL                        AMSG.446
1463    <macro>                          $UNTIL (R!%W2 = R!%W3)         .UNTIL FOUND                        AMSG.447
1466   030446  024446                      R!MDAL = (R!MDAL)            .NEXT DAL                           AMSG.448
1470    <macro>                            $PUNTIF (A = 0)              .IF NO MATCH                        AMSG.449
1472    <macro>                            GET R!%W3,DA@SQN,R!MDAL      .SEQUENCE NUMBER                    AMSG.450
1500    <macro>                          $ENDTIL                                                            AMSG.451
                                                                                                             AMSG.452
                                          *  Process response.                                              AMSG.453
```

```
                        *  If this IOP owns the MOS DAL and the function is DAF$WBFR,      AMSG.454
                        *  then it is an early write behind reply.  Otherwise,             AMSG.455
                        *  process as a final reply.                                       AMSG.456
                                                                                           AMSG.457
1501  072127            R = ERRCHK                    .Check for errors                    AMSG.458
1502   <macro>          GET A,DA@FC,R!DAL             .Function code                       AMSG.459
1507   <macro>          $IF (A = DAF$WBFR)            .Write behind with final reply       AMSG.460
                                                                        .>>>>> 0              $IF      1
1512   <macro>            $IF (R!NUM # R!%MYID)       .If other IOP owns MOS DAL           AMSG.461
                                                                        .>>>>> 1              $IF      1
1516  077000 /002300        R = RESPOND               .Process final responses             AMSG.462
1520   <macro>            $ELSE                                                            AMSG.463
1521   <macro>              GET R!NUM,DA@TRQ,R!DAL    .Total requests sent                 AMSG.464
1530   <macro>              FLDADD 1,DA@ERS,R!MDAL    .Update early responses              AMSG.465
1555   <macro>              $IF (A = R!NUM)           .If all early responses in           AMSG.466
                                                                        .>>>>> 2              $IF      1
1560   <macro>                COPY R!MDAL,R!DAL,DA@@LH+DA@@LE .Copy MDAL                   AMSG.467
1577   <macro>                SEND R!DAL              .Send to mainframe                   AMSG.468
1613   <macro>                STORE 0,DA@ERC,R!MDAL   .Clear errcount for next reply       AMSG.469
1620   <macro>              $ELSE                                                          AMSG.470
1621  077000 /002267          R = RELDAL              .Release local DAL                   AMSG.471
1623   <macro>              $ENDIF                                                         AMSG.472
                                                                        .<<<<< 2              $ENDIF   1
1623   <macro>            $ENDIF                                                           AMSG.473
                                                                        .<<<<< 1              $ENDIF   1
1623   <macro>          $ELSE                                                              AMSG.474
1624  077000 /002300        R = RESPOND               .Process final responses             AMSG.475
1626   <macro>          $ENDIF                                                             AMSG.476
                                                                        .<<<<< 0              $ENDIF   1
1626  075000 /000006     P = MIMM                      .CHECK FOR MORE                      AMSG.477
                                                                                           AMSG.478
                     ************************** SUBROUTINES ******************************   AMSG.479
                                                                                           AMSG.480
                        *                                                                  AMSG.481
                        * CHECK FOR DEVICE ERRORS                                          AMSG.482
                        * KEEP TRACK OF UNIT THAT ERROR OCCURRED ON                        AMSG.483
                        *                                                                  AMSG.484
                                                                                           AMSG.485
1630                 ERRCHK *                                                              AMSG.486
                                                                                           AMSG.487
                        * CHECK FOR ERROR                                                  AMSG.488
                                                                                           AMSG.489
```

```
1630    <macro>                     GET R!%W5,DA@RC,R!                .Response code                          AMSG.490
1636   103002  001000               EXIT, A = DAR$OK                  .If no error                            AMSG.491
1640    <macro>                     $IF (R!%W5 # DAR$REC),AND,(R!%W5 # DAR$COR) .Not recoverable             AMSG.492
                                                                                        .>>>>> 0                $IF      1
1650    <macro>                       GET R!%W1,DA@FC,R!DAL           .Function code                        AMSG.493
1656    <macro>                       $IF (R!%W1 = DAF$WBFR),AND,(R!NUM = R!%MYID) .If early reply          AMSG.494
                                                                                        .>>>>> 1                $IF      1
1666    <macro>                         FLDSUB 1,DA@TRQ,R!MDAL        .Don't expect late reply              AMSG.495
1713    <macro>                         STORE 1,DA@NFR,R!MDAL         .Don't send final response            AMSG.496
1724    <macro>                       $ENDIF                                                                AMSG.497
                                                                                        .<<<<< 1                $ENDIF   1
1724    <macro>                     $ENDIF                                                                  AMSG.498
                                                                                        .<<<<< 0                $ENDIF   1
1724    <macro>                     GET R!%W1,DA@ERC,R!DAL            .Recovered error count                AMSG.499
1731    <macro>                     FLDADD R!%W1,DA@ERC,R!MDAL        .Sum of all units                     AMSG.500
                                                                                                              AMSG.501
                                    * FIND PHYSICAL UNITS' RELATIVE POSITION                                  AMSG.502
                                    * IN GROUP.                                                               AMSG.503
                                                                                                              AMSG.504
1737    <macro>                     GET R!%W1,BD@PUT,R!DCB            .PHYSICAL UNIT TABLE                  AMSG.505
1744    <macro>                     GET R!%W2,DA@CHN,R!DAL            .CHANNEL NUMBER FROM DAL              AMSG.506
1752   010000  024425               R!%W3 = 0                         .CHANNEL TO MATCH                     AMSG.507
1754   010000  024426               R!%W4 = 0                         .RELATIVE POSITION                    AMSG.508
1756    <macro>                     $UNTIL (R!%W3 = R!%W2)            .UNTIL POSITION FOUND                 AMSG.509
1761   020423  022426  024425         R!%W3 = R!%W1 + R!%W4           .PHYSICAL UNIT ENTRY                  AMSG.510
1764   030425  024425                 R!%W3 = (R!%W3)                 .                                     AMSG.511
1766    <macro>                       RGET R!%W3,PUT@CH,R!%W3         .Channel                              AMSG.512
1771    <macro>                       $IF (R!%W3 # R!%W2)             .IF NO MATCH                          AMSG.513
                                                                                        .>>>>> 0                $IF      1
1775   026426                           R!%W4 = R!%W4 + 1             .INCREMENT OFFSET                     AMSG.514
1776    <macro>                       $ENDIF                                                                AMSG.515
                                                                                        .<<<<< 0                $ENDIF   1
1776    <macro>                     $ENDTIL                                                                 AMSG.516
                                                                                                              AMSG.517
                                    *  If new error channel is lower in order or first error,                 AMSG.518
                                    *  enter error data in MDAL.                                              AMSG.519
                                    *  Unrecovered status has priority over recovered.                        AMSG.520
                                                                                                              AMSG.521
1777    <macro>                     GET R!%W1,DA@LUE,R!MDAL           .Last error                           AMSG.522
2006    <macro>                     GET R!%W2,DA@RC,R!MDAL            .Current response                     AMSG.523
2014    <macro>                     $IF (R!%W2 # DAR$OK)              .If previous error                    AMSG.524
                                                                                        .>>>>> 0                $IF      1
```

```
2017    <macro>                 $IF (R!%W2 = DAR$REC),OR,(R!%W2 = DAR$COR) .Recovered          AMSG.525
                                                                        .>>>>> 1        $IF      1
2026    <macro>                   $IF (R!%W1 <= R!%W4)      .If new error higher in order   AMSG.526
                                                                        .>>>>> 2        $IF      1
2034    <macro>                     $IF (R!%W5 = DAR$REC),OR,(R!%W5 = DAR$COR)          AMSG.527
                                                                        .>>>>> 3        $IF      1
2043   001000                         EXIT                .And recovered, ignore          AMSG.528
2044    <macro>                     $ENDIF                                              AMSG.529
                                                                        .<<<<< 3        $ENDIF   1
2044    <macro>                   $ENDIF                                                AMSG.530
                                                                        .<<<<< 2        $ENDIF   1
2044    <macro>                 $ELSE                       .If previous error unrecovered  AMSG.531
2045    <macro>                   $IF (R!%W1 <= R!%W4)      .If new error higher in order   AMSG.532
                                                                        .>>>>> 2        $IF      1
2053   001000                         EXIT                .Ignore                         AMSG.533
2054    <macro>                   $ELSEIF (R!%W5 = DAR$REC),OR,(R!%W5 = DAR$COR)          AMSG.534
2064   001000                         EXIT                .Or recovered, ignore           AMSG.535
2065    <macro>                   $ENDIF                                                AMSG.536
                                                                        .<<<<< 2        $ENDIF   1
2065    <macro>                 $ENDIF                                                  AMSG.537
                                                                        .<<<<< 1        $ENDIF   1
2065    <macro>               $ENDIF                                                    AMSG.538
                                                                        .<<<<< 0        $ENDIF   1
2065    <macro>               PUT R!%W5,DA@RC,R!MDAL      .Set new response               AMSG.539
2100    <macro>               PUT R!%W4,DA@LUE,R!MDAL     .NEW LAST                       AMSG.540
                                                                                            AMSG.541
                              * CONVERT PHYSICAL ERROR ADDRESS TO LOGICAL                   AMSG.542
                              * (SECTOR)X(UNITS)+(UNIT-OFFSET)                              AMSG.543
                                                                                            AMSG.544
2114    <macro>               GET R!%W1,BD@NUM,R!DCB      .UNITS IN GROUP                 AMSG.545
2121    <macro>               GET R!%W2,DA@SEC,R!DAL      .PHYSICAL SECTOR                AMSG.546
2127   010000  024425          R!%W3 = 0                   .                               AMSG.547
2131    <macro>               $UNTIL (R!%W1 = 0)          .MULTIPLY SECTOR BY UNITS       AMSG.548
2133   020425  022424  024425   R!%W3 = R!%W3 + R!%W2      .                               AMSG.549
2136   027423                   R!%W1 = R!%W1 - 1          .                               AMSG.550
2137    <macro>               $ENDTIL                                                     AMSG.551
2140   020425  022426  024425 R!%W3 = R!%W3 + R!%W4        .ADD UNIT OFFSET                AMSG.552
2143    <macro>               PUT R!%W3,DA@SEC,R!MDAL     .ENTER IN RESPONSE              AMSG.553
                                                                                            AMSG.554
                              * If unrecovered error, adjust length to number of actual     AMSG.555
                              * sectors done.                                               AMSG.556
                                                                                            AMSG.557
```

```
2156    <macro>                          $IF (R!%W5 = DAR$UN.   JR,(R!%W5 = DAR$UNC) .Unrecovered          AMSG.558
                                                                                      .>>>>> 0            $IF      1
2165    <macro>                            GET R!%W1,DA@LN0,R!DAL      .Successful transfer length       AMSG.559
2172    <macro>                            GET R!%W2,DA@LN1,R!DAL                                        AMSG.560
2177    <macro>                            DSUB R!%W1,R!%W2,D'512      .Minimum issued to all units      AMSG.561
2211    <macro>                            GET R!%W3,DA@TRQ,R!DAL      .Number of units I/Oed            AMSG.562
2220    010000  024451                     R!LN0 = 0                   .Initialize for product           AMSG.563
2222    024452                             R!LN1 = A                                                     AMSG.564
2223    <macro>                            $UNTIL (R!%W3 = 0)          .Multiply                         AMSG.565
2225    <macro>                              DADD R!LN0,R!LN1,R!%W1,R!%W2                                AMSG.566
2235    027425                               R!%W3 = R!%W3 - 1                                           AMSG.567
2236    <macro>                            $ENDTIL                                                       AMSG.568
2237    <macro>                            $IF (R!%W5 = DAR$UNC)       .If uncorrected read              AMSG.569
                                                                                      .>>>>> 1            $IF      1
2243    026426                               R!%W4 = R!%W4 + 1         .Include sector of error          AMSG.570
2244    <macro>                            $ENDIF                                                        AMSG.571
                                                                                      .<<<<< 1            $ENDIF   1
2244    020426  005011  024423             R!%W1 = R!%W4 < D'9         .Add remainder                    AMSG.572
2247    <macro>                            DADD R!LN0,R!LN1,R!%W1                                        AMSG.573
2254    <macro>                            PUT R!LN0,DA@LN0,R!MDAL     .Save for response                AMSG.574
2261    <macro>                            PUT R!LN1,DA@LN1,R!MDAL                                       AMSG.575
2266    <macro>                          $ENDIF                                                          AMSG.576
                                                                                      .<<<<< 0            $ENDIF   1
2266    001000                           EXIT                                                            AMSG.577
                                                                                                         AMSG.578
                                         *                                                               AMSG.579
                                         * RELEASE LOCAL DAL                                             AMSG.580
                                         *                                                               AMSG.581
                                                                                                         AMSG.582
2267                             RELDAL  *                                                               AMSG.583
2267    <macro>                          RELDAL R!DAL                                                    AMSG.584
2277    001000                           EXIT                                                            AMSG.585
                                                                                                         AMSG.586
                                         *                                                               AMSG.587
                                         * SEND RESPONSE TO MAINFRAME                                    AMSG.588
                                         *                                                               AMSG.589
                                                                                                         AMSG.590
2300                             RESPOND *                                                               AMSG.591
2300    <macro>                          GET R!NUM,DA@TRQ,R!MDAL     .Total requests sent              AMSG.592
2307    <macro>                          FLDADD 1,DA@FRS,R!MDAL      .Update final responses           AMSG.593
2332    <macro>                          $IF (A = R!NUM)             .If all final responses in        AMSG.594
                                                                                      .>>>>> 0            $IF      1
```

```
2335   010000  024423          R!%W1 = 0                    .Initialize for search          AMSG.595
2337   024424                  R!%W2 = A                                                    AMSG.596
2340    <macro>                GET R!%W3,BD@IQU,R!DCB       .First on queue                 AMSG.597
                                                                                            AMSG.598
                              *  Search for all complete MDAL's.  Replies must be sent      AMSG.599
                              *  to the mainframe in the same order in which they were      AMSG.600
                              *  received.                                                  AMSG.601
                                                                                            AMSG.602
2345    <macro>                $UNTIL (R!%W1 # R!%W2),OR,(R!%W3 = 0)                       AMSG.603
2352    <macro>                  GET R!%W1,DA@IRQ,R!%W3     .Total DAL's for request        AMSG.604
2361    <macro>                  GET R!%W2,DA@FRS,R!%W3     .Final responses                AMSG.605
2367    <macro>                GET R!%W4,DA@NFR,R!MDAL      .Flag for no final response     AMSG.606
2376   030425  024425            R!%W3 = (R!%W3)            .Next request                   AMSG.607
2400    <macro>                  $IF (R!%W1 = R!%W2)        .If all reponses in             AMSG.608
                                                                          .>>>>> 1              $IF      1
2404   002000  000000              I = 0                                                     AMSG.609
2406    <macro>                    ADDRESS R!DZ,BD@IQU,R!DCB .Queue address                  AMSG.610
2411    <macro>                    RKERN HGET               .Take first entry                AMSG.611
2413    <macro>                  $IF (R!%W4 = 0)            .If final response expected      AMSG.612
                                                                          .>>>>> 2              $IF      1
2416    <macro>                    RKERN IDALSND            .Send it                         AMSG.613
2420    <macro>                  $ELSE                                                       AMSG.614
2421    <macro>                    RKERN HONN               .Release DAL                     AMSG.615
2423    <macro>                  $ENDIF                                                      AMSG.616
                                                                          .<<<<< 2              $ENDIF   1
2423    <macro>                    FLDSUB 1,CPI@DP,R!%LSPI .Adjust disk request population   AMSG.617
2427   003000                      I = 1                                                     AMSG.618
2430    <macro>                  $ENDIF                                                      AMSG.619
                                                                          .<<<<< 1              $ENDIF   1
2430    <macro>                $ENDTIL                                                       AMSG.620
2431    <macro>              $ENDIF                                                          AMSG.621
                                                                          .<<<<< 0              $ENDIF   1
2431   073142                R = RELDAL                   .Release local DAL                  AMSG.622
2432   001000                EXIT                                                             AMSG.623
                                                                                              AMSG.624


                              END                                                             AMSG.626
```

0%	%ACTIVE		3:11 D	6:16	1[illegible]	17:18 D	17:18				
0%	%B		1: 4 D	4:23 B	4:26 B	4:29 B	4:32 B	11:12 B	11:31 B	11:35 B	
			11:38 B	13: 2 B	13:40 B	15: 4 B	17:11 B	17:21 B	17:27 B	17:30 B	
0%	%EX		6:14 D	6:14	7:39 D	7:39	9: 9 D	9: 9	11:22 D	11:22	
			12: 6 D	12: 6	17:18 D	17:18	20:31 D	20:31			
10	%GBLNUM	$APTEXT	2:12 F	2:12							
400	%GBLREG	$APTEXT	2:12								
0	%KERNEL	$APTEXT	4: 6 D	4: 6 B	4:19 D	4:19 B	4:42 D	4:42 B	6: 7 D	6: 7 B	
			6:28 D	6:28 B	6:31 D	6:31 B	6:33 D	6:33 B	7:12 D	7:12 B	
			7:23 D	7:23 B	7:25 D	7:25 B	8: 1 D	8: 1 B	9:30 D	9:30 B	
			12:35 D	12:35 B	12:39 D	12:39 B	13: 1 D	13: 1 B	13:19 D	13:19 B	
			13:38 D	13:38 B	14:24 D	14:24 B	14:34 D	14:34 B	15: 1 D	15: 1 B	
			21:18 D	21:18 B	21:21 D	21:21 B	21:23 D	21:23 B			
0%	%LSPI		3:12 D	13:39	21:26						
0%	%LSPO		3:13 D	5:12							
0%	%MYID		3:14 D	17:10	18: 7						
0%	%PUNT		4:14 D	4:14	4:40 D	4:40	6:15 D	6:15	7: 7 D	7: 7	
			9:10 D	9:10	9:26 D	9:26	16:38 D	16:38			
177777	AD$NODEM	$APTEXT	6:14	9: 9							
20	AD@FU@N	$APTEXT	9:25	9:25 F							
12	AD@FU@P	$APTEXT	9:25 F	9:25							
0	AD@FU@S	$APTEXT	9:25	9:25 F							
20	AD@P1@N	$APTEXT	6:16	6:16 F							
14	AD@P1@P	$APTEXT	6:16 F	6:16							
0	AD@P1@S	$APTEXT	6:16	6:16 F							
17	AD@PRI@M	$APTEXT	11: 8	11:36							
4	AD@PRI@N	$APTEXT	11: 8	11:36							
2	AD@PRI@P	$APTEXT	11: 8 F	11: 8	11:36 F	11:36					
14	AD@PRI@S	$APTEXT	11: 8	11: 8 F	11:36	11:36 F					
20	AD@RC@N	$APTEXT	7:10 F	9:29 F	17:18	17:18 F					
13	AD@RC@P	$APTEXT	7:10 F	7:10	9:29 F	9:29	17:18 F	17:18			
0	AD@RC@S	$APTEXT	7:10 F	9:29 F	17:18	17:18 F					
20	AM$ALERT	$APTEXT	4:34								
24	AM$AWAKE	$APTEXT	4:36								
42	AM$BDRSP	$APTEXT	4:23								
40	AM$COSTO	$APTEXT	4:25	12:22							
21	AM$CRTDN	$APTEXT	4:35	6:22							
46	AM$HSPDN	$APTEXT	4:32	12:27							
45	AM$HSPTO	$APTEXT	4:31	12:26							
41	AM$TODON	$APTEXT	4:26	12:23							
33	AM$OUTC	$APTEXT	4:38								
34	AM$OUTDN	$APTEXT	4:39								

25	AM$RSPND	$APTEXT	4:37									
44	AM$XFRDN	$APTEXT	4:29		12:25							
43	AM$XFRIO	$APTEXT	4:28		12:24							
10	AMSG$LIM	$APTEXT	11: 7									
0X	APNTR		3:15	D	12: 4		12: 7					
20	BD@IQU@N	$APTEXT	21: 3		21: 3	F						
3	BD@IQU@P	$APTEXT	16:33		21: 3	F	21: 3		21:17			
0	BD@IQU@S	$APTEXT	21: 3		21: 3	F						
20	BD@NUM@N	$APTEXT	19:30		19:30	F						
1	BD@NUM@P	$APTEXT	19:30	F	19:30							
0	BD@NUM@S	$APTEXT	19:30		19:30	F						
20	BD@PUT@N	$APTEXT	18:20		18:20	F						
2	BD@PUT@P	$APTEXT	18:20	F	18:20							
0	BD@PUT@S	$APTEXT	18:20		18:20	F						
0X	CA		3:16	D	7:22		13:16		14:19		14:31	
0X	CB		3:17	D	7:17		13:17		14:20		14:32	
0X	CC		3:18	D	7:20		13:18		14:21		14:33	
0X	CD		3:19	D	14:22							
0X	CE		3:20	D	14:23							
0X	COMIO		14:24	D	14:24							
20	CPI@DP@N	$APTEXT	13:39	F	21:26	F						
12	CPI@DP@P	$APTEXT	13:39	F	13:39		21:26	F	21:26			
777	CPO@CN@M	$APTEXT	5:12									
11	CPO@CN@N	$APTEXT	5:12									
0	CPO@CN@P	$APTEXT	5:12	F								
7	CPO@CN@S	$APTEXT	5:12		5:12	F						
30	DA@@LE	$APTEXT	17:17									
10	DA@@LH	$APTEXT	17:17									
20	DA@ACI@N	$APTEXT	9:24		9:24	F	14:40		14:40	F		
4	DA@ACI@P	$APTEXT	7: 1		8:16		9:24	F	9:24		14:40	F 14:40
0	DA@ACI@S	$APTEXT	9:24		9:24	F	14:40		14:40	F		
20	DA@ARC@N	$APTEXT	9:28		9:28	F						
11	DA@ARC@P	$APTEXT	9:28	F	9:28							
0	DA@ARC@S	$APTEXT	9:28		9:28	F						
20	DA@BMO@N	$APTEXT	11: 2		11: 2	F	11:41	F	14:20		14:20	F 14:32 14:32 F
34	DA@BMO@P	$APTEXT	11: 2	F	11: 2		11:41	F	11:41		14:20	F 14:20 14:32 F 14:32
0	DA@BMO@S	$APTEXT	11: 2		11: 2	F	11:41	F	14:20		14:20	F 14:32 14:32 F
20	DA@BM1@N	$APTEXT	11: 3		11: 3	F	11:42	F	14:21		14:21	F 14:33 14:33 F
35	DA@BM1@P	$APTEXT	11: 3	F	11: 3		11:42	F	11:42		14:21	F 14:21 14:33 F 14:33
0	DA@BM1@S	$APTEXT	11: 3		11: 3	F	11:42	F	14:21		14:21	F 14:33 14:33 F
177	DA@CHN@M	$APTEXT	18:21									
7	DA@CHN@N	$APTEXT	18:21									

27	DA@CHN@P	$APTEXT	18:21 F	18:21						
11	DA@CHN@S	$APTEXT	18:21	18:21 F						
20	DA@DAT@N	$APTEXT	10:29 F	13:31	13:31 F					
13	DA@DAT@P	$APTEXT	10:29 F	10:29	13:31 F	13:31				
0	DA@DAT@S	$APTEXT	10:29 F	13:31	13:31 F					
20	DA@DID@N	$APTEXT	13:23	13:23 F	13:26 F					
10	DA@DID@P	$APTEXT	13:23 F	13:23	13:26 F	13:26				
0	DA@DID@S	$APTEXT	13:23	13:23 F	13:26 F					
20	DA@DLO@N	$APTEXT	13:17	13:17 F						
2	DA@DLO@P	$APTEXT	13:17 F	13:17						
0	DA@DLO@S	$APTEXT	13:17	13:17 F						
20	DA@DL1@N	$APTEXT	13:18	13:18 F						
3	DA@DL1@P	$APTEXT	13:18 F	13:18						
0	DA@DL1@S	$APTEXT	13:18	13:18 F						
20	DA@ERC@N	$APTEXT	17:19 F	18:14	18:14 F	18:15 F				
17	DA@ERC@P	$APTEXT	17:19 F	17:19	18:14 F	18:14	18:15 F	18:15		
0	DA@ERC@S	$APTEXT	17:19 F	18:14	18:14 F					
17	DA@ERS@M	$APTEXT	17:14							
4	DA@ERS@N	$APTEXT	17:14 F	17:14						
12	DA@ERS@P	$APTEXT	17:14 F	17:14						
4	DA@ERS@S	$APTEXT	17:14	17:14 F						
377	DA@FC@M	$APTEXT	10:18 F	10:20 F						
10	DA@FC@N	$APTEXT	10:15	10:15 F	10:18 F	10:18	10:20 F	10:20	10:40	10:40 F
			14:15	14:15 F	17: 6	17: 6 F	18: 5	18: 5 F		
26	DA@FC@P	$APTEXT	10:15 F	10:15	10:18 F	10:18	10:20 F	10:20	10:40 F	10:40
			14:15 F	14:15	17: 6 F	17: 6	18: 5 F	18: 5		
0	DA@FC@S	$APTEXT	10:15	10:15 F	10:18 F	10:18	10:20 F	10:20	10:40	10:40 F
			14:15	14:15 F	17: 6	17: 6 F	18: 5	18: 5 F		
17	DA@FRS@M	$APTEXT	20:40							
4	DA@FRS@N	$APTEXT	20:40 F	20:40	21:11	21:11 F				
12	DA@FRS@P	$APTEXT	20:40 F	20:40	21:11 F	21:11				
0	DA@FRS@S	$APTEXT	20:40	20:40 F	21:11	21:11 F				
6	DA@HPO@P	$APTEXT	6:12	6:25	7: 8	7:33				
20	DA@IFC@N	$APTEXT	4:21	4:21 F	12:20	12:20 F	12:23 F	12:25 F	12:27 F	
1	DA@IFC@P	$APTEXT	4:21 F	4:21	6:22	12:20 F	12:20	12:23 F	12:23	12:25 F
			12:25	12:27 F	12:27					
0	DA@IFC@S	$APTEXT	4:21	4:21 F	12:20	12:20 F	12:23 F	12:25 F	12:27 F	
3	DA@IOP@M	$APTEXT	12:38							
2	DA@IOP@N	$APTEXT	12:38	12:38 F						
27	DA@IOP@P	$APTEXT	12:38 F	12:38						
5	DA@IOP@S	$APTEXT	12:38	12:38 F						
6	DA@LCH@N	$APTEXT	16:14	16:14 F						

21	DA@LCH@P	$APTEXT	16:14 F	16:14						
0	DA@LCH@S	$APTEXT	16:14	16:14 F						
20	DA@LNO@N	$APTEXT	10:26	10:26 F	11: 4	11: 4 F	12: 1 F	13:32 F	20: 3	20: 3 F
			20:20 F							
32	DA@LNO@P	$APTEXT	10:26 F	10:26	11: 4 F	11: 4	12: 1 F	12: 1	13:32 F	13:32
			20: 3 F	20: 3	20:20 F	20:20				
0	DA@LNO@S	$APTEXT	10:26	10:26 F	11: 4	11: 4 F	12: 1 F	13:32 F	20: 3	20: 3 F
			20:20 F							
20	DA@LN1@N	$APTEXT	10:27	10:27 F	11: 5	11: 5 F	12: 2 F	13:33 F	14:23	14:23 F
			20: 4	20: 4 F	20:21 F					
33	DA@LN1@P	$APTEXT	10:27 F	10:27	11: 5 F	11: 5	12: 2 F	12: 2	13:33 F	13:33
			14:23 F	14:23	20: 4 F	20: 4	20:21 F	20:21		
0	DA@LN1@S	$APTEXT	10:27	10:27 F	11: 5	11: 5 F	12: 2 F	13:33 F	14:23	14:23 F
			20: 4	20: 4 F	20:21 F					
20	DA@LOC@N	$APTEXT	14:22	14:22 F						
37	DA@LOC@P	$APTEXT	14:22 F	14:22						
0	DA@LOC@S	$APTEXT	14:22	14:22 F						
17	DA@LUE@M	$APTEXT	18:39	19:25						
4	DA@LUE@N	$APTEXT	18:39	18:39 F	19:25 F	19:25				
12	DA@LUE@P	$APTEXT	18:39 F	18:39	19:25 F	19:25				
10	DA@LUE@S	$APTEXT	18:39	18:39 F	19:25 F	19:25				
20	DA@MES@N	$APTEXT	16:26	16:26 F						
5	DA@MES@P	$APTEXT	16:26 F	16:26						
0	DA@MES@S	$APTEXT	16:26	16:26 F						
1	DA@NFR@M	$APTEXT	18: 9 F	21:12						
1	DA@NFR@N	$APTEXT	18: 9 F	18: 9	21:12	21:12 F				
12	DA@NFR@P	$APTEXT	18: 9 F	18: 9	21:12 F	21:12				
14	DA@NFR@S	$APTEXT	18: 9 F	18: 9	21:12	21:12 F				
3	DA@NUM@M	$APTEXT	16:24							
2	DA@NUM@N	$APTEXT	16:24	16:24 F						
5	DA@NUM@P	$APTEXT	16:24 F	16:24						
2	DA@NUM@S	$APTEXT	16:24	16:24 F						
377	DA@RC@M	$APTEXT	12:31 F	18: 1	18:40	19:24				
10	DA@RC@N	$APTEXT	12:31 F	18: 1	18:40	19:24 F				
26	DA@RC@P	$APTEXT	12:31 F	12:31	18: 1 F	18: 1	18:40 F	18:40	19:24 F	19:24
10	DA@RC@S	$APTEXT	12:31 F	18: 1	18: 1 F	18:40	18:40 F	19:24 F		
177400	DA@RC@X	$APTEXT	12:31	19:24						
177	DA@SEC@M	$APTEXT	19:38							
7	DA@SEC@N	$APTEXT	19:31	19:31 F	19:38 F	19:38				
31	DA@SEC@P	$APTEXT	19:31 F	19:31	19:38 F	19:38				
0	DA@SEC@S	$APTEXT	19:31	19:31 F	19:38 F	19:38				
20	DA@SEQ@N	$APTEXT	10:28 F	13:30	13:30 F					

12	DA@SEQ@P	$APTEXT	10:28 F	10:28	1. F	13:30				
0	DA@SEQ@S	$APTEXT	10:28 F	13:30	13:30 F					
20	DA@SID@N	$APTEXT	13:24	13:24 F	13:25 F					
11	DA@SID@P	$APTEXT	13:24 F	13:24	13:25 F	13:25				
0	DA@SID@S	$APTEXT	13:24	13:24 F	13:25 F					
377	DA@SQN@M	$APTEXT	16:34	16:39						
10	DA@SQN@N	$APTEXT	16:34	16:39						
20	DA@SQN@P	$APTEXT	16:34 F	16:34	16:39 F	16:39				
10	DA@SQN@S	$APTEXT	16:34	16:34 F	16:39	16:39 F				
37777	DA@TMO@M	$APTEXT	10:42	11:39						
16	DA@TMO@N	$APTEXT	10:42	11:39 F						
24	DA@TMO@P	$APTEXT	10:42 F	10:42	11:39 F	11:39				
2	DA@TMO@S	$APTEXT	10:42	10:42 F	11:39 F					
140000	DA@TMO@X	$APTEXT	11:39							
20	DA@TM1@N	$APTEXT	11: 1	11: 1 F	11:40 F					
25	DA@TM1@P	$APTEXT	11: 1 F	11: 1	11:40 F	11:40				
0	DA@TM1@S	$APTEXT	11: 1	11: 1 F	11:40 F					
2	DA@TM@N	$APTEXT	10:41	10:41 F						
24	DA@TM@P	$APTEXT	10:41 F	10:41						
0	DA@TM@S	$APTEXT	10:41	10:41 F						
17	DA@TRQ@M	$APTEXT	17:13	18: 8	20: 6	20:39	21:10			
4	DA@TRQ@N	$APTEXT	17:13	17:13 F	18: 8 F	18: 8	20: 6	20: 6 F	20:39	20:39 F
			21:10	21:10 F						
20	DA@TRQ@P	$APTEXT	17:13 F	17:13	18: 8 F	18: 8	20: 6 F	20: 6	20:39 F	20:39
			21:10 F	21:10						
4	DA@TRQ@S	$APTEXT	17:13	17:13 F	18: 8	18: 8 F	20: 6	20: 6 F	20:39	20:39 F
			21:10	21:10 F						
5	DA@TYP@N	$APTEXT	16:21	16:21 F						
27	DA@TYP@P	$APTEXT	16:21 F	16:21						
0	DA@TYP@S	$APTEXT	16:21	16:21 F						
1	DAF$READ	$APTEXT	10:17							
6	DAF$WBFR	$APTEXT	17: 8	18: 7						
13	DAR$COR	$APTEXT	18: 4	19: 2	19: 6	19:16				
0	DAR$OK	$APTEXT	18: 2	18:42						
1	DAR$REC	$APTEXT	18: 4	19: 2	19: 6	19:16				
14	DAR$UNC	$APTEXT	20: 2	20:14						
15	DAR$UNR	$APTEXT	20: 2							
0X	DCCB		3:34 D	16:15						
0X	DS		3:21 D	6:27	12:34					
1	DSK$DD19	$APTEXT	16:23							
2	DSK$DD29	$APTEXT	16:23							
0X	DSUM		6:28 D	6:28	12:35 D	12:35				

0X	DY		3:22	D	4:20	6:32	7:24	12:42	13:37			
0X	DZ		3:23	D	4:17	4:18	21:17					
0X	ECPQ		3:24	D	11:32							
0X	ED		3:25	D	4: 5	7:42						
0X	EH		3:26	D	4: 7	6:29	7:11	7:41	9:24	9:25	9:29	
			14:40		14:42							
0X	EI		3:27	D	6:29	12:38						
0X	EMSGIOP		6:31	D	6:31	12:39	D	12:39				
0X	ENQU		8: 1	D	8: 1							
0X	EQCP		7:12	D	7:12	9:30	D	9:30	15: 1	D	15: 1	
0X	ESWP		4:42	D	4:42							
0X	ETOQ		4: 6	D	4: 6							
15	F$ALERT	$APTEXT	7: 3									
16	F$AWAKE	$APTEXT	8:15									
55	F$CREATE	$APTEXT	6:14		9: 9							
4	F$GIVEUP	$APTEXT	12: 6									
37	F$OUTCAL	$APTEXT	9:26									
2	F$POP	$APTEXT	7:39									
27	F$RDAL	$APTEXT	20:31									
34	F$SEND	$APTEXT	17:18									
45	F$TRANSF	$APTEXT	11:22									
0	FS$IN	$APTEXT	10:20		14:17							
1	FS$OUT	$APTEXT	10:18									
0	FS$WAIT	$APTEXT	11:22									
0X	HB		3:28	D	6: 6							
0X	HC		3:29	D	6: 8							
0X	HGET		4:19	D	4:19	21:18	D	21:18				
0X	HMA		6: 7	D	6: 7							
0X	HONN		6:33	D	6:33	7:25	D	7:25	13: 1	D	13: 1	21:23 D 21:23
0X	IDALSND		13:38	D	13:38	21:21	D	21:21				
0X	MIMC		3:30	D	4: 9							
0X	MIMDALQ		3:31	D	4:17							
0X	MIMMQ		3:32	D	4: 3							
1	MIO$BMR	$APTEXT	14:19									
0X	MPOPCELL		3:33	D	6:19							
3	MQ$DLAY	$APTEXT	4:13									
0X	O$AMSG		1: 4	D	1: 4							
0X	O$OUTCAL		9: 9	D	9: 9							
17	OV@NUM@N	$APTEXT	1: 4									
1	OV@TYP@N	$APTEXT	1: 4									
20	PC@ACT@N	$APTEXT	6:17	F								
1	PC@ACT@P	$APTEXT	6:17	F	6:17							

0	PC@ACI@S	$APTEXT	6:17 F							
4	PC@DAL@P	$APTEXT	7:42							
6	PC@LEN@P	$APTEXT	6: 6	6:11						
2	PC@PSH@P	$APTEXT	7:35							
0	PT$HUH	$APTEXT	4:14	4:40	6:15	7: 7	9:10	9:26	16:38	
77	PUT@CH@M	$APTEXT	18:27							
6	PUT@CH@N	$APTEXT	18:27							
12	PUT@CH@S	$APTEXT	18:27	18:27 F						
0X	PUTDAL		7:23 D	7:23	13:19 D	13:19				
0X	PUTDISK		14:34 D	14:34						
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
11	WRD$BUF	$APTEXT	11: 7							

SYMBOL QUALIFIER IS AMSG

1	$CONST	10:18 D	10:18 F	10:18	10:20 D	10:20 F	10:20	12:23 D	12:23 F	
		12:23	12:25 D	12:25 F	12:25	12:27 D	12:27 F	12:27	12:31 D	
		12:31 F	17:19 D	17:19 F	18: 9 D	18: 9 F	18: 9			
453	$REGORG	2:12 D	2:12 F	2:13	2:13 D	2:14	2:14 D	2:15	2:15 D	
		2:16	2:16 D	2:17	2:17 D	2:17 F	2:18	2:18 D	2:19	
		2:19 D	2:20	2:20 D	2:21	2:21 D	2:22	2:22 D	2:23	
		2:23 D	2:23 F	2:24	2:24 D	2:25	2:25 D	2:26	2:26 D	
		2:27	2:27 D	2:28	2:28 D	2:28 F	2:29	2:29 D	2:30	
		2:30 D	2:31	2:31 D	2:32	2:32 D	2:33	2:33 D	2:34	
		2:34 D	2:35	2:35 D	2:36	2:36 D	2:37	2:37 D	2:38	
		2:38 D	2:39	2:39 D	2:40	2:40 D	2:41	2:41 D	2:42	
		2:42 D	3: 1	3: 1 D	3: 2	3: 2 D	3: 3	3: 3 D	3: 4	
		3: 4 D	3: 5	3: 5 D	3: 5 F	6:14	7:39	9: 9	11:22	
		11:22 F	12: 6	17:18	20:31					
0	%NP	1: 4	2:28 D							
430	%P	1: 4	2:28 D	2:28	11:22 F	11:22				
410	%S1	2:12	2:13 D	4: 9	4:21	6:12	6:16	6:17	6:22	
		6:23	6:25	6:26	7: 1	7: 5	7: 8	7:10	7:14	
		7:15	7:33	8:16	9:24	9:25	9:28	9:29	10:15	
		10:18	10:20	10:26	10:27	10:28	10:29	10:40	10:41	
		10:42	11: 1	11: 2	11: 3	11: 4	11: 5	11: 8	11:32	
		11:36	11:39	11:40	11:41	11:42	12: 1	12: 2	12: 4	
		12: 5	12: 7	12:20	12:23	12:25	12:27	12:31	12:38	
		13:17	13:18	13:23	13:24	13:25	13:26	13:30	13:31	
		13:32	13:33	13:39	14:15	14:20	14:21	14:22	14:23	
		14:32	14:33	14:40	16:14	16:21	16:24	16:26	16:34	

		16:39	17: 6	17:13	17:18	17:19	18: 1	18: 5	18: 9
		18:14	18:15	18:20	18:21	18:39	18:40	19:24	19:25
		19:30	19:31	19:38	20: 3	20: 4	20: 6	20:20	20:21
		20:39	21: 3	21:10	21:11	21:12	21:26		
411	%S2	2:12	2:14 D						
412	%S3	2:12	2:15 D						
413	%S4	2:12	2:16 D						
414	%S5	2:12	2:17 D						
415	%T1	2:18 D	6:11	17:14	17:17	18: 8	20:40		
416	%T2	2:19 D	6:11	17:14	17:17	18: 8	20:40		
417	%T3	2:20 D	11:26	17:14	17:17	18: 8	20: 5	20:40	
420	%T4	2:21 D							
421	%T5	2:22 D							
422	%T6	2:23 D							
423	%W1	2:24 D	6:13	6:14	6:16	6:17	9:28	9:29	11:33
		11:35	11:36	16:14	16:15	16:16	16:17	16:21	16:23
		18: 5	18: 7	18:14	18:15	18:20	18:25	18:39	19: 4
		19:14	19:30	19:33	19:35	20: 3	20: 5	20:10	20:18
		20:19	21: 1	21: 9	21:10	21:15			
424	%W2	2:25 D	16:34	16:36	18:21	18:24	18:29	18:40	18:42
		19: 2	19:31	19:34	20: 4	20: 5	20:10	21: 2	21: 9
		21:11	21:15						
425	%W3	2:26 D	16:35	16:36	16:39	18:22	18:24	18:25	18:26
		18:27	18:29	19:32	19:34	19:37	19:38	20: 6	20: 9
		20:11	21: 3	21: 9	21:10	21:11	21:13		
426	%W4	2:27 D	18:23	18:25	18:30	19: 4	19:14	19:25	19:37
		20:15	20:18	21:12	21:20				
427	%W5	2:28 D	18: 1	18: 4	19: 6	19:16	19:24	20: 2	20:14
1376	BIGDISK	4:23	16:12 D						
1177	BMRDN	4:26	13:12 D						
510	BMRIO	4:25	10:11 D						
133	CPUDLAY	4:13	5:11 D						
430	DAL	2:29 D	3: 8	4:20	4:21	9: 9	9:24	9:28	10:15
		10:18	10:20	10:26	10:27	10:28	10:29	10:40	10:41
		10:42	11: 1	11: 2	11: 3	11: 4	11: 5	11:39	11:40
		11:41	11:42	12: 1	12: 2	12: 4	12: 8	12:20	12:23
		12:25	12:27	12:31	12:34	12:38	12:42	13:17	13:18
		13:23	13:24	13:25	13:26	13:30	13:31	13:32	13:33
		13:37	14:15	14:20	14:21	14:22	14:23	14:32	14:33
		14:40	16:14	16:21	16:24	16:26	16:34	17: 6	17:13
		17:17	17:18	18: 1	18: 5	18:14	18:21	19:31	20: 3
		20: 4	20: 6	20:31					

447	DCB	3: 2 D	16:17	16	18:20	19:30	21: 3	21:17	
1630	ERRCHK	17: 5	17:39 D						
1277	HSPDN	4:32	14:11 D						
570	HSPTO	4:31	10:32 D						
451	LN0	3: 4 D	20: 7	20:10	20:19	20:20			
452	LN1	3: 5 D	20: 8	20:10	20:19	20:21			
570	LOOP	10:34 D	12: 9						
364	MAWAKE	4:36	7:32 D						
431	MB	2:30 D	4: 3	4: 5					
250	MCFIN	6: 9	6:24 D						
36	MCHKDAL	4: 8	4:16 D						
303	MCOMMN	7: 4 D	8:18						
150	MCREA	4:34	6: 5 D						
274	MCREDN	4:35	6:42 D						
432	MD	2:31 D	4: 7	4:11					
446	MDAI	3: 1 D	16:33	16:37	16:39	17:14	17:17	17:19	18: 8
		18: 9	18:15	18:39	18:40	19:24	19:25	19:38	20:20
		20:21	20:39	20:40	21:12				
433	ME	2:32 D	4:21	4:23	4:25	4:26	4:28	4:29	4:31
		4:32	4:34	4:35	4:36	4:37	4:38	4:39	12:20
		12:22	12:24	12:26					
430	MF	3: 8 D	6:12	6:22	6:25	6:27	6:32	7: 1	7: 8
		7:14	7:16	7:24	7:33	7:39	7:41	8:16	
332	MFREED	7:13 D	8:19	9:31	15: 4				
434	MG	2:33 D	4:11	4:13	6: 8	6:11	6:14	6:17	6:20
		6:21	6:25	10:15	10:17	10:40	11:22	14:15	14:17
435	MH	2:34 D	6:19	6:20	6:21	7: 2	7: 5	7:10	7:11
		8:17	8:18	10:41	11:22				
436	MI	2:35 D	7: 9	7:10	7:34	7:35	7:42	10:42	11:22
		11:24	11:39						
131	MIDLE	4:18	4:41 D						
6	MIMM	4: 2 D	5:19	6:34	7:26	8: 5	9:11	13: 2	13:40
		17:30							
437	MJ	2:36 D	7: 6	7: 7	7:35	7:38	7:39	11: 1	11:22
		11:24	11:40						
440	MK	2:37 D	7: 3	7: 7	8:15	11: 2	11:22	11:25	11:41
441	ML	2:38 D	7:16	7:17	7:18	7:19	7:20	7:21	11: 3
		11:22	11:25	11:42					
442	MM	2:39 D	10:26	10:28	11: 4	11:12	11:15	11:26	11:31
		12: 1	13:23	13:25	13:30	13:32			
443	MN	2:40 D	10:27	10:29	11: 5	11:12	11:15	11:18	11:26
		11:31	12: 2	13:24	13:26	13:31	13:33		

444	MO	2:41 D	11:16	11:18	11:22	11:24	11:25	11:26	
445	MP	2:42 D	11: 8	11:38					
422	MRESPD	4:37	8:14 D						
450	NUM	3: 3 D	11: 7	11:15	11:16	16:24	16:27	17:10	17:13
		17:16	18: 7	20:39	20:42				
434	OUTCALL	4:38	9: 8 D						
457	OUTRESP	4:39	9:23 D						
2267	RELDAL	17:21	20:30 D	21:33					
2300	RESPOND	17:11	17:27	20:38 D					
1362	XFRDN	4:29	14:38 D						
570	XFRIO	4:28	10:31 D						

```
0     <macro>                    OVERLAY   BEGIN                                                    BEGIN.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
6     <macro>                          REGDEFS   (CRT),,(CA,CL,MA,PA,PB,PC,PD,PE,PF,PG,PH,PI,PJ,PK,PL,    BEGIN.5
                                                   PM,PN,PO,PP,PQ,PR,TO,T1,T2,T3,T4,TM,TBL,ICOM,MU,ML)     BEGIN.6
                 400      .CRT      EQUALS    $REGORG                                                          $REG     3
                 410      %S1       EQUALS    $REGORG                                                          $REG     3
                 411      %S2       EQUALS    $REGORG                                                          $REG     3
                 412      %S3       EQUALS    $REGORG                                                          $REG     3
                 413      %S4       EQUALS    $REGORG                                                          $REG     3
                 414      %S5       EQUALS    $REGORG                                                          $REG     3
                 415      %T1       EQUALS    $REGORG                                                          $REG     3
                 416      %T2       EQUALS    $REGORG                                                          $REG     3
                 417      %T3       EQUALS    $REGORG                                                          $REG     3
                 420      %T4       EQUALS    $REGORG                                                          $REG     3
                 421      %T5       EQUALS    $REGORG                                                          $REG     3
                 422      %T6       EQUALS    $REGORG                                                          $REG     3
                 423      %W1       EQUALS    $REGORG                                                          $REG     3
                 424      %W2       EQUALS    $REGORG                                                          $REG     3
                 425      %W3       EQUALS    $REGORG                                                          $REG     3
                 426      %W4       EQUALS    $REGORG                                                          $REG     3
                 427      %W5       EQUALS    $REGORG                                                          $REG     3
                 430      CA        EQUALS    $REGORG                                                          $REG     3
                 431      CL        EQUALS    $REGORG                                                          $REG     3
                 432      MA        EQUALS    $REGORG                                                          $REG     3
                 433      PA        EQUALS    $REGORG                                                          $REG     3
                 434      PB        EQUALS    $REGORG                                                          $REG     3
                 435      PC        EQUALS    $REGORG                                                          $REG     3
                 436      PD        EQUALS    $REGORG                                                          $REG     3
                 437      PE        EQUALS    $REGORG                                                          $REG     3
                 440      PF        EQUALS    $REGORG                                                          $REG     3
                 441      PG        EQUALS    $REGORG                                                          $REG     3
                 442      PH        EQUALS    $REGORG                                                          $REG     3
                 443      PI        EQUALS    $REGORG                                                          $REG     3
                 444      PJ        EQUALS    $REGORG                                                          $REG     3
                 445      PK        EQUALS    $REGORG                                                          $REG     3
                 446      PL        EQUALS    $REGORG                                                          $REG     3
                 447      PM        EQUALS    $REGORG                                                          $REG     3
                 450      PN        EQUALS    $REGORG                                                          $REG     3
                 451      PO        EQUALS    $REGORG                                                          $REG     3
                 452      PP        EQUALS    $REGORG                                                          $REG     3
                 453      PQ        EQUALS    $REGORG                                                          $REG     3
                 454      PR        EQUALS    $REGORG                                                          $REG     3
                 455      TO        EQUALS    $REGORG                                                          $REG     3
                 456      T1        EQUALS    $REGORG                                                          $REG     3
```

```
457     T2      EQUALS  $REGORG                                          $REG    3
460     T3      EQUALS  $REGORG                                          $REG    3
461     T4      EQUALS  $REGORG                                          $REG    3
462     TM      EQUALS  $REGORG                                          $REG    3
463     TBL     EQUALS  $REGORG                                          $REG    3
464     ICOM    EQUALS  $REGORG                                          $REG    3
465     MU      EQUALS  $REGORG                                          $REG    3
466     ML      EQUALS  $REGORG                                          $REG    3
                                                                    BEGIN.7
 21     PARL    EQUALS  PR-PA              .Maximum parameter length    BEGIN.8

                EXT     %KCONS                                      BEGIN.10
                EXT     %MYID                                       BEGIN.11
                EXT     %TRANQ                                      BEGIN.12
                EXT     DCCB                                        BEGIN.13
                EXT     LBFW                                        BEGIN.14
                EXT     LBPT                                        BEGIN.15
                EXT     MATX                                        BEGIN.16
                EXT     MIAQ                                        BEGIN.17
                EXT     TMCB                                        BEGIN.18
                EXT     O$ACOM                                      BEGIN.19
                EXT     O$ADEM                                      BEGIN.20
                EXT     O$AMSG                                      BEGIN.21
                EXT     O$BCOM                                      BEGIN.22
                EXT     O$CDEM                                      BEGIN.23
                EXT     O$CLOCK                                     BEGIN.24
                EXT     O$CRTDEM                                    BEGIN.25
                EXT     O$D4DEM                                     BEGIN.26
                EXT     O$DISK                                      BEGIN.27
                EXT     O$ICOM                                      BEGIN.28
                EXT     O$TRANSF                                    BEGIN.29
                EXT     D$ACOM                                      BEGIN.30
                EXT     D$ADEM                                      BEGIN.31
                EXT     D$AMSG                                      BEGIN.32
                EXT     D$BCOM                                      BEGIN.33
                EXT     D$CDEM                                      BEGIN.34
                EXT     D$CLOCK                                     BEGIN.35
                EXT     D$CRTDEM                                    BEGIN.36
                EXT     D$D4DEM                                     BEGIN.37
                EXT     D$DISK                                      BEGIN.38
                EXT     D$ICOM                                      BEGIN.39
```

```
EXT     D$TRANSF                                        BEGIN.40
EXT     EMEMSIZ                                         BEGIN.41
```

```
  6   020000  004001  011003          R!CRT = R!%KCONS >     0'3    .CRT index for CALL                BEGIN.43
      024400
 12   010000                           A = 0                         .Initialize DKERR buffer address   BEGIN.44
 13   024465                           R!MU = A                                                         BEGIN.45
 14   024466                           R!ML = A                                                         BEGIN.46
                                                                                                        BEGIN.47
                                *       Process Deadstart commands.                                      BEGIN.48
                                                                                                        BEGIN.49
 15   010000  024432                   MA = 0                                                           BEGIN.50
 17    <macro>                         FIND      AMAP,T3,T4,T2       .Get ready to read AMAP            BEGIN.51
 36   127000 /001277                   P = BEG1, A # 0               .If AMAP not found                 BEGIN.52
 40   020457  013512  100004           P = BEG1, T2 > 512            .If bigger than 1 buffer           BEGIN.53
      102003  075000 /001277
 46    <macro>                         BGET     MA                   .Get AMAP buffer                   BEGIN.54
 56   127000 /001277                   P = BEG1, A # 0               .If no space available             BEGIN.55
 60    <macro>                         MOSR      T3,T4,MA,T2         .Load AMAP                         BEGIN.56
                                                                                                        BEGIN.57
100   010000  024464                   R!ICOM = 0                    .ICOM setup flag                   BEGIN.58
                                                                                                        BEGIN.59
                                       *                                                                BEGIN.60
                                       * INITIALIZE TARGET MEMORY CONTROL TABLES (TMCB, TM@xxxx)        BEGIN.61
                                       *                                                                BEGIN.62
                                       *         R!MA = AMAP base address                               BEGIN.63
                                       *                                                                BEGIN.64
                                                                                                        BEGIN.65
102   014000 /000000  024463           R!TBL= TMCB                   .Target memory control block       BEGIN.66
                                                                                                        BEGIN.67
                                       * Set up control table for BMR.  Search through logical         BEGIN.68
                                       * device entries to see if BMR is defined.                       BEGIN.69
                                                                                                        BEGIN.70
105   020432  012017  024410           R!T1 = (R!MA + 0'17)          .Pointer to logical devices        BEGIN.71
      030410  024456
112    <macro>                         $IF (A # 0)                   .If any                            BEGIN.72
                                                                                        .>>>>> 0          $IF      1
114   020456  022432  024456             R!T1 = R!T1 + R!MA          .Add AMAP base                     BEGIN.73
117   010000  024457                     R!T2 = 0                    .Initialize                        BEGIN.74
121    <macro>                           $UNTIL (R!T2 = LDV$END)     .Until all searched                BEGIN.75
125   030456  024457                       R!T2 = (R!T1)             .Device type                       BEGIN.76
127    <macro>                             $IF (A = LDV$BMR)         .BMR device                        BEGIN.77
                                                                                        .>>>>> 1          $IF      1
132   010006  024460                         R!T3 = TM@LT            .Table length                      BEGIN.78
134   014000 /052115  024423                 R!%W1 = 'TM'R           .ASCII header                      BEGIN.79
```

```
137   014000 /040102  024424          R!%W2 = '@B'R                                          BEGIN.80
142   014000 /046522  024425          R!%W3 = 'MR'R                                          BEGIN.81
145   077000 /002512                  R = GETBUF                .Allocate memory             BEGIN.82
147   020463  012001  024410          (R!TBL + TM$BMR) = R!T4 .Save pointer to TM table      BEGIN.83
      020461  034410
154   026456                          R!T1 = R!T1 + 1           .Channel                     BEGIN.84
155    <macro>                        $IF (R!%MYID = MIOP)      .If MIOP                     BEGIN.85
                                                                                .>>>>> 2          $IF      1
160   030456  013020  016000            R!T0 = (R!T1) - O'20 + DCCB .Place for DCB pointer   BEGIN.86
     /000000  024455
165   020461  034455                    (R!T0) = R!T4           .Use same table for DCB      BEGIN.87
167    <macro>                          STORE BIOP,TM@IOP,R!T4 .BIOP handles BMR             BEGIN.88
174    <macro>                        $ELSE                                                  BEGIN.89
175    <macro>                          PUT R!%MYID,TM@IOP,R!T4 .Processor number            BEGIN.90
202    <macro>                        $ENDIF                                                 BEGIN.91
                                                                                .<<<<< 2          $ENDIF   1
202    <macro>                        PUT TM$BMR,TM@TYP,R!T4  .Memory type                   BEGIN.92
204   020432  012020  024410          R!%W3 = (R!MA + O'20)   .End of IOS buffer memory      BEGIN.93
      030410  024425
211    <macro>                        PUT R!%W3,TM@FWO,R!T4   .BMR Base address              BEGIN.94
216   026456                          R!T1 = R!T1 + 1         .Number of 65K segments        BEGIN.95
217   030456  022425  024425          R!%W3 = (R!T1) + R!%W3  .Add to base                   BEGIN.96
222    <macro>                        PUT R!%W3,TM@LMO,R!T4   .BMR Limit address            BEGIN.97
227   070026                          P = TMCMEM              .Skip to Central memory        BEGIN.98
230    <macro>                      $ELSEIF (R!T2 = LDV$DSK)  .Striped disk group            BEGIN.99
235   010004  025456                  R!T1 = R!T1 + 4         .Physical device list          BEGIN.100
237    <macro>                        $UNTIL ((R!T1) = 0)     .For all units defined         BEGIN.101
241   026456                            R!T1 = R!T1 + 1       .Next                          BEGIN.102
242    <macro>                        $ENDTIL                                                BEGIN.103
243   026456                          R!T1 = R!T1 + 1         .Point to next entry           BEGIN.104
244    <macro>                      $ELSEIF (R!T2 # LDV$END)  .Illegal value                 BEGIN.105
252    <macro>                        $PUNTIF CODE=PT$INIT                                   BEGIN.106
254    <macro>                      $ENDIF                                                   BEGIN.107
                                                                                .<<<<< 1          $ENDIF   1
254    <macro>                    $ENDTIL                                                    BEGIN.108
255    <macro>                  $ENDIF                                                       BEGIN.109
                                                                                .<<<<< 0          $ENDIF   1
                                                                                              BEGIN.110
                                *  Set up control table for Central Memory.  Search          BEGIN.111
                                *  channel entries for high speed to CMEM.                   BEGIN.112
                                                                                              BEGIN.113
255                     TMCMEM  *                                                             BEGIN.114
```

```
255   020000  024456                    R!T1 = R!%MYID                  .Start with this IOP                BEGIN.115
257   010035  024455                    R!IO = CH$CH                    .High speed channel                 BEGIN.116
261   010000  024462                    R!TM = TM$CMEM                  .Target Central Memory              BEGIN.117
263   070004                            P = CMSRCH                      .Start search                       BEGIN.118
264    <macro>                          $UNTIL (R!T1 = R!%MYID)         .Search all IOP's                   BEGIN.119
267                             CMSRCH  *                                                                   BEGIN.120
267   077000 /001644                      R = CHSCAN                    .Search for channel                 BEGIN.121
271    <macro>                            $IF (R!T2 = 0)                .If found                           BEGIN.122
                                                                                        .>>>>> 0                $IF      1
274   010006  024460                        R!T3 = TM@LT                .Get space for TM table             BEGIN.123
276   014000 /052115  024423                R!%W1 = 'TM'R               .ASCII header                       BEGIN.124
301   014000 /040103  024424                R!%W2 = '@C'R                                                   BEGIN.125
304   014000 /042516  024425                R!%W3 = 'EN'R                                                   BEGIN.126
307   077000 /002512                        R =  GETBUF                                                     BEGIN.127
311   020463  012000  024410                (R!TBL + TM$CMEM) = R!T4    .Save pointer to TM table           BEGIN.128
      020461  034410
316    <macro>                              STORE TM$CMEM,TM@TYP,R!T4 .Memory type                          BEGIN.129
320    <macro>                              PUT R!T1,TM@IOP,R!T4      .Processor number                     BEGIN.130
325    <macro>                              STORE CRAY@SIZ,TM@LMO,R!T4 .Limit address upper                 BEGIN.131
332   070013                                P = TMSSD                   .Skip to SSD                        BEGIN.132
333    <macro>                            $ENDIF                                                           BEGIN.133
                                                                                        .<<<<< 0                $ENDIF   1
333   026456                             R!T1 = R!T1 + 1               .Next IOP                           BEGIN.134
334   041000  013003  100004             R!I1 = MIOP, A > XIOP         .Wrap                               BEGIN.135
      102003  010000  024456
342    <macro>                          $ENDTIL                                                            BEGIN.136
343    <macro>                          $PUNTIF CODE=PT$NOHSP         .High speed must be configured      BEGIN.137
                                                                                                            BEGIN.138
                                        *  Set up control table for SSD.  Search channel                   BEGIN.139
                                        *  entries for high speed to SSD.                                  BEGIN.140
                                                                                                            BEGIN.141
345                             TMSSD   *                                                                   BEGIN.142
345   020000  024456                    R!T1 = R!%MYID                  .Start with this IOP                BEGIN.143
347   010035  024455                    R!IO = CH$CH                    .High speed channel                 BEGIN.144
351   010002  024462                    R!TM = TM$SSD                   .Target SSD                         BEGIN.145
353   070004                            P = SSDSRCH                     .Start search                       BEGIN.146
354    <macro>                          $UNTIL (R!T1 = R!%MYID)         .Search all IOP's                   BEGIN.147
357                             SSDSRCH *                                                                   BEGIN.148
357   077000 /001644                      R = CHSCAN                    .Search for channel                 BEGIN.149
361    <macro>                            $IF (R!T2 = 0)                .If found                           BEGIN.150
                                                                                        .>>>>> 0                $IF      1
364   010006  024460                        R!T3 = TM@LT                .Get space for TM table             BEGIN.151
```

```
366   014000 /052115  024423          R!%W1 = 'TM'R              .ASCII header                    BEGIN.152
371   014000 /040123  024424          R!%W2 = '@S'R                                               BEGIN.153
374   014000 /051504  024425          R!%W3 = 'SD'R                                               BEGIN.154
377   077000 /002512                  R =  GETBUF                                                 BEGIN.155
401   020463  012002  024410          (R!TBL + TM$SSD) = R!T4    .Save pointer to TM table        BEGIN.156
      020461  034410
406    <macro>                        STORE TM$SSD,TM@TYP,R!T4   .Memory type                     BEGIN.157
410    <macro>                        PUT R!T1,TM@IOP,R!T4       .Processor number                BEGIN.158
415    <macro>                        STORE SSD@SIZ,TM@LMO,R!T4  .Limit address upper             BEGIN.159
423   070011                          P = TMEND                  .End of TM setup                 BEGIN.160
424    <macro>                      $ENDIF                                                         BEGIN.161
                                                                                .<<<<< 0              $ENDIF    1
424   026456                        R!T1 = R!T1 + 1            .Next IOP                        BEGIN.162
425   041000  013003  100004        R!T1 = MIOP, A > XIOP      .Wrap                            BEGIN.163
      102003  010000  024456
433    <macro>                    $ENDTIL                                                        BEGIN.164
434                        TMEND   *                                                              BEGIN.165
                                                                                                  BEGIN.166
                                   *                                                              BEGIN.167
                                   * CREATE DEMON ACTIVITIES FOR THIS IOP                        BEGIN.168
                                   *                                                              BEGIN.169
                                                                                                  BEGIN.170
                                   * DEMONS FOR ALL IOPS                                         BEGIN.171
                                                                                                  BEGIN.172
434   014000 /000000  024455        R!T0 = O$ACOM              .ACOM                            BEGIN.173
437   010000  024456                R!T1 = 0                   .PRIORITY                        BEGIN.174
441   014000 /000000  024457        R!T2 = D$ACOM              .DEMON POINTER                   BEGIN.175
444   077000 /001724                R = DEMON                  .CREATE                          BEGIN.176
446   014000 /000000  024455        R!T0 = O$AMSG              .AMSG                            BEGIN.177
451   010000  024456                R!T1 = 0                   .PRIORITY                        BEGIN.178
453   014000 /000000  024457        R!T2 = D$AMSG              .POINTER                         BEGIN.179
456   077000 /001724                R = DEMON                  .CREATE                          BEGIN.180
460   014000 /000000  024455        R!T0 = O$CLOCK             .CLOCK DEMON                     BEGIN.181
463   010007  024456                R!T1 = 7                   .PRIORITY                        BEGIN.182
465   014000 /000000  024457        R!T2 = D$CLOCK             .POINTER                         BEGIN.183
470   077000 /001724                R = DEMON                  .                                BEGIN.184
472   014000 /000000  024455        R!T0 = O$CRTDEM            .CRTDEM                          BEGIN.185
475   010007  024456                R!T1 = 7                   .PRIORITY                        BEGIN.186
477   014000 /000000  024457        R!T2 = D$CRTDEM            .POINTER                         BEGIN.187
502   077000 /001724                R = DEMON                  .                                BEGIN.188
                                                                                                  BEGIN.189
                                   * MIOP DEMONS                                                  BEGIN.190
```

```
                                                                                                BEGIN.191
504    <macro>                          $IF (R!%MYID = MIOP)          .                         BEGIN.192
                                                                                   .>>>>> 0          $IF      1
507    014000 /000000  024455             R!T0 = O$CDEM               .CDEM                     BEGIN.193
512    010000  024456                     R!T1 = 0                    .PRIORITY                 BEGIN.194
514    014000 /000000  024457             R!T2 = D$CDEM               .POINTER                  BEGIN.195
517    077000 /001724                     R = DEMON                   .CREATE                   BEGIN.196
                                                                                                BEGIN.197
521    014000 /000000  024455             R!T0 = O$ADEM                                         BEGIN.198
524    010000  024456                     R!T1 = 0                                              BEGIN.199
526    014000 /000000  024457             R!T2 = D$ADEM                                         BEGIN.200
531    077000 /001724                     R = DEMON                   .Create ADEM demon        BEGIN.201
                                                                                                BEGIN.202
                                          * BLOCK MUX                                           BEGIN.203
                                                                                                BEGIN.204
533    <macro>                            $IF (I@BMXSYS # 0)          .IF DEFINED               BEGIN.205
                                                                                   .>>>>> 1          $IF      1
536    014000 /000000  024455               R!T0 = O$BCOM             .CREATE BCOM              BEGIN.206
541    010000  024456                       R!T1 = 0                  .PRIORITY                 BEGIN.207
543    014000 /000000  024457               R!T2 = D$BCOM             .DEMON POINTER            BEGIN.208
546    077000 /001724                       R = DEMON                 .                         BEGIN.209
550    <macro>                            $ENDIF                                                BEGIN.210
                                                                                   .<<<<< 1          $ENDIF   1
                                                                                                BEGIN.211
                                          * DCU5                                                BEGIN.212
                                                                                                BEGIN.213
                                          * Create ICOM if any DCU5 disks in the entire system. BEGIN.214
                                          * Create other DCU5 demons as needed.                 BEGIN.215
                                                                                                BEGIN.216
550    010001  024456                     R!T1 = BIOP                 .Start with BIOP          BEGIN.217
552    010001  024460                     R!T3 = 1                    .Force pass               BEGIN.218
554    <macro>                            $UNTIL (R!T1 > XIOP),OR,(R!T3 = 0) .Until some found  BEGIN.219
563    077000 /002274                       R = DCU5                  .Search for DCU5 disks    BEGIN.220
565    <macro>                              $IF (R!T3 = 0)            .If some found            BEGIN.221
                                                                                   .>>>>> 1          $IF      1
570    077000 /002074                         R = ICOMSET             .Initialize ICOM          BEGIN.222
572    <macro>                              $ENDIF                                              BEGIN.223
                                                                                   .<<<<< 1          $ENDIF   1
572    026456                               R!T1 = R!T1 + 1           .Next IOP                 BEGIN.224
573    <macro>                            $ENDUNTIL                                             BEGIN.225
574    077000 /002134                     R = INDCU5                  .Create other DCU5 demons BEGIN.226
                                                                                                BEGIN.227
```

```
                                  * BIOP DEMONS                                                   BEGIN.228
                                                                                                  BEGIN.229
576    <macro>                    $ELSEIF (R!%MYID = BIOP)      .                                  BEGIN.230
                                                                                                  BEGIN.231
                                    * BLOCK MUX                                                   BEGIN.232
                                                                                                  BEGIN.233
604    <macro>                      $IF (I@BMXSYS # 0)          .IF EFINED                       BEGIN.234
                                                                                .>>>>> 1               $IF      1
607    014000 /000000  024455         R!T0 = O$BCOM             .CREATE BCOM                     BEGIN.235
612    010000  024456                 R!T1 = 0                  .PRIORITY 0                      BEGIN.236
614    014000 /000000  024457         R!T2 = D$BCOM             .DEMON POINTER                   BEGIN.237
617    077000 /001724                 R = DEMON                 .                                BEGIN.238
621    <macro>      .               $ENDIF                                                        BEGIN.239
                                                                                .<<<<< 1               $ENDIF   1
                                                                                                  BEGIN.240
                                    * DD-19/29                                                    BEGIN.241
                                                                                                  BEGIN.242
621    010001  024455               R!T0 = CH$D1                .DD-19                           BEGIN.243
623    010001  024456               R!T1 = BIOP                 .BIOP                            BEGIN.244
625    077000 /001644               R = CHSCAN                  .SCAN                            BEGIN.245
627    <macro>                      $IF (R!T2 # 0)              .IF NONE                         BEGIN.246
                                                                                .>>>>> 1               $IF      1
632    010002  024455                 R!T0 = CH$D2              .CHECK FOR DD-29                 BEGIN.247
634    077000 /001644                 R = CHSCAN                .CHECK                           BEGIN.248
636    <macro>                      $ENDIF                                                        BEGIN.249
                                                                                .<<<<< 1               $ENDIF   1
636    <macro>                      $IF (R!T2 # 0)              .If none found yet               BEGIN.250
                                                                                .>>>>> 1               $IF      1
641    010101  024455                 R!T0 = CH$D1S             .Striped DD19                    BEGIN.251
643    077000 /001644                 R = CHSCAN                .Search                          BEGIN.252
645    <macro>                        $IF (R!T2 # 0)            .If none found                   BEGIN.253
                                                                                .>>>>> 2               $IF      1
650    010102  024455                   R!T0 = CH$D2S           .Striped DD29                    BEGIN.254
652    077000 /001644                   R = CHSCAN              .Search                          BEGIN.255
654    <macro>                        $ENDIF                                                        BEGIN.256
                                                                                .<<<<< 2               $ENDIF   1
654    <macro>                      $ENDIF                                                        BEGIN.257
                                                                                .<<<<< 1               $ENDIF   1
654    <macro>                      $IF (R!T2 = 0)              .IF DISK DEMON NEEDED            BEGIN.258
                                                                                .>>>>> 1               $IF      1
657    014000 /000000  024455         R!T0 = O$DISK             .OVERLAY                         BEGIN.259
662    010000  024456                 R!T1 = 0                  .PRIORITY                        BEGIN.260
```

```
664   014000 /000000  024457      R!T2 = D$DISK          .POINTER                          BEGIN.261
667   077000 /001724              R = DEMON              .CREATE                           BEGIN.262
671    <macro>                    CALL INDD29,(R!MA)     .INITIALIZE DD-19/29              BEGIN.263
705   077000 /001743              R = DKBUF              .Allocate DKERR buffer            BEGIN.264
707    <macro>                  $ENDIF                                                     BEGIN.265
                                                                        .<<<<< 1                $ENDIF   1
                                                                                            BEGIN.266
                                 * DCU5                                                     BEGIN.267
                                                                                            BEGIN.268
707   077000 /002134              R = INDCU5             .Initialize DCU5 disks            BEGIN.269
                                                                                            BEGIN.270
                                * DIOP DEMONS                                               BEGIN.271
                                                                                            BEGIN.272
711    <macro>                  $ELSEIF (R!%MYID = DIOP)                                   BEGIN.273
                                                                                            BEGIN.274
                                 * BLOCK MUX AND HIGH-SPEED CHANNEL                         BEGIN.275
                                                                                            BEGIN.276
720    <macro>                  $IF (I@BMXSYS # 0)     .BMX DEFINED                       BEGIN.277
                                                                        .>>>>> 1                $IF      1
723   010035  024455              R!T0 = CH$CH           .CHECK FOR HIGH-SPEED CHANNEL     BEGIN.278
725   010002  024456              R!T1 = DIOP            .IN DIOP                          BEGIN.279
727   077000 /001644              R = CHSCAN             .CHECK                            BEGIN.280
731    <macro>                    $IF (R!T2 = 0)         .IF SO                            BEGIN.281
                                                                        .>>>>> 2                $IF      1
734   014000 /000000  024455        R!T0 = O$BCOM        .CREATE BCOM                      BEGIN.282
737   010000  024456                R!T1 = 0             .PRIORITY 0                       BEGIN.283
741   014000 /000000  024457        R!T2 = D$BCOM        .DEMON POINTER                    BEGIN.284
744   077000 /001724                R = DEMON            .                                 BEGIN.285
746    <macro>                    $ENDIF                                                     BEGIN.286
                                                                        .<<<<< 2                $ENDIF   1
746    <macro>                  $ENDIF                                                     BEGIN.287
                                                                        .<<<<< 1                $ENDIF   1
                                                                                            BEGIN.288
                                 * DD-19/29                                                 BEGIN.289
                                                                                            BEGIN.290
746   010001  024455              R!T0 = CH$D1           .DD-19                            BEGIN.291
750   010002  024456              R!T1 = DIOP            .IN DIOP                          BEGIN.292
752   077000 /001644              R = CHSCAN             .CHECK                            BEGIN.293
754    <macro>                    $IF (R!T2 # 0)         .IF NONE                          BEGIN.294
                                                                        .>>>>> 1                $IF      1
757   010002  024455              R!T0 = CH$D2           .CHECK FOR DD-29                  BEGIN.295
761   077000 /001644              R = CHSCAN             .                                 BEGIN.296
```

```
  763    <macro>                    $ENDIF                                                    BEGIN.297
                                                                        .<<<<< 1                $ENDIF   1
  763    <macro>                    $IF (R!T2 # 0)          .If none found yet                BEGIN.298
                                                                        .>>>>> 1                $IF      1
  766   010101  024455                R!T0 = CH$D1S           .Striped DD19                     BEGIN.299
  770   077000 /001644                R = CHSCAN              .Search                           BEGIN.300
  772    <macro>                      $IF (R!T2 # 0)          .If none found                    BEGIN.301
                                                                        .>>>>> 2                $IF      1
  775   010102  024455                  R!T0 = CH$D2S         .Striped DD29                     BEGIN.302
  777   077000 /001644                  R = CHSCAN            .Search                           BEGIN.303
 1001    <macro>                      $ENDIF                                                    BEGIN.304
                                                                        .<<<<< 2                $ENDIF   1
 1001    <macro>                    $ENDIF                                                      BEGIN.305
                                                                        .<<<<< 1                $ENDIF   1
 1001    <macro>                    $IF (R!T2 = 0)          .IF ANY                           BEGIN.306
                                                                        .>>>>> 1                $IF      1
 1006   014000 /000000  024455        R!T0 = O$DISK           .DISK DEMON                       BEGIN.307
 1011   010000  024456                R!T1 = 0                .PRIORITY                         BEGIN.308
 1013   014000 /000000  024457        R!T2 = D$DISK           .POINTER                          BEGIN.309
 1016   072706                        R = DEMON               .CREATE                           BEGIN.310
 1017    <macro>                     CALL INDD29,(R!MA)      .INITIALIZE DD-19/29              BEGIN.311
 1033   072710                        R = DKBUF               .Allocate DKERR buffer            BEGIN.312
 1034    <macro>                    $ENDIF                                                      BEGIN.313
                                                                        .<<<<< 1                $ENDIF   1
                                                                                                BEGIN.314
                                    * DCU5                                                      BEGIN.315
                                                                                                BEGIN.316
 1034   077000 /002134               R = INDCU5              .Initialize DCU5 disks            BEGIN.317
                                                                                                BEGIN.318
                                  * XIOP DEMONS                                                 BEGIN.319
                                                                                                BEGIN.320
 1036    <macro>                  $ELSEIF (R!%MYID = XIOP)                                      BEGIN.321
                                                                                                BEGIN.322
                                    * BLOCK MUX                                                 BEGIN.323
                                                                                                BEGIN.324
 1043    <macro>                    $IF (I@BMXSYS # 0)      .IF DEFINED                       BEGIN.325
                                                                        .>>>>> 1                $IF      1
 1046   014000 /000000  024455        R!T0 = O$BCOM           .CREATE BCOM                      BEGIN.326
 1051   010002  024456                R!T1 = 2                .PRIORITY                         BEGIN.327
 1053   014000 /000000  024457        R!T2 = D$BCOM           .DEMON POINTER                    BEGIN.328
 1056   072646                        R = DEMON               .                                 BEGIN.329
 1057    <macro>                    $ENDIF                                                      BEGIN.330
```

```
                                                                                .<<<<< 1          $END.    1
                                                                                                BEGIN.331
                                  * DCU5                                                        BEGIN.332
                                                                                                BEGIN.333
1057   077000 /002134             R = INDCU5                 .Initialize DCU5 disks             BEGIN.334
                                                                                                BEGIN.335
1061    <macro>                  $ENDIF                                                        BEGIN.336
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                BEGIN.337
                          *       Initialize A-A channels                                       BEGIN.338
                                                                                                BEGIN.339
1061   014000 /000000  024455    R!T0 = MATX                                                 R16498AB.1
1064   012004  024456            R!T1 = A + 4                                                R16498AB.2
                                                                                              R16498AB.3
                          *       Synchronize all IOPs with MIOP, drain A-A input channels,   R16498AB.4
                          *       and initialize all A-A channels.                            R16498AB.5
                                                                                              R16498AB.6
1066    <macro>                  $IF (R!%MYID = MIOP)       .If MIOP                         R16498AB.7
                                                                                .>>>>> 0          $IF      1
1071    <macro>                    $UNTIL (R!T0 = R!T1)     .Loop for all channels           R16498AB.8
1074   030455  054000                B = (R!T0)                                                R16498AB.9
1076    <macro>                      $IF (A # 0)            .If IOP exists                   R16498AB.10
                                                                                .>>>>> 1          $IF      1
1100   056000                          B = B + 1            .Output channel                  R16498AB.11
1101   160000                          IOB: 0               .Clear it                        R16498AB.12
1102   042000  104001                  WAIT, IOB # DN       .Wait until IOP drains channel   R16498AB.13
1104   057000                          B = B - 1                                             R16498AB.14
1105   170000                          IOB: 10              .Drain input channel             R16498AB.15
1106   042000  104001                  WAIT, IOB # DN       .Wait for IOP to respond         R16498AB.16
1110   170000                          IOB: 10              .Drain it again                  R16498AB.17
1111   167000                          IOB: 7               .Enable input channel            R16498AB.18
1112   056000                          B = B + 1                                             R16498AB.19
1113   167000                          IOB: 7               .Enable output channel           R16498AB.20
1114    <macro>                      $ENDIF                                                  R16498AB.21
                                                                                .<<<<< 1          $ENDIF   1
1114   026455                         R!T0 = R!T0 + 1      .Next channel pair               R16498AB.22
1115    <macro>                    $ENDTIL                                                   R16498AB.23
1116    <macro>                  $ELSE                      .Not MIOP                        R16498AB.24
1117    <macro>                    $UNTIL (AOA = DN)        .Until MIOP is in sync           R16498AB.25
1121   150006                        AIA: 10                .Drain input channel             R16498AB.26
1122    <macro>                    $ENDTIL                                                   R16498AB.27
1123   140007                      AOA: 0                                                    R16498AB.28
```

```
1124    <macro>                          $UNTIL (R!TO = R!T1)        .Loop for all channels          R16498AB.29
1127   030455  054000                      B = (R!TO)                                                  R16498AB.30
1131    <macro>                            $IF (A # 0)                 .If IOP exists                  R16498AB.31
                                                                                        .>>>>> 1          $IF      1
1133   170000                                IOB: 10                   .Drain input channel            R16498AB.32
1134   167000                                IOB: 7                    .Enable input channel           R16498AB.33
1135   056000                                B = B + 1                                                 R16498AB.34
1136   167000                                IOB: 7                    .Enable output channel          R16498AB.35
1137    <macro>                            $ENDIF                                                      R16498AB.36
                                                                                        .<<<<< 1          $ENDIF   1
1137   026455                              R!TO = R!TO + 1             .Next channel pair              R16498AB.37
1140    <macro>                          $ENDTIL                                                     R16498AB.38
1141   154007                            AOA: 14                       .Kick MIOP                      R16498AB.39
1142    <macro>                        $ENDIF                                                         R16498AB.40
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        BEGIN.352
                                       * OUTPUT KERNEL BANNER                                           BEGIN.353
                                                                                                        BEGIN.354
1142    <macro>                        CALL CALL,(O'5015)                                               BEGIN.356
                                                                                                        BEGIN.357
                                 *     Send DKERR buffer address to MIOP                                BEGIN.358
                                                                                                        BEGIN.359
1157    <macro>                        $IF (R!MU # 0),OR,(R!ML # 0) .If disks on this IOP               BEGIN.360
                                                                                        .>>>>> 0          $IF      1
1164    <macro>                          OUTCALL MIOP,DKERR1,(R!%MYID,R!MU,R!ML)                        BEGIN.361
1212    <macro>                        $ENDIF                                                           BEGIN.362
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        BEGIN.363
                                       * PROCESS COMMANDS FROM AMAP                                     BEGIN.364
                                                                                                        BEGIN.365
1212   020432  022000  012013          T1 = MA + R!%MYID + O'13    .Find IOP table                     BEGIN.366
       024456
1216   030456  022432  012011          CA = (T1) + MA + O'11       .Command buffer                     BEGIN.367
       024430
1222   020430  012001  024431          CL = CA + 1                 .End of commands                    BEGIN.368
1225   030430  024430                  CA = (CA)                                                       BEGIN.369
1227   020430  102047                  P = BEG1, CA = 0            .If commands not available          BEGIN.370
1231   020430  022432  024430          CA = CA + MA                .Message location                   BEGIN.371
1234   030431  024431                  CL = (CL)                                                       BEGIN.372
1236   020431  102040                  P = BEG1, CL = 0            .If no end specified                BEGIN.373
1240   020431  022432  024431          CL = CL + MA                                                    BEGIN.374
1243    <macro>                        $UNTIL    (CA >= CL)                                            BEGIN.375
```

```
1246    <macro>                          MSG     R!%B+LFCR                                          BEGIN.376
1261    020430  012001  024455           10 = CA + 1                .Display command                BEGIN.377
1264    <macro>                          MSG     10                                                 BEGIN.378
1275    072112                           R = LCM                                                    BEGIN.379
1276    <macro>                        $ENDTIL                                                      BEGIN.380
                                                                                                    BEGIN.381
1277                            BEG1     *                                                          BEGIN.382
1277    <macro>                          $IF       (MA # 0)         .If buffer assigned             BEGIN.383
                                                                                   .>>>>> 0              $IF      1
1302    <macro>                          BRET    MA                 .Return buffer                  BEGIN.384
1312    <macro>                        $ENDIF                                                       BEGIN.385
                                                                                   .<<<<< 0              $ENDIF   1
                                                                                                    BEGIN.386
1312    <macro>                        CREATE SCRUB,,PRI=17                                         BEGIN.387
1330    <macro>                        $PUNTIF (A # 0)            .IF CAN'T CREATE                  BEGIN.388
                                                                                                    BEGIN.389
                                        *       IOP Dependent operations.                           BEGIN.390
                                                                                                    BEGIN.391
1332    <macro>                        $GOTO     (MIOPL,BIOPL,DIOPL,XIOPL),R!%MYID                 BEGIN.392
                                                                                                    BEGIN.393
1353                            MIOPL    *                                                          BEGIN.394
1353    <macro>                        CALL TIME,(0)               .GET TIME AND DATE              BEGIN.395
1367    070003                         P = BEGEND                                                   BEGIN.396
                                                                                                    BEGIN.397
1370                            BIOPL    *                                                          BEGIN.398
1370    070002                         P = BEGEND                                                   BEGIN.399
                                                                                                    BEGIN.400
1371                            DIOPL    *                                                          BEGIN.401
1371    070001                         P = BEGEND                                                   BEGIN.402
                                                                                                    BEGIN.403
1372                            XIOPL    *                                                          BEGIN.404
                                        *       P = BEGEND                                          BEGIN.405
                                                                                                    BEGIN.406
1372                            BEGEND   *                                                          BEGIN.407
1372    014000 /000000  024423         R!%W1 = EMEMSIZ                                              BEGIN.408
1375    013001  024424                 R!%W2 = A - 1                                                BEGIN.409
1377    030424  034423                 (R!%W1) = (R!%W2)          .Set available memory size        BEGIN.410
1401    <macro>                        RETURN                                                       BEGIN.411
        <edit>                         SUBTITLE  '"$OSUB" Subroutines.'                             BEGIN.412
```

```
                                **      LCM - Load command and excute it.                                   BEGIN.414
                                *       Entry     CA = Address of command.                                  BEGIN.415
                                *       Exit      CA = Address of next command.                             BEGIN.416
                                                                                                          BEGIN.417
                                                                                                          BEGIN.418
1407                            LCM     *                                                                 BEGIN.419
1407   010433  054000                   B = [PA]                                                          BEGIN.420
1411   010000  024455                   T0 = 0                                                            BEGIN.421
1413   010001  024456                   T1 = 1                          .Show not at end yet              BEGIN.422
1415   030430  024457                   T2 = (CA)                       .Display/Interactive command type BEGIN.423
1417   026430                           CA = CA + 1                     .Message start                    BEGIN.424
1420    <macro>                         $UNTIL    (T1 = 0),OR,(T0 = PARL)  .Load up command               BEGIN.425
1425   030430  064000                     (B) = (CA)                                                      BEGIN.426
1427   060000  024456                     T1 = (B)                         .Save for end of line check    BEGIN.427
1431   026430                             CA = CA + 1                                                     BEGIN.428
1432   056000                             B = B + 1                                                       BEGIN.429
1433   026455                             T0 = T0 + 1                                                     BEGIN.430
1434    <macro>                         $ENDTIL                                                           BEGIN.431
                                                                                                          BEGIN.432
1435   010000  064000                   (B) = 0                         .Ensure end of line               BEGIN.433
1437   057000                           B = B - 1                                                         BEGIN.434
1440   057000                           B = B - 1                                                         BEGIN.435
1441    <macro>                         $IF       (0 = (B) & O'377)     .Set end of command               BEGIN.436
                                                                                        .>>>>> 0            $IF      1
1450   010015  065000                     (B) = (B) + O'15              .Enter CR                         BEGIN.437
1452    <macro>                         $ELSE                                                             BEGIN.438
1453   056000                             B = B + 1                                                       BEGIN.439
1454   014000 /006400  064000             (B) = O'6400                  .Enter CR                         BEGIN.440
1457   026430                             CA = CA + 1                                                     BEGIN.441
1460    <macro>                         $ENDIF                                                            BEGIN.442
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                          BEGIN.443
1460   056000                           B = B + 1                                                         BEGIN.444
1461   010000  064000                   (B) = 0                                                           BEGIN.445
1463    <macro>                         $IF       (T2 = 0)                                                BEGIN.446
                                                                                        .>>>>> 0            $IF      1
1466    <macro>                           CALL    CALL,(PA,PB,PC,PD,PE,PF,PG,PH,PI,PJ,PK,PL,PM,PN,PO,PP   BEGIN.447
                                ,,PQ,PR)                                                                  BEGIN.448
1544    <macro>                           GIVEUP                                                          BEGIN.449
1553    <macro>                         $ELSEIF   (R!CRT # 0)           .If CRT available                 BEGIN.450
1557    <macro>                           CREATE  CALL,(PA,PB,PC,PD,PE,PF,PG,PH,PI,PJ,PK,PL,PM,PN,PO,PP   BEGIN.451
                                ,,PQ,PR)                                                                  BEGIN.452
```

```
1641    <macro>                          $PUNTIF (A # 0)              .IF ERROR                                BEGIN.453
1643    <macro>                        $ENDIF                                                                     BEGIN.454
                                                                                              .<<<<< 0                $ENDIF   1
                                                                                                                     BEGIN.455
1643    001000                         EXIT                                                                        BEGIN.456
                                                                                                                     BEGIN.457
                                       *                                                                           BEGIN.458
                                       * SCAN AMAP FOR A SPECIFIED CHANNEL TYPE                                   BEGIN.459
                                       * AND IOP.                                                                  BEGIN.460
                                       *                                                                           BEGIN.461
                                       *  Entry: R!T0 = Channel type                                               BEGIN.462
                                       *         R!T1 = IOP number                                                 BEGIN.463
                                       *         R!TM = Target memory if R!T0=CH$CH                                BEGIN.464
                                       *                                                                           BEGIN.465
                                       *  Exit:  R!T2 = 0 if match found                                           BEGIN.466
                                       *                                                                           BEGIN.467
                                                                                                                     BEGIN.468
1644                           CHSCAN  *                                                                           BEGIN.469
1644    020432  012006  012005         R!%W1 = R!MA + OV@ENI@P + 5 + R!T1  .POINTER TO INFO                        BEGIN.470
        022456  024423
1651    030423  022432  024423         R!%W1 = (R!%W1) + R!MA       .INFO AREA FOR IOP                             BEGIN.471
1654    023432  102044                 P = NOMATCH, A = R!MA        .IF IOP NOT CONFIGURED                         BEGIN.472
1656    010006  025423                 R!%W1 = R!%W1 + 6            .POINTER TO CHANNEL DESCRIPTORS                BEGIN.473
1660    030423  022432  024423         R!%W1 = (R!%W1) + R!MA       .                                              BEGIN.474
1663    030423  024424                 R!%W2 = (R!%W1)              .LAST CHANNEL NUMBER                           BEGIN.475
1665    010006  024425                 R!%W3 = 6                    .CURRENT CHANNEL                               BEGIN.476
1667    010000  024457                 R!T2 = 0                     .CLEAR ERROR FLAG                              BEGIN.477
1671    <macro>                        $UNTIL (R!%W3 = R!%W2)                                                      BEGIN.478
1674    026423                           R!%W1 = R!%W1 + 1          .NEXT DESCRIPTOR                               BEGIN.479
1675    030423  011777  024426           R!%W4 = (R!%W1) & O'777    .TYPE                                          BEGIN.480
1700    <macro>                          $IF (A = R!T0)             .If a match found                              BEGIN.481
                                                                                              .>>>>> 0                $IF      1
1703    <macro>                            $IF (R!T0 = CH$CH)       .If high speed channel                         BEGIN.482
                                                                                              .>>>>> 1                $IF      1
1707    030423  004011  011037               A = (R!%W1) > D'9 & O'37 .Target memory                                BEGIN.483
1712    023462  103002  001000               EXIT, A = R!TM           .If a match                                   BEGIN.484
1715    <macro>                            $ELSE                                                                    BEGIN.485
1716    001000                               EXIT                                                                   BEGIN.486
1717    <macro>                            $ENDIF                                                                   BEGIN.487
                                                                                              .<<<<< 1                $ENDIF   1
1717    <macro>                          $ENDIF                                                                     BEGIN.488
                                                                                              .<<<<< 0                $ENDIF   1
```

```
1717   026425                          R!%W3 = R!%W3 + 1          .CURRENT CHANNEL          BEGIN.489
1720    <macro>                       $ENDTIL                                              BEGIN.490
1721                          NOMATCH  *                                                    BEGIN.491
1721   010001  024457                 R!I2 = 1                   .INDICATE NO MATCH        BEGIN.492
1723   001000                         EXIT                                                 BEGIN.493
                                                                                            BEGIN.494
                                      *                                                     BEGIN.495
                                      * CREATE DEMON ACTIVITY                               BEGIN.496
                                      *                                                     BEGIN.497
                                                                                            BEGIN.498
1724                          DEMON   *                                                     BEGIN.499
1724    <macro>                       CREATE R!T0,,PRI=R!T1,DPTR=R!T2,TYPE=NUMBER          BEGIN.500
1740    <macro>                       $PUNTIF (A # 0)            .IF FAILURE               BEGIN.501
1742   001000                         EXIT                                                 BEGIN.502
                                                                                            BEGIN.503
                              ***                                                           BEGIN.504
                              *       DKBUF:  Allocate DKERR buffer if not already allocated. BEGIN.505
                              ***                                                           BEGIN.506
                                                                                            BEGIN.507
1743                          DKBUF   *                                                     BEGIN.508
1743    <macro>                       $IF (R!MU = 0),AND,(R!ML = 0) .If buffer not allocated BEGIN.509
                                                                                .>>>>> 0        $IF      1
1753    <macro>                         MGET R!MU,R!ML           .Get buffer               BEGIN.510
2001    <macro>                         $PUNTIF (A # 0),CODE=PT$NOMOS .If no MOS available BEGIN.511
2003   014000 /000000  024423           R!%W1 = LBFW                                       BEGIN.512
2006   020465  034423                   (R!%W1) = R!MU           .Upper address            BEGIN.513
2010   026423                           R!%W1 = R!%W1 + 1                                  BEGIN.514
2011   020466  034423                   (R!%W1) = R!ML           .Lower address            BEGIN.515
2013    <macro>                         MOSW R!MU,R!ML,LBPT,1    .Write IN and OUT         BEGIN.516
2034    <macro>                       $ENDIF                                               BEGIN.517
                                                                                .<<<<< 0        $ENDIF   1
2034   001000                         EXIT                                                 BEGIN.518
                                                                                            BEGIN.519
                              ***                                                           BEGIN.520
                              *       TRANSET:  Set up TRANSFR demon and tables.            BEGIN.521
                              ***                                                           BEGIN.522
                                                                                            BEGIN.523
2035                          TRANSET *                                                     BEGIN.524
2035   020000  102002  001000         EXIT, R!%TRANQ # 0         .If already initialized   BEGIN.525
2040   014000 /000000  024455         R!I0 = O$TRANSF            .TRANSFR demon            BEGIN.526
2043   010000  024456                 R!I1 = 0                   .Priority                 BEGIN.527
2045   014000 /000000  024457         R!I2 = D$TRANSF            .Demon pointer            BEGIN.528
```

```
2050   073124                          R = DEMON                      .Initialize                    BEGIN.529
2051   010004  024460                  R!T3 = TRA@LT                  .Table length                  BEGIN.530
2053   014000 /052122  024423          R!%W1 = 'TR'                   .ASCII header                  BEGIN.531
2056   014000 /040516  024424          R!%W2 = 'AN'                                                  BEGIN.532
2061   014000 /050440  024425          R!%W3 = 'Q '                                                  BEGIN.533
2064   072426                          R = GETBUF                     .Get a buffer                  BEGIN.534
2065   020461  024000                  R!%TRANQ = R!T4                .Save address                  BEGIN.535
2067    <macro>                        PUT R!T1,TRA@PR,R!%TRANQ       .Store priority                BEGIN.536
                                                                                                     BEGIN.537
                                ***                                                                  BEGIN.538
                                *        ICOMSET: Set up ICOM demon                                  BEGIN.539
                                ***                                                                  BEGIN.540
                                                                                                     BEGIN.541
2074                            ICOMSET  *                                                           BEGIN.542
2074   020464  102002  001000          EXIT, R!ICOM # 0               .If already initialized        BEGIN.543
2077   014000 /000000  024455          R!T0 = O$ICOM                  .Overlay                       BEGIN.544
2102   010000  024456                  R!T1 = 0                       .Priority                      BEGIN.545
2104   014000 /000000  024457          R!T2 = D$ICOM                  .Demon pointer                 BEGIN.546
2107   073163                          R = DEMON                      .Create it                     BEGIN.547
2110   010100  024460                  R!T3 = D4QSZI                  .Size of queue                 BEGIN.548
2112   014000 /046511  024423          R!%W1 = 'MI'R                  .ASCII header                  BEGIN.549
2115   014000 /040521  024424          R!%W2 = 'AQ'R                                                 BEGIN.550
2120   014000 /020040  024425          R!%W3 = '  'R                                                 BEGIN.551
2123   072342                          R = ABLD                       .Build ICOM msg queue          BEGIN.552
2124   014000 /000000  024410          (MIAQ) = R!T4                  .Save pointer                  BEGIN.553
       020461  034410
2131   010001  024464                  R!ICOM = 1                     .Indicate created              BEGIN.554
2133   001000                          EXIT                                                          BEGIN.555
                                                                                                     BEGIN.556
                                ***                                                                  BEGIN.557
                                *        INDCU5:   Initialize support in the current IOP for DCU5 disks.  BEGIN.558
                                ***                                                                  BEGIN.559
                                                                                                     BEGIN.560
2134                            INDCU5   *                                                           BEGIN.561
2134   020000  024456                  R!T1 = R!%MYID                 .This IOP                      BEGIN.562
2136   072136                          R = DCU5                       .Search for DCU5 disks         BEGIN.563
2137    <macro>                        $IF (R!T3 = 0)                 .If any                        BEGIN.564
                                                                                  .>>>>> 0     $IF    1
2142   073046                            R = ICOMSET                  .Initialize ICOM               BEGIN.565
2143   014000 /000000  024455            R!T0 = O$D4DEM               .Initialize demon              BEGIN.566
2146   010000  024456                    R!T1 = 0                     .Priority                      BEGIN.567
2150   014000 /000000  024457            R!T2 = D$D4DEM               .Demon pointer                 BEGIN.568
```

```
2153   073227                          R = DEMON                  .Initialize                          BEGIN.569
2154   073211                          R = DKBUF                  .Allocate DKERR buffer               BEGIN.570
2155    <macro>                       $ENDIF                                                          BEGIN.571
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                       BEGIN.572
                                       * If this IOP has a high speed path to Central memory, set up   BEGIN.573
                                       * demons to support it for DCU5 disks on other IOP's            BEGIN.574
                                                                                                       BEGIN.575
2155   010035  024455                  R!T0 = CH$CH               .High speed channel                  BEGIN.576
2157   020000  024456                  R!T1 = R!%MYID             .This IOP                            BEGIN.577
2161   010000  024462                  R!TM = TM$CMEM             .Target Central memory               BEGIN.578
2163   073317                          R = CHSCAN                 .Search                              BEGIN.579
2164    <macro>                        $IF (R!T2 = 0)             .If found                            BEGIN.580
                                                                                  .>>>>> 0                $IF      1
2167   026456                            R!T1 = R!T1 + 1          .Search other IOP's                  BEGIN.581
2170   041000  013003  100004            R!T1 = BIOP, A > XIOP    .Wrap                                BEGIN.582
       102003  010001  024456
2176    <macro>                          $UNTIL (R!T1 = R!%MYID)                                       BEGIN.583
2201   072073                              R = DCU5               .Look for DCU5 disks                 BEGIN.584
2202    <macro>                            $IF (R!T3 = 0)         .If found                            BEGIN.585
                                                                                  .>>>>> 1                $IF      1
2205   010035  024455                        R!T0 = CH$CH         .Check for CMEM path                 BEGIN.586
2207   073343                                R = CHSCAN                                                BEGIN.587
2210    <macro>                              $IF (R!T2 # 0)       .If none                             BEGIN.588
                                                                                  .>>>>> 2                $IF      1
2213   071156                                  P = TRANSET        .Set up demons, end search           BEGIN.589
2214    <macro>                              $ENDIF                                                    BEGIN.590
                                                                                  .<<<<< 2                $ENDIF   1
2214    <macro>                            $ENDIF                                                      BEGIN.591
                                                                                  .<<<<< 1                $ENDIF   1
2214   026456                              R!T1 = R!T1 + 1        .Next IOP                            BEGIN.592
2215   041000  013003  100004              R!T1 = MIOP, A > XIOP  .Wrap                            R16410GA.1
       102003  010000  024456
2223    <macro>                          $ENDTIL                                                       BEGIN.594
2224    <macro>                         $ENDIF                                                         BEGIN.595
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                       BEGIN.596
                                       * If this IOP has a high speed path to SSD, set up demons       BEGIN.597
                                       * to support it for DCU5 disks on other IOP's                   BEGIN.598
                                                                                                       BEGIN.599
2224   010035  024455                  R!T0 = CH$CH               .High speed channel                  BEGIN.600
2226   020000  024456                  R!T1 = R!%MYID             .This IOP                            BEGIN.601
```

```
2230  010002  024462                R!TM = TM$SSD                 .Target SSD memory                BEGIN.602
2232  073366                        R = CHSCAN                    .Search                           BEGIN.603
2233   <macro>                      $IF (R!T2 = 0)                .If found                         BEGIN.604
                                                                                   .>>>>> 0            $IF      1
2236  026456                          R!T1 = R!T1 + 1             .Search other IOP's               BEGIN.605
2237  041000  013003  100004          R!T1 = BIOP, A > XIOP       .Wrap                             BEGIN.606
      102003  010001  024456
2245   <macro>                        $UNTIL (R!T1 = R!%MYID)                                       BEGIN.607
2250  072024                            R = DCU5                  .Look for DCU5 disks              BEGIN.608
2251   <macro>                        $IF (R!T3 = 0)              .If found                         BEGIN.609
                                                                                   .>>>>> 1            $IF      1
2254  010035  024455                      R!TO = CH$CH            .Check for SSD path               BEGIN.610
2256  073412                            R = CHSCAN                                                  BEGIN.611
2257   <macro>                        $IF (R!T2 # 0)              .If none                          BEGIN.612
                                                                                   .>>>>> 2            $IF      1
2262  071225                              P = TRANSET             .Set up demons, end search        BEGIN.613
2263   <macro>                        $ENDIF                                                        BEGIN.614
                                                                                   .<<<<< 2            $ENDIF   1
2263   <macro>                        $ENDIF                                                        BEGIN.615
                                                                                   .<<<<< 1            $ENDIF   1
2263  026456                            R!T1 = R!T1 + 1           .Next IOP                         BEGIN.616
2264  041000  013003  100004          R!T1 = MIOP, A > XIOP       .Wrap                         R16410GA.2
      102003  010000  024456
2272   <macro>                        $ENDTIL                                                       BEGIN.618
2273   <macro>                      $ENDIF                                                          BEGIN.619
                                                                                   .<<<<< 0            $ENDIF   1
2273  001000                        EXIT                                                            BEGIN.620
                                                                                                     BEGIN.621
                                 ***                                                                 BEGIN.622
                                 *        DCU5:     Search the specified IOP for DCU5 disks.          BEGIN.623
                                 *                                                                   BEGIN.624
                                 *        Entry:    R!T1 = IOP number                                 BEGIN.625
                                 *        Exit:     R!T3 = 0 if any found                             BEGIN.626
                                 ***                                                                 BEGIN.627
                                                                                                     BEGIN.628
2274                             DCU5     *                                                         BEGIN.629
2274  010003  024455                      R!TO = CH$D4            .Channel type - DD49              BEGIN.630
2276  073432                            R = CHSCAN                .Search                           BEGIN.631
2277   <macro>                        $IF (R!T2 # 0)              .If none found                    BEGIN.632
                                                                                   .>>>>> 0            $IF      1
2302  010103  024455                      R!TO = CH$D4S           .Striped DD49                     BEGIN.633
2304  073440                            R = CHSCAN                                                  BEGIN.634
```

```
2305    <macro>                 $ENDIF                                                  BEGIN.635
                                                                  .<<<<< 0                $ENDIF   1
2305   020457  024460           R!T3 = R!T2                .Save indicator              BEGIN.636
2307    <macro>                 $IF (R!T2 = 0)             .If any DD49's               BEGIN.637
                                                                  .>>>>> 0                $IF      1
2312    <macro>                   $IF (R!T1 = R!%MYID)     .In the current IOP          BEGIN.638
                                                                  .>>>>> 1                $IF      1
2316    <macro>                     CALL INDD49,(R!MA)     .Initialize                  BEGIN.639
2332    <macro>                   $ENDIF                                                 BEGIN.640
                                                                  .<<<<< 1                $ENDIF   1
2332    <macro>                 $ENDIF                                                  BEGIN.641
                                                                  .<<<<< 0                $ENDIF   1
2332   010004  024455           R!T0 = CH$D3               .DD39                        BEGIN.642
2334   073470                   R = CHSCAN                                              BEGIN.643
2335    <macro>                 $IF (R!T2 # 0)             .If none found               BEGIN.644
                                                                  .>>>>> 0                $IF      1
2340   010104  024455             R!T0 = CH$D3S            .Striped DD39                BEGIN.645
2342   073476                     R = CHSCAN                                            BEGIN.646
2343    <macro>                 $ENDIF                                                  BEGIN.647
                                                                  .<<<<< 0                $ENDIF   1
2343    <macro>                 $IF (R!T2 = 0)             .If any DD39's               BEGIN.648
                                                                  .>>>>> 0                $IF      1
2346    <macro>                   $IF (R!T1 = R!%MYID)     .In the current IOP          BEGIN.649
                                                                  .>>>>> 1                $IF      1
2352    <macro>                     CALL INDD39,(R!MA)     .Initialize                  BEGIN.650
2366    <macro>                   $ENDIF                                                 BEGIN.651
                                                                  .<<<<< 1                $ENDIF   1
2366   010000  024460             R!T3 = 0                 .Indicator                   BEGIN.652
2370    <macro>                 $ENDIF                                                  BEGIN.653
                                                                  .<<<<< 0                $ENDIF   1
2370   010005  024455           R!T0 = CH$D5               .DD50                        BEGIN.654
2372   073526                   R = CHSCAN                                              BEGIN.655
2373    <macro>                 $IF (R!T2 # 0)             .If none found               BEGIN.656
                                                                  .>>>>> 0                $IF      1
2376   010105  024455             R!T0 = CH$D5S            .Striped DD50                BEGIN.657
2400   073534                     R = CHSCAN                                            BEGIN.658
2401    <macro>                 $ENDIF                                                  BEGIN.659
                                                                  .<<<<< 0                $ENDIF   1
2401    <macro>                 $IF (R!T2 = 0)             .If any DD50's               BEGIN.660
                                                                  .>>>>> 0                $IF      1
2404    <macro>                   $IF (R!T1 = R!%MYID)     .In the current IOP          BEGIN.661
                                                                  .>>>>> 1                $IF      1
```

```
2410    <macro>                          CALL INDD50,(R!                .Initialize                          BEGIN.662
2424    <macro>                        $ENDIF                                                                BEGIN.663
                                                                                        .<<<<< 1                $ENDIF   1
2424   010000  024460                    R!T3 = 0                       .Indicator                           BEGIN.664
2426    <macro>                        $ENDIF                                                                BEGIN.665
                                                                                        .<<<<< 0                $ENDIF   1
2426   010006  024455                    R!T0 = CH$D40                  .DD40                             R13739AC.1
2430   073564                            R = CHSCAN                                                       R13739AC.2
2431    <macro>                          $IF (R!T2 # 0)                 .If none found                    R13739AC.3
                                                                                        .>>>>> 0                $IF      1
2434   010106  024455                    R!T0 = CH$D40S                 .Striped DD40                     R13739AC.4
2436   073572                            R = CHSCAN                                                       R13739AC.5
2437    <macro>                          $ENDIF                                                           R13739AC.6
                                                                                        .<<<<< 0                $ENDIF   1
2437    <macro>                          $IF (R!T2 = 0)                 .If any DD40's                    R13739AC.7
                                                                                        .>>>>> 0                $IF      1
2442    <macro>                          $IF (R!T1 = R!%MYID)           .In the current IOP               R13739AC.8
                                                                                        .>>>>> 1                $IF      1
2446    <macro>                          CALL INDD40,(R!MA)             .Initialize                       R13739AC.9
2462    <macro>                        $ENDIF                                                           R13739AC.10
                                                                                        .<<<<< 1                $ENDIF   1
2462   010000  024460                    R!T3 = 0                       .Indicator                        R13739AC.11
2464    <macro>                          $ENDIF                                                           R13739AC.12
                                                                                        .<<<<< 0                $ENDIF   1
2464   001000                            EXIT                                                                 BEGIN.666
                                                                                                              BEGIN.667
                                 ***                                                                          BEGIN.668
                                 *        ABLD:     Initialize A-A message queue                              BEGIN.669
                                 *                                                                            BEGIN.670
                                 *        Entry:    R!T3 = Length of queue                                    BEGIN.671
                                 *                  R!%W1 = ASCII header for buffer                           BEGIN.672
                                 *                  R!%W2 = "                    "                            BEGIN.673
                                 *                  R!%W3 = "                    "                            BEGIN.674
                                 *                                                                            BEGIN.675
                                 *        Exit:     R!T4 = Message queue address                              BEGIN.676
                                 ***                                                                          BEGIN.677
                                                                                                              BEGIN.678
2465                             ABLD     *                                                                  BEGIN.679
2465   072025                            R = GETBUF                     .Get and clear table space         BEGIN.680
2466   020461  012012  034461            (R!T4) = R!T4 + O'12           .Message head                      BEGIN.681
2471   020461  012001  024410            (R!T4 + 1) = R!T4 + R!T3       .Message tail                      BEGIN.682
       020461  022460  034410
```

```
2477   020461  012002  024410          (R!T4 + 2) = (R!T4)          .Current                    BEGIN.683
       030461  034410
2504   020461  012003  024410          (R!T4 + 3) = (R!T4)          .Next                       BEGIN.684
       030461  034410
2511   001000                          EXIT                                                     BEGIN.685
                                                                                                BEGIN.686
                                 ***                                                            BEGIN.687
                                 *       GETBUF:  Allocate and clear local memory buffer.       BEGIN.688
                                 *                                                              BEGIN.689
                                 *       Entry:   R!T3 = Length to allocate                     BEGIN.690
                                 *                R!%W1 = ASCII header for buffer               BEGIN.691
                                 *                R!%W2 = "                     "               BEGIN.692
                                 *                R!%W3 = "                     "               BEGIN.693
                                 *                                                              BEGIN.694
                                 *       Exit:    R!T4 = Address of buffer allocated            BEGIN.695
                                 ***                                                            BEGIN.696
                                                                                                BEGIN.697
2512                             GETBUF  *                                                      BEGIN.698
2512    <macro>                          GETMEM R!T3,R!T4,R!%W1,R!%W2,R!%W3 .Allocate          BEGIN.699
2532    <macro>                          $PUNTIF (A # 0),CODE=PT$LOCAL .If not available       BEGIN.700
2534    <macro>                          CLEAR START=R!T4,COUNT=R!T3                           BEGIN.701
2547   001000                            EXIT                                                  BEGIN.702
                                                                                                BEGIN.703
        <edit>                           SUBTITLE  '"$OSUB" Data Storage.'                     BEGIN.704
```

```
                                  **      Debbugger command.                                    BEGIN.706
                                                                                                BEGIN.707
                                                                                                BEGIN.708
        2550                      DEBC    *                                                     BEGIN.709
        2550   000000                     0                                                     BEGIN.710
        2551                              'DEBUG'L                                              BEGIN.711
        2554   000000                     0                                                     BEGIN.712
                                                                                                BEGIN.713
                                                                                                BEGIN.714
        2555                      LFCR    *                                                     BEGIN.715
        2555   005015                     O'5015                    .LF/CR                      BEGIN.716
        2556   000000                     0                         .Ensure end-of-line         BEGIN.717
                                                                                                BEGIN.718
                                                                                                BEGIN.719
        2557                      END     *                                                     BEGIN.720
                                          END                                                   BEGIN.721
```

0X	%ACTIVE		14:25 D	14:25	18:23 D	18:23			
0X	%B		1: 4 D	5:11 B	5:13 B	5:15 B	6: 3 B	7: 7 B	7:14 B	7:39 B
			8: 4 B	8:28 B	8:32 B	8:36 B	8:40 B	9: 7 B	9:12 B	9:21 B
			9:33 B	9:36 B	9:41 B	10: 3 B	10:12 B	10:20 B	10:24 B	10:30 B
			10:34 B	11: 2 B	11: 4 B	11:10 B	11:14 B	11:22 B	11:28 B	11:38 B
			11:42 B	12: 6 B	12:10 B	12:16 B	12:28 B	13: 5 B	15: 1	15:19
			18:22 B							
0X	%EX		5:10 D	5:10	5:14 D	5:14	5:16 D	5:16	11: 3 D	11: 3
			12:21 D	12:21	14:19 D	14:19	14:25 D	14:25	15: 1 D	15: 1
			15: 3 D	15: 3	15:10 D	15:10	15:14 D	15:14	15:22 D	15:22
			15:38 D	15:38	16:38 D	16:38	16:39 D	16:39	16:42 D	16:42
			18:12 D	18:12	18:23 D	18:23	18:29 D	18:29	22: 8 D	22: 8
			22:25 D	22:25	23: 1 D	23: 1	23:19 D	23:19	24:19 D	24:19
10	%GBLNUM	$APTEXT	2: 3 F	2: 3						
400	%GBLREG	$APTEXT	2: 2							
0X	%KCONS		3:13 D	5: 1						
0X	%MYID		3:14 D	6: 8	6:14	7: 1	7: 5	7:33	7:37	9: 3
			10: 3	11:14	12:32	13:19	14:25	14:31	15:19	19:35
			20:10	20:18	20:42	21: 8	22: 7	22:24	22:42	23:18
0X	%PUNT		6:32 D	6:32	7:27 D	7:27	15:15 D	15:15	17: 1 D	17: 1
			18:13 D	18:13	18:24 D	18:24	24:20 D	24:20		
0X	%TRANQ		3:15 D	18:39	19: 7	19: 8				
177777	AD$NODEM	$APTEXT	15:14	16:42						
20	AD@P1@N	$APTEXT	18:23	18:23 F						
14	AD@P1@P	$APTEXT	18:23 F	18:23						
0	AD@P1@S	$APTEXT	18:23	18:23 F						
20	AD@P2@N	$APTEXT	18:23	18:23 F						
15	AD@P2@P	$APTEXT	18:23 F	18:23						
0	AD@P2@S	$APTEXT	18:23	18:23 F						
20	AD@RC@N	$APTEXT	14:25	14:25 F	18:23	18:23 F				
13	AD@RC@P	$APTEXT	14:25 F	14:25	18:23 F	18:23				
0	AD@RC@S	$APTEXT	14:25	14:25 F	18:23	18:23 F				
6	AIA	$APTEXT	13:40							
7	AOA	$APTEXT	13:39	13:42	14:13					
1	BIOP	$APTEXT	6:12	9:30	10: 3	10:19	20:17	21: 7		
35	CH$CH	$APTEXT	7: 2	7:34	11:20	17:34	20: 9	20:22	20:41	21:12
1	CH$D1	$APTEXT	10:18	11:36						
101	CH$D1S	$APTEXT	10:29	12: 5						
2	CH$D2	$APTEXT	10:23	11:41						
102	CH$D2S	$APTEXT	10:33	12: 9						
4	CH$D3	$APTEXT	22:13							
104	CH$D3S	$APTEXT	22:17							

3	CH$D4	$APTEXT	21:37								
6	CH$D40	$APTEXT	23: 7								
106	CH$D40S	$APTEXT	23:11								
103	CH$D4S	$APTEXT	21:41								
5	CH$D5	$APTEXT	22:31								
105	CH$D5S	$APTEXT	22:35								
400	CRAY@SIZ	$APTEXT	7:19								
0X	D$ACOM		3:33	D	8:27						
0X	D$ADEM		3:34	D	9:11						
0X	D$AMSG		3:35	D	8:31						
0X	D$BCOM		3:36	D	9:20	10:11	11:27	12:40			
0X	D$CDEM		3:37	D	9: 6						
0X	D$CLOCK		3:38	D	8:35						
0X	D$CRTDEM		3:39	D	8:39						
0X	D$D4DEM		3:40	D	19:42						
0X	D$DISK		3:41	D	11: 1	12:19					
0X	D$ICOM		3:42	D	19:18						
0X	D$TRANSF		4: 1	D	18:42						
100	D4QSZI	$APTEXT	19:20								
0X	DCCB		3:16	D	6: 9						
2	DIOP	$APTEXT	11:14		11:21	11:37					
0X	EMEMSIZ		4: 2	D	15:35						
32	F$BGET	$APTEXT	5:14								
33	F$BRET	$APTEXT	15:10								
50	F$CALL	$APTEXT	11: 3		12:21	14:19	15:22	16:38	22: 8	22:25	23: 1
			23:19								
55	F$CREATE	$APTEXT	15:14		16:42	18:12					
53	F$FIND	$APTEXT	5:10								
4	F$GIVEUP	$APTEXT	16:39								
30	F$GMEM	$APTEXT	24:19								
20	F$MESG	$APTEXT	15: 1		15: 3						
35	F$MGET	$APTEXT	18:23								
46	F$MOSR	$APTEXT	5:16								
47	F$MOSW	$APTEXT	18:29								
37	F$OUTCAL	$APTEXT	14:25								
52	F$RETURN	$APTEXT	15:38								
0	F$SWAIT	$APTEXT	5:16		18:29						
1	I@BMXSYS	$APTEXT	9:17		10: 8	11:19	12:37				
0X	LBFW		3:17	D	18:25						
0X	LBPI		3:18	D	18:29						
1	LDV$BMK	$APTEXT	5:40								
2	LDV$DSK	$APTEXT	6:25								

177777	LDV$END	$APTEXT	5:37		6:31						
0X	MATX		3:19	D	13:12						
0X	MIAQ		3:20	D	19:25						
0	MIOP	$APTEXT	6: 8		7:25	8:15	9: 3	13:19	14:25	20:33	21:23
0X	O$ACOM		3:22	D	8:25						
0X	O$ADEM		3:23	D	9: 9						
0X	O$AMAP		5:10	D	5:10						
0X	O$AMSG		3:24	D	8:29						
0X	O$BCOM		3:25	D	9:18	10: 9	11:25	12:38			
0X	O$BEGIN		1: 4	D	1: 4						
0X	O$CALL		14:19	D	14:19	16:38 D	16:38	16:42 D	16:42		
0X	O$CDEM		3:26	D	9: 4						
0X	O$CLOCK		3:27	D	8:33						
0X	O$CRTDEM		3:28	D	8:37						
0X	O$D4DEM		3:29	D	19:40						
0X	O$DISK		3:30	D	10:41	12:17					
0X	O$DKERR1		14:25	D	14:25						
0X	O$ICOM		3:31	D	19:16						
0X	O$INDD29		11: 3	D	11: 3	12:21 D	12:21				
0X	O$INDD39		22:25	D	22:25						
0X	O$INDD40		23:19	D	23:19						
0X	O$INDD49		22: 8	D	22: 8						
0X	O$INDD50		23: 1	D	23: 1						
0X	O$SCRUB		15:14	D	15:14						
0X	O$TIME		15:22	D	15:22						
0X	O$TRANSF		3:32	D	18:40						
6	OV@ENT@P	$APTEXT	17:19								
17	OV@NUM@N	$APTEXT	1: 4								
1	OV@TYP@N	$APTEXT	1: 4								
0	PT$HUH	$APTEXT	15:15		17: 1	18:13					
102	PT$INIT	$APTEXT	6:32								
14	PT$LOCAL	$APTEXT	24:20								
55	PT$NOHSP	$APTEXT	7:27								
15	PT$NOMOS	$APTEXT	18:24								
11	SM@EST@N	$APTEXT	1: 4								
7	SM@NUM@N	$APTEXT	1: 4								
20000	SSD@SIZ	$APTEXT	8: 9								
1	TM$BMR	$APTEXT	6: 4		6:17						
0	TM$CHEM	$APTEXT	7: 3		7:15	7:17	20:11				
2	TM$SSD	$APTEXT	7:35		8: 5	8: 7	21: 1				
20	TM@FWO@N	$APTEXT	6:20	F							
1	TM@FWO@P	$APTEXT	6:20	F	6:20						

0	TM@IWO@S	$APTEXT	6:20 F								
20	TM@IOP@N	$APTEXT	6:12 F	6:14 F	7:18 F	8: 8 F					
5	TM@IOP@P	$APTEXT	6:12 F	6:12	6:14 F	6:14	7:18 F	7:18	8: 8 F	8: 8	
0	TM@IOP@S	$APTEXT	6:12 F	6:14 F	7:18 F	8: 8 F					
20	TM@IMO@N	$APTEXT	6:23 F	7:19 F	8: 9 F						
3	TM@IMO@P	$APTEXT	6:23 F	6:23	7:19 F	7:19	8: 9 F	8: 9			
0	TM@IMO@S	$APTEXT	6:23 F	7:19 F	8: 9 F						
6	TM@LT	$APTEXT	5:41	7:10	7:42						
20	TM@TYP@N	$APTEXT	6:17 F	7:17 F	8: 7 F						
0	TM@TYP@P	$APTEXT	6:17 F	7:17 F	8: 7 F						
0	TM@TYP@S	$APTEXT	6:17 F	7:17 F	8: 7 F						
0×	TMCB		3:21 D	5:26							
4	TRA@JT	$APTEXT	19: 2								
20	TRA@PR@N	$APTEXT	19: 8 F								
3	TRA@PR@P	$APTEXT	19: 8 F	19: 8							
0	TRA@PR@S	$APTEXT	19: 8 F								
3	XIOP	$APTEXT	7:24	8:14	9:32	12:32	20:16	20:32	21: 6	21:22	

SYMBOL QUALIFIER IS BEGIN

20000	$CONST	6:12 D	6:12 F	6:12	7:17 D	7:17 F	7:19 D	7:19 F	7:19		
		8: 7 D	8: 7 F	8: 7	8: 9 D	8: 9 F	8: 9				
467	$REGORG	2: 2 D	2: 3	2: 3 D	2: 3 F	2: 4	2: 4 D	2: 5	2: 5 D		
		2: 6	2: 6 D	2: 7	2: 7 D	2: 8	2: 8 D	2: 8 F	2: 9		
		2: 9 D	2:10	2:10 D	2:11	2:11 D	2:12	2:12 D	2:13		
		2:13 D	2:14	2:14 D	2:14 F	2:15	2:15 D	2:16	2:16 D		
		2:17	2:17 D	2:18	2:18 D	2:19	2:19 D	2:19 F	2:20		
		2:20 D	2:21	2:21 D	2:22	2:22 D	2:23	2:23 D	2:24		
		2:24 D	2:25	2:25 D	2:26	2:26 D	2:27	2:27 D	2:28		
		2:28 D	2:29	2:29 D	2:30	2:30 D	2:31	2:31 D	2:32		
		2:32 D	2:33	2:33 D	2:34	2:34 D	2:35	2:35 D	2:36		
		2:36 D	2:37	2:37 D	2:38	2:38 D	2:39	2:39 D	2:40		
		2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2		
		3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6		
		3: 6 D	3: 7	3: 7 D	3: 8	3: 8 D	3: 8 F	5:10	5:14		
		5:16	11: 3	11: 3 F	12:21	12:21 F	14:19	14:19 F	14:25		
		14:25 F	15: 1	15: 1 F	15: 3	15: 3 F	15:10	15:14	15:22		
		15:22 F	15:38	16:38	16:38 F	16:39	16:39 F	16:42	18:12		
		18:23	18:29	22: 8	22: 8 F	22:25	22:25 F	23: 1	23: 1 F		
		23:19	23:19 F	24:19							
0	%NP	1: 4	2:19 D								
430	%P	1: 4	2:19 D	2:19	11: 3 F	11: 3	12:21 F	12:21	14:19 F		

		14:19	14:25 F	14:25	15: 1 F	15: 1	15: 3 F	15: 3	15:22 F
		15:22	16:38 F	16:38	16:39 F	16:39	22: 8 F	22: 8	22:25 F
		22:25	23: 1 F	23: 1	23:19 F	23:19			
410	%S1	2: 3	2: 4 D	5:31	6: 4	6: 5	6:12	6:14	6:18
		6:20	6:23	7:15	7:16	7:18	7:19	8: 5	8: 6
		8: 8	8: 9	14:25	16:24	18:23	19: 8	19:25	19:26
		23:41	23:42	24: 1	24: 2	24: 3	24: 4		
411	%S2	2: 3	2: 5 D						
412	%S3	2: 3	2: 6 D						
413	%S4	2: 3	2: 7 D						
414	%S5	2: 3	2: 8 D						
415	%T1	2: 9 D	15:19	24:21					
416	%T2	2:10 D	24:21						
417	%T3	2:11 D							
420	%T4	2:12 D							
421	%T5	2:13 D							
422	%T6	2:14 D							
423	%W1	2:15 D	5:42	7:11	8: 1	15:35	15:37	17:20	17:21
		17:23	17:24	17:25	17:29	17:30	17:35	18:25	18:26
		18:27	18:28	19: 3	19:21	24:19			
424	%W2	2:16 D	6: 1	7:12	8: 2	15:36	15:37	17:25	17:28
		19: 4	19:22	24:19					
425	%W3	2:17 D	6: 2	6:19	6:20	6:22	6:23	7:13	8: 3
		17:26	17:28	18: 1	19: 5	19:23	24:19		
426	%W4	2:18 D	17:30						
427	%W5	2:19 D							
2465	ABLD	19:24	23:38 D						
1277	BEG1	5:11	5:13	5:15	14:37	14:40	15: 7 D		
1372	BEGEND	15:23	15:26	15:29	15:34 D				
1370	BIOPI	15:19	15:25 D						
430	CA	2:20 D	14:33	14:35	14:36	14:37	14:38	14:42	15: 2
		16:10	16:11	16:13	16:15	16:29			
1644	CHSCAN	7: 7	7:39	10:20	10:24	10:30	10:34	11:22	11:38
		11:42	12: 6	12:10	17:18 D	20:12	20:23	21: 2	21:13
		21:38	21:42	22:14	22:18	22:32	22:36	23: 8	23:12
431	CL	2:21 D	14:35	14:39	14:40	14:41	14:42		
267	CMSRCH	7: 4	7: 6 D						
400	CRT	2: 3 D	5: 1	16:40					
2274	DCU5	9:33	19:36	20:19	21: 9	21:36 D			
2550	DEBC	25: 4 D							
1724	DEMON	8:28	8:32	8:36	8:40	9: 7	9:12	9:21	10:12
		11: 2	11:28	12:20	12:41	18:11 D	19: 1	19:19	20: 1

1371	DIOPL	15:19	15:28 D						
1743	DKBUF	11: 4	12:22	18:20 D	20: 2				
2557	END	25:15 D							
2512	GETBUF	6: 3	7:14	8: 4	19: 6	23:39	24:18 D		
464	ICOM	3: 6 D	5:18	19:15	19:27				
2074	ICOMSET	9:36	19:14 D	19:39					
2134	INDCU5	9:41	11:10	12:28	13: 5	19:34 D			
1407	LCM	15: 4	16: 6 D						
2555	LFCR	15: 1	25:10 D						
432	MA	2:22 D	5: 9	5:14	5:16	5:31	5:35	6:18	11: 3
		12:21	14:31	14:33	14:38	14:41	15: 9	15:10	17:19
		17:21	17:22	17:24	22: 8	22:25	23: 1	23:19	
1353	MIOPL	15:19	15:21 D						
466	ML	3: 8 D	5: 5	14:24	14:25	18:22	18:23	18:28	18:29
465	MU	3: 7 D	5: 4	14:24	14:25	18:22	18:23	18:26	18:29
1721	NOMATCH	17:22	18: 3 D						
433	PA	2:23 D	3:10	16: 7	16:38	16:42			
21	PARL	3:10 D	16:12						
434	PB	2:24 D	16:38	16:42					
435	PC	2:25 D	16:38	16:42					
436	PD	2:26 D	16:38	16:42					
437	PE	2:27 D	16:38	16:42					
440	PF	2:28 D	16:38	16:42					
441	PG	2:29 D	16:38	16:42					
442	PH	2:30 D	16:38	16:42					
443	PI	2:31 D	16:38	16:42					
444	PJ	2:32 D	16:38	16:42					
445	PK	2:33 D	16:38	16:42					
446	PL	2:34 D	16:38	16:42					
447	PM	2:35 D	16:38	16:42					
450	PN	2:36 D	16:38	16:42					
451	PO	2:37 D	16:38	16:42					
452	PP	2:38 D	16:38	16:42					
453	PQ	2:39 D	16:38	16:42					
454	PR	2:40 D	3:10	16:38	16:42				
357	SSDSRCH	7:36	7:38 D						
455	TO	2:41 D	6:10	6:11	7: 2	7:34	8:25	8:29	8:33
		8:37	9: 4	9: 9	9:18	10: 9	10:18	10:23	10:29
		10:33	10:41	11:20	11:25	11:36	11:41	12: 5	12: 9
		12:17	12:38	13:12	13:20	13:21	13:36	14: 1	14: 2
		14:11	15: 2	15: 3	16: 8	16:12	16:17	17:32	17:34
		18:12	18:40	19:16	19:40	20: 9	20:22	20:41	21:12

		21:37	21:41	22:13	22:17	22:31	22:35	23: 7	23:11
456	T1	2:42 D	5:32	5:35	5:38	6: 6	6: 9	6:21	6:22
		6:26	6:27	6:28	6:30	7: 1	7: 5	7:18	7:23
		7:25	7:33	7:37	8: 8	8:13	8:15	8:26	8:30
		8:34	8:38	9: 5	9:10	9:19	9:30	9:32	9:39
		10:10	10:19	10:42	11:21	11:26	11:37	12:18	12:39
		13:13	13:20	14: 1	14:31	14:33	16: 9	16:12	16:14
		17:19	18:12	18:41	19: 8	19:17	19:35	19:41	20:10
		20:15	20:17	20:18	20:31	20:33	20:42	21: 5	21: 7
		21: 8	21:21	21:23	22: 7	22:24	22:42	23:18	
457	T2	3: 1 D	5:10	5:12	5:16	5:36	5:37	5:38	6:25
		6:31	7: 9	7:41	8:27	8:31	8:35	8:39	9: 6
		9:11	9:20	10:11	10:22	10:28	10:32	10:40	11: 1
		11:24	11:27	11:40	12: 4	12: 8	12:16	12:19	12:40
		16:10	16:36	17:27	18: 4	18:12	18:42	19:18	19:42
		20:14	20:25	21: 4	21:15	21:40	22: 3	22: 5	22:16
		22:22	22:34	22:40	23:10	23:16			
460	T3	3: 2 D	5:10	5:16	5:41	7:10	7:42	9:31	9:32
		9:35	19: 2	19:20	19:38	20:21	21:11	22: 3	22:28
		23: 4	23:22	23:42	24:19	24:21			
461	T4	3: 3 D	5:10	5:16	6: 4	6:11	6:12	6:14	6:17
		6:20	6:23	7:15	7:17	7:18	7:19	8: 5	8: 7
		8: 8	8: 9	19: 7	19:25	23:40	23:41	24: 1	24: 3
		24:19	24:21						
463	TBL	3: 5 D	5:26	6: 4	7:15	8: 5			
462	TM	3: 4 D	7: 3	7:35	17:36	20:11	21: 1		
255	TMCMEM	6:24	6:42 D						
434	TMEND	8:10	8:17 D						
345	TMSSD	7:20	7:32 D						
2035	TRANSET	18:38 D	20:26	21:16					
1372	XIOPL	15:19	15:31 D						

```
0     <macro>                    OVERLAY   CALL                                                    CALL.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    **      CALL - Process a command entered at a KERNEL console keyboard.      CALL.5


                    **              CALL is entered when a command is keyed in at one of        CALL.7
                    *       the consoles allocated to the KERNEL.  It isolates the first        CALL.8
                    *       set of characters and validates it against a command table.         CALL.9
                    *       If a match occurs, CALL proceses the command locally or by          CALL.10
                    *       passing control to a designated overlay.                            CALL.11


                    **      Entry   Character string from keyboard.                             CALL.13
                    *               Registers IA thru IQ contain character string.              CALL.14


                    **      Messages:                                                           CALL.16
                    *                                                                           CALL.17
                    *       *COMMAND ERROR* - Invalid command.                                  CALL.18
                    *       *PARAMETER ERROR* - Invalid parameter.                              CALL.19


                            BASE    D                                                           CALL.21
                                                                                                CALL.22
6   <macro>                 REGDEFS (CRT),(IA,IB,IC,ID,IE,IF,IG,IH,II,IJ,IK,IL,IM,IN,IO,        CALL.23
                    ,               IP,IQ,IR,IS,IT,IU,IV,IW,IX,IY,IZ,IAA,IAB,IAC,IAD,IAE,       CALL.24
                    ,               IAF,IAG,IAH,IAI,IAJ,IAK,IAL,IAM,IAN),(NA,NB,NC,ND,ER,       CALL.25
                    ,               T0,T1,T2,T3,T4)                                             CALL.26
            400     CRT     EQUALS  $REGORG                                                     $REG    3
            410     %S1     EQUALS  $REGORG                                                     $REG    3
            411     %S2     EQUALS  $REGORG                                                     $REG    3
            412     %S3     EQUALS  $REGORG                                                     $REG    3
            413     %S4     EQUALS  $REGORG                                                     $REG    3
            414     %S5     EQUALS  $REGORG                                                     $REG    3
            415     %T1     EQUALS  $REGORG                                                     $REG    3
            416     %T2     EQUALS  $REGORG                                                     $REG    3
```

```
417     %T3     EQUALS    $REGORG                                                    $REG    3
420     %T4     EQUALS    $REGORG                                                    $REG    3
421     %T5     EQUALS    $REGORG                                                    $REG    3
422     %T6     EQUALS    $REGORG                                                    $REG    3
423     %W1     EQUALS    $REGORG                                                    $REG    3
424     %W2     EQUALS    $REGORG                                                    $REG    3
425     %W3     EQUALS    $REGORG                                                    $REG    3
426     %W4     EQUALS    $REGORG                                                    $REG    3
427     %W5     EQUALS    $REGORG                                                    $REG    3
430     IA      EQUALS    $REGORG                                                    $REG    3
431     IB      EQUALS    $REGORG                                                    $REG    3
432     IC      EQUALS    $REGORG                                                    $REG    3
433     ID      EQUALS    $REGORG                                                    $REG    3
434     IE      EQUALS    $REGORG                                                    $REG    3
435     IF      EQUALS    $REGORG                                                    $REG    3
436     IG      EQUALS    $REGORG                                                    $REG    3
437     IH      EQUALS    $REGORG                                                    $REG    3
440     II      EQUALS    $REGORG                                                    $REG    3
441     IJ      EQUALS    $REGORG                                                    $REG    3
442     IK      EQUALS    $REGORG                                                    $REG    3
443     IL      EQUALS    $REGORG                                                    $REG    3
444     IM      EQUALS    $REGORG                                                    $REG    3
445     IN      EQUALS    $REGORG                                                    $REG    3
446     IO      EQUALS    $REGORG                                                    $REG    3
447     IP      EQUALS    $REGORG                                                    $REG    3
450     IQ      EQUALS    $REGORG                                                    $REG    3
451     IR      EQUALS    $REGORG                                                    $REG    3
452     IS      EQUALS    $REGORG                                                    $REG    3
453     IT      EQUALS    $REGORG                                                    $REG    3
454     IU      EQUALS    $REGORG                                                    $REG    3
455     IV      EQUALS    $REGORG                                                    $REG    3
456     IW      EQUALS    $REGORG                                                    $REG    3
457     IX      EQUALS    $REGORG                                                    $REG    3
460     IY      EQUALS    $REGORG                                                    $REG    3
461     IZ      EQUALS    $REGORG                                                    $REG    3
462     IAA     EQUALS    $REGORG                                                    $REG    3
463     IAB     EQUALS    $REGORG                                                    $REG    3
464     IAC     EQUALS    $REGORG                                                    $REG    3
465     IAD     EQUALS    $REGORG                                                    $REG    3
466     IAE     EQUALS    $REGORG                                                    $REG    3
467     IAF     EQUALS    $REGORG                                                    $REG    3
470     IAG     EQUALS    $REGORG                                                    $REG    3
```

```
471     IAH     EQUALS  $REGORG                                          $REG    3
472     IAI     EQUALS  $REGORG                                          $REG    3
473     IAJ     EQUALS  $REGORG                                          $REG    3
474     IAK     EQUALS  $REGORG                                          $REG    3
475     IAL     EQUALS  $REGORG                                          $REG    3
476     IAM     EQUALS  $REGORG                                          $REG    3
477     IAN     EQUALS  $REGORG                                          $REG    3
500     NA      EQUALS  $REGORG                                          $REG    3
501     NB      EQUALS  $REGORG                                          $REG    3
502     NC      EQUALS  $REGORG                                          $REG    3
503     ND      EQUALS  $REGORG                                          $REG    3
504     ER      EQUALS  $REGORG                                          $REG    3
505     T0      EQUALS  $REGORG                                          $REG    3
506     T1      EQUALS  $REGORG                                          $REG    3
507     T2      EQUALS  $REGORG                                          $REG    3
510     T3      EQUALS  $REGORG                                          $REG    3
511     T4      EQUALS  $REGORG                                          $REG    3
                                                                    CALL.27
 50     REGL    EQUALS  IAN-IA+1          .REGISTER COUNT           CALL.28
120     CARL    EQUALS  REGL*2            .CHARACTORS               CALL.29

                EXT     %ACTIVE                                     CALL.31
                EXT     %DEBBA                                      CALL.32
                EXT     %DEBUG                                      CALL.33
                EXT     %MYID                                       CALL.34
                EXT     %SVPT                                       CALL.35
                EXT     DEB$INIT                                    CALL.36
                EXT     DEBLEN                                      CALL.37
                EXT     DEBSTART                                    CALL.38
                EXT     EC$COMER                                    CALL.39
                EXT     EC$PARM                                     CALL.40
                EXT     EXPDCT                                      CALL.41
                EXT     XCRT                                        CALL.42
```

```
                    ****************************  ****************************************  CALL.44
                    *                                                                     *  CALL.45
                    *        COME       Build command table entry                         *  CALL.46
                    *                                                                     *  CALL.47
                    *        COME       command,FROM=iopi                                 *  CALL.48
                    *                                                                     *  CALL.49
                    *        command    Keyboard command; also the name of a subroutine   *  CALL.50
                    *                   which is called to process the command.            *  CALL.51
                    *                                                                     *  CALL.52
                    *        iopi       IOPs for which this command is valid.  It is       *  CALL.53
                    *                   specified as of the parameters IOP0, IOP1, IOP2,   *  CALL.54
                    *                   and IOP3; the default value is IOP0.               *  CALL.55
                    *                                                                     *  CALL.56
                    ***********************************************************************  CALL.57
                                                                                             CALL.58
                             MACRO                                                           CALL.59
<prototype>                  COME       COMMAND,FROM=IOP0                                    CALL.60
<definition>                 'COMMAND'8L                                                     CALL.61
<definition>                 PDATA      COMMAND                                              CALL.62
<definition>                 VWD        4/FROM,12/0                                          CALL.63
                    COME     ENDM                                                            CALL.64


                    **       OCOME - Build command table entry for overlay calls.            CALL.66
                    *        Call       OCOME   name,OVL=overlay,FROM=iopn                   CALL.67
                    *                   name - Command name.                                 CALL.68
                    *                   overlay - Overlay to call.  If not specified, name   CALL.69
                    *                             is used.                                   CALL.70
                    *                   iopn - IOP's on which this command may be issued.    CALL.71
                    *                          It is specified as a sum of IOP0, IOP1, IOP2  CALL.72
                    *                          and IOP3.  If not entered, IOP0 is used.      CALL.73
                                                                                             CALL.74
                             MACRO                                                           CALL.75
<prototype>                  OCOME      NAME,OVL=,FROM=IOP0                                  CALL.76
                             LOCAL      $OVN                                                 CALL.77
<definition>        OCOME1   IFC        !_OVL_!,EQ,,                                         CALL.78
<definition>        $OVN     $CONCAT    (O$,NAME),D'8                                        CALL.79
<definition>        OCOME1   ELSE                                                           CALL.80
<definition>        $OVN     $CONCAT    (O$,OVL),D'8                                         CALL.81
<definition>        OCOME1   ENDIF                                                          CALL.82
```

```
        <definition>                EXT       "$OVN"                                          CALL.83
        <definition>                'NAME'8L                                                  CALL.84
        <definition>                PDATA     OCALL                                           CALL.85
        <definition>                VWD       4/FROM,12/"$OVN"                                CALL.86
                            OCOME   ENDM                                                      CALL.87


                            **      COMI - Command table description.                         CALL.89
                                                                                              CALL.90
  6     <macro>           KC@SUB    FIELD     4                    . Subroutine offset address. CALL.91
  6     <macro>           KC@IOP    FIELD     5,0,4                . IOP bit map.              CALL.92
  6     <macro>           KC@OVL    FIELD     5,4,12               . Overlay number.           CALL.93
  6     <macro>           KC@LEN    FIELD     6,*                  . Entry length.             CALL.94


                     10     IOP0    EQUALS    0'10                 . IOP0.                     CALL.96
                      4     IOP1    EQUALS    0'04                 . IOP1.                     CALL.97
                      2     IOP2    EQUALS    0'02                 . IOP2.                     CALL.98
                      1     IOP3    EQUALS    0'01                 . IOP3.                     CALL.99
                                                                                              CALL.100
        <edit>                      SUBTITLE '"$OSUB" Main Program.'                          CALL.101
```

```
 6                          CALL     *                                                          CALL.102
 6   010000                          A = 0                                                      CALL.103
 7   024500                          NA = A                        . Clear overlay name.        CALL.104
10   024501                          NB = A                                                     CALL.105
11   024502                          NC = A                                                     CALL.106
12   024503                          ND = A                                                     CALL.107
13   010430  024505                  T0 = [IA]                     . Input character pointers.  CALL.108
15   010000  024506                  T1 = 0                                                     CALL.109
17   010500  024510                  T3 = [NA]                     . Command keyword pointer.   CALL.110
21   010000  024511                  T4 = 0                                                     CALL.111
23   010000  024504                  ER = 0                                                     CALL.112
                                                                                                 .

                            *        Delete leading spaces.                                     CALL.114
                                                                                                 CALL.115
25   077000 /001604         CALL20   R = GOC                       . Get a character.           CALL.116
27   020507  013040  106004          P = CALL20, T2 = ' 'R         . If space.                  CALL.117
32   020507  103004                  P = CALL40, T2 # 0            .If not a null line          CALL.118
                                                                                                 CALL.119
                            *        Process Null line - Display CRT header.                    CALL.120
                                                                                                 CALL.121
34   077000 /001655                  R = PNL                       .Process null line           CALL.122
36   070047                          P = ERR                       .Exit from overlay           CALL.123


                            *        Move out command keyword.                                  CALL.125
                                                                                                 CALL.126
37                          CALL40   *                                                          CALL.127
37   020507  013040  102020          P = CALL50, T2 = ' 'R         . If end of name.            CALL.128
42   020507  102016                  P = CALL50, T2 = 0            . If end of input.           CALL.129
44   <macro>                         $IF        (T4 = 8)           . If name is too long.       CALL.130
                                                                                  .>>>>> 0          $IF      1
50   014000 /000000  024504            R!ER = EC$COMER                                            CALL.131
53   070032                            P = ERR                                                    CALL.132
54   <macro>                         $ENDIF                                                       CALL.133
                                                                                  .<<<<< 0          $ENDIF   1
54   077000 /002005                  R = SOC                       . Store one character.       CALL.134
56   077000 /001604                  R = GOC                       . Get next character.        CALL.135
```

```
 60   071021                          P = CALL40                                                    CALL.136


                                *      Eliminate command name from character string.                 CALL.138
                                                                                                     CALL.139
 61   010430  024510            CALL50 T3 = [IA]                      . Move destination pointer.    CALL.140
 63   010000  024511                   T4 = 0                                                        CALL.141
 65   020507  137000 /001604    CALL60 R = GOC, T2 # 0               .GET NEXT CHAR IF NOT EOL      CALL.142
 70   020507  013040  106005           P = CALL60, T2 = ' 'R        . If leading space.             CALL.143
                                                                                                     CALL.144
 73                             CALL70 *                                                            CALL.145
 73   020511  013120  102007           P = CALL90, T4 = CARL        . If character string moved.    CALL.146
 76   077000 /002005                   R = SOC                      . Store the character.          CALL.147
100   020507  137000 /001604           R = GOC, T2 # 0              .IF NOT EOL                     CALL.148
103   071010                           P = CALL70                                                   CALL.149


                                *      Complete command/overlay process.                            CALL.151
                                                                                                     CALL.152
104                             CALL90 *                                                            CALL.153
104   072034                           R = PLC                      . Process command.              CALL.154


                                **     ERR - Process error.                                         CALL.156
                                *      Entry     ER = Error code.                                   CALL.157
                                                                                                     CALL.158
105                             ERR    *                                                            CALL.159
105   <macro>                          $IF       (ER # 0)           . If error message to be sent.  CALL.160
                                                                                     .>>>>> 0          $IF      1
110   <macro>                            CALL    INFORM,(R!ER,0,0,0)                                CALL.161
132   <macro>                          $ENDIF                                                       CALL.162
                                                                                     .<<<<< 0          $ENDIF   1
                                                                                                     CALL.163
132   <macro>                          RETURN                                                       CALL.164
      <edit>                           SUBTITLE  '"$OSUB" Process Commands.'                        CALL.165
```

```
                                **      PLC - Process comm.                                          CALL.166
                                *       Entry     Registers NA thru ND has command name.             CALL.167
                                *       Exit      If command not found.                              CALL.168
                                *                 ER = Error code if command error.                  CALL.169
                                                                                                     CALL.170
                                                                                                     CALL.171
140                             PLC     *                                                            CALL.172
140   020000  016000 /000225            R!%W1 = R!%B+COMT              . Command table address.      CALL.173
      024423
                                                                                                     CALL.174
144    <macro>                          $UNTIL    ((R!%W1) = 0)        . Until end of table is reached. CALL.175
146   010500  054000                      B = [NA]                     . Start of command.           CALL.176
150   020423  024424                      R!%W2 = R!%W1                . Entry keyword address.      CALL.177
152   010004  024425                      R!%W3 = 4                    . Check up to 8 characters.   CALL.178
                                                                                                     CALL.179
154    <macro>                            $UNTIL  (R!%W3 = 0)          . Until 8 characters are checked. CALL.180
156   060000  033424  103036              P = NEXT, (B) # (R!%W2)      . If no match.                CALL.181
161   056000                              B = B+1                                                    CALL.182
162   026424                              R!%W2 = R!%W2+1                                            CALL.183
163   027425                              R!%W3 = R!%W3-1                                            CALL.184
164    <macro>                            $ENDTIL                                                    CALL.185
                                                                                                     CALL.186
165    <macro>                            ADDRESS R!%W2,KC@IOP,R!%W1   . Point to IOP bit map.       CALL.187
170    <macro>                            GETBIT  A,R!%MYID,R!%W2                                    CALL.188
203   102013                              P = NEXT, A = 0              . If invalid for this IOP.    CALL.189
                                                                                                     CALL.190
204    <macro>                            GET     R!%W2,KC@SUB,R!%W1                                 CALL.191
211   020424  022000  024424              R!%W2 = R!%W2+R!%B           . Address of subroutine.      CALL.192
214   076424                              R = R!%W2                    . Process the command.        CALL.193
215   001000                              EXIT                                                       CALL.194
                                                                                                     CALL.195
216                             NEXT      *                                                          CALL.196
216   010006  025423                      R!%W1 = R!%W1+KC@LEN@P       . Advance to next entry.      CALL.197
220    <macro>                          $ENDTIL                                                    CALL.198
                                                                                                     CALL.199
221   014000 /000000  024504            R!ER = EC$COMER                .Invalid command              CALL.200
224   001000                            EXIT                                                         CALL.201
```

```
225                         COMT    *                                                          CALL.203
225    <macro>                      COME    ABORT                                              CALL.204
233    <macro>                      COME    BUGOFF,FROM=IOP0+IOP1+IOP2+IOP3                    CALL.205
241    <macro>                      COME    DEBUG,FROM=IOP0+IOP1+IOP2+IOP3                     CALL.206
247    <macro>                      COME    DISABLE                                            CALL.207
255    <macro>                      COME    ENABLE                                             CALL.208
263    <macro>                      COME    ERROR                                              CALL.209
271    <macro>                      COME    RESTART                                            CALL.210
277    <macro>                      COME    RESUME                                             CALL.211

305    <macro>                      OCOME   ASD                                        R17260AC.1
313    <macro>                      OCOME   CHNTEST,OVL=CHNTES,FROM=IOP0+IOP1+IOP2+IOP3        CALL.213
321    <macro>                      OCOME   CLEAR                                              CALL.214
327    <macro>                      OCOME   CONC                                               CALL.215
335    <macro>                      OCOME   CONFIG,FROM=IOP0+IOP1+IOP2+IOP3                    CALL.216
343    <macro>                      OCOME   COPY                                               CALL.217
351    <macro>                      OCOME   CPTEST,OVL=CPSPIN,FROM=IOP3                        CALL.218
357    <macro>                      OCOME   CRAY                                               CALL.219
365    <macro>                      OCOME   DDUMP                                              CALL.220
373    <macro>                      OCOME   DEF                                                CALL.221
401    <macro>                      OCOME   DELETE                                             CALL.222
407    <macro>                      OCOME   DKDMP                                              CALL.223
415    <macro>                      OCOME   DLOAD                                              CALL.224
423    <macro>                      OCOME   DSTAT                                              CALL.225
431    <macro>                      OCOME   ECHOCP                                             CALL.226
437    <macro>                      OCOME   EDIT                                               CALL.227
445    <macro>                      OCOME   ENDCONC                                            CALL.228
453    <macro>                      OCOME   ERRDMP                                             CALL.229
461    <macro>                      OCOME   FDUMP                                              CALL.230
467    <macro>                      OCOME   FLAW                                               CALL.231
475    <macro>                      OCOME   FLOAD                                              CALL.232
503    <macro>                      OCOME   FORMAT                                             CALL.233
511    <macro>                      OCOME   FSTAT                                              CALL.234
517    <macro>                      OCOME   FSCOPY                                             CALL.235
525    <macro>                      OCOME   GETIME,OVL=WATCH                                   CALL.236
533    <macro>                      OCOME   HELP,FROM=IOP0+IOP1+IOP2+IOP3                      CALL.237
541    <macro>                      OCOME   HSPTEST,OVL=HSPTES                                 CALL.238
547    <macro>                      OCOME   IACON                                              CALL.239
555    <macro>                      OCOME   IAIOP                                              CALL.240
563    <macro>                      OCOME   INIT                                               CALL.241
571    <macro>                      OCOME   LISTO                                              CALL.242
```

```
  577   <macro>          OCOME    LISTP                                          CALL.243
  605   <macro>          OCOME    MFCLR                                          CALL.244
  613   <macro>          OCOME    MFINIT                                         CALL.245
  621   <macro>          OCOME    MOSTEST,OVL=MOSTES                             CALL.246
  627   <macro>          OCOME    NSC                                            CALL.247
  635   <macro>          OCOME    NSCEND                                         CALL.248
  643   <macro>          OCOME    PART                                           CALL.249
  651   <macro>          OCOME    PATCH,FROM=IOP0+IOP1+IOP2+IOP3                 CALL.250
  657   <macro>          OCOME    PLOT                                       R15159AB.1
  665   <macro>          OCOME    PROC                                           CALL.252
  673   <macro>          OCOME    PRTAPE                                         CALL.253
  701   <macro>          OCOME    SETIME                                         CALL.254
  707   <macro>          OCOME    RENAME                                         CALL.255
  715   <macro>          OCOME    SSDTEST,OVL=SSDTES                             CALL.256
  723   <macro>          OCOME    START                                          CALL.257
  731   <macro>          OCOME    STATION                                        CALL.258
  737   <macro>          OCOME    STOP,FROM=IOP0+IOP1+IOP2+IOP3                  CALL.259
  745   <macro>          OCOME    SYSTEM,OVL=OPSYS                               CALL.260
  753   <macro>          OCOME    TRACE,FROM=IOP0+IOP1+IOP2+IOP3                 CALL.261
  761   <macro>          OCOME    TTY                                            CALL.262
  767   <macro>          OCOME    UBTAPE                                         CALL.263
  775   <macro>          OCOME    XDK                                            CALL.264
 1003   <macro>          OCOME    XMT                                            CALL.265
 1011   <macro>          OCOME    XPR                                            CALL.266

 1017   000000           0                        . End of table.                CALL.268
```

```
                        *       Call the overlay designated by command table entry.          CALL.270
                        *                                                                    CALL.271
                        *       Entry:    R!%W1 points to the command table entry.           CALL.272
                                                                                             CALL.273
1020                    OCALL   *                                                            CALL.274
1020    <macro>                 GEI       R!%W2,KC@OVL,R!%W1  . Get the overlay number.      CALL.275
                                                                                             CALL.276
1027    <macro>                 GOTO      R!%W2,(IA,IB,IC,ID,IE,IF,IG,IH,II,IJ,IK,IL,IM,IN,IO,I  CALL.277
                        ,P,IQ,IR,IS,IT,IU,IV,IW,IX,IY,IZ,R!IAA,R!IAB,R!IAC,R!IAD,R!IAE,R!IAF,R!I  CALL.278
                        ,AG,R!IAH,R!IAI,R!IAJ,R!IAK,R!IAL,R!IAM,R!IAN),TYPE=NUMBER           CALL.279
```

```
                                 ***************************  ****************************************   CALL.281
                                 *                                                                   *   CALL.282
                                 *        DEBUG     Enter the Debugger                               *   CALL.283
                                 *                                                                   *   CALL.284
                                 *********************************************************************   CALL.285
                                                                                                         CALL.286
1157                             DEBUG    *                                                             CALL.287
1157   <macro>                            $IF       (R!%DEBUG = 0)       .If debugger not in memory    CALL.288
                                                                                        .>>>>> 0          $IF      1
1162   072411                               R = GBF                                                    CALL.289
1163   102002  001000                       EXIT, A # 0                   .If no buffer available      CALL.290
1165   <macro>                              MOSR    0,DEBSTART,R!NA,DEBLEN .Read into memory           CALL.291
1207   020500  024000                       R!%DEBUG = R!NA               .Establish base address      CALL.292
1211   014000 /000000  005002               R!%W1 = DEBSTART < 2          . TEMP ***************       CALL.293
       024423
1215   020000  023423  024000               R!%DEBBA = R!%DEBUG - R!%W1 . TEMP ***************         CALL.294
1220   <macro>                            $ENDIF                                                       CALL.295
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                         CALL.296
1220   014000 /000000  024410             (XCRT) = R!CRT < 1 + O'41     .Originating CRT channel number CALL.297
       020400  005001  012041
       034410
1227   <macro>                            RKERN     DEBUGGER            .Enter the debugger          CALL.298
1231   003000                             I = 1                                                        CALL.299
1232   001000                             EXIT                                                         CALL.300


                                 *********************************************************************   CALL.302
                                 *                                                                   *   CALL.303
                                 *        BUGOFF    Release the Debugger                             *   CALL.304
                                 *                                                                   *   CALL.305
                                 *********************************************************************   CALL.306
                                                                                                         CALL.307
1233                             BUGOFF   *                                                             CALL.308
1233   <macro>                            $IF       (R!%DEBUG # 0)       .If debugger in memory        CALL.309
                                                                                        .>>>>> 0          $IF      1
1236   <macro>                              RKERN   DEBUGGER             .Remove all breakpoints      CALL.310
1240   <macro>                              $NOOP   DEB$INIT             .Inline function code        CALL.311
1241   020000  024500                       R!NA = R!%DEBUG                                            CALL.312
1243   010000  024000                       R!%DEBUG = 0                                               CALL.313
```

```
1245   003000                    I = 1                                          CALL.314
1246   072523                    R = RBF          .Release local memory buffer  CALL.315
1247    <macro>                 $ENDIF                                          CALL.316
                                                              .<<<<< 0              $ENDIF    1
                                                                                CALL.317
1247   001000                  EXIT                                            CALL.318
```

```
                        *       Device commands.                                                CALL.320
                                                                                                CALL.321
                                * ABORT DEVICE                                                  CALL.322
                                                                                                CALL.323
1250                    ABORT   *                                                               CALL.324
1250  072162                    R = DEVICE                      .FIND DEVICE TABLE              CALL.325
1251   <macro>                  GET R!%W1,XD@IPN,R!%W2          .CHECK FOR DEVICE ACTIVITY      CALL.326
1257   <macro>                  GET R!%W3,XD@IRT,R!%W2          .                               CALL.327
1265   <macro>                  $IF (R!%W1 = 0) AND (R!%W3 = 0)                                 CALL.328
                                                                                >>>>> 0             $IF      1
1273   <macro>                    GET A,XD@OPR,R!%W2            .SEE IF WAITING FOR RESPONSE    CALL.329
1300   <macro>                    $IF (A # R$RESP)              .IF NOT                         CALL.330
                                                                                >>>>> 1             $IF      1
1303   <macro>                      GET A,XD@KEY,R!%W2          .IF HUNG                        CALL.331
1307   <macro>                      $IF (A # 0)                 .                               CALL.332
                                                                                >>>>> 2             $IF      1
1311   <macro>                        $IF (R!IA = 'MT')    .IF TAPE DEVICE                      CALL.333
                                                                                >>>>> 3             $IF      1
1316   <macro>                          STORE XD$READ,XD@MOD,R!%W2 .DON'T ALLOW EOF WRITE       CALL.334
1323   <macro>                        $ENDIF                                                   CALL.335
                                                                                <<<<< 3             $ENDIF   1
1323   <macro>                        PUT R!%ACTIVE,XD@USR,R!%W2  .USURP DEVICE                CALL.336
1330   <macro>                        CLOSE DCT=R!%W2             .CLOSE IT                    CALL.337
1350   <macro>                      $ENDIF                                                     CALL.338
                                                                                <<<<< 2             $ENDIF   1
1350  001006                        EXIT                                                       CALL.339
1351   <macro>                    $ENDIF                                                       CALL.340
                                                                                <<<<< 1             $ENDIF   1
1351   <macro>                  $ENDIF                                                         CALL.341
                                                                                <<<<< 0             $ENDIF   1
1351   <macro>                  STORE R$ABRT,XD@OPR,R!%W2       .SET ABORT FLAG                 CALL.342
1361  001006                    EXIT                                                            CALL.343
                                                                                                CALL.344
                                * RESTART DEVICE                                                CALL.345
                                                                                                CALL.346
1362                    RESTART *                                                               CALL.347
1362  072050                    R = DEVICE                      .FIND DEVICE TABLE              CALL.348
1363   <macro>                  STORE R$RSTRT,XD@OPR,R!%W2      .ENTER RESTART FLAG             CALL.349
1373  001006                    EXIT                                                            CALL.350
                                                                                                CALL.351
                                * RESUME DEVICE                                                 CALL.352
                                                                                                CALL.353
```

```
1374                          RESUME   *                                                          CALL.354
1374   072036                          R = DEVICE                    .FIND DEVICE TABLE           CALL.355
1375    <macro>                        STORE R$RSME,XD@OPR,R!%W2     .ENTER RESUME FLAG           CALL.356
1405   001000                          EXIT                                                       CALL.357
                                                                                                  CALL.358
                                       * DISABLE (DOWN) DEVICE                                    CALL.359
                                                                                                  CALL.360
1406                          DISABLE  *                                                          CALL.361
1406   072024                          R = DEVICE                    .FIND DEVICE TABLE           CALL.362
1407    <macro>                        STORE 1,XD@OFF,R!%W2          .SET DISABLED                CALL.363
1420   001000                          EXIT                                                       CALL.364
                                                                                                  CALL.365
                                       * ENABLE (UP) DEVICE                                       CALL.366
                                                                                                  CALL.367
1421                          ENABLE   *                                                          CALL.368
1421   072011                          R = DEVICE                    .FIND DEVICE TABLE           CALL.369
1422    <macro>                        STORE 0,XD@OFF,R!%W2          .CLEAR DISABLE               CALL.370
1431   001000                          EXIT                                                       CALL.371
                                                                                                  CALL.372
                                       *                                                          CALL.373
                                       * FIND DEVICE TABLE ADDRESS                                CALL.374
                                       *                                                          CALL.375
                                                                                                  CALL.376
1432                          DEVICE   *                                                          CALL.377
1432   020430  015000 /177400          R!%W1 = R!IA & O'177400 > D'8 .GET DEVICE DESIGNATOR       CALL.378
       004010  024423
1437    <macro>                        $IF (R!%W1 # '@'R)            .IF ERROR                    CALL.379
                                                                                   .>>>>> 0          $IF      1
1443   014000 /000000  024504            R!ER = EC$PARM              .Set parameter error         CALL.380
1446   075000 /000105                    P = ERR                     .REPORT IT                   CALL.381
1450    <macro>                        $ENDIF                                                     CALL.382
                                                                                   .<<<<< 0          $ENDIF   1
                                                                                                  CALL.383
1450    <macro>                        LSHIFT R!IA,R!IB,D'8          .STRIP DESIGNATOR            CALL.384
1467   020432  015000 /177400          R!%W1 = R!IC & O'177400       .GET HIGH BYTE               CALL.385
       024423
1473   020431  022423  024431          R!IB = R!IB + R!%W1           .SHIFT INTO UNIT NUMBER      CALL.386
                                                                                                  CALL.387
1476   014000 /000000  024423          R!%W1 = EXPDCT                .TABLE POINTERS              CALL.388
1501    <macro>                        $UNTIL ((R!%W1) = 0)            .SCAN TABLE                CALL.389
1503   030423  024424                    R!%W2 = (R!%W1)             .TABLE ADDRESS               CALL.390
1505    <macro>                          $IF (IA = (R!%W2)) AND (IB = (R!%W2 + 1))                CALL.391
```

```
                                                                                  .>>>>> 0          $IF      1
1520   001000                          EXIT                    .FOUND IT                        CALL.392
1521    <macro>                      $ENDIF                                                      CALL.393
                                                                                  .<<<<< 0          $ENDIF   1
1521   026423                        R!%W1 = R!%W1 + 1         .NEXT DESCRIPTOR                 CALL.394
1522    <macro>                    $ENDTIL                                                       CALL.395
1523   014000 /000000  024504      R!ER = EC$PARM              .Set parameter error             CALL.396
1526   075000 /000105              P = ERR                     .REPORT IT                       CALL.397
                                                                                                 CALL.398
        <edit>                     SUBTITLE  '"$OSUB" Subroutines.'                              CALL.399
```

```
                                        * Turn ON or OFF Error channel                                  CALL.401
1530                            ERROR   *                                                               CALL.402
                                                                                                        CALL.403
1530    <macro>                         $IF       (R!IA = 'ON'H)      .If an enable request             CALL.404
                                                                                        .>>>>> 0            $IF      1
1535    <macro>                           STORE   0,ER@FLG,R!%SVPT    .Enable error logging            CALL.405
1542    <macro>                           STORE   0,ER@CNT,R!%SVPT    .Reset error count               CALL.406
1547   147016                             ERA: 7                      .Enable error channel interrupts CALL.407
1550    <macro>                         $ELSEIF   (R!IA = 'OF'H)      .If a disable request            CALL.408
1556    <macro>                           STORE   1,ER@FLG,R!%SVPT    .Disable logging                 CALL.409
1563   146016                             ERA: 6                      .Disable error channel interrupts CALL.410
1564    <macro>                         $ELSE                                                           CALL.411
1565   014000 /000000  024504             R!ER = EC$PARM              .Invalid parameter               CALL.412
1570   075000 /000105                     P = ERR                                                      CALL.413
1572    <macro>                         $ENDIF                                                          CALL.414
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                        CALL.415
1572   001000                          EXIT                                                            CALL.416


                                ***     GBF - Get disk Buffer.                                          CALL.418
                                *       Exit      NA = Buffer address.                                  CALL.419
                                *                 A = 0 if a buffer allocated                           CALL.420
                                                                                                        CALL.421
                                                                                                        CALL.422
1573                            GBF     *                                                               CALL.423
1573    <macro>                         BGET      NA                                                    CALL.424
1603   001000                           EXIT                                                            CALL.425


                                ***     GOC - Get one character from a register buffer.                 CALL.427
                                *       Entry     T0 = Register buffer pointer.                         CALL.428
                                *                 T1 = Character pointer.                               CALL.429
                                *       Exit      T2 = The character.                                   CALL.430
                                                                                                        CALL.431
                                                                                                        CALL.432
1604                            GOC     *                                                               CALL.433
1604   010000  024507                   T2 = 0                                                          CALL.434
```

```
1606   020506  013120  103002          EXIT, T1 = CARL                 . If end of input.                CALL.435
       001000
1612   020506  004001  022505          B = T1 > 1 + T0                 . Parcel address.                 CALL.436
       054000
1616   060000  024507                  T2 = (B)                                                          CALL.437
1620   020506  011001  024410          T2 = T2 > 8, 0 = T1 & 1         . If upper character.             CALL.438
       010000  023410  103004
       020507  004010  024507
                                                                                                         CALL.439
1631   020507  011377  024507          T2 = T2 & O'377                 . Isolate the character.          CALL.440
1634   026506                          T1 = T1 + 1                     . Character pointer incremented.  CALL.441
1635   020507  013012  103003          T2 = 0, T2 = O'012              . If LF.                          CALL.442
       010000  024507
1642   020507  013015  103003          T2 = 0, T2 = O'015              . If CR.                          CALL.443
       010000  024507
1647   020507  013033  103003          T2 = 0, T2 = O'33               .IF ESCAPE                        CALL.444
       010000  024507
1654   001000                          EXIT                                                              CALL.445


                               **       PNL - Process Null Line.                                          CALL.447
                                                                                                         CALL.448
                                                                                                         CALL.449
1655                           PNL      *                                                                 CALL.450
1655    <macro>                         FIND      AMAP,T3,T4,T2         .Get ready to read AMAP           CALL.451
1674   102002  001000                   EXIT, A # 0                     .If AMAP not found                CALL.452
1676   020507  017000 /001000          EXIT, T2 > 512                  .If AMAP larger than a buffer     CALL.453
       100003  102002  001000
1704   073111                          R = GBF                         .Get space for AMAP               CALL.454
1705   102002  001000                   EXIT, A # 0                     .If no buffer allocated           CALL.455
1707    <macro>                         MOSR      T3,T4,NA,T2           .Load AMAP                        CALL.456
                                                                                                         CALL.457
                               *        Display CRT Banner Message.                                       CALL.458
                                                                                                         CALL.459
1727   020500  022000  012013          T1 = NA + R!%MYID + O'13        .Find IOP table                   CALL.460
       024506
1733   030506  022500  012007          T1 = (T1) + NA + 7              .Title address                    CALL.461
       024506
1737   030506  024506                  T1 = (T1)                                                         CALL.462
1741   020506  102025                  P = PNL1, T1 = 0                .If message not available         CALL.463
```

```
1743   020506  022500  024507          T2 = T1 + NA                  .Message location        CALL.464
1746   020507  012003  024510          T3 = T2 + 3                   .Set IOP number          CALL.465
1751   020000  005010  016000          (T3) = R!%MYID < 8 + '0 'R                             CALL.466
      /030040  034510
1756    <macro>                        MSG       T2                  .Post the message        CALL.467
                                                                                              CALL.468
1767                           PNL1    *                                                      CALL.469
1767   072002                          R = RBF                       .Return buffer           CALL.470
1770   001000                          EXIT                                                   CALL.471


                               **      RBF - Return disk Buffer.                              CALL.473
                               *       Entry     NA = Address of buffer to return.            CALL.474
                                                                                              CALL.475
                                                                                              CALL.476
1771                           RBF     *                                                      CALL.477
1771   020500  103002  001000          EXIT, NA = 0                  .If no buffer            CALL.478
1774    <macro>                        BRET      NA                                           CALL.479
2004   001000                          EXIT                                                   CALL.480


                               **      SOC - Store one charcter into a register buffer.       CALL.482
                               *       Entry     T3 = Register buffer pointer.               CALL.483
                               *                 T4 = Character pointer.                      CALL.484
                               *                 T2 = Character.                              CALL.485
                                                                                              CALL.486
                                                                                              CALL.487
2005                           SOC     *                                                      CALL.488
2005   020511  004001  022510          B = T4 > 1 + T3               . Parcel address.        CALL.489
       054000
2011   020511  011001  024410          P = SOC10, 0 # T4&1           . If lower character.    CALL.490
       010000  023410  127000
      /002024
2020   020507  005010  064000          (B) = T2 < 8                  . Upper character.       CALL.491
2023   070004                          P = SOC20                                              CALL.492
                                                                                              CALL.493
2024   060000  022507  064000   SOC10  (B) = (B) + T2                 . Lower character.       CALL.494
2027   026511                   SOC20  T4 = T4 + 1                                            CALL.495
```

```
2030   001000                    EXIT                                          CALL.49c

                                 END                                           CALL.498
```

0X	%ACTIVE		4:23 D	15:22						
0X	%B		1: 4 D	7:18 B	7:24 B	7:41 B	7:42 B	8:10 B	8:15 B	8:16 B
			9: 8	9:28	16:30 B	17: 8 B	18:14 B	20:36 B		
0X	%DEBBA		4:24 D	13:16						
0X	%DEBUG		4:25 D	13: 9	13:13	13:16	13:38	13:41	13:42	
0X	%EX		8:36 D	8:36	8:40 D	8:40	12:10 D	12:10	13:12 D	13:12
			15:23 D	15:23	18:29 D	18:29	19:27 D	19:27	19:33 D	19:33
			20: 5 D	20: 5	20:20 D	20:20				
10	%GBLNUM	$APTEXT	2:35 F	2:35						
400	%GBLREG	$APTEXT	2:34							
0	%KERNEL	$APTEXT	13:23 D	13:23 B	13:39 D	13:39 B				
0X	%MYID		4:26 D	9:24	19:37	20: 3				
0X	%SVPT		4:27 D	18: 6	18: 7	18:10				
0X	DEBSINIT		4:28 D	13:40						
0X	DEBLEN		4:29 D	13:12						
0X	DEBSTART		4:30 D	13:12	13:14					
0X	DEBUGGER		13:23 D	13:23	13:39 D	13:39				
0X	EC$COMER		4:31 D	7:37	9:36					
0X	EC$PARM		4:32 D	16:29	17: 7	18:13				
20	ER@CNT@N	$APTEXT	18: 7 F							
2	ER@CNT@P	$APTEXT	18: 7 F	18: 7						
0	ER@CNT@S	$APTEXT	18: 7 F							
20	ER@FLG@N	$APTEXT	18: 6 F	18:10 F						
1	ER@FLG@P	$APTEXT	18: 6 F	18: 6	18:10 F	18:10				
0	ER@FLG@S	$APTEXT	18: 6 F	18:10 F						
16	ERA	$APTEXT	18: 8	18:11						
0X	EXPDCT		4:33 D	16:39						
32	F$BGET	$APTEXT	18:29							
33	F$BRET	$APTEXT	20:20							
50	F$CALL	$APTEXT	8:36	15:23						
6	F$CLSRW	$APTEXT	15:23							
53	F$FIND	$APTEXT	19:27							
51	F$GOTO	$APTEXT	12:10							
20	F$MESG	$APTEXT	20: 5							
46	F$MOSR	$APTEXT	13:12	19:33						
52	F$RETURN	$APTEXT	8:40							
0	F$SWAIT	$APTEXT	13:12	19:33						
0X	O$AMAP		19:27 D	19:27						
0X	O$ASD		10:12 D	10:12						
0X	O$CALL		1: 4 D	1: 4						
0X	O$CHNTES		10:13 D	10:13						
0X	O$CLEAR		10:14 D	10:14						

0X	O$CONC	10:15	D	10:15
0X	O$CONFIG	10:16	D	10:16
0X	O$COPY	10:17	D	10:17
0X	O$CPSPIN	10:18	D	10:18
0X	O$CRAY	10:19	D	10:19
0X	O$DDUMP	10:20	D	10:20
0X	O$DEL	10:21	D	10:21
0X	O$DELETE	10:22	D	10:22
0X	O$DLDMP	10:23	D	10:23
0X	O$DLOAD	10:24	D	10:24
0X	O$DSTAT	10:25	D	10:25
0X	O$ECHOCP	10:26	D	10:26
0X	O$EDIT	10:27	D	10:27
0X	O$ENDCON	10:28	D	10:28
0X	O$ERRDMP	10:29	D	10:29
0X	O$FDUMP	10:30	D	10:30
0X	O$FLAW	10:31	D	10:31
0X	O$FLOAD	10:32	D	10:32
0X	O$FORMAT	10:33	D	10:33
0X	O$FSCOPY	10:35	D	10:35
0X	O$FSTAT	10:34	D	10:34
0X	O$HELP	10:37	D	10:37
0X	O$HSPTES	10:38	D	10:38
0X	O$IACON	10:39	D	10:39
0X	O$IAIOP	10:40	D	10:40
0X	O$INFORM	8:36	D	8:36
0X	O$INIT	10:41	D	10:41
0X	O$LISTO	10:42	D	10:42
0X	O$LISTP	11: 1	D	11: 1
0X	O$MFCLR	11: 2	D	11: 2
0X	O$MFINIT	11: 3	D	11: 3
0X	O$MOSTFS	11: 4	D	11: 4
0X	O$NSC	11: 5	D	11: 5
0X	O$NSCEND	11: 6	D	11: 6
0X	O$OPSYS	11:18	D	11:18
0X	O$PART	11: 7	D	11: 7
0X	O$PATCH	11: 8	D	11: 8
0X	O$PLOT	11: 9	D	11: 9
0X	O$PROC	11:10	D	11:10
0X	O$PRTAPE	11:11	D	11:11
0X	O$RENAME	11:13	D	11:13
0X	O$SETIME	11:12	D	11:12

0X	O$SSDIES		11:14 D	11:14								
0X	O$START		11:15 D	11:15								
0X	O$STATIO		11:16 D	11:16								
0X	O$STOP		11:17 D	11:17								
0X	O$TRACE		11:19 D	11:19								
0X	O$TTY		11:20 D	11:20								
0X	O$UBTAPE		11:21 D	11:21								
0X	O$WATCH		10:36 D	10:36								
0X	O$XCLS		15:23 D	15:23								
0X	O$XDK		11:22 D	11:22								
0X	O$XMT		11:23 D	11:23								
0X	O$XPR		11:24 D	11:24								
17	OV@NUM@N	$APTEXT	1: 4									
1	OV@TYP@N	$APTEXT	1: 4									
2	R$ABRT	$APTEXT	15:31									
1	R$RESP	$APTEXT	15:13									
4	R$RSME	$APTEXT	16: 3									
3	R$RSTRT	$APTEXT	15:38									
11	SM@FST@N	$APTEXT	1: 4									
7	SM@NUM@N	$APTEXT	1: 4									
0X	XCRT		4:34 D	13:20								
1	XD$READ	$APTEXT	15:19									
10	XD@IPN@N	$APTEXT	15: 7	15: 7 F								
21	XD@IPN@P	$APTEXT	15: 7 F	15: 7								
0	XD@IPN@S	$APTEXT	15: 7	15: 7 F								
377	XD@IRT@M	$APTEXT	15: 8									
10	XD@IRT@N	$APTEXT	15: 8									
21	XD@IRT@P	$APTEXT	15: 8 F	15: 8								
10	XD@IRT@S	$APTEXT	15: 8	15: 8 F								
20	XD@KEY@N	$APTEXT	15:14	15:14 F								
15	XD@KEY@P	$APTEXT	15:14 F	15:14								
0	XD@KEY@S	$APTEXT	15:14	15:14 F								
20	XD@MOD@N	$APTEXT	15:19 F									
14	XD@MOD@P	$APTEXT	15:19 F	15:19								
0	XD@MOD@S	$APTEXT	15:19 F									
17	XD@OFT@M	$APTEXT	16:10 F	16:17 F								
4	XD@OFT@N	$APTEXT	16:10 F	16:10	16:17 F							
13	XD@OFT@P	$APTEXT	16:10 F	16:10	16:17 F	16:17						
10	XD@OFT@S	$APTEXT	16:10 F	16:10	16:17 F							
177417	XD@OFT@X	$APTEXT	16:17									
377	XD@OPR@M	$APTEXT	15:31 F	15:38 F	16: 3 F							
10	XD@OPR@N	$APTEXT	15:11	15:11 F	15:31 F	15:31	15:38 F	15:38	16: 3 F	16: 3		

13	XD@OPR@P	$APIEXT	15:11 F	15:11	15 F	15:31	15:38 F	15:38	16: 3 F	16: 3	
0	XD@OPR@S	$APIEXT	15:11	15:11 F	15:31 F	15:31	15:38 F	15:38	16: 3 F	16: 3	
20	XD@USR@N	$APIEXT	15:22 F								
10	XD@USR@P	$APIEXT	15:22 F	15:22							
0	XD@USR@S	$APIEXT	15:22 F								

SYMBOL QUALIFIER IS CALL

1	$CONST	15:19 D	15:19 F	15:19	15:31 D	15:31 F	15:31	15:38 D	15:38 F	
		15:38	16: 3 D	16: 3 F	16: 3	16:10 D	16:10 F	16:10	16:17 D	
		16:17 F	18: 6 D	18: 6 F	18: 7 D	18: 7 F	18:10 D	18:10 F	18:10	
512	$REGORG	2:34 D	2:35	2:35 D	2:35 F	2:36	2:36 D	2:37	2:37 D	
		2:38	2:38 D	2:39	2:39 D	2:40	2:40 D	2:40 F	2:41	
		2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2	3: 2 D	3: 3	
		3: 3 D	3: 4	3: 4 D	3: 4 F	3: 5	3: 5 D	3: 6	3: 6 D	
		3: 7	3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3: 9 F	3:10	
		3:10 D	3:11	3:11 D	3:12	3:12 D	3:13	3:13 D	3:14	
		3:14 D	3:15	3:15 D	3:16	3:16 D	3:17	3:17 D	3:18	
		3:18 D	3:19	3:19 D	3:20	3:20 D	3:21	3:21 D	3:22	
		3:22 D	3:23	3:23 D	3:24	3:24 D	3:25	3:25 D	3:26	
		3:26 D	3:27	3:27 D	3:28	3:28 D	3:29	3:29 D	3:30	
		3:30 D	3:31	3:31 D	3:32	3:32 D	3:33	3:33 D	3:34	
		3:34 D	3:35	3:35 D	3:36	3:36 D	3:37	3:37 D	3:38	
		3:38 D	3:39	3:39 D	3:40	3:40 D	3:41	3:41 D	3:42	
		3:42 D	4: 1	4: 1 D	4: 2	4: 2 D	4: 3	4: 3 D	4: 4	
		4: 4 D	4: 5	4: 5 D	4: 6	4: 6 D	4: 7	4: 7 D	4: 7 F	
		4: 8	4: 8 D	4: 9	4: 9 D	4:10	4:10 D	4:11	4:11 D	
		4:12	4:12 D	4:13	4:13 D	4:14	4:14 D	4:15	4:15 D	
		4:16	4:16 D	4:17	4:17 D	4:17 F	8:36	8:36 F	8:40	
		12:10	13:12	15:23	15:23 F	18:29	19:27	19:33	20: 5	
		20: 5 F	20:20							
50	%NP	1: 4	4: 7 D							
430	%P	1: 4	3: 9 D	4: 7	8:36 F	8:36	15:23 F	15:23	20: 5 F	
		20: 5								
410	%S1	2:35	2:36 D	9:27	12: 6	13:20	13:21	15: 7	15: 8	
		15:11	15:14	15:19	15:22	15:31	15:38	16: 3	16:10	
		16:17	17: 1	18: 6	18: 7	18:10	19: 6	19: 7	20:35	
		20:36								
411	%S2	2:35	2:37 D							
412	%S3	2:35	2:38 D							
413	%S4	2:35	2:39 D							
414	%S5	2:35	2:40 D							

415	%T1	2:41 D	9:24	15:23	16:34				
416	%T2	2:42 D							
417	%T3	3: 1 D							
420	%T4	3: 2 D							
421	%T5	3: 3 D							
422	%T6	3: 4 D							
423	%W1	3: 5 D	9: 8	9:11	9:13	9:23	9:27	9:33	12: 6
		13:14	13:16	15: 7	15:10	16:26	16:28	16:35	16:37
		16:39	16:40	16:41	17: 5				
424	%W2	3: 6 D	9:13	9:17	9:19	9:23	9:24	9:27	9:28
		9:29	12: 6	12:10	15: 7	15: 8	15:11	15:14	15:19
		15:22	15:23	15:31	15:38	16: 3	16:10	16:17	16:41
		17: 1							
425	%W3	3: 7 D	9:14	9:16	9:20	15: 8	15:10		
426	%W4	3: 8 D							
427	%W5	3: 9 D							
1250	ABORT	10: 2	15: 5 D						
1233	BUGOFF	10: 3	13:36 D						
6	CALL	7: 1 D							
25	CALL20	7:18 D	7:19						
37	CALL40	7:20	7:32 D	8: 1					
61	CALL50	7:33	7:34	8: 8 D					
65	CALL60	8:10 D	8:11						
73	CALL70	8:13 D	8:17						
104	CALL90	8:14	8:24 D						
120	CARL	4:20 D	8:14	19: 1					
225	COMT	9: 8	10: 1 D						
400	CRT	2:35 D	13:20						
1157	DEBUG	10: 4	13: 7 D						
1432	DEVICE	15: 6	15:37	16: 2	16: 9	16:16	16:24 D		
1406	DISABLE	10: 5	16: 8 D						
1421	ENABLE	10: 6	16:15 D						
504	ER	4:12 D	7:11	7:37	8:35	8:36	9:36	16:29	17: 7
		18:13							
105	ERR	7:25	7:38	8:33 D	16:30	17: 8	18:14		
1530	ERROR	10: 7	18: 2 D						
1573	GBF	13:10	18:28 D	19:31					
1604	GOC	7:18	7:42	8:10	8:16	18:41 D			
430	IA	3:10 D	4:19	7: 7	8: 8	12:10	15:18	16:25	16:34
		17: 1	18: 5	18: 9					
462	IAA	3:36 D	12:10						
463	IAB	3:37 D	12:10						

464	IAC	3:38 D	12:10							
465	IAD	3:39 D	12:10							
466	IAE	3:40 D	12:10							
467	IAF	3:41 D	12:10							
470	IAG	3:42 D	12:10							
471	IAH	4: 1 D	12:10							
472	IAI	4: 2 D	12:10							
473	IAJ	4: 3 D	12:10							
474	IAK	4: 4 D	12:10							
475	IAL	4: 5 D	12:10							
476	IAM	4: 6 D	12:10							
477	IAN	4: 7 D	4:19	12:10						
431	IB	3:11 D	12:10	16:34	16:37	17: 1				
432	IC	3:12 D	12:10	16:35						
433	ID	3:13 D	12:10							
434	IE	3:14 D	12:10							
435	IF	3:15 D	12:10							
436	IG	3:16 D	12:10							
437	IH	3:17 D	12:10							
440	II	3:18 D	12:10							
441	IJ	3:19 D	12:10							
442	IK	3:20 D	12:10							
443	IL	3:21 D	12:10							
444	IM	3:22 D	12:10							
445	IN	3:23 D	12:10							
446	IO	3:24 D	12:10							
10	IOP0	6:20 D	10: 2	10: 3	10: 4	10: 5	10: 6	10: 7	10: 8	
		10: 9	10:12	10:13	10:14	10:15	10:16	10:17	10:19	
		10:20	10:21	10:22	10:23	10:24	10:25	10:26	10:27	
		10:28	10:29	10:30	10:31	10:32	10:33	10:34	10:35	
		10:36	10:37	10:38	10:39	10:40	10:41	10:42	11: 1	
		11: 2	11: 3	11: 4	11: 5	11: 6	11: 7	11: 8	11: 9	
		11:10	11:11	11:12	11:13	11:14	11:15	11:16	11:17	
		11:18	11:19	11:20	11:21	11:22	11:23	11:24		
4	IOP1	6:21 D	10: 3	10: 4	10:13	10:16	10:37	11: 8	11:17	
		11:19								
2	IOP2	6:22 D	10: 3	10: 4	10:13	10:16	10:37	11: 8	11:17	
		11:19								
1	IOP3	6:23 D	10: 3	10: 4	10:13	10:16	10:18	10:37	11: 8	
		11:17	11:19							
447	IP	3:25 D	12:10							
450	IQ	3:26 D	12:10							

451	IR	3:27 D	12:10						
452	IS	3:28 D	12:10						
453	IT	3:29 D	12:10						
454	IU	3:30 D	12:10						
455	IV	3:31 D	12:10						
456	IW	3:32 D	12:10						
457	IX	3:33 D	12:10						
460	IY	3:34 D	12:10						
461	IZ	3:35 D	12:10						
17	KC@IOP@M	6:13 D	6:13						
4	KC@IOP@N	6:13 D							
5	KC@IOP@P	6:13 D	9:23						
0	KC@IOP@S	6:13 D							
7777	KC@IOP@X	6:13 D							
6	KC@LEN@P	6:15 D	9:33						
7777	KC@OVL@M	6:14 D	6:14	12: 6					
14	KC@OVL@N	6:14 D	12: 6						
5	KC@OVL@P	6:14 D	12: 6 F	12: 6					
4	KC@OVL@S	6:14 D	12: 6	12: 6 F					
170000	KC@OVL@X	6:14 D							
20	KC@SUB@N	6:12 D	9:27	9:27 F					
4	KC@SUB@P	6:12 D	9:27 F	9:27					
0	KC@SUB@S	6:12 D	9:27	9:27 F					
500	NA	4: 8 D	7: 3	7: 9	9:12	13:12	13:13	13:41	18:29
		19:33	19:37	19:39	20: 1	20:19	20:20		
501	NB	4: 9 D	7: 4						
502	NC	4:10 D	7: 5						
503	ND	4:11 D	7: 6						
216	NEXT	9:17	9:25	9:32 D					
1020	OCALL	10:12 D	10:13 D	10:14 D	10:15 D	10:16 D	10:17 D	10:18 D	10:19 D
		10:20 D	10:21 D	10:22 D	10:23 D	10:24 D	10:25 D	10:26 D	10:27 D
		10:28 D	10:29 D	10:30 D	10:31 D	10:32 D	10:33 D	10:34 D	10:35 D
		10:36 D	10:37 D	10:38 D	10:39 D	10:40 D	10:41 D	10:42 D	11: 1 D
		11: 2 D	11: 3 D	11: 4 D	11: 5 D	11: 6 D	11: 7 D	11: 8 D	11: 9 D
		11:10 D	11:11 D	11:12 D	11:13 D	11:14 D	11:15 D	11:16 D	11:17 D
		11:18 D	11:19 D	11:20 D	11:21 D	11:22 D	11:23 D	11:24 D	12: 5 D
140	PLC	8:25	9: 7 D						
1655	PNL	7:24	19:26 D						
1767	PNL1	19:42	20: 7 D						
1771	RBF	14: 2	20: 8	20:18 D					
50	REGL	4:19 D	4:20						
1362	RESTART	10: 8	15:36 D						

1374	RESUME	10: 9	16: 1 D						
2005	SOC	7:41	8:15	20:32 D					
2024	SOC10	20:36	20:41 D						
2027	SOC20	20:39	20:42 D						
505	I0	4:13 D	7: 7	19: 3					
506	I1	4:14 D	7: 8	19: 1	19: 3	19: 6	19:11	19:37	19:39
		19:41	19:42	20: 1					
507	I2	4:15 D	7:19	7:20	7:33	7:34	8:10	8:11	8:16
		18:42	19: 5	19: 7	19: 8	19:10	19:12	19:13	19:14
		19:15	19:16	19:17	19:27	19:29	19:33	20: 1	20: 2
		20: 5	20:38	20:41					
510	I3	4:16 D	7: 9	8: 8	19:27	19:33	20: 2	20: 4	20:33
511	I4	4:17 D	7:10	7:36	8: 9	8:14	19:27	19:33	20:33
		20:35	20:42						

```
0     <macro>                    OVERLAY   CDEM                                                    CDEM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ***************************  ****************************************   CDEM.5
                    *                                                                     *   CDEM.6
                    *        CDEM processes disk requests (A-packets) and tape requests   *   CDEM.7
                    *        (D-packets) received from the cpu.  Tape requests are sent   *   CDEM.8
                    *        to the XIOP for processing.  Disk requests destined for IOPs *   CDEM.9
                    *        other than the MIOP are sent to the target iop for handling. *   CDEM.10
                    *        The MIOP disk requests, for striped disks and buffer memory  *   CDEM.11
                    *        resident datasets, are processed in this overlay.            *   CDEM.12
                    *                                                                     *   CDEM.13
                    ***********************************************************************   CDEM.14
                                                                                                .

6   <macro>                     REGDEFS   ,,(DAL,IOP,FC,DCB,BMU,BML,NUM,MAXHEAD,MAXSEC,LIST      CDEM.16
                    ,                     ,CPU,CPL,HEAD,LN0,LN1,WDSU,WDSL,SECTOR,OFFSET,COUNT  CDEM.17
                    ,                     ,CYL,EXTRA)                                          CDEM.18
410                 %S1       EQUALS      $REGORG                                                  $REG    3
411                 %S2       EQUALS      $REGORG                                                  $REG    3
412                 %S3       EQUALS      $REGORG                                                  $REG    3
413                 %S4       EQUALS      $REGORG                                                  $REG    3
414                 %S5       EQUALS      $REGORG                                                  $REG    3
415                 %T1       EQUALS      $REGORG                                                  $REG    3
416                 %T2       EQUALS      $REGORG                                                  $REG    3
417                 %T3       EQUALS      $REGORG                                                  $REG    3
420                 %T4       EQUALS      $REGORG                                                  $REG    3
421                 %T5       EQUALS      $REGORG                                                  $REG    3
422                 %T6       EQUALS      $REGORG                                                  $REG    3
423                 %W1       EQUALS      $REGORG                                                  $REG    3
424                 %W2       EQUALS      $REGORG                                                  $REG    3
425                 %W3       EQUALS      $REGORG                                                  $REG    3
426                 %W4       EQUALS      $REGORG                                                  $REG    3
427                 %W5       EQUALS      $REGORG                                                  $REG    3
430                 DAL       EQUALS      $REGORG                                                  $REG    3
431                 IOP       EQUALS      $REGORG                                                  $REG    3
432                 FC        EQUALS      $REGORG                                                  $REG    3
433                 DCB       EQUALS      $REGORG                                                  $REG    3
434                 BMU       EQUALS      $REGORG                                                  $REG    3
435                 BML       EQUALS      $REGORG                                                  $REG    3
436                 NUM       EQUALS      $REGORG                                                  $REG    3
437                 MAXHEAD   EQUALS      $REGORG                                                  $REG    3
440                 MAXSEC    EQUALS      $REGORG                                                  $REG    3
441                 LIST      EQUALS      $REGORG                                                  $REG    3
442                 CPU       EQUALS      $REGORG                                                  $REG    3
```

```
443     CPL     EQUALS   $REGORG                                    $REG    3
444     HEAD    EQUALS   $REGORG                                    $REG    3
445     LNO     EQUALS   $REGORG                                    $REG    3
446     LN1     EQUALS   $REGORG                                    $REG    3
447     WDSU    EQUALS   $REGORG                                    $REG    3
450     WDSL    EQUALS   $REGORG                                    $REG    3
451     SECTOR  EQUALS   $REGORG                                    $REG    3
452     OFFSET  EQUALS   $REGORG                                    $REG    3
453     COUNT   EQUALS   $REGORG                                    $REG    3
454     CYL     EQUALS   $REGORG                                    $REG    3
455     EXTRA   EQUALS   $REGORG                                    $REG    3

                EXT      %LSPI                                   CDEM.20
                EXT      DCCB                                    CDEM.21
                EXT      DS                                      CDEM.22
                EXT      DY                                      CDEM.23
                EXT      DZ                                      CDEM.24
                EXT      EI                                      CDEM.25
```

```
                        *       Get next entry from  put queue and branch per the packet      CDEM.27
                        *       type.                                                          CDEM.28
                                                                                               CDEM.29
 6                      CDEM    *                                                              CDEM.30
 6   002000  000000             I = 0                                                          CDEM.31
10    <macro>                   ADDRESS   R!DZ,CPI@CQ,R!%LSPI  .CDEM input queue              CDEM.32
13    <macro>                   RKERN     HGET                 .Get top entry                 CDEM.33
                                                                                               CDEM.34
15    <macro>                   $IF       (R!DY = 0)           .If queue is empty             CDEM.35
                                                                                .>>>>> 0           $IF      1
20    <macro>                     PKERN   ESWP                 .Wait for a new message        CDEM.36
22    <macro>                   $ENDIF                                                         CDEM.37
                                                                                .<<<<< 0           $ENDIF   1
                                                                                               CDEM.38
22   020000  024430             R!DAL = R!DY                   .Request packet address        CDEM.39
24   003000                     I = 1                                                          CDEM.40
25    <macro>                   GET       R!%W1,DA@DIL,R!DAL   .Packet type                   CDEM.41
                                                                                               CDEM.42
33    <macro>                   $IF       (R!%W1 = RQ$DISK)    .If a disk request             CDEM.43
                                                                                .>>>>> 0           $IF      1
37   070053                       P = DISK                                                     CDEM.44
40    <macro>                   $ELSEIF   (R!%W1 = RQ$BMXO)    .If a tape request             CDEM.45
45   070004                       P = TAPE                                                     CDEM.46
46    <macro>                   $ELSE                          .If illegal packet type        CDEM.47
47    <macro>                     $PUNTIF                                                      CDEM.48
51    <macro>                   $ENDIF                                                         CDEM.49
                                                                                .<<<<< 0           $ENDIF   1
```

```
                                 ***********************************************************************   CDEM.51
                                 *                                                                     *   CDEM.52
                                 *        Tape request                                                 *   CDEM.53
                                 *                                                                     *   CDEM.54
                                 ***********************************************************************   CDEM.55
                                                                                                           CDEM.56
 51                              TAPE     *                                                                CDEM.57
 51   010001  024432                      R!FC = MC$MSTIO                .Function code                    CDEM.58
 53   010003  024431                      R!IOP = XIOP                   .Packet destination               CDEM.59
                                                                                                           CDEM.60
 55                              SEND     *                                                                CDEM.61
 55   072012                              R = TRANSMIT                                                     CDEM.62
 56    <macro>                            RELDAL    R!DAL                .Release the packet               CDEM.63
 66   071060                              P = CDEM                                                         CDEM.64


                                 *        Send packet to destination iop:                                  CDEM.66
                                 *                                                                         CDEM.67
                                 *        R!DAL     Packet address                                         CDEM.68
                                 *        R!IOP     Target iop                                             CDEM.69
                                 *        R!FC      Function code                                          CDEM.70
                                                                                                           CDEM.71
 67                              TRANSMIT *                                                                CDEM.72
 67    <macro>                            PUT       R!FC,DA@IFC,R!DAL    .Enter function code              CDEM.73
 74   002000  000000                      I = 0                                                            CDEM.74
 76   020430  024000                      R!DS = R!DAL                   .Packet address                   CDEM.75
100    <macro>                            RKERN     DDALGETA             .Allocate MOS DAL                 CDEM.76
102    <macro>                            RKERN     DSUM                 .Write packet to MOS              CDEM.77
104   020431  024000                      R!EI = R!IOP                   .Iop number                       CDEM.78
106    <macro>                            RKERN    EMSGIOP               .Send message to destination iop  CDEM.79
110   003000                              I   1                                                            CDEM.80
111   001000                              EXIT                                                             CDEM.81
```

```
                                 **************************  ****************************************   CDEM.83
                                 *                                                                *   CDEM.84
                                 *        Disk i/o request                                        *   CDEM.85
                                 *                                                                *   CDEM.86
                                 ******************************************************************   CDEM.87
                                                                                                     CDEM.88
112                              DISK     *                                                          CDEM.89
112     <macro>                           GET       R!%W1,DL@TYP,R!DAL  .Device type                 CDEM.90
                                                                                                     CDEM.91
120     <macro>                           $IF (R!%W1 # DSK$DD19) AND (R!%W1 # DSK$DD29) .If DCU5   CDEM.92
                                                                                        .>>>>> 0         $IF      1
130     <macro>                             STORE   RQ$DD49,DA@DIL,R!DAL .Change packet type       CDEM.93
140     <macro>                           $ENDIF                                                   CDEM.94
                                                                                        .<<<<< 0         $ENDIF   1
                                                                                                     CDEM.95
140     <macro>                           STORE     1,DA@UNS,R!DAL      .Set number of physical units  CDEM.96
150     <macro>                           GET       R!IOP,DA@IOP,R!DAL  .Target iop                  CDEM.97
                                                                                                     CDEM.98
157     <macro>                           $IF       (A # MIOP)          .If destination not MIOP     CDEM.99
                                                                                        .>>>>> 0         $IF      1
161   010001  024432                        R!FC = MC$MSTIO             .Function code               CDEM.100
163   071106                                P = SEND                                                 CDEM.101
                                                                                                     CDEM.102
164     <macro>                           $ELSE                         .If a logical device         CDEM.103
165     <macro>                             GET     R!%W1,DA@CHN,R!DAL  .Channel number              CDEM.104
                                                                                                     CDEM.105
173     <macro>                             $IF     (R!%W1 < O'20),OR,(R!%W1 > O'37) .If invalid channel  CDEM.106
                                                                                        .>>>>> 1         $IF      1
204   070031                                  P = SOS                                                CDEM.107
205     <macro>                             $ENDIF                                                   CDEM.108
                                                                                        .<<<<< 1         $ENDIF   1
                                                                                                     CDEM.109
205   020423  013020  016000                R!%W1 = R!%W1 - O'20 + DCCB                            CDEM.110
      /000000 024423
212   030423  024433                        R!DCB = (R!%W1)             .DCB address                 CDEM.111
214   102021                                P = SOS, A = 0              .If device not defined       CDEM.112
                                                                                                     CDEM.113
215     <macro>                             GET     R!%W1,LDV@TY,R!DCB  .Get device type             CDEM.114
                                                                                                     CDEM.115
217     <macro>                             $IF     (R!%W1 = LDV$BMR)   .If buffer memory            CDEM.116
                                                                                        .>>>>> 1         $IF      1
223   070111                                  P = BMR                                                CDEM.117
```

```
224    <macro>                    $ELSEIF (R!%W1 = LDV$DSK)   .If a striped disk                CDEM.118
231   070247                        P = BIGDISK                                                  CDEM.119
232    <macro>                    $ELSE                       .If an illegal type               CDEM.120
233    <macro>                      $PUNTIF                                                      CDEM.121
235    <macro>                    $ENDIF                                                         CDEM.122
                                                                          .<<<<< 1                 $ENDIF    1
                                                                                                  CDEM.123
235    <macro>                  $ENDIF                                                           CDEM.124
                                                                          .<<<<< 0                 $ENDIF    1


                          *       Return an error for an invalid request.                         CDEM.126
                                                                                                  CDEM.127
235                       SOS     *                                                               CDEM.128
235    <macro>                    STORE     DAR$PAR,DA@RC,R!DAL .Return status                   CDEM.129
245    <macro>                    GET       R!%W1,DL@TYP,R!DAL  .Device type                     CDEM.130
253    <macro>                    $IF       (R!%W1 # DSK$DD19),AND,(R!%W1 # DSK$DD29) .If DCU5   CDEM.131
                                                                          .>>>>> 0                 $IF       1
263    <macro>                      STORE   RQ$DISK,DA@DIL,R!DAL .Restore packet type             CDEM.132
273    <macro>                    $ENDIF                                                         CDEM.133
                                                                          .<<<<< 0                 $ENDIF    1
273    <macro>                    GET       R!%W1,DA@SID,R!DAL                                   CDEM.134
300    <macro>                    GET       R!%W2,DA@DID,R!DAL                                   CDEM.135
305    <macro>                    PUT       R!%W2,DA@SID,R!DAL  .Swap sid and did                CDEM.136
312    <macro>                    PUT       R!%W1,DA@DID,R!DAL                                   CDEM.137
317    <macro>                    SEND      R!DAL                                                CDEM.138
333   071325                      P = CDEM                                                       CDEM.139
```

```
                                 ***************************************************************** CDEM.141
                                 *                                                               * CDEM.142
                                 *        Buffer memory i/o request                              * CDEM.143
                                 *                                                               * CDEM.144
                                 ***************************************************************** CDEM.145
                                                                                                   CDEM.146
                                 *        Validate parameters and translate the memory address.  CDEM.147
                                 *                                                               CDEM.148
                                 *        Entry:    R!DAL - Request DAL address                  CDEM.149
                                 *                  R!DCB - DCB address for logical device       CDEM.150
                                                                                                   CDEM.151
334                              BMR      *                                                      CDEM.152
334     <macro>                          GET       R!%W1,DA@TM,R!DAL    .Target memory of request CDEM.153
342    107105                            P = SOS, A # TM$CMEM           .Must be Central memory  CDEM.154
                                                                                                   CDEM.155
                                 *        Add the BMR base address to the requested memory address. CDEM.156
                                                                                                   CDEM.157
343     <macro>                          GET       R!BMU,DA@DV0,R!DAL   .Request address         CDEM.158
350     <macro>                          GET       R!BML,DA@DV1,R!DAL                            CDEM.159
355     <macro>                          GET       R!%W1,TM@FW0,R!DCB   .BMR base address        CDEM.160
362     <macro>                          GET       R!%W2,TM@FW1,R!DCB                            CDEM.161
367     <macro>                          DADD      R!BMU,R!BML,R!%W1,R!%W2 .Include the base address CDEM.162
377     <macro>                          PUT       R!BMU,DA@BM0,R!DAL   .Save the address in the DAL CDEM.163
404     <macro>                          PUT       R!BML,DA@BM1,R!DAL                            CDEM.164
                                                                                                   CDEM.165
                                 *        Verify that the request does not extend past the area allo- CDEM.166
                                 *        cated for COS files.                                   CDEM.167
                                                                                                   CDEM.168
411     <macro>                          GET       R!%W1,DA@LN0,R!DAL   .Get the transfer length CDEM.169
416     <macro>                          GET       R!%W2,DA@LN1,R!DAL                            CDEM.170
423     <macro>                          DADD      R!BMU,R!BML,R!%W1,R!%W2 .Calculate ending address CDEM.171
433     <macro>                          GET       R!%W1,TM@LM0,R!DCB   .Limit address           CDEM.172
440     <macro>                          GET       R!%W2,TM@LM1,R!DCB                            CDEM.173
                                                                                                   CDEM.174
445     <macro>                          $IF      (R!BMU > R!%W1)       .If overrun              CDEM.175
                                                                                     .>>>>> 0          $IF      1
453    071216                              P = SOS                                                 CDEM.176
454     <macro>                          $ELSEIF  (R!BMU = R!%W1) AND (R!BML > R!%W2) .If overrun CDEM.177
467    071232                              P = SOS                                                 CDEM.178
470     <macro>                          $ENDIF                                                  CDEM.179
                                                                                     .<<<<< 0          $ENDIF   1
                                                                                                   CDEM.180
```

```
470    <macro>                          GEI       R!IOP,TM@IOP,R!DCB  .Target IOP                    CDEM.181
475    010040   024432                  R!FC = AM$COSIO               .Function code for ACOM         CDEM.182
477    071422                           P = SEND                                                      CDEM.183
```

```
                                 ***************************************************************** CDEM.185
                                 *                                                               * CDEM.186
                                 *        Striped disk i/o request                               * CDEM.187
                                 *                                                               * CDEM.188
                                 ***************************************************************** CDEM.189
                                                                                                   CDEM.190
                                 *        Queue request in device table.                         CDEM.191
                                                                                                   CDEM.192
500                              BIGDISK  *                                                       CDEM.193
500   002000  000000                      I = 0                                                   CDEM.194
502   020430  024000                      R!DY = R!DAL                                            CDEM.195
504    <macro>                            ADDRESS   R!DZ,BD@IQU,R!DCB   .Queue address           CDEM.196
507    <macro>                            RKERN     HPUT                                          CDEM.197
511   003000                              I = 1                                                   CDEM.198
                                                                                                   CDEM.199
                                 *        Set up disk parameters.                                CDEM.200
                                                                                                   CDEM.201
512    <macro>                            GET       R!NUM,BD@NUM,R!DCB  .Count of units in group  CDEM.202
517    <macro>                            PUT       R!NUM,DA@UNS,R!DAL  .Enter in DAL             CDEM.203
532    <macro>                            GET       R!%W1,DA@CHN,R!DAL  .Save logical channel     CDEM.204
540    <macro>                            PUT       R!%W1,DA@LCH,R!DAL                            CDEM.205
553    <macro>                            GET       R!MAXHEAD,BD@HDS,R!DCB .Physical heads per cylinder CDEM.206
560    <macro>                            GET       R!MAXSEC,BD@SPT,R!DCB .Physical sectors per track CDEM.207
565    <macro>                            GET       R!LIST,BD@PUT,R!DCB .Physical unit table      CDEM.208
572    <macro>                            GET       R!CPU,DA@TMO,R!DAL  .Starting address         CDEM.209
601    <macro>                            GET       R!CPL,DA@TM1,R!DAL  .                         CDEM.210
606    <macro>                            GET       R!CYL,DA@CYL,R!DAL  .Cylinder                 CDEM.211
614    <macro>                            GET       R!HEAD,DA@HED,R!DAL .Head                     CDEM.212
622    <macro>                            GET       R!LNO,DA@LNO,R!DAL  .Word count               CDEM.213
627    <macro>                            GET       R!LN1,DA@LN1,R!DAL                            CDEM.214
634   011777  024410  010000              P = SOS, 0 # A & 777          .If not a sector multiple CDEM.215
      023410  107403
641    <macro>                            FLDADD    1,BD@SQN,R!DCB      .Update sequence number   CDEM.216
664   024423                              R!%W1 = A                                               CDEM.217
665    <macro>                            PUT       R!%W1,DA@SQN,R!DAL  .Save in DAL (for responses) CDEM.218
                                                                                                   CDEM.219
                                 *        Compute words per unit.                                CDEM.220
                                                                                                   CDEM.221
700   020445  024423                      R!%W1 = R!LNO                                           CDEM.222
702   020446  024424                      R!%W2 = R!LN1                                           CDEM.223
704    <macro>                            RSHIFT    R!%W1,R!%W2,D'9     .Count of sectors         CDEM.224
723   020423  107467                      P = SOS, R!%W1 # 0            .Assume less than 65k sectors CDEM.225
```

```
 725   010000  024450                R!WDSL = 0                                                  CDEM.226
 727    <macro>                      $UNTIL     (R!%W2 < R!NUM)      .Determine sectors per unit  CDEM.227
 732   020424  023436  024424          R!%W2 = R!%W2 - R!NUM                                    CDEM.228
 735   026450                          R!WDSL = R!WDSL + 1                                      CDEM.229
 736    <macro>                      $ENDTIL                                                     CDEM.230
                                                                                                 CDEM.231
 737    <macro>                      $IF        (R!WDSL = 0)         .Determine count of disks to i/o  CDEM.232
                                                                                  .>>>>> 0          $IF      1
 742   020424  024453                  R!COUNT = R!%W2                .Set to actual               CDEM.233
 744    <macro>                      $ELSE                                                       CDEM.234
 745   020436  024453                  R!COUNT = R!NUM                .All                         CDEM.235
 747    <macro>                      $ENDIF                                                      CDEM.236
                                                                                  .<<<<< 0          $ENDIF   1
 747    <macro>                      PUT        R!COUNT,DA@TRQ,R!DAL .Total responses expected   CDEM.237
 763   020424  024455                R!EXTRA = R!%W2                .Remainder                   CDEM.238
 765   010000  024447                R!WDSU = 0                     .Words per device            CDEM.239
 767    <macro>                      LSHIFT     R!WDSU,R!WDSL,D'9                                CDEM.240
                                                                                                 CDEM.241
                              *      Find starting physical sector and offset into unit table.   CDEM.242
                                                                                                 CDEM.243
1006    <macro>                      GET        R!%W1,DA@SEC,R!DAL   .Logical sector            CDEM.244
1014   010000  024451                R!SECTOR = 0                                                CDEM.245
1016    <macro>                      $UNTIL     (R!%W1 < R!NUM)      .Determine starting sector  CDEM.246
1021   020423  023436  024423          R!%W1 = R!%W1 - R!NUM                                    CDEM.247
1024   026451                          R!SECTOR = R!SECTOR + 1                                  CDEM.248
1025    <macro>                      $ENDTIL                                                     CDEM.249
1026   020423  024452                R!OFFSET = R!%W1               .First disk                  CDEM.250
                                                                                                 CDEM.251
                              *      Spawn off a master DAL for each unit in the group.          CDEM.252
                                                                                                 CDEM.253
1030    <macro>                      $UNTIL (R!COUNT = 0)                                        CDEM.254
                                                                                                 CDEM.255
                              *        Enter iop number and channel.                             CDEM.256
                                                                                                 CDEM.257
1032   020441  022452  024423          R!%W1 = R!LIST + R!OFFSET                                CDEM.258
1035   030423  024423                  R!%W1 = (R!%W1)                .Put entry                   CDEM.259
1037    <macro>                        RGET       R!IOP,PUT@PN,R!%W1   .IOP number                CDEM.260
1043    <macro>                        RGET       R!%W1,PUT@CH,R!%W1   .Channel number            CDEM.261
1046    <macro>                        PUT        R!IOP,DA@IOP,R!DAL   .Enter in DAL              CDEM.262
1062    <macro>                        PUT        R!%W1,DA@CHN,R!DAL                              CDEM.263
                                                                                                 CDEM.264
                              *        Enter cylinder, head, and sector.                         CDEM.265
```

```
                                                                                                  CDEM.266
1075    <macro>                         PUT     R!CYL,DA@CYL,R!DAL  .Cylinder                     CDEM.267
1111    <macro>                         PUT     R!HEAD,DA@HED,R!DAL .Head                         CDEM.268
1124    <macro>                         PUT     R!SECTOR,DA@SEC,R!DAL .Sector                     CDEM.269
                                                                                                  CDEM.270
                                *       Enter length of data area                                 CDEM.271
                                                                                                  CDEM.272
1137    <macro>                         PUT     R!WDSU,DA@LN0,R!DAL .Upper                        CDEM.273
1144    <macro>                         PUT     R!WDSL,DA@LN1,R!DAL .Lower                        CDEM.274
1151    <macro>                         $IF     (R!EXTRA # 0)       .If any remainders            CDEM.275
                                                                                    .>>>>> 0          $IF      1
1154    <macro>                           FLDADD D'512,DA@LN1,R!DAL                               CDEM.276
1162    <macro>                           $IF   (A = 0)             .If overflow                  CDEM.277
                                                                                    .>>>>> 1          $IF      1
1164    <macro>                             FLDADD 1,DA@LN0,R!DAL                                 CDEM.278
1170    <macro>                           $ENDIF                                                  CDEM.279
                                                                                    .<<<<< 1          $ENDIF   1
1170    027455                            R!EXTRA = R!EXTRA - 1     .Decrement remainder          CDEM.280
1171    <macro>                         $ENDIF                                                    CDEM.281
                                                                                    .<<<<< 0          $ENDIF   1
                                                                                                  CDEM.282
                                *       Enter starting cpu address.                               CDEM.283
                                                                                                  CDEM.284
1171    <macro>                         PUT     R!CPU,DA@TM0,R!DAL  .Upper                        CDEM.285
1205    <macro>                         PUT     R!CPL,DA@TM1,R!DAL  .Lower                        CDEM.286
1212    <macro>                         DADD    R!CPU,R!CPL,D'512   .Next                         CDEM.287
                                                                                                  CDEM.288
                                *       Send DAL to target iop.                                   CDEM.289
                                                                                                  CDEM.290
1217    010001  024432                  R!FC = MC$MSTIO             .Function code                CDEM.291
1221    077000 /000067                  R = TRANSMIT                                              CDEM.292
                                                                                                  CDEM.293
                                *       Update pointers.                                          CDEM.294
                                                                                                  CDEM.295
1223    026452                          R!OFFSET = R!OFFSET + 1     .List pointer                 CDEM.296
1224    <macro>                         $IF     (A = R!NUM)         .Check for wrap               CDEM.297
                                                                                    .>>>>> 0          $IF      1
1227    010000  024452                    R!OFFSET = 0                                            CDEM.298
1231    026451                            R!SECTOR = R!SECTOR + 1   .Next sector                  CDEM.299
1232    <macro>                           $IF   (A = R!MAXSEC)      .If next track                CDEM.300
                                                                                    .>>>>> 1          $IF      1
1235    010000  024451                      R!SECTOR = 0            .Reset sector                 CDEM.301
```

```
1237   026444                          R!HEAD = R!HEAD + 1      .Next head                          CDEM.302
1240    <macro>                        $IF (A = R!MAXHEAD)      .If next cylinder                   CDEM.303
                                                                                  .>>>>> 2              $IF      1
1243   010000  024444                    R!HEAD = 0             .Reset head                         CDEM.304
1245   026454                            R!CYL = R!CYL + 1      .Next cylinder                      CDEM.305
1246    <macro>                        $ENDIF                                                        CDEM.306
                                                                                  .<<<<< 2              $ENDIF   1
1246    <macro>                        $ENDIF                                                        CDEM.307
                                                                                  .<<<<< 1              $ENDIF   1
1246    <macro>                        $ENDIF                                                        CDEM.308
                                                                                  .<<<<< 0              $ENDIF   1
1246    <macro>                        FLDADD  1,DA@BLN,R!DAL   .Advance start block number          CDEM.309
                                                                                                      CDEM.310
1273   027453                          R!COUNT = R!COUNT - 1    .Left to do                          CDEM.311
1274    <macro>                       $ENDTIL                                                        CDEM.312
                                                                                                      CDEM.313
                                *      Prepare M-DAL for reply.                                       CDEM.314
                                                                                                      CDEM.315
1275    <macro>                       STORE     0,DA@FRS,R!DAL     .Final responses                  CDEM.316
1306    <macro>                       STORE     0,DA@LUE,R!DAL     .Clear for reported errors        CDEM.317
1315    <macro>                       PUT       R!LN0,DA@LN0,R!DAL .Transfer length                  CDEM.318
1322    <macro>                       PUT       R!LN1,DA@LN1,R!DAL                                   CDEM.319
1327    <macro>                       STORE     MIOP,DA@IOP,R!DAL  .Reset IOP                        CDEM.320
1336    <macro>                       GET       R!%W1,DA@LCH,R!DAL .Channel                          CDEM.321
1344    <macro>                       PUT       R!%W1,DA@CHN,R!DAL                                   CDEM.322
1357    <macro>                       GET       R!%W1,DA@DID,R!DAL .Swap SID and DID                 CDEM.323
1364    <macro>                       GET       R!%W2,DA@SID,R!DAL                                   CDEM.324
1371    <macro>                       PUT       R!%W1,DA@SID,R!DAL                                   CDEM.325
1376    <macro>                       PUT       R!%W2,DA@DID,R!DAL                                   CDEM.326
1403    <macro>                       GET       R!%W1,DA@SIL,R!DAL                                   CDEM.327
1411    <macro>                       $IF       (R!%W1 = RQ$DD49)  .If DD49 request                  CDEM.328
                                                                                  .>>>>> 0              $IF      1
1415    <macro>                        STORE   RQ$DISK,DA@SIL,R!DAL .Change id                       CDEM.329
1425    <macro>                       $ENDIF                                                         CDEM.330
                                                                                  .<<<<< 0              $ENDIF   1
1425   075000 /000006                 P = CDEM                                                       CDEM.331
                                                                                                      CDEM.332
                                      END                                                            CDEM.333
```

0X	%ACTIVE		7:28 D	7:28								
0X	%B		1: 4 D	12:31 B	13:36 B							
0X	%EX		5:13 D	5:13	7:28 D	7:28						
10	%GBLNUM	$APTEXT	2:15 F	2:15								
400	%GBLREG	$APTEXT	2:15									
0	%KERNEL	$APTEXT	4: 7 D	4: 7 B	4:11 D	4:11 B	5:29 D	5:29 B	5:30 D	5:30 B		
			5:32 D	5:32 B	10:13 D	10:13 B						
0X	%LSPI		3:14 D	4: 6								
0X	%PUNT		4:25 D	4:25	7: 4 D	7: 4						
20	AD@RC@N	$APTEXT	7:28	7:28 F								
13	AD@RC@P	$APTEXT	7:28 F	7:28								
0	AD@RC@S	$APTEXT	7:28	7:28 F								
40	AM$COSTO	$APTEXT	9: 2									
20	BD@HDS@N	$APTEXT	10:22	10:22 F								
6	BD@HDS@P	$APTEXT	10:22 F	10:22								
0	BD@HDS@S	$APTEXT	10:22	10:22 F								
3	BD@IQU@P	$APTEXT	10:12									
20	BD@NUM@N	$APTEXT	10:18	10:18 F								
1	BD@NUM@P	$APTEXT	10:18 F	10:18								
0	BD@NUM@S	$APTEXT	10:18	10:18 F								
20	BD@PUT@N	$APTEXT	10:24	10:24 F								
2	BD@PUT@P	$APTEXT	10:24 F	10:24								
0	BD@PUT@S	$APTEXT	10:24	10:24 F								
20	BD@SPI@N	$APTEXT	10:23	10:23 F								
7	BD@SPI@P	$APTEXT	10:23 F	10:23								
0	BD@SPI@S	$APTEXT	10:23	10:23 F								
377	BD@SQN@M	$APTEXT	10:33									
10	BD@SQN@N	$APTEXT	10:33 F	10:33								
10	BD@SQN@P	$APTEXT	10:33 F	10:33								
0	BD@SQN@S	$APTEXT	10:33	10:33 F								
7	CPI@CQ@P	$APTEXT	4: 6									
77777	DA@BLN@M	$APTEXT	13:12									
17	DA@BLN@N	$APTEXT	13:12 F	13:12								
16	DA@BLN@P	$APTEXT	13:12 F	13:12								
1	DA@BLN@S	$APTEXT	13:12	13:12 F								
100000	DA@BLN@X	$APTEXT	13:12									
20	DA@BMO@N	$APTEXT	8:23 F									
34	DA@BMO@P	$APTEXT	8:23 F	8:23								
0	DA@BMO@S	$APTEXT	8:23 F									
20	DA@BM1@N	$APTEXT	8:24 F									
35	DA@BM1@P	$APTEXT	8:24 F	8:24								
0	DA@BM1@S	$APTEXT	8:24 F									

177	DA@CHN@M	$APTEXT	6:25	10:20	11:40	13:25					
7	DA@CHN@N	$APTEXT	6:25	10:20	11:40 F	13:25 F					
27	DA@CHN@P	$APTEXT	6:25 F	6:25	10:20 F	10:20	11:40 F	11:40	13:25 F	13:25	
11	DA@CHN@S	$APTEXT	6:25	6:25 F	10:20	10:20 F	11:40 F	13:25 F			
177600	DA@CHN@X	$APTEXT	11:40	13:25							
3777	DA@CYL@M	$APTEXT	12: 2								
13	DA@CYL@N	$APTEXT	10:27	10:27 F	12: 2 F	12: 2					
30	DA@CYL@P	$APTEXT	10:27 F	10:27	12: 2 F	12: 2					
0	DA@CYL@S	$APTEXT	10:27	10:27 F	12: 2 F	12: 2					
20	DA@DID@N	$APTEXT	7:25	7:25 F	7:27 F	13:26	13:26 F	13:29 F			
10	DA@DID@P	$APTEXT	7:25 F	7:25	7:27 F	7:27	13:26 F	13:26	13:29 F	13:29	
0	DA@DID@S	$APTEXT	7:25	7:25 F	7:27 F	13:26	13:26 F	13:29 F			
377	DA@DIL@M	$APTEXT	4:17	6:12 F	7:21 F						
10	DA@DIL@N	$APTEXT	4:17	6:12 F	7:21 F						
10	DA@DIL@P	$APTEXT	4:17 F	4:17	6:12 F	6:12	7:21 F	7:21			
10	DA@DIL@S	$APTEXT	4:17	4:17 F	6:12 F	7:21 F					
177400	DA@DIL@X	$APTEXT	6:12	7:21							
20	DA@DVO@N	$APTEXT	8:18	8:18 F							
30	DA@DVO@P	$APTEXT	8:18 F	8:18							
0	DA@DVO@S	$APTEXT	8:18	8:18 F							
20	DA@DV1@N	$APTEXT	8:19	8:19 F							
31	DA@DV1@P	$APTEXT	8:19 F	8:19							
0	DA@DV1@S	$APTEXT	8:19	8:19 F							
17	DA@FRS@M	$APTEXT	13:19 F								
4	DA@FRS@N	$APTEXT	13:19 F	13:19							
12	DA@FRS@P	$APTEXT	13:19 F	13:19							
0	DA@FRS@S	$APTEXT	13:19 F	13:19							
37	DA@HED@M	$APTEXT	10:28	12: 3							
5	DA@HED@N	$APTEXT	10:28	12: 3 F							
30	DA@HED@P	$APTEXT	10:28 F	10:28	12: 3 F	12: 3					
13	DA@HED@S	$APTEXT	10:28	10:28 F	12: 3 F						
177740	DA@HED@X	$APTEXT	12: 3								
20	DA@IFC@N	$APTEXT	5:26 F								
1	DA@IFC@P	$APTEXT	5:26 F	5:26							
0	DA@IFC@S	$APTEXT	5:26 F								
3	DA@IOP@M	$APTEXT	6:17	11:39	13:23 F						
2	DA@IOP@N	$APTEXT	6:17	6:17 F	11:39 F	11:39	13:23 F				
27	DA@IOP@P	$APTEXT	6:17 F	6:17	11:39 F	11:39	13:23 F	13:23			
5	DA@IOP@S	$APTEXT	6:17	6:17 F	11:39 F	11:39	13:23 F				
174777	DA@IOP@X	$APTEXT	13:23								
77	DA@LCH@M	$APTEXT	10:21								
6	DA@LCH@N	$APTEXT	10:21 F	10:21	13:24	13:24 F					

21	DA@LCH@P	$APTEXT	10:21 F	10:21	F	13:24				
0	DA@LCH@S	$APTEXT	10:21 F	10:21	13:24	13:24 F				
20	DA@LNO@N	$APTEXT	8:29	8:29 F	10:29	10:29 F	12: 8 F	12:15 F	13:21 F	
32	DA@LNO@P	$APTEXT	8:29 F	8:29	10:29 F	10:29	12: 8 F	12: 8	12:15 F	12:15
			13:21 F	13:21						
0	DA@LNO@S	$APTEXT	8:29	8:29 F	10:29	10:29 F	12: 8 F	13:21 F		
20	DA@LN1@N	$APTEXT	8:30	8:30 F	10:30	10:30 F	12: 9 F	12:12 F	13:22 F	
33	DA@LN1@P	$APTEXT	8:30 F	8:30	10:30 F	10:30	12: 9 F	12: 9	12:12 F	12:12
			13:22 F	13:22						
0	DA@LN1@S	$APTEXT	8:30	8:30 F	10:30	10:30 F	12: 9 F	13:22 F		
17	DA@LUE@M	$APTEXT	13:20 F							
4	DA@LUE@N	$APTEXT	13:20 F							
12	DA@LUE@P	$APTEXT	13:20 F	13:20						
10	DA@LUE@S	$APTEXT	13:20 F							
177417	DA@LUE@X	$APTEXT	13:20							
377	DA@RC@M	$APTEXT	7:17 F							
10	DA@RC@N	$APTEXT	7:17 F							
26	DA@RC@P	$APTEXT	7:17 F	7:17						
10	DA@RC@S	$APTEXT	7:17 F							
177400	DA@RC@X	$APTEXT	7:17							
177	DA@SEC@M	$APTEXT	12: 4							
7	DA@SEC@N	$APTEXT	11:21	11:21 F	12: 4 F	12: 4				
31	DA@SEC@P	$APTEXT	11:21 F	11:21	12: 4 F	12: 4				
0	DA@SEC@S	$APTEXT	11:21	11:21 F	12: 4 F	12: 4				
20	DA@SID@N	$APTEXT	7:24	7:24 F	7:26 F	13:27	13:27 F	13:28 F		
11	DA@SID@P	$APTEXT	7:24 F	7:24	7:26 F	7:26	13:27 F	13:27	13:28 F	13:28
0	DA@SID@S	$APTEXT	7:24	7:24 F	7:26 F	13:27	13:27 F	13:28 F		
377	DA@SIL@M	$APTEXT	13:30	13:33 F						
10	DA@SIL@N	$APTEXT	13:30	13:33 F						
11	DA@SIL@P	$APTEXT	13:30 F	13:30	13:33 F	13:33				
10	DA@SIL@S	$APTEXT	13:30	13:30 F	13:33 F					
177400	DA@SIL@X	$APTEXT	13:33							
377	DA@SQN@M	$APTEXT	10:35							
10	DA@SQN@N	$APTEXT	10:35 F							
20	DA@SQN@P	$APTEXT	10:35 F	10:35						
10	DA@SQN@S	$APTEXT	10:35 F							
177400	DA@SQN@X	$APTEXT	10:35							
37777	DA@TMO@M	$APTEXT	10:25	12:24						
16	DA@TMO@N	$APTEXT	10:25	12:24 F						
24	DA@TMO@P	$APTEXT	10:25 F	10:25	12:24 F	12:24				
2	DA@TMO@S	$APTEXT	10:25	10:25 F	12:24 F					
140000	DA@TMO@X	$APTEXT	12:24							

20	DA@IM1@N	$APTEXT	10:26		10:26	F	12:25	F			
25	DA@IM1@P	$APTEXT	10:26	F	10:26		12:25	F	12:25		
0	DA@IM1@S	$APTEXT	10:26		10:26	F	12:25	F			
2	DA@IM@N	$APTEXT	8:13		8:13	F					
24	DA@IM@P	$APTEXT	8:13	F	8:13						
0	DA@IM@S	$APTEXT	8:13		8:13	F					
17	DA@IRQ@M	$APTEXT	11:14								
4	DA@IRQ@N	$APTEXT	11:14	F	11:14						
20	DA@IRQ@P	$APTEXT	11:14	F	11:14						
4	DA@IRQ@S	$APTEXT	11:14	F	11:14						
17	DA@UNS@M	$APTEXT	6:16	F	10:19						
4	DA@UNS@N	$APTEXT	6:16	F	6:16		10:19	F	10:19		
20	DA@UNS@P	$APTEXT	6:16	F	6:16		10:19	F	10:19		
0	DA@UNS@S	$APTEXT	6:16	F	6:16		10:19	F	10:19		
20	DARSPAR	$APTEXT	7:17								
0X	DCCB		3:15	D	6:33						
0X	DDALGETA		5:29	D	5:29						
5	DL@TYP@N	$APTEXT	6: 8		6: 8	F	7:18		7:18	F	
27	DL@TYP@P	$APTEXT	6: 8	F	6: 8		7:18	F	7:18		
0	DL@TYP@S	$APTEXT	6: 8		6: 8	F	7:18		7:18	F	
0X	DS		3:16	D	5:28						
1	DSK$DD19	$APTEXT	6:11		7:20						
2	DSK$DD29	$APTEXT	6:11		7:20						
0X	DSUM		5:30	D	5:30						
0X	DY		3:17	D	4:10		4:15		10:11		
0X	DZ		3:18	D	4: 6		10:12				
0X	EI		3:19	D	5:31						
0X	EMSGIOP		5:32	D	5:32						
0X	ESWP		4:11	D	4:11						
27	F$RDAL	$APTEXT	5:13								
34	F$SEND	$APTEXT	7:28								
0X	HGET		4: 7	D	4: 7						
0X	HPUT		10:13	D	10:13						
1	LDV$BMR	$APTEXT	6:41								
2	LDV$DSK	$APTEXT	7: 1								
20	LDV@IY@N	$APTEXT	6:38		6:38	F					
0	LDV@IY@P	$APTEXT	6:38	F							
0	LDV@IY@S	$APTEXT	6:38		6:38	F					
1	MC$MSIIO	$APTEXT	5: 8		6:21		12:30				
0	MIOP	$APTEXT	6:20		13:23						
0X	OSCDEM		1: 4	D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4								

1	OV@TYP@N	$APTEXT	1: 4		
0	PTSHUH	$APTEXT	4:25	7: 4	
77	PUT@CH@M	$APTEXT	11:38		
6	PUT@CH@N	$APTEXT	11:38		
12	PUT@CH@S	$APTEXT	11:38	11:38 F	
3	PUT@PN@M	$APTEXT	11:37		
2	PUT@PN@N	$APTEXT	11:37	11:37 F	
10	PUT@PN@S	$APTEXT	11:37	11:37 F	
104	RQ$BMX0	$APTEXT	4:22		
127	RQ$DD49	$APTEXT	6:12	13:32	
101	RQ$DISK	$APTEXT	4:20	7:21	13:33
11	SM@EST@N	$APTEXT	1: 4		
7	SM@NUM@N	$APTEXT	1: 4		
0	TM$CMEM	$APTEXT	8:14		
20	TM@FW0@N	$APTEXT	8:20	8:20 F	
1	TM@FW0@P	$APTEXT	8:20 F	8:20	
0	TM@FW0@S	$APTEXT	8:20	8:20 F	
20	TM@FW1@N	$APTEXT	8:21	8:21 F	
2	TM@FW1@P	$APTEXT	8:21 F	8:21	
0	TM@FW1@S	$APTEXT	8:21	8:21 F	
20	TM@IOP@N	$APTEXT	9: 1	9: 1 F	
5	TM@IOP@P	$APTEXT	9: 1 F	9: 1	
0	TM@IOP@S	$APTEXT	9: 1	9: 1 F	
20	TM@LM0@N	$APTEXT	8:32	8:32 F	
3	TM@LM0@P	$APTEXT	8:32 F	8:32	
0	TM@LM0@S	$APTEXT	8:32	8:32 F	
20	TM@LM1@N	$APTEXT	8:33	8:33 F	
4	TM@LM1@P	$APTEXT	8:33 F	8:33	
0	TM@LM1@S	$APTEXT	8:33	8:33 F	
3	XIOP	$APTEXT	5: 9		

SYMBOL QUALIFIER IS CDEM

101	$CONST	6:12 D	6:12 F	6:12	6:16 D	6:16 F	6:16	7:17 D	7:17 F
		7:17	7:21 D	7:21 F	7:21	13:19 D	13:19 F	13:19	13:20 D
		13:20 F	13:23 D	13:23 F	13:33 D	13:33 F	13:33		
456	$REGORG	2:15 D	2:15 F	2:16	2:16 D	2:17	2:17 D	2:18	2:18 D
		2:19	2:19 D	2:20	2:20 D	2:20 F	2:21	2:21 D	2:22
		2:22 D	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D	2:26
		2:26 D	2:26 F	2:27	2:27 D	2:28	2:28 D	2:29	2:29 D
		2:30	2:30 D	2:31	2:31 D	2:31 F	2:32	2:32 D	2:33
		2:33 D	2:34	2:34 D	2:35	2:35 D	2:36	2:36 D	2:37

		2:37 D	2:38	2:38 D	2:39	2:39 D	2:40	2:40 D	2:41
		2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2	3: 2 D	3: 3
		3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6	3: 6 D	3: 7
		3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3:10	3:10 D	3:11
		3:11 D	3:11 F	5:13	7:28				
0	%NP	1: 4	2:31 D						
430	%P	1: 4	2:31 D	2:31					
410	%S1	2:15	2:16 D	4:17	5:26	6: 8	6:12	6:16	6:17
		6:25	7:17	7:18	7:21	7:24	7:25	7:26	7:27
		7:28	8:13	8:18	8:19	8:20	8:21	8:23	8:24
		8:29	8:30	8:32	8:33	9: 1	10:18	10:19	10:20
		10:21	10:22	10:23	10:24	10:25	10:26	10:27	10:28
		10:29	10:30	10:31	10:35	11:14	11:21	11:39	11:40
		12: 2	12: 3	12: 4	12: 8	12: 9	12:12	12:15	12:24
		12:25	13:19	13:20	13:21	13:22	13:23	13:24	13:25
		13:26	13:27	13:28	13:29	13:30	13:33		
411	%S2	2:15	2:17 D						
412	%S3	2:15	2:18 D						
413	%S4	2:15	2:19 D						
414	%S5	2:15	2:20 D						
415	%T1	2:21 D	10:33	10:41	11:17	13:12			
416	%T2	2:22 D	10:33	13:12					
417	%T3	2:23 D	10:33	13:12					
420	%T4	2:24 D							
421	%T5	2:25 D							
422	%T6	2:26 D							
423	%W1	2:27 D	4:17	4:20	4:22	6: 8	6:11	6:25	6:28
		6:33	6:34	6:35	6:38	6:41	7: 1	7:18	7:20
		7:24	7:27	8:13	8:20	8:22	8:29	8:31	8:32
		8:36	8:38	10:20	10:21	10:34	10:35	10:39	10:41
		10:42	11:21	11:23	11:24	11:27	11:35	11:36	11:37
		11:38	11:40	13:24	13:25	13:26	13:28	13:30	13:32
424	%W2	2:28 D	7:25	7:26	8:21	8:22	8:30	8:31	8:33
		8:38	10:40	10:41	11: 2	11: 3	11: 9	11:15	13:27
		13:29							
425	[illegible]	2:29 D							
426	[illegible]	2:30 D							
427	%W5	2:31 D							
500	BIGDISK	7: 2	10: 9 D						
435	BML	2:37 D	8:19	8:22	8:24	8:31	8:38		
334	BMR	6:42	8:12 D						
434	BMU	2:36 D	8:18	8:22	8:23	8:31	8:36	8:38	

6	CDEM	4: 4 D	5:14	.	13:36				
453	COUNT	3: 9 D	11: 9	11:11	11:14	11:31	13:14		
443	CPI	3: 1 D	10:26	12:25	12:26				
442	CPU	2:42 D	10:25	12:24	12:26				
454	CYL	3:10 D	10:27	12: 2	13: 5				
430	DAL	2:32 D	4:15	4:17	5:13	5:26	5:28	6: 8	6:12
		6:16	6:17	6:25	7:17	7:18	7:21	7:24	7:25
		7:26	7:27	7:28	8:13	8:18	8:19	8:23	8:24
		8:29	8:30	10:11	10:19	10:20	10:21	10:25	10:26
		10:27	10:28	10:29	10:30	10:35	11:14	11:21	11:39
		11:40	12: 2	12: 3	12: 4	12: 8	12: 9	12:12	12:15
		12:24	12:25	13:12	13:19	13:20	13:21	13:22	13:23
		13:24	13:25	13:26	13:27	13:28	13:29	13:30	13:33
433	DCB	2:35 D	6:35	6:38	8:20	8:21	8:32	8:33	9: 1
		10:12	10:18	10:22	10:23	10:24	10:33		
112	DISK	4:21	6: 7 D						
455	EXTRA	3:11 D	11:15	12:11	12:18				
432	FC	2:34 D	5: 8	5:26	6:21	9: 2	12:30		
444	HEAD	3: 2 D	10:28	12: 3	13: 1	13: 4			
431	TOP	2:33 D	5: 9	5:31	6:17	9: 1	11:37	11:39	
441	LIST	2:41 D	10:24	11:35					
445	LN0	3: 3 D	10:29	10:39	13:21				
446	LN1	3: 4 D	10:30	10:40	13:22				
437	MAXHEAD	2:39 D	10:22	13: 3					
440	MAXSEC	2:40 D	10:23	12:41					
436	NUM	2:38 D	10:18	10:19	11: 2	11: 3	11:11	11:23	11:24
		12:37							
452	OFFSET	3: 8 D	11:27	11:35	12:35	12:38			
451	SECTOR	3: 7 D	11:22	11:25	12: 4	12:39	12:42		
55	SEND	5:11 D	6:22	9: 3					
235	SOS	6:29	6:36	7:16 D	8:14	8:37	8:39	10:32	10:42
51	TAPE	4:23	5: 7 D						
67	TRANSMIT	5:12	5:25 D	12:31					
450	WDSL	3: 6 D	11: 1	11: 4	11: 8	11:17	12: 9		
447	WDSU	3: 5 D	11:16	11:17	12: 8				

```
0     <macro>          OVERLAY   CLOCK                                          CLOCK.4

                       COMMENT   'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                                  *****************  ****************************************  CLOCK.5
                                  *                                                        *  CLOCK.6
                                  * CLOCK:                                                 *  CLOCK.7
                                  *   Demon overlay activated at one second intervals.  Used  *  CLOCK.8
                                  *   to deselect disks which are not in use and are still  *  CLOCK.9
                                  *   reserved by the IOP disk software.  Disks which have had *  CLOCK.10
                                  *   no activity in last second are deselected.            *  CLOCK.11
                                  *                                                        *  CLOCK.12
                                  ************************************************************  CLOCK.13
                                                                                                CLOCK.14
6    <macro>                      REGDEFS   ,,(IOP,DA,DB,DC,MA,MB,MC,MD,ME)                   CLOCK.15
                410     %S1       EQUALS    $REGORG                                                $REG    3
                411     %S2       EQUALS    $REGORG                                                $REG    3
                412     %S3       EQUALS    $REGORG                                                $REG    3
                413     %S4       EQUALS    $REGORG                                                $REG    3
                414     %S5       EQUALS    $REGORG                                                $REG    3
                415     %T1       EQUALS    $REGORG                                                $REG    3
                416     %T2       EQUALS    $REGORG                                                $REG    3
                417     %T3       EQUALS    $REGORG                                                $REG    3
                420     %T4       EQUALS    $REGORG                                                $REG    3
                421     %T5       EQUALS    $REGORG                                                $REG    3
                422     %T6       EQUALS    $REGORG                                                $REG    3
                423     %W1       EQUALS    $REGORG                                                $REG    3
                424     %W2       EQUALS    $REGORG                                                $REG    3
                425     %W3       EQUALS    $REGORG                                                $REG    3
                426     %W4       EQUALS    $REGORG                                                $REG    3
                427     %W5       EQUALS    $REGORG                                                $REG    3
                430     IOP       EQUALS    $REGORG                                                $REG    3
                431     DA        EQUALS    $REGORG                                                $REG    3
                432     DB        EQUALS    $REGORG                                                $REG    3
                433     DC        EQUALS    $REGORG                                                $REG    3
                434     MA        EQUALS    $REGORG                                                $REG    3
                435     MB        EQUALS    $REGORG                                                $REG    3
                436     MC        EQUALS    $REGORG                                                $REG    3
                437     MD        EQUALS    $REGORG                                                $REG    3
                440     ME        EQUALS    $REGORG                                                $REG    3

                                  EXT       %CLKL                                               CLOCK.17
                                  EXT       %HSLBZA                                             CLOCK.18
                                  EXT       %HSLBZB                                             CLOCK.19
                                  EXT       %HSUBZA                                             CLOCK.20
```

```
EXT      %HSUBZB                                    CLOCK.21
EXT      %IDLEL                                     CLOCK.22
EXT      %IDLEU                                     CLOCK.23
EXT      %IDLMXL                                    CLOCK.24
EXT      %IDLMXU                                    CLOCK.25
EXT      %LSPO                                      CLOCK.26
EXT      %MSLBZA                                    CLOCK.27
EXT      %MSLBZB                                    CLOCK.28
EXT      %MSUBZA                                    CLOCK.29
EXT      %MSUBZB                                    CLOCK.30
EXT      %MYID                                      CLOCK.31
EXT      %OLDMXL                                    CLOCK.32
EXT      %OLDMXU                                    CLOCK.33
EXT    %TENTHS                                      CLOCK.34
EXT      %TIME                                      CLOCK.35
EXT      D$D4DEM                                  R13739AD.1
EXT      DAY                                        CLOCK.36
EXT      DCCB                                       CLOCK.37
EXT      DEMONS                                   R13739AD.2
EXT      DT                                       R13739AD.3
EXT      EC$TIME                                    CLOCK.38
EXT      EH                                         CLOCK.39
EXT      EI                                         CLOCK.40
EXT      HOUR                                       CLOCK.41
EXT      MATX                                       CLOCK.42
EXT      MINUTE                                     CLOCK.43
EXT      MONTH                                      CLOCK.44
EXT    RTCQUE                                       CLOCK.45
EXT      SECOND                                     CLOCK.46
EXT      SYNCUP                                     CLOCK.47
EXT      YEAR                                       CLOCK.48
```

```
                                        BLOCK TABLE                                                     CLOCK.50
                                ***************************.  ****************************************    CLOCK.51
                                                                                                        CLOCK.52
                                        *                                                               CLOCK.53
                                        * MONTH TABLE                                                   CLOCK.54
                                        *                                                               CLOCK.55
                                                                                                        CLOCK.56
1130                            MTBLE   *                                                               CLOCK.57
1130   000000                           0                                                               CLOCK.58
1131   000040                           D'32                          .JANUARY                          CLOCK.59
1132   000035                           D'29                          .FEBRUARY                         CLOCK.60
1133   000040                           D'32                          .MARCH                            CLOCK.61
1134   000037                           D'31                          .APRIL                            CLOCK.62
1135   000040                           D'32                          .MAY                              CLOCK.63
1136   000037                           D'31                          .JUNE                             CLOCK.64
1137   000040                           D'32                          .JULY                             CLOCK.65
1140   000040                           D'32                          .AUGUST                           CLOCK.66
1141   000037                           D'31                          .SEPTEMBER                        CLOCK.67
1142   000040                           D'32                          .OCTOBER                          CLOCK.68
1143   000037                           D'31                          .NOVEMBER                         CLOCK.69
1144   000040                           D'32                          .DECEMBER                         CLOCK.70
                                                                                                        CLOCK.71
                                        BLOCK *                                                         CLOCK.72
                                ***********************************************************************CLOCK.73
                                                                                                        CLOCK.74
                                                                                                        CLOCK.75
                                        *                                                               CLOCK.76
                                        * CHECK FOR EVENT TIMEOUTS                                      CLOCK.77
                                        *                                                               CLOCK.78
                                                                                                        CLOCK.79
   6   002000  000000                   I = 0                                                           CLOCK.80
  10   014000 /000000  024410           R!MA = (RTCQUE)               .TIMER QUEUE                      CLOCK.81
       030410  024434
  15    <macro>                         $UNTIL (R!MA = RTCQUE)        .UNTIL SCANNED                    CLOCK.82
  21    <macro>                           $PUNTIF (R!MA = 0)          .IF 0 LINK ENCOUNTERED            CLOCK.83
  24    <macro>                           FLDSUB 1,TMR@TM,R!MA        .DECREMENT TIME                   CLOCK.84
  30   024424                             R!%W2 = A                   .SAVE RESULT                      CLOCK.85
  31   020434  024423                     R!%W1 = R!MA                .ENTRY ADDRESS                    CLOCK.86
  33   030434  024434                     R!MA = (R!MA)               .NEXT LINK                        CLOCK.87
  35   020424  112663                     R = TIMEOUT, R!%W2 = 0      .IF TIMEOUT                       CLOCK.88
  37    <macro>                         $ENDTIL                                                         CLOCK.89
  40   003000                           I = 1                                                           CLOCK.90
```

```
                                                                                CLOCK.91
                                 *                                              CLOCK.92
                                 * CHECK FOR 1 SECOND INTERVAL                  CLOCK.93
                                 *                                              CLOCK.94
                                                                                CLOCK.95
 41   026000                     R!%TENTHS = R!%TENTHS + 1     .UPDATE TENTHS OF A SECOND   CLOCK.96
 42    <macro>                   $IF (A = D'10)                .IF 1 SECOND EXPIRED         CLOCK.97
                                                                        .>>>>> 0                $IF      1
 45   010000  024000               R!%TENTHS = 0                 .CLEAR                       CLOCK.98
 47    <macro>                   $ELSE                                                      CLOCK.99
 50    <macro>                     PKERN ESWP                  .RELEASE CONTROL             CLOCK.100
 52    <macro>                   $ENDIF                                                     CLOCK.101
                                                                        .<<<<< 0                $ENDIF   1
                                                                                             CLOCK.102
                                 *                                              CLOCK.103
                                 * MAINTAIN CPU IDLE TIME STATS                 CLOCK.104
                                 *                                              CLOCK.105
                                                                                CLOCK.106
 52    <macro>                   $IF (R!%IDLMXU > R!%IDLEU)    .IF CURRENT MAX IS STILL MAX CLOCK.107
                                                                        .>>>>> 0                $IF      1
 60   070020                       P = STILLBIG                                               CLOCK.108
 61    <macro>                   $ELSEIF (R!%IDLMXU = R!%IDLEU) AND (R!%IDLMXL >= R!%IDLEL) CLOCK.109
 72   070006                       P = STILLBIG                                               CLOCK.110
 73    <macro>                   $ELSE                         .NEW MAX IDLE                CLOCK.111
 74   020000  024000               R!%IDLMXU = R!%IDLEU          .SET                         CLOCK.112
 76   020000  024000               R!%IDLMXL = R!%IDLEL          .                            CLOCK.113
100    <macro>                   $ENDIF                                                     CLOCK.114
                                                                        .<<<<< 0                $ENDIF   1
100                      STILLBIG *                                             CLOCK.115
100   020000  024000               R!%OLDMXU = R!%IDLEU          .SAVE LAST INTERVAL          CLOCK.116
102   020000  024000               R!%OLDMXL = R!%IDLEL          .                            CLOCK.117
104   010000  024000               R!%IDLEU = 0                  .RESET INTERVAL KEEPER       CLOCK.118
106   024000                       R!%IDLEL = A                  .                            CLOCK.119
                                                                                             CLOCK.120
                                 *                                              CLOCK.121
                                 * MAINTAIN HSP/MOS CHANNEL TRANSFERS           CLOCK.122
                                 *                                              CLOCK.123
                                                                                CLOCK.124
107   020000  024000               R!%MSUBZB = R!%MSUBZA         .TRANSFERS FOR LAST SECOND   CLOCK.125
111   020000  024000               R!%MSLBZB = R!%MSLBZA         .                            CLOCK.126
113   010000  024000               R!%MSUBZA = 0                 .RESET FOR NEXT SECOND       CLOCK.127
115   024000                       R!%MSLBZA = A                 .                            CLOCK.128
```

```
116   020000  024000                R!%HSUBZB = R!%HS\          .HSP TRANSFERS LAST SECOND          CLOCK.12
120   020000  024000                R!%HSLBZB = R!%HSLbZA       .                                   CLOCK.130
122   010000  024000                R!%HSUBZA = 0               .RESET FOR NEXT SECOND              CLOCK.131
124   024000                        R!%HSLBZA = A               .                                   CLOCK.132
                                                                                                    CLOCK.133
                                    *                                                               CLOCK.134
                                    * CHECK FOR CPU TIMEOUT                                         CLOCK.135
                                    *  - GET REPORT TO OPERATORS CONSOLE                            CLOCK.136
                                    *                                                               CLOCK.137
                                                                                                    CLOCK.138
125    <macro>                      GET A,CPO@IO,R!%LSPO        .CHECK FOR CHANNEL TIMEOUT          CLOCK.139
130    <macro>                      $IF (A # 0)                 .IF YES                             CLOCK.140
                                                                                .>>>>> 0                $IF      1
132   014000 /177777  024430          R!IOP = O'177777          .INDICATE CPU TIMEOUT               CLOCK.141
135   072543                          R = NOBEAT                .TELL THE WORLD                     CLOCK.142
136    <macro>                        STORE 0,CPO@TO,R!%LSPO    .CLEAR                              CLOCK.143
142    <macro>                      $ENDIF                                                          CLOCK.144
                                                                                .<<<<< 0                $ENDIF   1


                                 *  The MIOP polls each configured IOP every minute.  It sends      CLOCK.146
                                 *  a warning message to the operator if a given IOP does not       CLOCK.147
                                 *  respond in the subsequent one minute interval.  The poll        CLOCK.148
                                 *  bits set by this routine are cleared by the interrupt handler   CLOCK.149
                                 *  when a response is received.                                    CLOCK.150
                                                                                                    CLOCK.151
142    <macro>                      $IF       (R!%MYID = MIOP),AND,((SECOND) = 0) .Every minute     CLOCK.152
                                                                                .>>>>> 0                $IF      1
153   010000  024000                  R!%CLKL = 0               .Reset millisecond clock            CLOCK.153
155   010001  024430                  R!IOP = 1                                                     CLOCK.154
                                                                                                    CLOCK.155
157    <macro>                        $UNTIL  (R!IOP = 4)       .Until all IOPs polled              CLOCK.156
                                                                                                    CLOCK.157
162    <macro>                          $IF   ((R!IOP + MATX) # 0) .If IOP is configured          CLOCK.158
                                                                                .>>>>> 1                $IF      1
171   002000  000000                     I = 0                                                      CLOCK.159
173   020430  024000                     R!EI = R!IOP                                               CLOCK.160
175   010027  024000                     R!EH = M$SYNCH                                             CLOCK.161
177    <macro>                           RKERN EMSGIOP           .Send poll message                 CLOCK.162
201   020430  013001  054000             B = R!IOP - 1                                              CLOCK.163
```

```
204   010001  045000  024423          R!%W1 = 1 < B          .Set mask for given IOP          CLOCK.164
207   014000 /000000  024410          R!%W2 = (SYNCUP)                                        CLOCK.165
      030410  024424
214    <macro>   .                    OR  R!%W1,R!%W2,((SYNCUP)) .Set poll bit for this IOP   CLOCK.166
225   003000                          I = 1                                                   CLOCK.167
226   020423  021424  024410          R = NOBEAT, 0 # R!%W1 & R!%W2 .If no reply to previous  CLOCK.168
      010000  023410  113445
234    <macro>                      $ENDIF                                                    CLOCK.169
                                                                        .<<<<< 1                $ENDIF   1
                                                                                                CLOCK.170
234   026430                          R!IOP = R!IOP + 1                                       CLOCK.171
235    <macro>                    $ENDTIL                                                    CLOCK.172
236    <macro>                  $ENDIF                                                        CLOCK.173
                                                                        .<<<<< 0                $ENDIF   1


                                  *                                                             CLOCK.175
                                  * UPDATE TIME AND DATE                                       CLOCK.176
                                  *                                                             CLOCK.177
                                                                                                CLOCK.178
236    <macro>                    $IF       (R!%MYID = MIOP)    .If MIOP only                 CLOCK.179
                                                                        .>>>>> 0                $IF      1
241   014000 /000000  024410        (SECOND) = (SECOND) + 1     .SECOND                       CLOCK.180
      036410
245    <macro>                      $IF (A = D'60)              .MINUTE?                      CLOCK.181
                                                                        .>>>>> 1                $IF      1
250   014000 /000000  024410          (SECOND) = 0              .CLEAR SECONDS                CLOCK.182
      010000  034410
255   014000 /000000  024410          (MINUTE) = (MINUTE) + 1   .INCREMENT MINUTE             CLOCK.183
      036410
261    <macro>                        $IF (0 = A & 7)           .8 MINUTE INTERVAL            CLOCK.184
                                                                        .>>>>> 2                $IF      1
267    <macro>                          CREATE CLKSNC           .SYNC SYSTEM WITH DAY CLOCK   CLOCK.185
305    <macro>                        $ENDIF                                                   CLOCK.186
                                                                        .<<<<< 2                $ENDIF   1
305   014000 /000000  024410          A = (MINUTE)              .RESTORE A                    CLOCK.187
      030410
311    <macro>                        $IF (A = D'60)            .HOUR?                        CLOCK.188
                                                                        .>>>>> 2                $IF      1
314   014000 /000000  024410          (MINUTE) = 0              .CLEAR MINUTES                CLOCK.189
```

```
        010000  034410
321     014000 /000000  024410          (HOUR) = (HOUR) + 1      .INCREMENT HOUR                    CLOCK.190
        036410
325      <macro>                        $IF (A = D'24)           .DAY?                              CLOCK.191
                                                                                   .>>>>> 3          $IF      1
330     014000 /000000  024410            (HOUR) = 0             .CLEAR HOURS                       CLOCK.192
        010000  034410
335     014000 /000000  024410            (DAY) = (DAY) + 1      .INCREMENT DAY                     CLOCK.193
        036410
341      <macro>                          $IF (A > D'28)         .MONTH?                            CLOCK.194
                                                                                   .>>>>> 4          $IF      1
347     014000 /000000  024410              DC = (DAY)           .SAVE DAY                          CLOCK.195
        030410  024433
354     014000 /000000  024410              DA = MTBLE + (MONTH) + R!%B   .TABLE LOOK UP            CLOCK.196
        014000 /001130  032410
        022000  024431
364     030431  024431                      DA = (DA)            .ENTRY                             CLOCK.197
366      <macro>                            $IF ((MONTH) = 2)    .FEBRUARY?                         CLOCK.198
                                                                                   .>>>>> 5          $IF      1
375     014000 /000000  024410                DB = (YEAR) & 7     .LEAP YEAR?                       CLOCK.199
        030410  011007  024432
403      <macro>                              $IF (DB = 0) OR (DB = 4) .YES                          CLOCK.200
                                                                                   .>>>>> 6          $IF      1
411     026431                                 DA = DA + 1       .LAST DAY +1                        CLOCK.201
412      <macro>                              $ENDIF                                                 CLOCK.202
                                                                                   .<<<<< 6          $ENDIF   1
412      <macro>                            $ENDIF                                                   CLOCK.203
                                                                                   .<<<<< 5          $ENDIF   1
412      <macro>                            $IF (DA = DC)        .IF MONTH CHANGED                  CLOCK.204
                                                                                   .>>>>> 5          $IF      1
416     014000 /000000  024410                (MONTH) = (MONTH) + 1                                  CLOCK.205
        036410
422     014000 /000000  024410                (DAY) = 1          .DAY 1                              CLOCK.206
        010001  034410
427      <macro>                            $ENDIF                                                   CLOCK.207
                                                                                   .<<<<< 5          $ENDIF   1
427      <macro>                            $IF ((MONTH) = D'13) .YEAR?                             CLOCK.208
                                                                                   .>>>>> 5          $IF      1
436     014000 /000000  024410                (MONTH) = 1        .JANUARY                            CLOCK.209
        010001  034410
443     014000 /000000  024410                (YEAR) = (YEAR) + 1                                    CLOCK.210
        036410
```

```
447   014000 /000000  024410                    (YEAR) = 0, (YEAR) = D'100 .FLIP CENTURY          CLOCK.211
      030410  013144  103003
      010000  034410
457    <macro>                                     $ENDIF                                          CLOCK.212
                                                                                     .<<<<< 5          $ENDIF   1
457    <macro>                                    $ENDIF                                           CLOCK.213
                                                                                     .<<<<< 4          $ENDIF   1
457    <macro>                                   $ENDIF                                            CLOCK.214
                                                                                     .<<<<< 3          $ENDIF   1
457    <macro>                                  $ENDIF                                             CLOCK.215
                                                                                     .<<<<< 2          $ENDIF   1
457    <macro>                                 $ENDIF                                              CLOCK.216
                                                                                     .<<<<< 1          $ENDIF   1
                                                                                                     CLOCK.217
                                         * WRITE TIME AND DATE TO MOS - NO WAIT                      CLOCK.218
                                                                                                     CLOCK.219
457    <macro>                           MOSW 0,R!%TIME>2,R!%TIME,2,NOWAIT                           CLOCK.220
500    <macro>                           $ENDIF                                                      CLOCK.221
                                                                                     .<<<<< 0          $ENDIF   1
                                                                                                     CLOCK.222
                                                                                                     CLOCK.223
                                         *                                                           CLOCK.224
                                         * DE-SELECT INACTIVE DISKS                                  CLOCK.225
                                         *                                                           CLOCK.226
                                                                                                     CLOCK.227
500    <macro>                           $IF (R!%MYID = MIOP)          .NOT IN MIOP                  CLOCK.228
                                                                                     .>>>>> 0          $IF      1
503    <macro>                             PKERN ESWP                  .                             CLOCK.229
505    <macro>                           $ENDIF                                                      CLOCK.230
                                                                                     .<<<<< 0          $ENDIF   1
                                                                                                     CLOCK.231
505   014000 /000000  024434             MA = DCCB                   .Disk Control Blocks            CLOCK.232
510                            NXTDK     *                                                           CLOCK.233
510   030434  024435                     MB = (MA)                   .First pointer                  CLOCK.234
512   103013                             P = DONOW, A # 0            .Channel exists                 CLOCK.235
513                            MORE      *                                                           CLOCK.236
513   026434                             MA = MA + 1                 .Next channel                   CLOCK.237
514   014000 /000000  012020             P = NXTDK, MA # DCCB + 20 .End of table?                    CLOCK.238
      024410  020434  023410
      107012
                                                                                                     CLOCK.239
                                         * Exit to idle loop                                         CLOCK.240
```

```
524     <macro>                         PKERN    ESWP          .Exit                                   CLOCK.24
                                                                                                         CLOCK.242
525                             DONOW   *                                                               CLOCK.243
525   030435  107013                    P = MORE, (MB) # 0       .Channel busy, don't deselect          CLOCK.244
                                        ERRIF     DK@SEL@P,NE,DD@CHN  .Keep these fields the same     R13739AD.4
527   020435  012007  024410            MC = (MB + DD@CHN) & 100000  .Check selected bit                CLOCK.245
      030410  015000 /100000
      024436
536   020436  107024                    P = MORE, MC # 0         .Not selected?                         CLOCK.246
540   020435  012007  024410            MC = (MB + DD@CHN) & 40000  .Flag cleared by DK int answ.?      CLOCK.247
      030410  015000 /040000
      024436
547   103010                            P = DSLCT, A # 0         .Any interrupts in last second?        CLOCK.248
550   020435  012007  024410            (MB + DD@CHN) = (MB + DD@CHN) + 40000 .Flag for IDID to clear   CLOCK.249
      014000 /040000  035410
556   071043                            P = MORE                 .Don't deselect                        CLOCK.250
557                             DSLCT   *                                                               CLOCK.251
                                                                                                         CLOCK.252
                                        * Deselect a disk channel                                       CLOCK.253
557   002000  000000                    I = 0                    .                                      CLOCK.254
                                        ERRIF     DK@TYP@P,NE,DB@TYP@P .The following code won't work  R13739AD.5
                                        ERRIF     DK@TYP@S,NE,DB@TYP@S . if these fields are not in    R13739AD.6
                                        ERRIF     DK@TYP@N,NE,DB@TYP@N . the same place                R13739AD.7
561     <macro>                         GET R!MC,DK@TYP,R!MB          .Check type                     R13739AD.8
570   013001  102032                    P = DDREL, A = DSK$DD19       .If DD19                        R13739AD.9
572   020436  013002  102027            P = DDREL, R!MC = DSK$DD29    .If DD29                        R13739AD.10
575   014000 /000000  016000            R!MC = D$D4DEM + DEMONS                                       R13739AD.11
     /000000  024436
602   030436                            A = (R!MC)                                                    R13739AD.12
603     <macro>                         $IF (A # 0)                                                   R13739AD.13
                                                                                  .>>>>> 0                $IF      1
605     <macro>                           STORE 1,DKS@UR,R!MB         .Unit release flag              R13739AD.14
616   020435  024000                      R!DT = R!MB                 .DCB address                    R13739AD.15
620     <macro>                           RKERN D4ACT                 .Activate D4DEM                 R13739AD.16
622     <macro>                         $ENDIF                                                        R13739AD.17
                                                                                  .<<<<< 0                $ENDIF   1
622   071107                            P = MORE                    .Next disk                      R13739AD.18
                                                                                                      R13739AD.19
623                             DDREL   *                                                             R13739AD.20
623   020435  012007  024410            MD = (MB + DD@CHN) & 777  .Get channel                          CLOCK.255
      030410  011777  024437
631   014000 /007777  024440            ME = 7777                 .Loop counter for wait                CLOCK.256
```

```
634   020437  054000                B = MD                    .Channel                                  CLOCK.257
636   166000                        IOB: 6                    .Turn off interrupts                      CLOCK.258
637   010000                        A = 0                     .Deselect                                 CLOCK.259
640   161000                        IOB: 1                    .Send                                     CLOCK.260
641                         DKWAIT  *                                                                   CLOCK.261
641   003000                        I = 1                                                               CLOCK.262
642   012010                        A = A + 10                .Allow ints                               CLOCK.263
643   027440                        ME = ME - 1               .Count down                               CLOCK.264
644   020440  102007                P = DKCLRR, ME = 0        .Clear chanl and go if deselect fails     CLOCK.265
646   002000  000000                I = 0                                                               CLOCK.266
650   020437  054000                B = MD                    .Channel                                  CLOCK.267
652   042000  104012                P = DKWAIT, IOB # DN      .Finished?                                CLOCK.268
654                         DKCLRR  *                                                                   CLOCK.269
654   160000                        IOB: 0                                                              CLOCK.270
655   000000                        PASS                                                                CLOCK.271
656   167000                        IOB: 7                    .Turn on ints                             CLOCK.272
657   020435  012007  024410        (MB + DD@CHN) = (MB +DD@CHN) & 37777 + 100000  .Deselected        CLOCK.273
      030410  015000 /037777
      016000 /100000  034410
670   020435  012001  024410        (MB + DD@NOW) = 177777    .Set to illegal cyl, head                 CLOCK.274
      014000 /177777  034410
676   003000                        I = 1                     .                                         CLOCK.275
677   071164                        P = MORE                  .                                         CLOCK.276
                                                                                                        CLOCK.277
                                                                                                        CLOCK.278
                            ***********************************************************************     CLOCK.279
                                                                                                        CLOCK.280
                                    *                                                                   CLOCK.281
                                    * ACTIVATE NOBEAT ACTIVITY                                         CLOCK.282
                                    * - MD CONTAINS PARAMETER TO PASS                                  CLOCK.283
                                    *                                                                   CLOCK.284
                                                                                                        CLOCK.285
700                         NOBEAT  *                                                                   CLOCK.286
700    <macro>                      CREATE    NOBEAT,(R!IOP)                                           CLOCK.287
720   001000                        EXIT                                                                CLOCK.288
                                                                                                        CLOCK.289
                                    *                                                                   CLOCK.290
                                    * PROCESS TIMEOUT OF AN EVENT                                      CLOCK.291
                                    *                                                                   CLOCK.292
                                                                                                        CLOCK.293
721                         TIMEOUT *                                                                   CLOCK.294
721   020423  024000                R!EH = R!%W1                  .TIMER ENTRY ADDRESS                  CLOCK.295
```

```
 72     <macro>                    UNLINK R!%W1,TMR@      R@BL     .REMOVE FROM QUEUE                    CLOCK.29
 741    <macro>                    STORE 0,TMR@FL,R!%...           .CLEAR LINK                           CLOCK.29
 743    <macro>                    GET R!%W2,TMR@RT,R!%W1          .TIMEOUT HANDLER                      CLOCK.298
                                                                                                          CLOCK.299
                                   ************************                                               CLOCK.300
                                   *        TRACE         *                                               CLOCK.301
                                   *                      *                                               CLOCK.302
                                   * 1. TIMER ENTRY       *                                               CLOCK.303
                                   * 2. TIMEOUT ROUTINE   *                                               CLOCK.304
                                   * 3. 0                 *                                               CLOCK.305
                                   * 4. 0                 *                                               CLOCK.306
                                   * 5. 0                 *                                               CLOCK.307
                                   ************************                                               CLOCK.308
                                                                                                          CLOCK.309
 750    <macro>                    KRECORD TF$TMO,(R!%W1,R!%W2,0,0,0)                                    CLOCK.310
                                                                                                          CLOCK.311
 767    <macro>                    $IF (R!%W2 # 0)                 .IF SPECIFIED                         CLOCK.312
                                                                                   .>>>>> 0                $IF      1
 772    076424                       R = R!%W2                     .LET IT EXECUTE                       CLOCK.313
 773    <macro>                    $ELSE                           .MUST BE AN ACTIVITY                  CLOCK.314
 775    020423  013014  024427       R!%W5 = R!%W1 - AD@P1@P       .ACTIVITY ADDRESS                     CLOCK.315
1000    <macro>                      GET R!%W2,AD@RC,R!%W5         .CHECK FOR EVENT QUEUE                CLOCK.316
1005    <macro>                      STORE EC$TIME,AD@RC,R!%W5     .SET TIMEOUT RESPONSE                 CLOCK.317
                                                                                                          CLOCK.318
                                     * IF ACTIVITY TIMED OUT DO TO A POLL FUNCTION                       CLOCK.319
                                     * THE QUEUE POINTED TO IS A DAL QUEUE RATHER                        CLOCK.320
                                     * THAN AN ACTIVITY QUEUE                                            CLOCK.321
                                                                                                          CLOCK.322
1013    <macro>                      $IF (R!%W2 # 0)               .IF QUEUE TO CHECK                    CLOCK.323
                                                                                   .>>>>> 1                $IF      1
1016    020424  024425                 R!%W3 = R!%W2               .SAVE POINTER TO QUEUE                CLOCK.324
1020    030424  024426                 R!%W4 = (R!%W2)             .FIRST ENTRY ON QUEUE                 CLOCK.325
1022    <macro>                        GET A,AD@FU,R!%W5           .CURRENT FUNCTION                     CLOCK.326
1026    <macro>                        $IF (A # F$POLL)            .IF NOT FROM POLL                     CLOCK.327
                                                                                   .>>>>> 2                $IF      1
1031    <macro>                          $UNTIL (R!%W4 = R!%W5) OR (R!%W4 = 0) .SCAN FOR ACTIVITY        CLOCK.328
1036    020426  024424                     R!%W2 = R!%W4           .SAVE LAST ENTRY                      CLOCK.329
1040    030424  024426                     R!%W4 = (R!%W2)         .NEXT ENTRY                           CLOCK.330
1042    <macro>                          $ENDTIL                                                         CLOCK.331
1043    072040                           R = DELINK              .REMOVE FROM QUEUE                    CLOCK.332
                                                                                                          CLOCK.333
                                         * IF ACTIVITY PUSHED AS A RESULT OF A POLL                      CLOCK.334
```

```
                                                                                                                CLOCK.335
1044    <macro>                          $ELSE                                                                  CLOCK.336
1045    <macro>                            GET R!DA,DA@ACT,R!%W4    .ACTIVITY                                  CLOCK.337
1052    <macro>                            $UNTIL (R!DA = R!%W5) OR (R!%W4 = 0)                                  CLOCK.338
1057    020426  024424                       R!%W2 = R!%W4          .SAVE LAST                                 CLOCK.339
1061    030424  024426                       R!%W4 = (R!%W2)        .NEXT                                      CLOCK.340
1063    <macro>                              GET R!DA,DA@ACT,R!%W4 .ACTIVITY                                   CLOCK.341
1070    <macro>                            $ENDTIL                                                               CLOCK.342
1071    072012                             R = DELINK               .REMOVE FROM QUEUE                         CLOCK.343
1072    020425  012002  024410             (R!%W3 + 2) = (R!%W3 + 2) - 1 .DECREMENT QUEUE COUNT                  CLOCK.344
        037410
1076    <macro>                          $ENDIF                                                                  CLOCK.345
                                                                                                  .<<<<< 2            $ENDIF    1
                                                                                                                CLOCK.346
1076    <macro>                        $ENDIF                                                                    CLOCK.347
                                                                                                  .<<<<< 1            $ENDIF    1
1076    020427  024000                 R!EH = R!%W5                .ACTIVITY                                    CLOCK.348
1100    <macro>                        RKERN EQCP                  .                                            CLOCK.349
1102    <macro>                      $ENDIF                                                                     CLOCK.350
                                                                                                  .<<<<< 0            $ENDIF    1
1102    001000                       EXIT                                                                         CLOCK.351
                                                                                                                CLOCK.352
                                     *                                                                          CLOCK.353
                                     * REMOVE ENTRY FROM QUEUE                                                  CLOCK.354
                                     *                                                                          CLOCK.355
                                                                                                                CLOCK.356
1103                         DELINK  *                                                                          CLOCK.357
1103    <macro>                      $PUNTIF (R!%W4 = 0)         .IF ACTIVITY NOT FOUND                         CLOCK.358
1106    030426  034424               (R!%W2) = (R!%W4)           .DE-LINK ENTRY                                 CLOCK.359
1110    020425  012001  024410       (R!%W3 + 1) = R!%W2, (R!%W3 + 1) = R!%W4 .SET NEW LAST                     CLOCK.360
        030410  023426  103003
        020424  034410
1120    030425  103006  020425       (R!%W3 + 1) = 0, (R!%W3) = 0  .IF QUEUE EMPTY                              CLOCK.361
        012001  024410  010000
        034410
1127    001000                       EXIT                                                                         CLOCK.362
                                                                                                                CLOCK.363
                                     END                                                                        CLOCK.364
```

0X	%B		1: 4 D	8:15	B								
0X	%CLKI		2:39 D	6:31									
0X	%EX		7:35 D	7:35	9:17 D	9:17	11:34 D	11:34					
10	%GBLNUM	$APTEXT	2:11 F	2:11									
400	%GBLREG	$APTEXT	2:11										
0X	%HSLBZA		2:40 D	6: 2	6: 4								
0X	%HSLBZB		2:41 D	6: 2									
0X	%HSUBZA		2:42 D	6: 1	6: 3								
0X	%HSUBZB		3: 1 D	6: 1									
0X	%IDLEL		3: 2 D	5:22	5:26	5:31	5:33						
0X	%IDLEU		3: 3 D	5:20	5:22	5:25	5:30	5:32					
0X	%IDLMXL		3: 4 D	5:22	5:26								
0X	%IDLMXU		3: 5 D	5:20	5:22	5:25							
0	%KERNEL	$APTEXT	5:11 D	5:11 B	6:41 D	6:41 B	9:28 D	9:28 B	10: 1 D	10: 1 B			
			10:34 D	10:34 B	13:18 D	13:18 B							
0X	%LSPO		3: 6 D	6:11	6:16								
0X	%MSLBZA		3: 7 D	5:40	5:42								
0X	%MSLBZB		3: 8 D	5:40									
0X	%MSUBZA		3: 9 D	5:39	5:41								
0X	%MSUBZB		3:10 D	5:39									
0X	%MYID		3:11 D	6:30	7:24	9:27							
0X	%OLDMXL		3:12 D	5:31									
0X	%OLDMXU		3:13 D	5:30									
0X	%PUNT		4:35 D	4:35	13:28 D	13:28							
0X	%TENTHS		3:14 D	5: 6	5: 9								
0X	%TIME		3:15 D	9:17									
0X	%TR1		12:15 D	12:15									
0X	%TR2		12:15 D	12:15									
0X	%TR3		12:15 D	12:15									
0X	%TR4		12:15 D	12:15									
0X	%TR5		12:15 D	12:15									
0X	%TRAC		12:15 D	12:15									
0X	%TVNT		12:15 D	12:15									
177777	AD$NODLM	$APTEXT	7:35	11:34									
20	AD@FU@N	$APTEXT	12:33	12:33 F									
12	AD@FU@P	$APTEXT	12:33 F	12:33									
0	AD@FU@S	$APTEXT	12:33	12:33 F									
14	AD@P1@P	$APTEXT	12:21										
20	AD@RC@N	$APTEXT	12:22	12:22 F	12:23 F								
13	AD@RC@P	$APTEXT	12:22 F	12:22	12:23 F	12:23							
0	AD@RC@S	$APTEXT	12:22	12:22 F	12:23 F								
1	CPO@IO@M	$APTEXT	6:11	6:16 F									

1	CPO@IO@N	$APTEXT	6:11		6:11	F	6:16	F			
0	CPO@IO@P	$APTEXT	6:11	F	6:16	F					
2	CPO@IO@S	$APTEXT	6:11		6:11	F	6:16	F			
157777	CPO@IO@X	$APTEXT	6:16								
0X	D$D4DEM		3:16	D	10:27						
0X	D4ACT		10:34	D	10:34						
20	DA@ACT@N	$APTEXT	13: 3		13: 3	F	13: 7		13: 7	F	
4	DA@ACT@P	$APTEXT	13: 3	F	13: 3		13: 7	F	13: 7		
0	DA@ACT@S	$APTEXT	13: 3		13: 3	F	13: 7		13: 7	F	
0X	DAY		3:17	D	8: 8		8:12		8:33		
5	DB@TYP@N	$APTEXT	10:23	F							
7	DB@TYP@P	$APTEXT	10:21	F							
2	DB@TYP@S	$APTEXT	10:22	F							
0X	DCCB		3:18	D	9:32		9:38				
7	DD@CHN	$APTEXT	10: 5	F	10: 6		10:10		10:14	10:40	11:17
1	DD@NOW	$APTEXT	11:20								
0X	DEMONS		3:19	D	10:27						
7	DK@SEL@P	$APTEXT	10: 5	F							
37	DK@TYP@M	$APTEXT	10:24								
5	DK@TYP@N	$APTEXT	10:23	F	10:24		10:24	F			
7	DK@TYP@P	$APTEXT	10:21	F	10:24	F	10:24				
2	DK@TYP@S	$APTEXT	10:22	F	10:24		10:24	F			
1	DKS@UR@M	$APTEXT	10:32	F							
1	DKS@UR@N	$APTEXT	10:32	F	10:32						
2	DKS@UR@P	$APTEXT	10:32	F	10:32						
15	DKS@UR@S	$APTEXT	10:32	F	10:32						
1	DSK$DD19	$APTEXT	10:25								
2	DSK$DD29	$APTEXT	10:26								
0X	DT		3:20	D	10:33						
0X	EC$TIME		3:21	D	12:23						
0X	EH		3:22	D	6:40		11:42		13:17		
0X	EI		3:23	D	6:39						
0X	EMSGIOP		6:41	D	6:41						
0X	EQCP		13:18	D	13:18						
0X	ESWP		5:11	D	5:11		9:28	D	9:28	10: 1 D	10: 1
55	F$CREATE	$APTEXT	7:35		11:34						
47	F$MOSW	$APTEXT	9:17								
44	F$POLL	$APTEXT	12:35								
1	FS$NWAIT	$APTEXT	9:17								
0X	HOUR		3:24	D	8: 2		8: 6				
27	M$SYNCH	$APTEXT	6:40								
0X	MATX		3:25	D	6:37						

0X	MINUTE		3:26 D	7:31		7:42				
0	MIOP	$APTEXT	6:30	7:24	9:27					
0X	MONTH		3:27 D	8:14	8:19	8:31	8:38	8:39		
0X	O$CLKSNC		7:35 D	7:35						
0X	O$CLOCK		1: 4 D	1: 4						
0X	O$NOBEAT		11:34 D	11:34						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
0	PT$HUH	$APTEXT	4:35	13:28						
0X	RTCQUE		3:28 D	4:32	4:34					
0X	SECOND		3:29 D	6:30	7:25	7:29				
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
0X	SYNCUP		3:30 D	7: 2	7: 4					
47	TFSTMO	$APTEXT	12:15							
20	TMR@BL@N	$APTEXT	12: 1	12: 1 F						
1	TMR@BL@P	$APTEXT	12: 1 F	12: 1						
0	TMR@BL@S	$APTEXT	12: 1	12: 1 F						
20	TMR@FL@N	$APTEXT	12: 1	12: 1 F	12: 2 F					
0	TMR@FL@P	$APTEXT	12: 1 F	12: 2 F						
0	TMR@FL@S	$APTEXT	12: 1	12: 1 F	12: 2 F					
20	TMR@RI@N	$APTEXT	12: 3	12: 3 F						
2	TMR@RI@P	$APTEXT	12: 3 F	12: 3						
0	TMR@RI@S	$APTEXT	12: 3	12: 3 F						
20	TMR@TM@N	$APTEXT	4:36 F							
3	TMR@TM@P	$APTEXT	4:36 F	4:36						
0X	YEAR		3:31 D	8:20	8:41	9: 1				

SYMBOL QUALIFIER IS CLOCK

0X	$CONST	6:16 D	6:16 F	10:32 D	10:32 F	10:32	12: 2 D	12: 2 F	12:23 D	
		12:23 F	12:23							
441	$REGORG	2:11 D	2:11 F	2:12	2:12 D	2:13	2:13 D	2:14	2:14 D	
		2:15	2:15 D	2:16	2:16 D	2:16 F	2:17	2:17 D	2:18	
		2:18 D	2:19	2:19 D	2:20	2:20 D	2:21	2:21 D	2:22	
		2:22 D	2:22 F	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D	
		2:26	2:26 D	2:27	2:27 D	2:27 F	2:28	2:28 D	2:29	
		2:29 D	2:30	2:30 D	2:31	2:31 D	2:32	2:32 D	2:33	
		2:33 D	2:34	2:34 D	2:35	2:35 D	2:36	2:36 D	2:36 F	
		7:35	9:17	11:34						
0	%NP	1: 4	2:27 D							
430	%P	1: 4	2:27 D	2:27						

410	%S1	2:11	2:12 D	4:32	4:36	6:30	6:37	7: 2	7: 4
		7: 6	7: 7	7:25	7:29	7:30	7:31	7:34	7:38
		7:42	8: 1	8: 2	8: 6	8: 7	8: 8	8:12	8:14
		8:15	8:19	8:20	8:31	8:33	8:34	8:38	8:39
		8:40	8:41	9: 1	9: 3	9:38	9:39	10: 6	10:10
		10:14	10:15	10:24	10:32	10:40	11:17	11:19	11:20
		11:21	12: 1	12: 3	12:22	12:23	12:33	13: 3	13: 7
		13:10	13:30	13:32	13:34				
411	%S2	2:11	2:13 D						
412	%S3	2:11	2:14 D						
413	%S4	2:11	2:15 D						
414	%S5	2:11	2:16 D						
415	%T1	2:17 D	12: 1						
416	%T2	2:18 D	12: 1						
417	%T3	2:19 D							
420	%T4	2:20 D							
421	%T5	2:21 D							
422	%T6	2:22 D							
423	%W1	2:23 D	4:38	7: 1	7: 4	7: 6	11:42	12: 1	12: 2
		12: 3	12:15	12:21					
424	%W2	2:24 D	4:37	4:40	7: 3	7: 4	7: 6	12: 3	12:15
		12:18	12:19	12:22	12:30	12:31	12:32	12:37	12:38
		13: 5	13: 6	13:29	13:31				
425	%W3	2:25 D	12:31	13:10	13:30	13:33			
426	%W4	2:26 D	12:32	12:36	12:37	12:38	13: 3	13: 4	13: 5
		13: 6	13: 7	13:28	13:29	13:31			
427	%W5	2:27 D	12:21	12:22	12:23	12:33	12:36	13: 4	13:17
431	DA	2:29 D	8:16	8:17	8:24	8:30	13: 3	13: 4	13: 7
432	DB	2:30 D	8:21	8:23					
433	DC	2:31 D	8:13	8:30					
623	DDREL	10:25	10:26	10:39 D					
1103	DELINk	12:40	13: 9	13:27 D					
654	DKCLRR	11: 9	11:13 D						
641	DKWAIT	11: 5 D	11:12						
525	DONOW	9:35	10: 3 D						
557	DSLCT	10:13	10:17 D						
430	IOP	2:28 D	6:14	6:32	6:34	6:37	6:39	6:42	7:11
		11:34							
434	MA	2:32 D	4:33	4:34	4:35	4:36	4:38	4:39	9:32
		9:34	9:37	9:39					
435	MB	2:33 D	9:34	10: 4	10: 6	10:10	10:14	10:24	10:32
		10:33	10:40	11:17	11:20				

436	MC	2:34 D	10: 7	1	10:11	10:24	10:26	10:28	10:29
437	MD	2:35 D	10:41	11: 1	11:11				
440	ME	2:36 D	10:42	11: 8	11: 9				
513	MORE	9:36 D	10: 4	10: 9	10:16	10:37	11:23		
1130	MTBLE	4: 8 D	8:15						
700	NOBEAT	6:15	7: 7	11:33 D					
510	NXTDK	9:33 D	9:39						
100	STILLBIG	5:21	5:23	5:29 D					
721	TIMEOUT	4:40	11:41 D						

```
0     <macro>                    OVERLAY   CRAY                                                    CRAY.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                                 *************************  ****************************************  CRAY.5
                                 *  THE CRAY OVERLAY IS CALLED TO ESTABLISH COMMUNICATION BETWEEN THE   *  CRAY.6
                                 *  THE IOP AND THE MAINFRAME. 2 PACKETS CONTAINING INITIALIZTION IDS   *  CRAY.7
                                 *  MUST BE SUCCESSFULLY LOOPED THROUGH THE MAINFRAME BEFORE THE        *  CRAY.8
                                 *  INTERFACE IS CONSIDERED INITITIALIZED.                              *  CRAY.9
                                 ************************************************************************  CRAY.10
                                                                                                          CRAY.11
                                          EXT     %LSPI                                                   CRAY.12
                                          EXT     %LSPO                                                   CRAY.13
                                          EXT        DY                                                   CRAY.14
                                                                                                          CRAY.15
                                          BLOCK MESSAGE                                                   CRAY.16
                                 ************************************************************************  CRAY.17
                                                                                                          CRAY.18
1170                             MSG1     *                                                               CRAY.19
1170                                      'CPU <-> MIOP CHANNEL INIT'                                     CRAY.20
1205                             MSG2     *                                                               CRAY.21
1205                                      'CPU <-> MIOP LINKAGE COMPLETE'                                 CRAY.22
1224                             MSG3     *                                                               CRAY.23
1224                                      'CPU <-> MIOP LINKAGE FAILED'                                   CRAY.24
1242                             MSG4     *                                                               CRAY.25
1242                                      'CPU NOT RESPONDING - RETRY?'                                   CRAY.26
1260                             MSG5     *                                                               CRAY.27
1260                                      'LOW SPEED CHANNEL DATA ERROR'                                  CRAY.28
1277                             MSG6     *                                                               CRAY.29
1277                                      '     DATA EXPECTED:'                                           CRAY.30
1311                             MSG7     *                                                               CRAY.31
1311                                      '     DATA RECEIVED:'                                           CRAY.32
1323                             MSG8     *                                                               CRAY.33
1323                                      'CONTINUE?'                                                     CRAY.34
                                                                                                          CRAY.35
                                          * CONSTANTS                                                     CRAY.36
                                                                                                          CRAY.37
                        32       MSG1L    EQUALS    D'26                                                  CRAY.38
                        36       MSG2L    EQUALS    D'30                                                  CRAY.39
                        34       MSG3L    EQUALS    D'28                                                  CRAY.40
                        34       MSG4L    EQUALS    D'28                                                  CRAY.41
                        35       MSG5L    EQUALS    D'29                                                  CRAY.42
                        23       MSG6L    EQUALS    D'19                                                  CRAY.43
                        23       MSG7L    EQUALS    D'19                                                  CRAY.44
                        12       MSG8L    EQUALS    D'10                                                  CRAY.45
                                                                                                          CRAY.46
```

```
                 177777     INTERNAL EQUALS   O'177777            .INTERNAL REQUEST CODE          CRAY.47
                     25     DTOFF    EQUALS   MSG6L+2             .OFFSET FOR DATA                CRAY.48
                                                                                                  CRAY.49
                     24     MSGSIZ   EQUALS   D'20                .MESSAGE BUFFER                 CRAY.50
                      4     MSGRSIZ  EQUALS   4                   .RESPONSE BUFFER                CRAY.51
                     30     MEMSIZ   EQUALS   MSGSIZ+MSGRSIZ      .MEMORY REQUIREMENT             CRAY.52
                                                                                                  CRAY.53
                     50     $$X      SET      MSGSIZ*2            .BYTE LENGTH                    CRAY.54
                     50     MSGCHR   EQUALS   $$X                                                 CRAY.55
                                                                                                  CRAY.56
                            ************************************************************************CRAY.57
                                    BLOCK *                                                       CRAY.58
                                                                                                  CRAY.59
6     <macro>                       REGDEFS ,(SOURCE),(MSG,MSGR,DAL,RESP,EXP,REC,SID,FAIL)        CRAY.60
                    410     %S1      EQUALS   $REGORG                                                $REG    3
                    411     %S2      EQUALS   $REGORG                                                $REG    3
                    412     %S3      EQUALS   $REGORG                                                $REG    3
                    413     %S4      EQUALS   $REGORG                                                $REG    3
                    414     %S5      EQUALS   $REGORG                                                $REG    3
                    415     %T1      EQUALS   $REGORG                                                $REG    3
                    416     %T2      EQUALS   $REGORG                                                $REG    3
                    417     %T3      EQUALS   $REGORG                                                $REG    3
                    420     %T4      EQUALS   $REGORG                                                $REG    3
                    421     %T5      EQUALS   $REGORG                                                $REG    3
                    422     %T6      EQUALS   $REGORG                                                $REG    3
                    423     %W1      EQUALS   $REGORG                                                $REG    3
                    424     %W2      EQUALS   $REGORG                                                $REG    3
                    425     %W3      EQUALS   $REGORG                                                $REG    3
                    426     %W4      EQUALS   $REGORG                                                $REG    3
                    427     %W5      EQUALS   $REGORG                                                $REG    3
                    430     SOURCE   EQUALS   $REGORG                                                $REG    3
                    431     MSG      EQUALS   $REGORG                                                $REG    3
                    432     MSGR     EQUALS   $REGORG                                                $REG    3
                    433     DAL      EQUALS   $REGORG                                                $REG    3
                    434     RESP     EQUALS   $REGORG                                                $REG    3
                    435     EXP      EQUALS   $REGORG                                                $REG    3
                    436     REC      EQUALS   $REGORG                                                $REG    3
                    437     SID      EQUALS   $REGORG                                                $REG    3
                    440     FAIL     EQUALS   $REGORG                                                $REG    3
                                                                                                  CRAY.61
                                    *                                                             CRAY.62
                                    * TEST FOR INITIALIZATION IN PROGRESS                         CRAY.63
```

```
                                  *                                                                  CRAY.64
                                                                                                     CRAY.65
 6   010000  024440               R!FAIL = 0                  .CLEAR FAIL FLAG                       CRAY.66
10   010000  024431               R!MSG = 0                   .CLEAR MEMORY ADDRESS                  CRAY.67
12   010000  024433               R!DAL = 0                   .DAL ADDRESS                           CRAY.68
14    <macro>                     GET A,CPO@LK,R!%LSPO        .LOCK                                  CRAY.69
17   103402                       P = DONEO, A # 0            .IF INITIALIZATION IN PROGRESS         CRAY.70
                                                                                                     CRAY.71
                                  *                                                                  CRAY.72
                                  * ALLOCATE RESOURSES                                               CRAY.73
                                  *                                                                  CRAY.74
                                                                                                     CRAY.75
20    <macro>                     GETMEM MEMSIZ,R!MSG         .REQUEST                               CRAY.76
40   103325                       P = DONE, A # 0             .IF NONE                               CRAY.77
41   020431  012024  024432       R!MSGR = R!MSG + MSGSIZ     .MESSAGE RESPONSE BUFFER               CRAY.78
44   072735                       R = CLEAR                   .CLEAR MESSAGE BUFFER                  CRAY.79
45    <macro>                     GETDAL R!DAL                .ALLOCATE DAL                          CRAY.80
55   103310                       P = DONE, A # 0             .IF NONE                               CRAY.81
                                                                                                     CRAY.82
                                  *                                                                  CRAY.83
                                  * SEE IF ALREADY INITIALIZED                                       CRAY.84
                                  *                                                                  CRAY.85
                                                                                                     CRAY.86
56    <macro>                     GET A,CPO@ON,R!%LSPO        .CHECK CHANNEL ON                      CRAY.87
60    <macro>                     $IF (A # 0)                 .IF SO                                 CRAY.88
                                                                                  .>>>>> 0               $IF      1
62    <macro>                       $IF (R!SOURCE = INTERNAL) .IF INTERNAL REQUEST                   CRAY.89
                                                                                  .>>>>> 1               $IF      1
67   070276                         P = DONE                  .RETURN                                CRAY.90
70    <macro>                       $ELSE                                                            CRAY.91
71   072634                         R = SUCCESS               .SEND MESSAGE TO CONSOLE               CRAY.92
72   070273                         P = DONE                  .RETURN                                CRAY.93
73    <macro>                       $ENDIF                                                           CRAY.94
                                                                                  .<<<<< 1               $ENDIF   1
73    <macro>                     $ENDIF                                                             CRAY.95
                                                                                  .<<<<< 0               $ENDIF   1
                                                                                                     CRAY.96
                                  *                                                                  CRAY.97
                                  * SET UP                                                           CRAY.98
                                  *                                                                  CRAY.99
                                                                                                     CRAY.100
73    <macro>                     STORE 1,CPO@LK,R!%LSPO      .SET LOCK BIT                          CRAY.101
```

```
101   010001  024423          R!%W1 = 1                    .WAIT IF DEADSTART GOING          CRAY.102
103    <macro>                $UNTIL (R!%W1 = 0)                                             CRAY.103
105    <macro>                  GET R!%W1,CPO@DS,R!%LSPO   .DEADSTART FLAG                   CRAY.104
111    <macro>                  $IF (A # 0)                .IF SET                            CRAY.105
                                                                          .>>>>> 0                $IF      1
113    <macro>                    PAUSE 1                  .WAIT                              CRAY.106
124    <macro>                  $ENDIF                                                       CRAY.107
                                                                          .<<<<< 0                $ENDIF   1
124    <macro>                $ENDTIL                                                        CRAY.108
                                                                                              CRAY.109
                              *                                                               CRAY.110
                              * OUTPUT INITIALIZATION IN PROGRESS                             CRAY.111
                              *                                                               CRAY.112
                                                                                              CRAY.113
125   072567                  R = STARTED                                                     CRAY.114
                                                                                              CRAY.115
                              *                                                               CRAY.116
                              * CLEAR CHANNELS                                                CRAY.117
                              *                                                               CRAY.118
                                                                                              CRAY.119
126                     START *                                                               CRAY.120
126    <macro>                GET B,CPO@CN,R!%LSPO         .OUTPUT CHANNEL                    CRAY.121
131   166000                  IOB:6                        .DISABLE INTERRUPTS                CRAY.122
132   163000                  IOB:3                        .CLEAR ERROR FLAGS                 CRAY.123
133   010000                  A = 0                        .CLEAR CONTROL SIGNALS             CRAY.124
134   164000                  IOB:4                        .                                  CRAY.125
135   160000                  IOB:0                        .CLEAR CHANNEL                     CRAY.126
136   000000                  PASS                                                            CRAY.127
137   167000                  IOB:7                        .ENABLE INTERRUPTS                 CRAY.128
140    <macro>                GET B,CPI@CN,R!%LSPI         .INPUT CHANNEL                     CRAY.129
143   160000                  IOB:0                        .CLEAR CHANNEL                     CRAY.130
144   000000                  PASS                                                            CRAY.131
                                                                                              CRAY.132
                              * INPUT ANY LEFT OVER DATA OFF CHANNEL                          CRAY.133
                                                                                              CRAY.134
145   163000                  IOB:3                        .CLEAR ERROR FLAGS                 CRAY.135
146   166000                  IOB:6                        .DISABLE INTERRUPTS                CRAY.136
147   010001  024423          R!%W1 = 1                    .STATUS                            CRAY.137
151    <macro>                $UNTIL (R!%W1 = 0)                                             CRAY.138
153   010004                    A = 4                      .ONE CRAY WORD                     CRAY.139
154   162000                    IOB:2                      .                                  CRAY.140
155   020432                    A = R!MSGR                 .LOCATION                          CRAY.141
```

```
156   061000                    IOB:1                          .TRANSFER                                CRAY.142
157   010001  024424            R!%W2 = 1                      .COUNTER                                 CRAY.143
161   <macro>                   $UNTIL (R!%W2 = 0) OR (IOB = DN)                                        CRAY.144
165   026424                      R!%W2 = R!%W2 + 1            .COUNT                                   CRAY.145
166   <macro>                   $ENDTIL                                                                 CRAY.146
167   <macro>                   $IF (R!%W2 = 0)                .IF TIMED OUT - CHANNEL CLEAN            CRAY.147
                                                                                        .>>>>> 0            $IF      1
172   010000  024423              R!%W1 = 0                    .TERMINATE                               CRAY.148
174   <macro>                   $ENDIF                                                                  CRAY.149
                                                                                        .<<<<< 0            $ENDIF   1
174   <macro>                 $ENDTIL                                                                   CRAY.150
175   160000                  IOB:0                            .CLEAR                                   CRAY.151
176   <macro>                 STORE 0,CPI@ON,R!%LSPI           .SET CHANNEL OFF                         CRAY.152
204   <macro>                 GET R!%W1,CPI@DA,R!%LSPI         .CHECK FOR INPUT DAL                     CRAY.153
211   <macro>                 $IF (A # 0)                                                               CRAY.154
                                                                                        .>>>>> 0            $IF      1
213   <macro>                   RELDAL R!%W1                   .RETURN DAL                              CRAY.155
223   <macro>                   STORE 0,CPI@DA,R!%LSPI         .CLEAR                                   CRAY.156
230   <macro>                 $ENDIF                                                                    CRAY.157
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                          CRAY.158
                                                                                                          CRAY.159
                              *                                                                           CRAY.160
                              * SEND INITIAL PACKET                                                      CRAY.161
                              *                                                                           CRAY.162
                                                                                                          CRAY.163
230   <macro>                 CLEAR START=R!DAL,COUNT=DA@@LH+DA@@LE .0 FILL DAL                         CRAY.164
244   010111  024437          R!SID = RQ$INIT1                 .SOURCE ID                               CRAY.165
246   072161                  R = POLL                         .SEND TO CPU                             CRAY.166
247   <macro>                 $IF (A # 0)                      .IF TIMEOUT                              CRAY.167
                                                                                        .>>>>> 0            $IF      1
251   072306                    R = NORESP                     .SEND MESSAGE                            CRAY.168
252   071124                    P = START                      .IF REQUEST TO RETRY                     CRAY.169
253   <macro>                 $ELSE                                                                     CRAY.170
                                                                                                          CRAY.171
                                * VALIDATE PACKET                                                         CRAY.172
                                                                                                          CRAY.173
254   072204                    R = VALIDATE                   .VALIDATE THE PACKET                     CRAY.174
255   <macro>                 $ENDIF                                                                    CRAY.175
                                                                                        .<<<<< 0            $ENDIF   1
                                                                                                          CRAY.176
                              *                                                                           CRAY.177
```

```
                                                                                CRAY.178
                                 * SEND 2ND INITIALIZATION PACKET               CRAY.179
                                 *                                              CRAY.180
                                                                                CRAY.181
                                 * FILL PACKET WITH ALL-ONES                    CRAY.182
                                                                                CRAY.183
255   020433  012010  024423     R!%W1 = R!DAL + DA@@LH        .START           CRAY.184
260   020423  012030  024424     R!%W2 = R!%W1 + DA@@LE        .END             CRAY.185
263    <macro>                   $UNTIL (R!%W1 = R!%W2)        .UNTIL FILLED    CRAY.186
266   014000 /177777  034423       (R!%W1) = 177777            .FILL            CRAY.187
271   026423                       R!%W1 = R!%W1 + 1           .                CRAY.188
272    <macro>                   $ENDTIL                                        CRAY.189
273   010112  024437             R!SID = RQ$INIT2              .SOURCE ID       CRAY.190
275   072132                     R = POLL                      .ISSUE           CRAY.191
276    <macro>                   $IF (A # 0)                   .IF TIMED OUT    CRAY.192
                                                                      .>>>>> 0      $IF      1
300   072257                       R = NORESP                  .SEND MESSAGE    CRAY.193
301   071153                       P = START                   .IF RETRY REQUESTED CRAY.194
302    <macro>                   $ELSE                                          CRAY.195
                                                                                CRAY.196
                                   * VALIDATE PACKET                            CRAY.197
                                                                                CRAY.198
303   072155                       R = VALIDATE                .                CRAY.199
304    <macro>                   $ENDIF                                         CRAY.200
                                                                      .<<<<< 0      $ENDIF   1
                                                                                CRAY.201
                                 *                                              CRAY.202
                                 * INITIALIZE ERROR LOGGING                     CRAY.203
                                 *                                              CRAY.204
                                                                                CRAY.205
304    <macro>                   CLEAR START=R!DAL,COUNT=DA@@LH+DA@@LE .CLEAR PACKET CRAY.206
320    <macro>                   STORE RQ$CPU,DA@DID,R!DAL     .DESTINATION     CRAY.207
326    <macro>                   STORE RQ$ERRR,DA@SID,R!DAL    .SOURCE TYPE     CRAY.208
333   020433  012010  024423     R!%W1 = R!DAL + DA@@LH        .POINT PAST HEADER CRAY.209
336    <macro>                   STORE ER@ON,ER$TYP,R!%W1      .SET CODE TO ENABLE LOGGING CRAY.210
346   002000  000000             I = 0                                          CRAY.211
350   020433  024000             R!DY = R!DAL                  .SEND IT         CRAY.212
352    <macro>                   RKERN IDALSND                 .                CRAY.213
354   003000                     I = 1                                          CRAY.214
355   010000  024433             R!DAL = 0                     .RETURNED BY OUTPUT INTERRUPT CRAY.215
357    <macro>                   STORE 1,CPO@ON,R!%LSPO        .SET CHANNEL ON  CRAY.216
364   072341                     R = SUCCESS                   .REPORT INITIALIZATION DONE CRAY.217
```

```
                                                                                                CRAY.21
                                        *                                                       CRAY.215
                                        * MISSION COMPLETE                                      CRAY.220
                                        *                                                       CRAY.221
                                                                                                CRAY.222
365                             DONE    *                                                       CRAY.223
365     <macro>                         STORE 0,CPO@LK,R!%LSPO          .CLEAR THE CHANNEL LOCK CRAY.224
371     <macro>                         $IF (R!DAL # 0)                 .RELEASE DAL IF NEEDED  CRAY.225
                                                                                  .>>>>> 0          $IF      1
374     <macro>                           RELDAL R!DAL                                          CRAY.226
404     <macro>                         $ENDIF                                                  CRAY.227
                                                                                  .<<<<< 0          $ENDIF   1
404     <macro>                         $IF (R!MSG # 0)                 .RELEASE MEMORY IF NEEDED CRAY.228
                                                                                  .>>>>> 0          $IF      1
407   020440  113326                      R = FAILURE, R!FAIL # 0       .OUTPUT FAIL MESSAGE    CRAY.229
411     <macro>                           RELMEM R!MSG                  .                       CRAY.230
421     <macro>                         $ENDIF                                                  CRAY.231
                                                                                  .<<<<< 0          $ENDIF   1
421                             DONE0   *                                                       CRAY.232
421     <macro>                         RETURN                                                  CRAY.233
                                                                                                CRAY.234
                                ****************************** SUBROUTINES ***************************CRAY.235
                                                                                                CRAY.236
                                        *                                                       CRAY.237
                                        * POLL THE CPU                                          CRAY.238
                                        *                                                       CRAY.239
                                                                                                CRAY.240
427                             POLL    *                                                       CRAY.241
427     <macro>                         STORE RQ$CPU,DA@DID,R!DAL       .DESTINATION            CRAY.242
435     <macro>                         PUT R!SID,DA@SID,R!DAL          .SOURCE ID              CRAY.243
442     <macro>                         POLL R!DAL                      .ISSUE                  CRAY.244
457   001000                            EXIT                                                    CRAY.245
                                                                                                CRAY.246
                                        *                                                       CRAY.247
                                        * VALIDATE PACKET                                       CRAY.248
                                        *                                                       CRAY.249
                                                                                                CRAY.250
460                             VALIDATE *                                                      CRAY.251
460     <macro>                         GET R!RESP,DA@HPO,R!DAL         .RESPONSE FROM POLL     CRAY.252
465   014000 /041461  024435            R!EXP = RQ$CPU                  .CPU ID                 CRAY.253
470     <macro>                         GET R!REC,DA@SID,R!RESP         .CHECK SID/DID          CRAY.254
475   020436  023435  103112            P = DATAERR, R!REC # R!EXP      .IF MISCOMPARE          CRAY.255
```

```
500   020437  024435                    R!EXP = R!SID                  .EXPECTED                          CRAY.256
502    <macro>                          GET R!REC,DA@DID,R!RESP        .RECIEVED                          CRAY.257
507   020436  023435  103100            P = DATAERR, R!REC # R!EXP     .IF MISCOMPARE                     CRAY.258
512    <macro>                          $IF (R!SID = RQ$INIT1)         .IF FIRST PACKET                   CRAY.259
                                                                                          .>>>>> 0          $IF      1
516   010000  024435                      R!EXP = 0                    .CHECK FOR OS                      CRAY.260
520    <macro>                          $ELSE                                                             CRAY.261
521   014000 /177777  024435              R!EXP = 177777               .ALL ONES ON 2ND                   CRAY.262
524    <macro>                          $ENDIF                                                            CRAY.263
                                                                                          .<<<<< 0          $ENDIF   1
524    <macro>                          ADDRESS R!%W1,DA@SID,R!RESP    .STARTING ADDRESS                  CRAY.264
527   026423                            R!%W1 = R!%W1 + 1              .                                  CRAY.265
530   020434  012010  012030            R!%W2 = R!RESP + DA@@LH + DA@@LE .END                             CRAY.266
      024424
534    <macro>                          $UNTIL (R!%W1 = R!%W2)                                            CRAY.267
537   030423  024436                      R!REC = (R!%W1)              .DATA RECEIVED                     CRAY.268
541   020436  023435  103046              P = DATAERR, R!REC # R!EXP   .IF MISMATCH                       CRAY.269
544   026423                              R!%W1 = R!%W1 + 1            .NEXT                              CRAY.270
545    <macro>                          $ENDTIL                                                           CRAY.271
546    <macro>                          RELDAL R!RESP                  .RETURN DAL                        CRAY.272
556   001000                            EXIT                                                              CRAY.273
                                                                                                          CRAY.274
                                        *                                                                 CRAY.275
                                        * REPORT CPU TIMEOUT                                             CRAY.276
                                        *                                                                 CRAY.277
                                                                                                          CRAY.278
557                           NORESP    *                                                                 CRAY.279
557   014000 /001242  024423            R!%W1 = MSG4                   .ADDRESS                           CRAY.280
562   010034  024424                    R!%W2 = MSG4L                  .LENGTH                            CRAY.281
564   077000 /001100                    R = COPY                       .MOVE TO BUFFER                    CRAY.282
566   072174                            R = MSGRESP                    .OUTPUT - WAIT FOR RESPONSE        CRAY.283
567   030432  004010  024423            R!%W1 = (R!MSGR) > D'8         .RESPONSE                          CRAY.284
572    <macro>                          $IF (R!%W1 = 'Y'R)             .                                  CRAY.285
                                                                                          .>>>>> 0          $IF      1
576   001000                              EXIT                         .RETURN CODE                       CRAY.286
577    <macro>                          $ELSEIF (R!%W1 = 'N'R)         .IF NO                             CRAY.287
604   010001  024440                      R!FAIL = 1                   .SET FAILURE                       CRAY.288
606   071221                              P = DONE                     .ABORT                             CRAY.289
607    <macro>                          $ELSE                                                             CRAY.290
610   071031                              P = NORESP                   .REPEAT                            CRAY.291
611    <macro>                          $ENDIF                                                            CRAY.292
                                                                                          .<<<<< 0          $ENDIF   1
```

```
                                                                                                CRAY.29.
                                        *                                                       CRAY.294
                                        * REPORT DATA ERROR                                     CRAY.295
                                        *                                                       CRAY.296
                                                                                                CRAY.297
611                             DATAERR *                                                       CRAY.298
611   <macro>                           RELDAL R!RESP                 .RELEASE INPUT DAL      CRAY.299
621   014000 /001260  024423            R!%W1 = MSG5                  .MESSAGE ADDRESS        CRAY.300
624   010035  024424                    R!%W2 = MSG5L                 .LENGTH                 CRAY.301
626   077000 /001100                    R = COPY                      .MOVE TO BUFFER         CRAY.302
630   072117                            R = MSGOUT                    .SEND                   CRAY.303
631   014000 /001277  024423            R!%W1 = MSG6                  .DATA EXPECTED          CRAY.304
634   010023  024424                    R!%W2 = MSG6L                 .LENGTH                 CRAY.305
636   077000 /001100                    R = COPY                      .MOVE IT                CRAY.306
640   020435  024423                    R!%W1 = R!EXP                 .DATA EXPECTED          CRAY.307
642   077000 /001016                    R = BTO                       .ENTER IT               CRAY.308
644   072103                            R = MSGOUT                    .SEND                   CRAY.309
645   014000 /001311  024423            R!%W1 = MSG7                  .DATA RECEIVED          CRAY.310
650   010023  024424                    R!%W2 = MSG7L                 .LENGTH                 CRAY.311
652   077000 /001100                    R = COPY                      .MOVE IT                CRAY.312
654   020436  024423                    R!%W1 = R!REC                 .DATA                   CRAY.313
656   077000 /001016                    R = BTO                       .ENTER                  CRAY.314
660   072067                            R = MSGOUT                    .SEND                   CRAY.315
661                             DATAERRO *                                                      CRAY.316
661   014000 /001323  024423            R!%W1 = MSG8                  .ASK FOR RETRY          CRAY.317
664   010012  024424                    R!%W2 = MSG8L                 .LENGTH                 CRAY.318
666   077000 /001100                    R = COPY                      .                       CRAY.319
670   072072                            R = MSGRESP                   .ISSUE                  CRAY.320
671   030432  004010  024423            R!%W1 = (R!MSGR) > D'8        .ANSWER                 CRAY.321
674   <macro>                           $IF (R!%W1 = 'Y'R)            .IF YES                 CRAY.322
                                                                                   .>>>>> 0        $IF      1
700   010000                              A = 0                       .                       CRAY.323
701   <macro>                           $ELSEIF (R!%W1 = 'N'R)        .NO                     CRAY.324
706   010001  024440                      R!FAIL = 1                  .SET FAIL               CRAY.325
710   071323                              P = DONE                    .                       CRAY.326
711   <macro>                           $ELSE                                                   CRAY.327
712   071031                              P = DATAERRO                .REISSUE                CRAY.328
713   <macro>                           $ENDIF                                                  CRAY.329
                                                                                   .<<<<< 0        $ENDIF   1
713   001000                            EXIT                                                    CRAY.330
                                                                                                CRAY.331
                                        *                                                       CRAY.332
```

```
                                        * REPORT INITIALIZATION STARTED                                CRAY.333
                                        *                                                              CRAY.334
                                                                                                       CRAY.335
714                             STARTED *                                                              CRAY.336
714   014000 /001170  024423            R!%W1 = MSG1                  .MESSAGE ADDRESS                 CRAY.337
717   010032  024424                    R!%W2 = MSG1L                 .LENGTH                          CRAY.338
721   077000 /001100                    R = COPY                      .MOVE RO BUFFER                  CRAY.339
723   072024                            R = MSGOUT                    .SEND                            CRAY.340
724   001000                            EXIT                                                           CRAY.341
                                                                                                       CRAY.342
                                        *                                                              CRAY.343
                                        * OUTPUT INITIALIZATION COMPLETE                               CRAY.344
                                        *                                                              CRAY.345
                                                                                                       CRAY.346
725                             SUCCESS *                                                              CRAY.347
725   014000 /001205  024423            R!%W1 = MSG2                  .MESSAGE ADDRESS                 CRAY.348
730   010036  024424                    R!%W2 = MSG2L                 .LENGTH                          CRAY.349
732   077000 /001100                    R = COPY                      .MOVE                            CRAY.350
734   072013                            R = MSGOUT                    .SEND IT                         CRAY.351
735   001000                            EXIT                                                           CRAY.352
                                                                                                       CRAY.353
                                        *                                                              CRAY.354
                                        * OUTPUT INITIALIZATION FAILED                                 CRAY.355
                                        *                                                              CRAY.356
                                                                                                       CRAY.357
736                             FAILURE *                                                              CRAY.358
736   014000 /001224  024423            R!%W1 = MSG3                  .ADDRESS                         CRAY.359
741   010034  024424                    R!%W2 = MSG3L                 .LENGTH                          CRAY.360
743   077000 /001100                    R = COPY                      .MOVE                            CRAY.361
745   072002                            R = MSGOUT                    .SEND IT                         CRAY.362
746   001000                            EXIT                                                           CRAY.363
                                                                                                       CRAY.364
                                        *                                                              CRAY.365
                                        * OUTPUT MESSAGE                                               CRAY.366
                                        *                                                              CRAY.367
                                                                                                       CRAY.368
747                             MSGOUT  *                                                              CRAY.369
747    <macro>                          MSG R!MSG                     .ISSUE CONSOLE MESSAGE           CRAY.370
760   072021                            R = CLEAR                     .CLEAR MESSAGE BUFFER            CRAY.371
761   001000                            EXIT                                                           CRAY.372
                                                                                                       CRAY.373
                                        *                                                              CRAY.374
```

```
                                        * OUTPUT MESSAGE -     FOR RESPONSE                          CRAY.375
                                        *                                                            CRAY.376
                                                                                                     CRAY.377
 762                            MSGRESP  *                                                            CRAY.378
 762    <macro>                         MSGR R!MSG,MSGRSIZ,R!MSGR      .WAIT FOR RESPONSE          CRAY.379
 777    072002                          R = CLEAR                      .CLEAR MESSAGE BUFFER       CRAY.380
1000    001000                          EXIT                                                       CRAY.381
                                                                                                     CRAY.382
                                        *                                                            CRAY.383
                                        * CLEAR MESSAGE BUFFER                                      CRAY.384
                                        *                                                            CRAY.385
                                                                                                     CRAY.386
1001                            CLEAR   *                                                            CRAY.387
1001    <macro>                         CLEAR START=R!MSG,COUNT=MSGSIZ,BLANKS=YES                  CRAY.388
1015    001000                          EXIT                                                       CRAY.389
                                                                                                     CRAY.390
                                        *                                                            CRAY.391
                                        * CONVERT BINARY TO ASCII OCTAL                             CRAY.392
                                        *                                                            CRAY.393
                                                                                                     CRAY.394
1016                            BTO     *                                                            CRAY.395
1016    020423  034432                  (R!MSGR) = R!%W1               .PARCEL TO TRANSLATE        CRAY.396
1020    <macro>                         CALL BTO,(R!MSGR,0,D'16,R!MSG,DTOFF,0)                     CRAY.397
1046    024423                          R!%W1 = A                      .NEW OFFSET                 CRAY.398
1047    <macro>                         PUTBYTE 0,R!%W1,R!MSG          .TERMINATE                  CRAY.399
1077    001000                          EXIT                                                       CRAY.400
                                                                                                     CRAY.401
                                        *                                                            CRAY.402
                                        * COPY MESSAGE TO BUFFER                                    CRAY.403
                                        *                                                            CRAY.404
                                                                                                     CRAY.405
1100                            COPY    *                                                            CRAY.406
1100    020423  022000  024423          R!%W1 = R!%W1 + R!%B           .ADD BASE                   CRAY.407
1103    010000  024426                  R!%W4 = 0                      .OFFSET                     CRAY.408
1105    <macro>                         $UNTIL (R!%W2 = 0)             .UNTIL COPIED               CRAY.409
1107    <macro>                           GETBYTE R!%W3,R!%W4,R!%W1    .SOURCE                     CRAY.410
1131    <macro>                           PUTBYTE R!%W3,R!%W4,R!MSG,INC=1                          CRAY.411
1162    027424                            R!%W2 = R!%W2 - 1            .COUNT                      CRAY.412
1163    <macro>                         $ENDTIL                                                    CRAY.413
1164    001000                          EXIT                                                       CRAY.414
                                                                                                     CRAY.415
                                        END                                                          CRAY.416
```

0X	%ACTIVE		8:31 D	8:31						
0X	%B		1: 4 D	9:30 B	10:10 B	10:14 B	10:16 B	10:20 B	10:22 B	10:27 B
			11: 7 B	11:18 B	11:29 B	12:33				
0X	%EX		4:13 D	4:13	4:17 D	4:17	5: 6 D	5: 6	6:17 D	6:17
			8:10 D	8:10	8:16 D	8:16	8:20 D	8:20	8:31 D	8:31
			9:20 D	9:20	10: 7 D	10: 7	11:38 D	11:38	12: 5 D	12: 5
			12:23 D	12:23						
10	%GBLNUM	$APTEXT	3:14 F	3:14						
400	%GBLREG	$APTEXT	3:14							
0	%KERNEL	$APTEXT	7:38 D	7:38 B						
0X	%LSPI		2: 8 D	5:30	6:13	6:14	6:18			
0X	%LSPO		2: 9 D	4: 6	4:24	4:42	5: 3	5:22	7:41	8: 7
20	AD@RC@N	$APTEXT	8:31	8:31 F						
13	AD@RC@P	$APTEXT	8:31 F	8:31						
0	AD@RC@S	$APTEXT	8:31	8:31 F						
777	CPI@CN@M	$APTEXT	5:30							
11	CPI@CN@N	$APTEXT	5:30							
0	CPI@CN@P	$APTEXT	5:30 F							
7	CPI@CN@S	$APTEXT	5:30	5:30 F						
20	CPI@DA@N	$APTEXT	6:14	6:14 F	6:18 F					
2	CPI@DA@P	$APTEXT	6:14 F	6:14	6:18 F	6:18				
0	CPI@DA@S	$APTEXT	6:14	6:14 F	6:18 F					
1	CPI@ON@M	$APTEXT	6:13 F							
1	CPI@ON@N	$APTEXT	6:13 F	6:13						
0	CPI@ON@P	$APTEXT	6:13 F							
0	CPI@ON@S	$APTEXT	6:13 F	6:13						
777	CPO@CN@M	$APTEXT	5:22							
11	CPO@CN@N	$APTEXT	5:22							
0	CPO@CN@P	$APTEXT	5:22 F							
7	CPO@CN@S	$APTEXT	5:22	5:22 F						
1	CPO@DS@M	$APTEXT	5: 3							
1	CPO@DS@N	$APTEXT	5: 3	5: 3 F						
0	CPO@DS@P	$APTEXT	5: 3 F							
3	CPO@DS@S	$APTEXT	5: 3	5: 3 F						
1	CPO@LK@M	$APTEXT	4: 6	4:42 F	8: 7 F					
1	CPO@LK@N	$APTEXT	4: 6	4: 6 F	4:42 F	4:42	8: 7 F			
0	CPO@LK@P	$APTEXT	4: 6 F	4:42 F	8: 7 F					
1	CPO@LK@S	$APTEXT	4: 6	4: 6 F	4:42 F	4:42	8: 7 F			
137777	CPO@LK@X	$APTEXT	8: 7							
1	CPO@ON@M	$APTEXT	7:41 F							
1	CPO@ON@N	$APTEXT	4:24	4:24 F	7:41 F	7:41				
0	CPO@ON@P	$APTEXT	4:24 F	7:41 F						

0	CPO@ON@S	$APTEXT	4:24		4:24	F		F	7:41					
30	DA@@LE	$APTEXT	6:27		7: 8		7:31		9:13					
10	DA@@LH	$APTEXT	6:27		7: 7		7:31		7:34		9:13			
20	DA@DID@N	$APTEXT	7:32	F	8:29	F	9: 2		9: 2	F				
10	DA@DID@P	$APTEXT	7:32	F	7:32		8:29	F	8:29		9: 2	F	9: 2	
0	DA@DID@S	$APTEXT	7:32	F	8:29	F	9: 2		9: 2	F				
20	DA@HPO@N	$APTEXT	8:39		8:39	F								
6	DA@HPO@P	$APTEXT	8:39	F	8:39									
0	DA@HPO@S	$APTEXT	8:39		8:39	F								
20	DA@SID@N	$APTEXT	7:33	F	8:30	F	8:41		8:41	F				
11	DA@SID@P	$APTEXT	7:33	F	7:33		8:30	F	8:30		8:41	F	8:41	9:11
0	DA@SID@S	$APTEXT	7:33	F	8:30	F	8:41		8:41	F				
0X	DY		2:10	D	7:37									
377	ER$TYP@M	$APTEXT	7:35	F										
10	ER$TYP@N	$APTEXT	7:35	F										
3	ER$TYP@P	$APTEXT	7:35	F	7:35									
10	ER$TYP@S	$APTEXT	7:35	F										
177400	ER$TYP@X	$APTEXT	7:35											
4	ER@ON	$APTEXT	7:35											
50	F$CALL	$APTEXT	12:23											
26	F$GDAL	$APTEXT	4:17											
30	F$GMEM	$APTEXT	4:13											
20	F$MESG	$APTEXT	11:38											
21	F$MSGR	$APTEXT	12: 5											
7	F$PAUSE	$APTEXT	5: 6											
44	F$POLL	$APTEXT	8:31											
27	F$RDAL	$APTEXT	6:17		8:10		9:20		10: 7					
52	F$RETURN	$APTEXT	8:20											
31	F$RMEM	$APTEXT	8:16											
0X	IDALSND		7:38	D	7:38									
0X	O$BTO		12:23	D	12:23									
0X	O$CRAY		1: 4	D	1: 4									
17	OV@NUM@N	$APTEXT	1: 4											
1	OV@TYP@N	$APTEXT	1: 4											
41461	RQ$CPU	$APTEXT	7:32		8:29		8:40							
103	RQ$ERRR	$APTEXT	7:33											
111	RQ$INIT1	$APTEXT	6:28		9: 5									
112	RQ$INIT2	$APTEXT	7:13											
11	SM@FST@N	$APTEXT	1: 4											
7	SM@NUM@N	$APTEXT	1: 4											

SYMBOL QUALIFIER IS CRAY

50	$$X	3: 8 D	3: 9						
41461	$CONST	4:42 D	4:42 F	4:42	6:13 D	6:13 F	6:13	6:18 D	6:18 F
		7:32 D	7:32 F	7:32	7:33 D	7:33 F	7:33	7:35 D	7:35 F
		7:35	7:41 D	7:41 F	7:41	8: 7 D	8: 7 F	8:29 D	8:29 F
		8:29							
441	$REGORG	3:14 D	3:14 F	3:15	3:15 D	3:16	3:16 D	3:17	3:17 D
		3:18	3:18 D	3:19	3:19 D	3:19 F	3:20	3:20 D	3:21
		3:21 D	3:22	3:22 D	3:23	3:23 D	3:24	3:24 D	3:25
		3:25 D	3:25 F	3:26	3:26 D	3:27	3:27 D	3:28	3:28 D
		3:29	3:29 D	3:30	3:30 D	3:30 F	3:31	3:31 D	3:31 F
		3:32	3:32 D	3:33	3:33 D	3:34	3:34 D	3:35	3:35 D
		3:36	3:36 D	3:37	3:37 D	3:38	3:38 D	3:39	3:39 D
		3:39 F	4:13	4:17	5: 6	5: 6 F	6:17	8:10	8:16
		8:20	8:31	8:31 F	9:20	10: 7	11:38	11:38 F	12: 5
		12: 5 F	12:23	12:23 F					
1	%NP	1: 4	3:31 D						
430	%P	1: 4	3:30 D	3:31	5: 6 F	5: 6	8:31 F	8:31	11:38 F
		11:38	12: 5 F	12: 5	12:23 F	12:23			
410	%S1	3:14	3:15 D	6:14	6:18	7:32	7:33	7:35	8:29
		8:30	8:31	8:39	8:41	9: 2	12:25	12:36	12:37
411	%S2	3:14	3:16 D						
412	%S3	3:14	3:17 D						
413	%S4	3:14	3:18 D						
414	%S5	3:14	3:19 D						
415	%T1	3:20 D	6:27	7:31	12:14	12:25	12:36	12:37	
416	%T2	3:21 D	6:27	7:31	12:14	12:25	12:37		
417	%T3	3:22 D							
420	%T4	3:23 D							
421	%T5	3:24 D							
422	%T6	3:25 D							
423	%W1	3:26 D	5: 1	5: 2	5: 3	5:38	5:39	6: 8	6:14
		6:17	7: 7	7: 8	7: 9	7:10	7:11	7:34	7:35
		9:11	9:12	9:15	9:16	9:18	9:28	9:32	9:34
		9:36	10: 8	10:12	10:15	10:18	10:21	10:25	10:29
		10:31	10:33	11: 5	11:16	11:27	12:22	12:24	12:25
		12:33	12:36						
424	%W2	3:27 D	6: 2	6: 3	6: 4	6: 7	7: 8	7: 9	9:13
		9:15	9:29	10: 9	10:13	10:19	10:26	11: 6	11:17
		11:28	12:35	12:38					
425	%W3	3:28 D	12:36	12:37					
426	%W4	3:29 D	12:34	12:36	12:37				

427	%W5	3:30 D							
1016	BTO	10:16	10:22	12:21 D					
1001	CLEAR	4:16	11:39	12: 6	12:13 D				
1100	COPY	9:30	10:10	10:14	10:20	10:27	11: 7	11:18	11:29
		12:32 D							
433	DAL	3:34 D	4: 5	4:17	6:27	7: 7	7:31	7:32	7:33
		7:34	7:37	7:40	8: 9	8:10	8:29	8:30	8:31
		8:39							
611	DATAERR	8:42	9: 3	9:17	10: 6 D				
661	DATAERRO	10:24 D	10:37						
365	DONE	4:14	4:18	4:29	4:32	8: 6 D	9:38	10:35	
421	DONEO	4: 7	8:19 D						
25	DTOFF	3: 2 D	12:23						
435	EXP	3:36 D	8:40	8:42	9: 1	9: 3	9: 6	9: 8	9:17
		10:15							
440	FAIL	3:39 D	4: 3	8:15	9:37	10:34			
736	FAILURE	8:15	11:26 D						
177777	INTERNAL	3: 1 D	4:28						
30	MEMSIZ	3: 6 D	4:13						
431	MSG	3:32 D	4: 4	4:13	4:15	8:14	8:16	11:38	12: 5
		12:14	12:23	12:25	12:37				
1170	MSG1	2:15 D	11: 5						
32	MSG1L	2:34 D	11: 6						
1205	MSG2	2:17 D	11:16						
36	MSG2L	2:35 D	11:17						
1224	MSG3	2:19 D	11:27						
34	MSG3L	2:36 D	11:28						
1242	MSG4	2:21 D	9:28						
34	MSG4L	2:37 D	9:29						
1260	MSG5	2:23 D	10: 8						
35	MSG5L	2:38 D	10: 9						
1277	MSG6	2:25 D	10:12						
23	MSG6L	2:39 D	3: 2	10:13					
1311	MSG7	2:27 D	10:18						
23	MSG7L	2:40 D	10:19						
1323	MSG8	2:29 D	10:25						
12	MSG8L	2:41 D	10:26						
50	MSGCHR	3: 9 D							
747	MSGOUT	10:11	10:17	10:23	11: 8	11:19	11:30	11:37 D	
432	MSGR	3:33 D	4:15	5:42	9:32	10:29	12: 5	12:22	12:23
762	MSGRLSP	9:31	10:28	12: 4 D					
4	MSGRSIZ	3: 5 D	3: 6	12: 5					

24	MSGSIZ	3: 4 D	3: 6	3: 8	4:15	12:14			
557	NORESP	6:32	7:17	9:27 D	9:40				
427	POLL	6:29	7:14	8:28 D					
436	REC	3:37 D	8:41	8:42	9: 2	9: 3	9:16	9:17	10:21
434	RESP	3:35 D	8:39	8:41	9: 2	9:11	9:13	9:20	10: 7
437	SID	3:38 D	6:28	7:13	8:30	9: 1	9: 5		
430	SOURCE	3:31 D	4:28						
126	START	5:21 D	6:33	7:18					
714	STARTED	5:15	11: 4 D						
725	SUCCESS	4:31	7:42	11:15 D					
460	VALIDATE	6:38	7:23	8:38 D					

```
<macro>                    OVERLAY   CRTDEM                                          CRTDEM.4

                           COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
*********************************************************************  CRTDEM.5
*  THE CRT DEMON IS ACTIVATED TO PROCESS INPUT AND OUTPUT INTERRUPTS   *  CRTDEM.6
*  FOR THE CRT CHANNELS (40 - 47). ENTRIES ARE PLACED ON A QUEUE FOR   *  CRTDEM.7
*  PROCESSING (CRTQU) IN RESPONSE TO CRT INTERRUPTS. ENTRIES ARE ALSO  *  CRTDEM.8
*  QUEUED BY THE KERNEL MESSAGE PROCESSOR TO INITIATE THE OUTPUT OF    *  CRTDEM.9
*  MESSAGES.                                                           *  CRTDEM.10
*                                                                      *  CRTDEM.11
*                QUEUE ENTRY:                                          *  CRTDEM.12
*                              BITS   0 - 7  ASCII INPUT               *  CRTDEM.13
*                                     8 - 15 CHANNEL NUMBER            *  CRTDEM.14
*                                                                      *  CRTDEM.15
*********************************************************************  CRTDEM.16

6    <macro>                    REGDEFS ,,(CHN,TBL,CHR,BFR,USB,MAX,NXT,LST,GSV,SC0,SC1,SC2,SC3)  CRTDEM.18
410  %S1    EQUALS   $REGORG                                                  $REG   3
411  %S2    EQUALS   $REGORG                                                  $REG   3
412  %S3    EQUALS   $REGORG                                                  $REG   3
413  %S4    EQUALS   $REGORG                                                  $REG   3
414  %S5    EQUALS   $REGORG                                                  $REG   3
415  %T1    EQUALS   $REGORG                                                  $REG   3
416  %T2    EQUALS   $REGORG                                                  $REG   3
417  %T3    EQUALS   $REGORG                                                  $REG   3
420  %T4    EQUALS   $REGORG                                                  $REG   3
421  %T5    EQUALS   $REGORG                                                  $REG   3
422  %T6    EQUALS   $REGORG                                                  $REG   3
423  %W1    EQUALS   $REGORG                                                  $REG   3
424  %W2    EQUALS   $REGORG                                                  $REG   3
425  %W3    EQUALS   $REGORG                                                  $REG   3
426  %W4    EQUALS   $REGORG                                                  $REG   3
427  %W5    EQUALS   $REGORG                                                  $REG   3
430  CHN    EQUALS   $REGORG                                                  $REG   3
431  TBL    EQUALS   $REGORG                                                  $REG   3
432  CHR    EQUALS   $REGORG                                                  $REG   3
433  BFR    EQUALS   $REGORG                                                  $REG   3
434  USB    EQUALS   $REGORG                                                  $REG   3
435  MAX    EQUALS   $REGORG                                                  $REG   3
436  NXT    EQUALS   $REGORG                                                  $REG   3
437  LST    EQUALS   $REGORG                                                  $REG   3
440  GSV    EQUALS   $REGORG                                                  $REG   3
441  SC0    EQUALS   $REGORG                                                  $REG   3
442  SC1    EQUALS   $REGORG                                                  $REG   3
```

```
443     SC2     EQUALS    $REGORG                                          $REG    3
444     SC3     EQUALS    $REGORG                                          $REG    3

                EXT       %EX                                              CRTDEM.20
                EXT       BCRT                                             CRTDEM.21
                EXT       BIQU                                             CRTDEM.22
                EXT       BOQU                                             CRTDEM.23
                EXT       CRTQU                                            CRTDEM.24
                EXT       ED                                               CRTDEM.25
                EXT       EH                                               CRTDEM.26
                EXT       O$CALL                                           CRTDEM.27
```

```
                                        *                                                       CRTDEM.29
                                        * GET ENTRY FROM QUEUE                                  CRTDEM.30
                                        *                                                       CRTDEM.31
                                                                                                CRTDEM.32
 6                              NEXT    *                                                       CRTDEM.33
 6   002000  000000                     I = 0                          .LOCK                    CRTDEM.34
10   014000 /000000  024000             ED = CRTQU                     .QUEUE ADDRESS           CRTDEM.35
13    <macro>                           RKERN ETOQ                     .POP NEXT ENTRY          CRTDEM.36
15   020000  024430                     R!CHN = EH                     .SAVE                    CRTDEM.37
17    <macro>                           $IF (A = 0)                    .IF NO MORE TO DO        CRTDEM.38
                                                                                   .>>>>> 0          $IF      1
21    <macro>                             PKERN ESWP                   .GO TO IDLE LOOP         CRTDEM.39
23    <macro>                           $ENDIF                                                  CRTDEM.40
                                                                                   .<<<<< 0          $ENDIF   1
23   003000                             I = 1                          .OPEN                    CRTDEM.41
                                                                                                CRTDEM.42
                                        *                                                       CRTDEM.43
                                        * GET CHANNEL AND MESSAGE                               CRTDEM.44
                                        *                                                       CRTDEM.45
                                                                                                CRTDEM.46
24   020430  004010  011377             R!CHR = R!CHN > 10 & 377       .MESSAGE                 CRTDEM.47
     024432
30   020430  011377  024430             R!CHN = R!CHN & 377            .CHANNEL                 CRTDEM.48
                                                                                                CRTDEM.49
                                        *                                                       CRTDEM.50
                                        * GET ADDRESS OF CRT TABLE                              CRTDEM.51
                                        *                                                       CRTDEM.52
                                                                                                CRTDEM.53
33   020430  004001  011003             R!TBL = R!CHN > 1 & 3 + BCRT .OFFSET                   CRTDEM.54
     016000 /000000  024431
41   030431  024431                     R!TBL = (R!TBL)                .TABLE ADDRESS           CRTDEM.55
                                                                                                CRTDEM.56
                                        *                                                       CRTDEM.57
                                        * GET POSITION POINTERS                                 CRTDEM.58
                                        *                                                       CRTDEM.59
                                                                                                CRTDEM.60
43    <macro>                           GET R!NXI,TT@PTO,R!TBL       .POINTER TO NEXT          CRTDEM.61
50    <macro>                           GET R!LST,TT@PT1,R!TBL       .POINTER TO LAST          CRTDEM.62
55    <macro>                           GET R!USB,TT@USB,R!TBL       .POINTER TO UNSOLICTED INPUT BUFF  CRTDEM.63
                                                                                                CRTDEM.64
                                        *                                                       CRTDEM.65
                                        * INPUT CHANNEL INTERRUPT                               CRTDEM.66
```

```
                                  *                                                          CRTDEM.67
                                                                                             CRTDEM.68
  62    <macro>                   $IF (0 = R!CHN & 1)          .INPUT INTERRUPT             CRTDEM.69
                                                                              .>>>>> 0          $IF      1
                                                                                             CRTDEM.70
                                    * IF CRT HAS BEEN USURPED BY AN ACTIVITY                 CRTDEM.71
                                                                                             CRTDEM.72
  71    <macro>                     GET R!SCO,TT@MOD,R!TBL      .GET MODE                  CRTDEM.73
  76    <macro>                     $IF (R!SCO = TT$OWN)        .IF CRT IS USURPED         CRTDEM.74
                                                                              .>>>>> 1          $IF      1
 102    002000  000000                I = 0                     .DEQUEUE USURPER           CRTDEM.75
 104    <macro>                       ADDRESS ED,TT@IQ,R!TBL    .QUEUE ADDRESS             CRTDEM.76
 107    <macro>                       RKERN EDQU                .ROUTINE TO DEQUEUE        CRTDEM.77
 111    <macro>                       $IF (EH = 0)              .IF NO ACTIVITY WAITING    CRTDEM.78
                                                                              .>>>>> 2          $IF      1
 114    <macro>                         PUT R!CHR,TT@ECH,R!TBL  .BUFFER IT                 CRTDEM.79
 127    <macro>                       $ELSE                     .RE-ACTIVATE               CRTDEM.80
 130    020000  024423                  R!%W1 = R!EH            .SAVE ACTIVITY ADDRESS     CRTDEM.81
 132    <macro>                         ADDRESS R!EH,AD@P1,R!EH .TIMER ENTRY               CRTDEM.82
 134    <macro>                         RKERN DQTIME            .DEQUEUE IT                CRTDEM.83
 136    020423  024000                  R!EH = R!%W1            .RESTORE ACTIVITY ADDRESS  CRTDEM.84
 140    <macro>                         RKERN EQCP              .QUEUE FOR EXECUTION       CRTDEM.85
 142    <macro>                         PUT R!CHR,AD@P1,R!EH    .PASS CHARACTOR            CRTDEM.86
 147    <macro>                         STORE 0,AD@RC,R!EH      .RESPONSE CODE             CRTDEM.87
 154    020430  054000                  B = R!CHN               .RE-ENABLE CHANNEL         CRTDEM.88
 156    167000                          IOB:7                                              CRTDEM.89
 157    071151                          P = NEXT                .CHECK FOR MORE            CRTDEM.90
 160    <macro>                       $ENDIF                                               CRTDEM.91
                                                                              .<<<<< 2          $ENDIF   1
                                                                                             CRTDEM.92
                                    * CRT BELONGS TO KERNEL                                  CRTDEM.93
                                    * - IGNORE INPUT IF IN MIDDLE OF OUTPUT                  CRTDEM.94
                                                                                             CRTDEM.95
 160    <macro>                     $ELSEIF (R!SCO # TT$OUT)                                 CRTDEM.96
                                                                                             CRTDEM.97
                                      * GET MESSAGE BUFFER ADDRESS AND LENGTH                CRTDEM.98
                                                                                             CRTDEM.99
 165    <macro>                       STORE TT$INP,TT@MOD,R!TBL .SET MODE TO INPUT         CRTDEM.100
 172    <macro>                       GET R!BFR,TT@IBF,R!TBL    .BUFFER ADDRESS            CRTDEM.101
 177    <macro>                       GET R!MAX,TT@SIZ,R!TBL    .BUFFER LENGTH             CRTDEM.102
 204    <macro>                       $IF (R!BFR = 0)           .IF NONE ASSIGNED          CRTDEM.103
                                                                              .>>>>> 2          $IF      1
```

```
207   020434  024433                R!BFR = R!USB          .USE BUFFER FOR UNSOLICITED INPUT  CRTDEM.104
211    <macro>                      PUT R!BFR,TT@IBF,R!TBL                                    CRTDEM.105
216   010117  024435                R!MAX = TT$MAX         .LENGTH OF BUFFER                  CRTDEM.106
220    <macro>                      STORE TT$MAX,TT@SIZ,R!TBL  .SET SIZE TO MAXIMUM LENGTH    CRTDEM.107
225    <macro>                    $ENDIF                                                      CRTDEM.108
                                                                          .<<<<< 2                $ENDIF   1
                                                                                                CRTDEM.109
                                  *                                                             CRTDEM.110
                                  * PROCESS CHARACTERS COMMON TO ALL CRT TYPES                  CRTDEM.111
                                  *                                                             CRTDEM.112
                                                                                                CRTDEM.113
                                  * CLEAR PROCESS FLAG                                          CRTDEM.114
                                                                                                CRTDEM.115
225   010000  024441              R!SCO = 0                                                    CRTDEM.116
                                                                                                CRTDEM.117
                                  * ESCAPE                                                      CRTDEM.118
                                  * - TERMINATE LINE / RETURN CHARACTER                         CRTDEM.119
                                                                                                CRTDEM.120
227    <macro>                    $IF (R!CHR = CHR$ES)                                         CRTDEM.121
                                                                          .>>>>> 2                $IF      1
233   077000 /002024                R = PUTCHAR            .ENTER IN BUFFER                   CRTDEM.122
235   010012  024432                R!CHR = CHR$LF         .DO LINE FEED                      CRTDEM.123
                                                                                                CRTDEM.124
                                  * CONTROL-U                                                   CRTDEM.125
                                  * - RETURN TO BEGINNING OF LINE                               CRTDEM.126
                                                                                                CRTDEM.127
237    <macro>                    $ELSEIF (R!CHR = CHR$CTU)                                    CRTDEM.128
244   020436  103003  010007        R!CHR = CHR$BEL, R!NXT = 0  .IF ALREADY THERE              CRTDEM.129
      024432
                                                                                                CRTDEM.130
                                  * BACK SPACE                                                  CRTDEM.131
                                  * - BACK UP 1 POSITION                                        CRTDEM.132
                                                                                                CRTDEM.133
250    <macro>                    $ELSEIF (R!CHR = CHR$BS)                                     CRTDEM.134
255    <macro>                      $IF (R!NXT = 0)        .IF ALREADY AT START               CRTDEM.135
                                                                          .>>>>> 3                $IF      1
260   010007  024432                  R!CHR = CHR$BEL      .SEND REMINDER                     CRTDEM.136
262    <macro>                      $ELSE                                                       CRTDEM.137
263   027436                          R!NXT = R!NXT - 1    .BACKUP                            CRTDEM.138
264    <macro>                      $ENDIF                                                      CRTDEM.139
                                                                          .<<<<< 3                $ENDIF   1
                                                                                                CRTDEM.140
```

```
                                    * CONTROL-A O,    NTROL-X                                   CRTDEM.14
                                    * - REPEAT THE LAST INPUT LINE                              CRTDEM.142
                                    * - EXECUTE IF CONTROL-X                                    CRTDEM.143
                                                                                                CRTDEM.144
264    <macro>                      $ELSEIF (R!CHR = CHR$CTA) OR (R!CHR = CHR$CTX)              CRTDEM.145
274    000000                   PASS                                                            CRTDEM.146
                                                                                                CRTDEM.147
                                    * CARRIAGE RETURN OR LINE FEED                              CRTDEM.148
                                                                                                CRTDEM.149
275    <macro>                      $ELSEIF (R!CHR = CHR$CR) OR (R!CHR = CHR$LF)                CRTDEM.150
305    010012  024432                 R!CHR = CHR$LF            .SET TO LINE FEED               CRTDEM.151
307    077000 /002024                 R = PUTCHAR               .TERMINATE BUFFER               CRTDEM.152
                                                                                                CRTDEM.153
                                    * RUBOUT                                                    CRTDEM.154
                                                                                                CRTDEM.155
311    <macro>                      $ELSEIF (R!CHR = CHR$RUB)                                   CRTDEM.156
316    020436  103003  010007         R!CHR = CHR$BEL, R!NXT = 0   .START WITH BACKSPACE        CRTDEM.157
       024432
                                                                                                CRTDEM.158
                                    * PRINTABLE CHARACTERS                                      CRTDEM.159
                                                                                                CRTDEM.160
322    <macro>                      $ELSEIF (R!CHR >= ' 'R),AND,(R!CHR <= '~'R)                 CRTDEM.161
335    077000 /002024                 R = PUTCHAR               .ENTER IN BUFFER                CRTDEM.162
                                                                                                CRTDEM.163
                                    * IF UNPROCESSED                                            CRTDEM.164
                                                                                                CRTDEM.165
337    <macro>                      $ELSE                                                       CRTDEM.166
340    010001  024441                 R!SCO = 1                 .SET FLAG                       CRTDEM.167
                                                                                                CRTDEM.168
342    <macro>                      $ENDIF                                                      CRTDEM.169
                                                                                .<<<<< 2              $ENDIF    1
                                                                                                CRTDEM.170
                                    *                                                           CRTDEM.171
                                    * PROCESS CHARACTERS UNIQUE TO CRT TYPE                     CRTDEM.172
                                    *                                                           CRTDEM.173
                                                                                                CRTDEM.174
342    <macro>                      GET R!SC1,TT@TYP,R!TBL        .GET CRT TYPE                 CRTDEM.175
                                                                                                CRTDEM.176
                                    * AMPEX                                                     CRTDEM.177
                                                                                                CRTDEM.178
347    <macro>                      $IF (R!SC1 = CF$AMPEX) AND (R!SCO = 1)   .AMPEX CRT         CRTDEM.179
                                                                                .>>>>> 2              $IF      1
```

```
357   010000  024441                R!SCO = 0              .CLEAR UNPROCESSED FLAG          CRTDEM.180
                                                                                            CRTDEM.181
                                     * FORWARD SPACE                                        CRTDEM.182
                                                                                            CRTDEM.183
361    <macro>                       $IF (R!CHR = CHR$CTL)                                  CRTDEM.184
                                                                          .>>>>> 3               $IF      1
365    <macro>                         $IF (R!NXT >= R!MAX - 1) .IF AT END OF INPUT LENGTH  R15961AA.1
                                                                          .>>>>> 4               $IF      1
374   010007  024432                    R!CHR = CHR$BEL     .RING THE BELL                   CRTDEM.186
376    <macro>                         $ELSE                                                  CRTDEM.187
377   026436                            R!NXT = R!NXT + 1   .MOVE FORWARD                    CRTDEM.188
400   020437  023436  101002            R!LST = R!LST + 1, R!LST < R!NXT                     CRTDEM.189
      026437
404    <macro>                         $ENDIF                                                 CRTDEM.190
                                                                          .<<<<< 4               $ENDIF   1
                                                                                            CRTDEM.191
                                     * SET UNPROCESSED FLAG                                 CRTDEM.192
                                                                                            CRTDEM.193
404    <macro>                       $ELSE                                                  CRTDEM.194
405   010001  024441                  R!SCO = 1                                              CRTDEM.195
                                                                                            CRTDEM.196
407    <macro>                       $ENDIF                                                 CRTDEM.197
                                                                          .<<<<< 3               $ENDIF   1
407    <macro>                     $ENDIF                                                   CRTDEM.198
                                                                          .<<<<< 2               $ENDIF   1
                                                                                            CRTDEM.199
                                     *                                                      CRTDEM.200
                                     * IF CHARACTER NOT PROCESSED                            CRTDEM.201
                                     *                                                      CRTDEM.202
                                                                                            CRTDEM.203
407    <macro>                     $IF (R!SCO # 0)                                          CRTDEM.204
                                                                          .>>>>> 2               $IF      1
412   010007  024432                  R!CHR = CHR$BEL         .BEEP                        CRTDEM.205
414    <macro>                     $ENDIF                                                   CRTDEM.206
                                                                          .<<<<< 2               $ENDIF   1
                                                                                            CRTDEM.207
                                     *                                                      CRTDEM.208
                                     * ECHO BACK CHARACTER                                   CRTDEM.209
                                     *                                                      CRTDEM.210
                                                                                            CRTDEM.211
414   077000 /001524                R = OUTCHAR             .ECHO CHARACTER                CRTDEM.212
                                                                                            CRTDEM.213
```

```
416    <macro>                    $ENDIF                                                    CRTDEM.21
                                                                        .<<<<< 1                $END     1
                                                                                              CRTDEM.215
                                  *                                                           CRTDEM.216
                                  * OUTPUT INTERRUPT HANDLING                                 CRTDEM.217
                                  *                                                           CRTDEM.218
                                                                                              CRTDEM.219
416    <macro>                    $ELSE                                                       CRTDEM.220
                                                                                              CRTDEM.221
                                    * IF CRT IS USURPED BY AN ACTIVITY                        CRTDEM.222
                                                                                              CRTDEM.223
420    <macro>                      GET R!SC0,T1@MOD,R!TBL       .MODE WORD                  CRTDEM.224
425    <macro>                      $IF (R!SC0 = TT$OWN)         .IF OWNED BY AN ACTIVITY    CRTDEM.225
                                                                                  .>>>>> 1          $IF      1
431   010001  024442                  R!SC1 = 1                   .SET NON ZERO               CRTDEM.226
433    <macro>                        $UNTIL (R!SC1 = 0)          .UNTIL QUEUE FLUSHED        CRTDEM.227
435    <macro>                          GET R!SC1,TT@OQ,R!TBL     .ACTIVITY ADDRESS           CRTDEM.228
442    <macro>                          $IF (R!SC1 # 0)           .IF ONE THERE               CRTDEM.229
                                                                                  .>>>>> 2          $IF      1
445    <macro>                            GET R!SC2,AD@P1,R!SC1 .GET CHARACTERS TO GO           CRTDEM.230
452    <macro>                            $IF (R!SC2 = 0)         .IF OUTPUT DONE               CRTDEM.231
                                                                                  .>>>>> 3          $IF      1
455    <macro>                              STORE 0,AD@RC,R!SC1 .RETURN CODE                    CRTDEM.232
462   002000  000000                        I = 0               .LOCK                           CRTDEM.233
464    <macro>                              ADDRESS ED,TT@OQ,R!TBL .QUEUE ADDRESS               CRTDEM.234
467    <macro>                              RKERN EDQU          .POP                            CRTDEM.235
471    <macro>                              RKERN EQCP          .QUEUE FOR EXECUTION            CRTDEM.236
473    <macro>                            $ELSE                                                 CRTDEM.237
474    <macro>                              GET R!SC2,AD@FU,R!SC1  .BUFFER ADDRESS              CRTDEM.238
501    <macro>                              GET R!SC3,AD@RC,R!SC1  .POINTER TO NEXT CHARACTER   CRTDEM.239
506   020430  054000                        B = R!CHN           .CHANNEL                        CRTDEM.240
510    <macro>                              GETBYTE A,R!SC3,R!SC2  .CHARACTER TO SEND           CRTDEM.241
531   174000                                IOB:14              .SEND IT                        CRTDEM.242
532    <macro>                              FLDADD 1,AD@RC,R!SC1 .UPDATE POINTER                CRTDEM.243
536    <macro>                              FLDSUB 1,AD@P1,R!SC1 .DECREMENT CHARACTER COUNT     CRTDEM.244
542   010000  024442                        R!SC1 = 0           .TERMINATE SEARCH               CRTDEM.245
544    <macro>                            $ENDIF                                                CRTDEM.246
                                                                                  .<<<<< 3          $ENDIF   1
544    <macro>                          $ENDIF                                                  CRTDEM.247
                                                                                  .<<<<< 2          $ENDIF   1
544    <macro>                        $ENDTIL                                                   CRTDEM.248
                                                                                              CRTDEM.249
```

```
                                  * RE-ENABLE CHANNEL                                    CRTDEM.250
                                                                                         CRTDEM.251
545   020430  054000              B = R!CHN                 .CHANNEL                     CRTDEM.252
547   167000                      IOB:7                                                  CRTDEM.253
550   071542                      P = NEXT                  .CHECK FOR MORE              CRTDEM.254
                                                                                         CRTDEM.255
                                * CONTINUE KERNEL OUTPUT OF MESSAGE                      CRTDEM.256
                                                                                         CRTDEM.257
551    <macro>                  $ELSEIF (R!SCO = TT$OUT)                                 CRTDEM.258
557    <macro>                    GET R!BFR,TT@OBF,R!TBL    .OUTPUT BUFFER ADDRESS       CRTDEM.259
564    <macro>                    $IF (R!BFR # 0)           .IF BUFFER EXISTS            CRTDEM.260
                                                                        .>>>>> 2                $IF      1
                                                                                         CRTDEM.261
                                      * CHECK FOR END OF MESSAGE                         CRTDEM.262
                                                                                         CRTDEM.263
567    <macro>                        GET R!CHR,TT@ECH,R!TBL  .LAST CHARACTER            CRTDEM.264
575    <macro>                        GET R!SCO,TT@SEQ,R!TBL   .SPECIAL CHARACTER PROCESS CRTDEM.265
                                                                                         CRTDEM.266
                                      * LINE FEED                                        CRTDEM.267
                                      * - DO CARRIAGE RETURN                             CRTDEM.268
                                                                                         CRTDEM.269
603    <macro>                        $IF (R!CHR = CHR$LF) AND (R!SCO = 1)  .IF LINE FEED CRTDEM.270
                                                                        .>>>>> 3                $IF      1
613   010015  024432                   R!CHR = CHR$CR         .DO CARRIAGE RETURN         CRTDEM.271
615   077000 /001524                   R = OUTCHAR            .SEND                       CRTDEM.272
                                                                                         CRTDEM.273
                                      * CARRIAGE RETURN                                  CRTDEM.274
                                      * - TERMINATE OUTPUT                               CRTDEM.275
                                                                                         CRTDEM.276
617    <macro>                        $ELSEIF (R!CHR = CHR$CR) AND (R!SCO = 1) .CARRIAGE RETURN CRTDEM.277
630    <macro>                          STORE 0,TT@SEQ,R!TBL  .CLEAR SPECIAL CHARACTER COUNT CRTDEM.278
637   077000 /001544                    R = MSGDN              .TERMINATE OUTPUT           CRTDEM.279
                                                                                         CRTDEM.280
                                      * OUTPUT NEXT CHARACTER                            CRTDEM.281
                                                                                         CRTDEM.282
641    <macro>                        $ELSE                                              CRTDEM.283
642    <macro>                          GETBYTE R!CHR,R!NXT,R!BFR,INC=1  .NEXT CHARACTER CRTDEM.284
                                  ***********************************************        CRTDEM.285
                                  * RECORD OUTPUT CHARACTERS TO CRT                *      CRTDEM.286
                                  *                                                 *      CRTDEM.287
                                  *   1. CHANNEL                                    *      CRTDEM.288
                                  *   2. CHARACTER IN ASCII                         *      CRTDEM.289
```

```
                                  *   3. BUFFER     ESS                          *          CRTDEM.290
                                  *   4. ACTIVITY DESC. ADDRESS                  *          CRTDEM.291
                                  *   5. 0                                       *          CRTDEM.292
                                  ***********************************************           CRTDEM.293
                                                                                            CRTDEM.294
665   020430  011006  016000      R!SCO = R!CHN & 6 + BOQU  . CALCULATE SPOT TO GET A.D.    CRTDEM.295
     /000000  024441
672   030441  024441              R!SCO = (R!SCO)           . ACTIVITY DESC.                CRTDEM.296
674    <macro>                    KRECORD TF$CHAR,(R!CHN,R!CHR,R!BFR,R!SCO,0)               CRTDEM.297
                                                                                            CRTDEM.298
                                                                                            CRTDEM.299
                                      * TERMINATE WITH LF-CR                                CRTDEM.300
                                                                                            CRTDEM.301
713    <macro>                        $IF (R!CHR = TT$EOM)  .IF TERMINATE CHARACTER         CRTDEM.302
                                                                          .>>>>> 4                $IF      1
716   010012  024432                   R!CHR = CHR$LF       .SEND LINE FEED                 CRTDEM.303
720   077000 /001524                   R = OUTCHAR          .SEND                           CRTDEM.304
722    <macro>                         STORE 1,TT@SEQ,R!TBL .MARK SPECIAL PROCESSING        CRTDEM.305
                                                                                            CRTDEM.306
                                      * TERMINATE - NO LF-CR                                CRTDEM.307
                                                                                            CRTDEM.308
732    <macro>                        $ELSEIF (R!CHR = TT$NLF) .TERMINATE WITH NO EOL       CRTDEM.309
737   077000 /001544                   R = MSGDN            .END OUTPUT MESSAGE             CRTDEM.310
                                                                                            CRTDEM.311
                                      * OUTPUT NEXT CHARACTER                               CRTDEM.312
                                                                                            CRTDEM.313
741    <macro>                        $ELSE                                                 CRTDEM.314
742   077000 /001524                   R = OUTCHAR          .SEND                           CRTDEM.315
744    <macro>                        $ENDIF                                                CRTDEM.316
                                                                          .<<<<< 4                $ENDIF   1
744    <macro>                        $ENDIF                                                CRTDEM.317
                                                                          .<<<<< 3                $ENDIF   1
744    <macro>                        $ENDIF                                                CRTDEM.318
                                                                          .<<<<< 2                $ENDIF   1
                                                                                            CRTDEM.319
                                  * INTERRUPT WAS RESULT OF ECHO                            CRTDEM.320
                                  * - CHECK FOR SPECIAL PROCESSING                          CRTDEM.321
                                                                                            CRTDEM.322
744    <macro>                    $ELSEIF (R!SCO = TT$INP)                                  CRTDEM.323
753   010117  024435              R!MAX = TT$MAX                                            CRTDEM.324
755    <macro>                    GET R!CHR,TT@ECH,R!TBL    .GET ECHOED CHARACTER           CRTDEM.325
763    <macro>                    GET R!SCO,TT@SEQ,R!TBL    .GET SEQUENCE NUMBER OF PROCESS CRTDEM.326
```

```
                                                                                CRTDEM.327
                                 * RUBOUT ONE CHARACTER                         CRTDEM.328
                                                                                CRTDEM.329
 771    <macro>                  $IF (R!CHR = CHR$RUB)                          CRTDEM.330
                                                                  .>>>>> 2            $IF      1
 775   077000 /001352              R = RUBOUT              .USE SUBROUTINE      CRTDEM.331
 777   010177  024432              R!CHR = CHR$RUB         .SAVE FOR NEXT INTERRUPT CRTDEM.332
                                                                                CRTDEM.333
                                 * CONTROL-U                                    CRTDEM.334
                                 * - DELETE LINE                                CRTDEM.335
                                                                                CRTDEM.336
1001    <macro>                  $ELSEIF (R!CHR = CHR$CTU)                      CRTDEM.337
1010   072342                      R = RUBOUT              .USE RUBOUT SUBROUTINE CRTDEM.338
1011    <macro>                    $IF (R!SCO = 0)         .IF RUBOUT COMPLETE  CRTDEM.339
                                                                  .>>>>> 3            $IF      1
1014    <macro>                      $IF (R!NXT # 0)       .IF NOT AT START     CRTDEM.340
                                                                  .>>>>> 4            $IF      1
1017   072333                          R = RUBOUT          .KEEP RUBBING        CRTDEM.341
1020    <macro>                      $ENDIF                                     CRTDEM.342
                                                                  .<<<<< 4            $ENDIF   1
1020    <macro>                    $ENDIF                                       CRTDEM.343
                                                                  .<<<<< 3            $ENDIF   1
1020   010025  024432              R!CHR = CHR$CTU         .SAVE FOR NEXT INTERRUPT CRTDEM.344
                                                                                CRTDEM.345
                                 * CONTROL-A OR CONTROL-X                       CRTDEM.346
                                 * - REPEAT LAST INPUT                          CRTDEM.347
                                 * - EXECUTE IT IF CONTROL-X                     CRTDEM.348
                                                                                CRTDEM.349
1022    <macro>                  $ELSEIF (R!CHR = CHR$CTA) OR (R!CHR = CHR$CTX) CRTDEM.350
1032   020432  024441              R!SCO = R!CHR           .SAVE                CRTDEM.351
1034    <macro>                    GET R!BFR,TT@IBF,R!TBL  .INPUT BUFFER ADDRESS CRTDEM.352
1041    <macro>                $IF (R!BFR # R!USB)                             CRTDEM.353
                                                                  .>>>>> 3            $IF      1
1045    <macro>                  GET R!MAX,TT@SIZ,R!TBL                         CRTDEM.354
1052    <macro>                $ENDIF                                           CRTDEM.355
                                                                  .<<<<< 3            $ENDIF   1
1052    <macro>                    GETBYTE R!CHR,R!NXT,R!USB .GET NEXT CHARACTER CRTDEM.356
1074    <macro>                    $IF (R!SCO = CHR$CTA)   .IF REPEATING ONLY   CRTDEM.357
                                                                  .>>>>> 3            $IF      1
1100    <macro>                      $IF (R!CHR # CHR$LF) AND (R!CHR # 0) .IF NOT END CRTDEM.358
                                                                  .>>>>> 4            $IF      1
1107   077000 /002024                  R = PUTCHAR         .SAVE THE CHARACTER  CRTDEM.359
```

```
111     072413                          R = OUT                 .DETECT END OF BUFFER                   CRTDEM.36
1112     <macro>                        $ENDIF                                                          CRTDEM.361
                                                                                   .<<<<< 4                $ENDIF    1
1112    020432  013012  102003          R!CHR = CHR$CTA, R!CHR # CHR$LF  .IF NOT END OF LINE            CRTDEM.362
        010001  024432
1117     <macro>                      $ELSE                   .CONTROL-X                              CRTDEM.363
1120     <macro>                        $IF (R!CHR = TT$EOM)  .TERMINATE LINE                         CRTDEM.364
                                                                                   .>>>>> 4                $IF       1
1123    010012  024432                  R!CHR = CHR$LF                                                  CRTDEM.365
1125     <macro>                        $ENDIF                                                          CRTDEM.366
                                                                                   .<<<<< 4                $ENDIF    1
1125    077000 /002024                  R = PUTCHAR             .SAVE THE CHARACTER                     CRTDEM.367
1127    072375                          R = OUTCHAR             .SEND                                   CRTDEM.368
1130    020432  013012  102003          R!CHR = CHR$CTX, R!CHR # CHR$LF .IF NOT END OF LINE             CRTDEM.369
        010030  024432
1135     <macro>                      $ENDIF                                                          CRTDEM.370
                                                                                   .<<<<< 3                $ENDIF    1
                                                                                                           CRTDEM.371
                                      * LINE FEED                                                         CRTDEM.372
                                      * - SEND CARRIAGE RETURN                                            CRTDEM.373
                                                                                                           CRTDEM.374
1135     <macro>                      $ELSEIF (R!CHR = CHR$LF)                                            CRTDEM.375
1142    010015  024432                  R!CHR = CHR$CR          .SET TO CARRIAGE RETURN                 CRTDEM.376
1144    072360                          R = OUTCHAR             .OUTPUT TO CRT                          CRTDEM.377
                                                                                                           CRTDEM.378
                                      * CARRIAGE RETURN                                                   CRTDEM.379
                                      * - INPUT COMPLETE / ACTIVATE RECEIVER                              CRTDEM.380
                                                                                                           CRTDEM.381
1145     <macro>                      $ELSEIF (R!CHR = CHR$CR)                                            CRTDEM.382
1152    010000  024436                  R!NXT = 0               .RESET POINTERS                         CRTDEM.383
1154    024437                          R!LST = A                                                       CRTDEM.384
1155     <macro>                        STORE TT$FREE,TT@MOD,R!TBL .FREE THE CRT                        CRTDEM.385
1162     <macro>                        GET R!BFR,TT@IBF,R!TBL  .GET BUFFER ADDRESS                     CRTDEM.386
1167     <macro>                        STORE 0,TT@IBF,R!TBL    .CLEAR INPUT BUFFER                     CRTDEM.387
1174     <macro>                        GET R!SCO,TT@INT,R!TBL  .INPUT INTERRUPT FLAG                   CRTDEM.388
1177     <macro>                        $IF (R!BFR = R!USB) AND (R!SCO = 0) .IF UNSOL. INPUT            CRTDEM.389
                                                                                   .>>>>> 3                $IF       1
                                                                                                           CRTDEM.390
                                        * PASS CRT NUMBER IN FIRST GLOBAL REGISTER                        CRTDEM.391
                                                                                                           CRTDEM.392
1206    010400  054000                  B = [%GBLREG]           .SAVE CURRENT VALUE                     CRTDEM.393
1210    060000  024440                  R!GSV = (B)             .                                       CRTDEM.394
```

```
1212   020430  004001  011003          (B) = R!CHN > 1 & 3      .CRT ORDINAL                    CRTDEM.395
       064000
                                                                                                 CRTDEM.396
                                         * COPY CONTENT OF BUFFER TO PARAMETER REGISTERS         CRTDEM.397
                                                                                                 CRTDEM.398
1216   010444  012001  054000            B = [SC3] + 1            .FIRST PARAMETER REGISTER       CRTDEM.399
1221   010117  012001  004001            R!SC0 = TT$MAX + 1 > 1 .PARCEL COUNT                    CRTDEM.400
       024441
1225    <macro>                          $UNTIL (R!SC0 = 0)       .                               CRTDEM.401
1227   030433  064000                      (B) = (R!BFR)          .COPY                           CRTDEM.402
1231   056000                              B = B + 1              .NEXT                           CRTDEM.403
1232   026433                              R!BFR = R!BFR + 1                                      CRTDEM.404
1233   027441                              R!SC0 = R!SC0 - 1      .DECREMENT COUNT                CRTDEM.405
1234    <macro>                          $ENDTIL                                                 CRTDEM.406
                                                                                                 CRTDEM.407
                                         * CREATE CALL                                           CRTDEM.408
                                                                                                 CRTDEM.409
1235   010055  024441                    R!SC0 = F$CREATE                                        CRTDEM.410
1237   014000 /000000  024442            R!SC1 = O$CALL                                          CRTDEM.411
1242   010010  024443                    R!SC2 = D'8              .Priority                       CRTDEM.412
1244   014000 /177777  024444            R!SC3 = AD$NODEM         .SPECIFY NOT A DEMON CREATE     CRTDEM.413
1247   010441  054000                    B = [SC0]                   .START REG                   CRTDEM.414
1251   010000                            A = 0                    .SPECIFY NO REG SAVE            CRTDEM.415
1252   076000                            R = R!%EX                .CALL CREATE ROUTINE            CRTDEM.416
1253    <macro>                          $PUNTIF (A # 0)          .IF CANT CREATE                 CRTDEM.417
                                                                                                 CRTDEM.418
                                       * RESTORE CURRENT VALUE IN FIRST GLOBAL REGISTER          CRTDEM.419
                                                                                                 CRTDEM.420
1255   010400  054000                  B = [%GBLREG]              .REGISTER NUMBER              CRTDEM.421
1257   020440  064000                  (B) = R!GSV                .RESTORE                      CRTDEM.422
                                                                                                 CRTDEM.423
                                       * RE-ACTIVATE WAITING RECEIVER                            CRTDEM.424
                                                                                                 CRTDEM.425
1261    <macro>                        $ELSEIF (R!BFR # R!USB) .IF NOT FROM INTERRUPTED INPUT    CRTDEM.426
1266   020430  011006  024441            R!SC0 = R!CHN & 6        .POINTER TO QUEUE               CRTDEM.427
1271   002000  000000                    I = 0                                                   CRTDEM.428
1273   014000 /000000  022441            ED = BIQU + R!SC0        .QUEUE                          CRTDEM.429
       024000
1277    <macro>                          RKERN EDQU               .DE-QUEUE                       CRTDEM.430
1301    <macro>                          RKERN EQCP               .QUEUE FOR EXECUTION            CRTDEM.431
1303   003000                            I = 1                                                   CRTDEM.432
                                                                                                 CRTDEM.433
```

```
                                  * START UP MESSAGE                                  CRTDEM.43
                                                                                      CRTDEM.435
1304   072327                     R = MSGNXT            .USE SUBROUTINE              CRTDEM.436
                                                                                      CRTDEM.437
                                 * INTERRUPTED INPUT - START UP QUEUED MESSAGE        CRTDEM.438
                                                                                      CRTDEM.439
1305    <macro>                  $ELSE                                                CRTDEM.440
1306   072325                     R = MSGNXT            .START UP MESSAGE            CRTDEM.441
                                                                                      CRTDEM.442
1307    <macro>                  $ENDIF                                               CRTDEM.443
                                                                        .<<<<< 3            $ENDIF   1
1307    <macro>                $ENDIF                                                 CRTDEM.444
                                                                        :<<<<< 2            $ENDIF   1
                                                                                      CRTDEM.445
                               * IF MESSAGE TO BE STARTED                             CRTDEM.446
                                                                                      CRTDEM.447
1307    <macro>                $ELSEIF (R!SCO = TT$INIT)                              CRTDEM.448
1314   072317                   R = MSGNXT            .START MESSAGE                 CRTDEM.449
                                                                                      CRTDEM.450
1315    <macro>                $ELSE                                                  CRTDEM.451
1316    <macro>                  $PUNTIF                                              CRTDEM.452
1320    <macro>                $ENDIF                                                 CRTDEM.453
                                                                        .<<<<< 1            $ENDIF   1
1320    <macro>              $ENDIF                                                   CRTDEM.454
                                                                        .<<<<< 0            $ENDIF   1
                                                                                      CRTDEM.455
                              * SAVE POINTERS                                         CRTDEM.456
                                                                                      CRTDEM.457
1320    <macro>               PUT R!NXT,TT@PTO,R!TBL                                  CRTDEM.458
1325    <macro>               PUT R!LST,TT@PT1,R!TBL                                  CRTDEM.459
                                                                                      CRTDEM.460
                              * SAVE CHARACTER FOR NEXT OUTPUT INTERRUPT              CRTDEM.461
                                                                                      CRTDEM.462
1332    <macro>               PUT R!CHR,TT@ECH,R!TBL        .SAVE CHARACTER           CRTDEM.463
                                                                                      CRTDEM.464
                              * ENABLE CHANNEL / CHECK QUEUE                          CRTDEM.465
                                                                                      CRTDEM.466
1345   020430  054000         B = R!CHN                     .CHANNEL                  CRTDEM.467
1347   167000                 IOB:7                                                   CRTDEM.468
1350   075000 /000006         P = NEXT                                                CRTDEM.469
                                                                                      CRTDEM.470
                        ************************** SUBROUTINES ******************************  CRTDEM.471
```

```
                                                                                CRTDEM.472
                                 *                                              CRTDEM.473
                                 * RUBOUT ONE CHARACTER                         CRTDEM.474
                                 * - UNIVERSAL SEQUENCE                         CRTDEM.475
                                 * - 0. BACKSPACE                               CRTDEM.476
                                 * - 1. SPACE                                   CRTDEM.477
                                 * - 2. BACKSPACE                               CRTDEM.478
                                 *                                              CRTDEM.479
                                                                                CRTDEM.480
1352                     RUBOUT  *                                              CRTDEM.481
1352   <macro>                   $IF (R!SCO = 0)              .SEQUENCE 0      CRTDEM.482
                                                                     .>>>>> 0        $IF      1
1355   010010  024432              R!CHR = CHR$BS             .BACK SPACE      CRTDEM.483
1357   072145                      R = OUTCHAR                .                CRTDEM.484
1360   <macro>                     FLDADD 1,TT@SEQ,R!TBL      .UPDATE SEQUENCE CRTDEM.485
1403   010001  024441              R!SCO = 1                  .SET NOT DONE    CRTDEM.486
1405   <macro>                   $ELSEIF (R!SCO = 1)          .SPACE DONE      CRTDEM.487
1412   010040  024432              R!CHR = ' 'R               .SPACE           CRTDEM.488
1414   072110                      R = OUTCHAR                .                CRTDEM.489
1415   <macro>                     FLDADD 1,TT@SEQ,R!TBL      .BUMP SEQUENCE   CRTDEM.490
1440   <macro>                   $ELSEIF (R!SCO = 2)          .SEQUENCE 2      CRTDEM.491
1445   010010  024432              R!CHR = CHR$BS             .BACK SPACE      CRTDEM.492
1447   072055                      R = OUTCHAR                .SEND            CRTDEM.493
1450   <macro>                     FLDADD 1,TT@SEQ,R!TBL      .NEXT SEQUENCE   CRTDEM.494
1473   <macro>                   $ELSE                                         CRTDEM.495
1474   <macro>                     STORE 0,TT@SEQ,R!TBL       .DONE - CLEAR SEQUENCER CRTDEM.496
1503   010000  024441              R!SCO = 0                  .SET DONE        CRTDEM.497
1505   027436                      R!NXT = R!NXT - 1          .BACK UP POINTER CRTDEM.498
1506   <macro>                     $IF (R!NXT < R!LST)        .IF BACKING OVER DATA CRTDEM.499
                                                                     .>>>>> 1        $IF      1
1512   027437                        R!LST = R!LST - 1        .BACK UP END POINTER CRTDEM.500
1513   <macro>                     $ENDIF                                      CRTDEM.501
                                                                     .<<<<< 1        $ENDIF   1
                                                                                CRTDEM.502
                                 *  If at the beginning of the line and not inputting a response CRTDEM.503
                                 *  for a MSGR, check for messages to be output.                CRTDEM.504
                                                                                CRTDEM.505
1513   <macro>                     $IF     (R!LST = 0)        .If at beginning of line CRTDEM.506
                                                                     .>>>>> 1        $IF      1
1516   <macro>                       GET   A,TT@IBF,R!TBL                      CRTDEM.507
1522   112111                        R = MSGNXT, A = 0        .If not a MSGR   CRTDEM.508
1523   <macro>                     $ENDIF                                      CRTDEM.509
```

```
                                                                                .<<<<< 1          $EN      1
                                                                                                CRTDEM.510
1523    <macro>                         $ENDIF                                                  CRTDEM.511
                                                                                .<<<<< 0          $ENDIF   1
1523    001000                          EXIT                                                    CRTDEM.512
                                                                                                CRTDEM.513
                                        *                                                       CRTDEM.514
                                        * OUTPUT CHARACTER                                      CRTDEM.515
                                        *                                                       CRTDEM.516
                                                                                                CRTDEM.517
1524                            OUTCHAR *                                                       CRTDEM.518
1524    <macro>                         $IF (0 = R!CHN & 1)          .IF FROM INPUT             CRTDEM.519
                                                                                .>>>>> 0          $IF      1
1533    020430  012001  054000            B = R!CHN + 1              .SET TO OUTPUT             CRTDEM.520
1536    <macro>                         $ELSE                                                   CRTDEM.521
1537    020430  054000                    B = R!CHN                  .FROM OUTPUT               CRTDEM.522
1541    <macro>                         $ENDIF                                                  CRTDEM.523
                                                                                .<<<<< 0          $ENDIF   1
1541    020432                          A = R!CHR                    .CHARACTER                 CRTDEM.524
1542    174000                          IOB:14                                                  CRTDEM.525
1543    001000                          EXIT                                                    CRTDEM.526
                                                                                                CRTDEM.527
                                        *                                                       CRTDEM.528
                                        * MESSAGE COMPLETED                                     CRTDEM.529
                                        * - RETURN CONTROL TO SENDER                            CRTDEM.530
                                        * - UNLESS A RESPONSE IS REQUESTED                      CRTDEM.531
                                        *                                                       CRTDEM.532
                                                                                                CRTDEM.533
1544                            MSGDN   *                                                       CRTDEM.534
1544    010000  024436                  R!NXT = 0                  .RESET BUFFER POINTERS       CRTDEM.535
1546    024437                          R!LST = A                                               CRTDEM.536
1547    <macro>                         STORE 0,TT@OBF,R!TBL       .CLEAR OUTPUT BUFFER POINTER CRTDEM.537
1554    <macro>                         STORE TI$FREE,TT@MOD,R!TBL .SET CRT FREE                CRTDEM.538
1561    020430  011006  024441          R!SCO = R!CHN & 6          .OFFSET TO QUEUE             CRTDEM.539
1564    002000  000000                  I = 0                      .DE-QUEUE FROM OUTPUT        CRTDEM.540
1566    014000 /000000  022441          ED = BOQU + R!SCO          .OUTPUT QUEUE ADDRESS        CRTDEM.541
        024000
1572    <macro>                         RKERN EDQU                 .DE-QUEUE                    CRTDEM.542
1574    <macro>                         $PUNTIF (EH = 0)           .IF PHANTOM MESSAGE SENDER   CRTDEM.543
1577    <macro>                         GET R!BFR,TT@IBF,R!TBL     .INPUT BUFFER                CRTDEM.544
1604    <macro>                         $IF (R!BFR # 0)            .IF A RESPONSE IS REQUESTED  CRTDEM.545
                                                                                .>>>>> 0          $IF      1
```

```
1607    <macro>                          SIMCONIN  R!CHN        .Tell CSIM to start input          CRTDEM.546
1611   014000 /000000  022441            ED = BIQU + R!SC0          .ADDRESS                      CRTDEM.547
       024000
1615    <macro>                          RKERN ENQU                                              CRTDEM.548
1617    <macro>                          STORE TT$INP,TT@MOD,R!TBL  .SET MODE TO INPUT            CRTDEM.549
1624   003000                            I = 1                                                   CRTDEM.550
1625   001000                            EXIT                                                    CRTDEM.551
1626    <macro>                        $ELSE                                                     CRTDEM.552
1627    <macro>                          RKERN EQCP                 .RE-ACTIVATE                  CRTDEM.553
1631   003000                            I = 1                                                   CRTDEM.554
1632   070001                            P = MSGNXT                 .START UP NEXT MESSAGE        CRTDEM.555
1633    <macro>                        $ENDIF                                                    CRTDEM.556
                                                                                .<<<<< 0          $ENDIF    1
                                                                                                  CRTDEM.557
                                       *                                                          CRTDEM.558
                                       * START UP NEXT MESSAGE                                   CRTDEM.559
                                       *                                                          CRTDEM.560
                                                                                                  CRTDEM.561
1633                          MSGNXT   *                                                          CRTDEM.562
1633   020430  011006  024441          R!SC0 = R!CHN & 6          .OFFSET TO QUEUE              CRTDEM.563
1636   014000 /000000  022441          R!SC1 = BOQU + R!SC0       .NEXT MESSAGE SENDER          CRTDEM.564
       024442
1642   030442  024442                  R!SC1 = (R!SC1)            .                             CRTDEM.565
1644    <macro>                        $IF (A # 0)                .IF THERE                     CRTDEM.566
                                                                                .>>>>> 0          $IF       1
1646    <macro>                          GET R!BFR,AD@RC,R!SC1    .GET BUFFER ADDRESS           CRTDEM.567
1653    <macro>                          PUT R!BFR,TT@OBF,R!TBL   .SAVE IN CRT TABLE            CRTDEM.568
1660    <macro>                          STORE TT$OUT,TT@MOD,R!TBL .SET MODE TO OUTPUT          CRTDEM.569
                                                                                                  CRTDEM.570
                                         * IF MESSAGE NEEDS RESPONSE                              CRTDEM.571
                                         * - SET UP RESPONSE PARAMETERS                           CRTDEM.572
                                                                                                  CRTDEM.573
1665    <macro>                          GET R!SC2,AD@FU,R!SC1    .FUNCTION CODE                CRTDEM.574
1672    <macro>                          $IF (R!SC2 = F$MSGR)                                   CRTDEM.575
                                                                                .>>>>> 1          $IF       1
1676    <macro>                            GET R!SC2,AD@P1,R!SC1  .RESPONSE BUFFER LENGTH        CRTDEM.576
1703    <macro>                            PUT R!SC2,TT@SIZ,R!TBL .SAVE                          CRTDEM.577
1710    <macro>                            GET R!SC2,AD@P2,R!SC1  .RESPONSE BUFFER               CRTDEM.578
1715    <macro>                            PUT R!SC2,TT@IBF,R!TBL .SAVE                          CRTDEM.579
1722    <macro>                          $ELSE                                                   CRTDEM.580
1723    <macro>                            STORE 0,TT@IBF,R!TBL   .CLEAR INPUT BUFFER            CRTDEM.581
1730    <macro>                          $ENDIF                                                  CRTDEM.582
```

```
                                                                                .<<<<< 1          $EN      1
                                                                                                CRTDEM.583
                                  * SEND FIRST CHARACTER                                        CRTDEM.584
                                                                                                CRTDEM.585
1730   <macro>                      GETBYTE R!CHR,R!NXT,R!BFR,INC=1 .CHARACTER                  CRTDEM.586
1753   073227                       R = OUTCHAR                 .SEND IT                        CRTDEM.587
1754   <macro>                    $ELSE                                                        CRTDEM.588
1756   <macro>                      STORE TT$FREE,TT@MOD,R!TBL  .SET MODE TO FREE               CRTDEM.589
1763   <macro>                      GET R!SC2,TT@INT,R!TBL      .CHECK IF INPUT WAS INTPTED     CRTDEM.590
1766   <macro>                      STORE 0,TT@INT,R!TBL        .CLEAR FLAG                     CRTDEM.591
1774   <macro>                      $IF (R!SC2 # 0)             .IF FLAG WAS SET                CRTDEM.592
                                                                                .>>>>> 1          $IF      1
2000   002000  000000                 I = 0                     .LOCK                           CRTDEM.593
2002   <macro>                        PUT R!USB,TT@IBF,R!TBL    .ENTER UNSOLICITED BUFFER ADDRESS CRTDEM.594
2007   010001  005010  022430         EH = CHR$CTA < 10 + R!CHN & 177776 .SEND THROUGH A CTL-A  CRTDEM.595
       015000 /177776  024000
2015   014000 /000000  024000         ED = CRTQU                .CRTDEM QUEUE                   CRTDEM.596
2020   <macro>                        RKERN EPOQ                .QUEUE IT                       CRTDEM.597
2022   003000                         I = 1                     .UNLOCK                         CRTDEM.598
2023   <macro>                      $ENDIF                                                       CRTDEM.599
                                                                                .<<<<< 1          $ENDIF   1
2023   <macro>                    $ENDIF                                                       CRTDEM.600
                                                                                .<<<<< 0          $ENDIF   1
2023   001000                     EXIT                                                         CRTDEM.601
                                                                                                CRTDEM.602
                                  *                                                             CRTDEM.603
                                  * ENTER CHARACTER INTO INPUT BUFFER                           CRTDEM.604
                                  *                                                             CRTDEM.605
                                                                                                CRTDEM.606
2024                       PUTCHAR *                                                            CRTDEM.607
2024   <macro>                    $IF (R!NXT >= R!MAX - 1)    .IF LAST CHARACTER POSITION     R15961AA.2
                                                                                .>>>>> 0          $IF      1
2033   010012  024432               R!CHR = CHR$LF              .TERMINATE MESSAGE              CRTDEM.609
2035   <macro>                    $ENDIF                                                       CRTDEM.610
                                                                                .<<<<< 0          $ENDIF   1
2035   020432  013012  103003     R!NXT = R!LST, R!CHR = CHR$LF .IF TERMINATING INPUT          CRTDEM.611
       020437  024436
2041   <macro>                    PUTBYTE R!CHR,R!NXT,R!USB   .SAVE COPY OF INPUT             CRTDEM.612
                                                                                                CRTDEM.613
2042   <macro>                    PUTBYTE R!CHR,R!NXT,R!BFR                                    CRTDEM.614
2049   020436  023435  101002     R!NXT = R!NXT + 1, R!NXT < R!MAX .Dont increment if at max   CRTDEM.615
       026436
```

```
2126   020437  023436  101002        R!LST = R!LST + 1, R!LST < R!NXT   .IF PAST END OF DATA        CRTDEM.616
       026437
2132   001000                        EXIT                                                          CRTDEM.617
                                                                                                    CRTDEM.618
                                     END                                                           CRTDEM.619
```

0X	%B		1: 4 D	6:21 B	B	7:23 B	8:41 B	9: 8 B	10: 9 B	10:25 B	
			10:32 B	11:17 B	11:23 B	11:28 B	11:39 B	12: 6 B	12:12 B	12:42 B	
			13:12 B	15:40 B	19: 7 B	19:12 B					
0X	%EX		3: 5 D	14:24							
10	%GBLNUM	$APTEXT	2:15 F	2:15							
400	%GBLREG	$APTEXT	2:15	13:41	14:29						
0	%KERNEL	$APTEXT	4: 8 D	4: 8 B	4:12 D	4:12 B	5:13 D	5:13 B	5:20 D	5:20 B	
			5:22 D	5:22 B	9:26 D	9:26 B	9:27 D	9:27 B	14:39 D	14:39 B	
			14:40 D	14:40 B	17:38 D	17:38 B	18: 4 D	18: 4 B	18: 9 D	18: 9 B	
			19:18 D	19:18 B							
0X	%PUNT		14:25 D	14:25	15:21 D	15:21	17:39 D	17:39			
0X	%TR1		11: 9 D	11: 9							
0X	%TR2		11: 9 D	11: 9							
0X	%TR3		11: 9 D	11: 9							
0X	%TR4		11: 9 D	11: 9							
0X	%TR5		11: 9 D	11: 9							
0X	%TRAC		11: 9 D	11: 9							
0X	%TVNT		11: 9 D	11: 9							
177777	AD$NODEM	$APTEXT	14:21								
20	AD@FU@N	$APTEXT	9:29	9:29 F	18:33	18:33 F					
12	AD@FU@P	$APTEXT	9:29 F	9:29	18:33 F	18:33					
0	AD@FU@S	$APTEXT	9:29	9:29 F	18:33	18:33 F					
20	AD@P1@N	$APTEXT	5:23 F	9:20	9:20 F	9:35 F	18:36	18:36 F			
14	AD@P1@P	$APTEXT	5:19	5:23 F	5:23	9:20 F	9:20	9:35 F	9:35	18:36 F	
			18:36								
0	AD@P1@S	$APTEXT	5:23 F	9:20	9:20 F	18:36	18:36 F				
20	AD@P2@N	$APTEXT	18:38	18:38 F							
15	AD@P2@P	$APTEXT	18:38 F	18:38							
0	AD@P2@S	$APTEXT	18:38	18:38 F							
20	AD@RC@N	$APTEXT	5:24 F	9:23 F	9:30	9:30 F	9:34 F	18:26	18:26 F		
13	AD@RC@P	$APTEXT	5:24 F	5:24	9:23 F	9:23	9:30 F	9:30	9:34 F	9:34	
			18:26 F	18:26							
0	AD@RC@S	$APTEXT	5:24 F	9:23 F	9:30	9:30 F	18:26	18:26 F			
0X	BCRT		3: 6 D	4:30							
0X	BIQU		3: 7 D	14:37	18: 2						
0X	BOQU		3: 8 D	11: 6	17:36	18:21					
3	CI$AMPEX	$APTEXT	7:42								
7	CHR$BEL	$APTEXT	6:28	6:37	7:17	8: 9	8:33				
10	CHR$BS	$APTEXT	6:34	16:13	16:22						
15	CHR$CR	$APTEXT	7:10	10:24	10:30	13:23	13:29				
1	CHR$CTA	$APTEXT	7: 5	12:29	12:39	13: 4	19:15				
14	CHR$CTL	$APTEXT	8: 6								

25	CHR$CTU	$APTEXT	6:27	12:12	12:23					
30	CHR$CTX	$APTEXT	7: 5	12:29	13:14					
33	CHR$ES	$APTEXT	6:20							
12	CHR$LF	$APTEXT	6:22	7:10	7:11	10:23	11:16	12:41	13: 4	13: 9
			13:14	13:22	19:33	19:36				
177	CHR$RUB	$APTEXT	7:16	12: 5	12: 7					
0X	CRTQU		3: 9 D	4: 7	19:17					
0X	DQTIME		5:20 D	5:20						
0X	ED		3:10 D	4: 7	5:12	9:25	14:37	17:36	18: 2	19:17
0X	EDQU		5:13 D	5:13	9:26 D	9:26	14:39 D	14:39	17:38 D	17:38
0X	EH		3:11 D	4: 9	5:15	5:18	5:19	5:21	5:23	5:24
			17:39	19:16						
0X	ENQU		18: 4 D	18: 4						
0X	EPOQ		19:18 D	19:18						
0X	EQCP		5:22 D	5:22	9:27 D	9:27	14:40 D	14:40	18: 9 D	18: 9
0X	ESWP		4:12 D	4:12						
0X	ETOQ		4: 8 D	4: 8						
55	F$CREATE	$APTEXT	14:18							
21	F$MSGR	$APTEXT	18:35							
0X	O$CALL		3:12 D	14:19						
0X	O$CRTDEM		1: 4 D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
0	PT$HUH	$APTEXT	14:25	15:21	17:39					
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
17	TF$CHAR	$APTEXT	11: 9							
0	TT$EOM	$APTEXT	11:15	13: 8						
0	TT$FREE	$APTEXT	13:32	17:33	19: 8					
3	TT$INIT	$APTEXT	15:17							
1	TT$INP	$APTEXT	5:38	11:39	18: 5					
117	TT$MAX	$APTEXT	6: 3	6: 4	11:40	14: 7				
377	TT$NLF	$APTEXT	11:22							
2	TT$OUT	$APTEXT	5:34	10: 9	18:28					
10	TT$OWN	$APTEXT	5:10	9:14						
377	TT@ECH@M	$APTEXT	5:16	15:34						
10	TT@ECH@N	$APTEXT	5:16 F	5:16	10:16	10:16 F	11:41	11:41 F	15:34 F	15:34
3	TT@ECH@P	$APTEXT	5:16 F	5:16	10:16 F	10:16	11:41 F	11:41	15:34 F	15:34
0	TT@ECH@S	$APTEXT	5:16 F	5:16	10:16	10:16 F	11:41	11:41 F	15:34 F	15:34
20	TT@IBF@N	$APTEXT	5:39	5:39 F	6: 2 F	12:31	12:31 F	13:33	13:33 F	13:34 F
			16:40	16:40 F	17:40	17:40 F	18:39 F	18:41 F	19:14 F	
2	TT@IBF@P	$APTEXT	5:39 F	5:39	6: 2 F	6: 2	12:31 F	12:31	13:33 F	13:33

			13:34 F	13:34	F	16:40	17:40 F	17:40	18:39 F	18:39
			18:41 F	18:41	19:14 F	19:14				
0	TT@IBI@S	$APIEXT	5:39	5:39 F	6: 2 F	12:31	12:31 F	13:33	13:33 F	13:34 F
			16:40	16:40 F	17:40	17:40 F	18:39 F	18:41 F	19:14 F	
1	TT@INI@M	$APIEXT	19:10 F							
1	TT@INI@N	$APIEXT	13:35	13:35 F	19: 9	19: 9 F	19:10 F	19:10		
0	TT@INI@P	$APIEXT	13:35 F	19: 9 F	19:10 F					
0	TT@INI@S	$APIEXT	13:35	13:35 F	19: 9	19: 9 F	19:10 F	19:10		
15	TT@IQ@P	$APIEXT	5:12							
20	TT@MOD@N	$APIEXT	5: 8	5: 8 F	5:38 F	9:12	9:12 F	13:32 F	17:33 F	18: 5 F
			18:28 F	19: 8 F						
4	TT@MOD@P	$APIEXT	5: 8 F	5: 8	5:38 F	5:38	9:12 F	9:12	13:32 F	13:32
			17:33 F	17:33	18: 5 F	18: 5	18:28 F	18:28	19: 8 F	19: 8
0	TT@MOD@S	$APIEXT	5: 8	5: 8 F	5:38 F	9:12	9:12 F	13:32 F	17:33 F	18: 5 F
			18:28 F	19: 8 F						
20	TT@OBI@N	$APIEXT	10:10	10:10 F	17:32 F	18:27 F				
1	TT@OBI@P	$APIEXT	10:10 F	10:10	17:32 F	17:32	18:27 F	18:27		
0	TT@OBI@S	$APIEXT	10:10	10:10 F	17:32 F	18:27 F				
20	TT@OQ@N	$APIEXT	9:17	9:17 F						
13	TT@OQ@P	$APIEXT	9:17 F	9:17	9:25					
0	TT@OQ@S	$APIEXT	9:17	9:17 F						
20	TT@PTO@N	$APIEXT	4:37	4:37 F	15:29 F					
6	TT@PTO@P	$APIEXT	4:37 F	4:37	15:29 F	15:29				
0	TT@PTO@S	$APIEXT	4:37	4:37 F	15:29 F					
20	TT@PT1@N	$APIEXT	4:38	4:38 F	15:30 F					
7	TT@PT1@P	$APIEXT	4:38 F	4:38	15:30 F	15:30				
0	TT@PT1@S	$APIEXT	4:38	4:38 F	15:30 F					
377	TT@SEQ@M	$APIEXT	10:17	10:31 F	11:18 F	11:42	16:15	16:20	16:24	16:26 F
10	TT@SEQ@N	$APIEXT	10:17	10:31 F	11:18 F	11:42	16:15 F	16:15	16:20 F	16:20
			16:24 F	16:24	16:26 F					
3	TT@SEQ@P	$APIEXT	10:17 F	10:17	10:31 F	10:31	11:18 F	11:18	11:42 F	11:42
			16:15 F	16:15	16:20 F	16:20	16:24 F	16:24	16:26 F	16:26
10	TT@SEQ@S	$APIEXT	10:17	10:17 F	10:31 F	11:18 F	11:42	11:42 F	16:15	16:15 F
			16:20	16:20 F	16:24	16:24 F	16:26 F			
177400	TT@SEQ@X	$APIEXT	10:31	11:18	16:15	16:20	16:24	16:26		
20	TT@SIZ@N	$APIEXT	5:40	5:40 F	6: 4 F	12:34	12:34 F	18:37 F		
5	TT@SIZ@P	$APIEXT	5:40 F	5:40	6: 4 F	6: 4	12:34 F	12:34	18:37 F	18:37
0	TT@SIZ@S	$APIEXT	5:40	5:40 F	6: 4 F	12:34	12:34 F	18:37 F		
20	TT@TYP@N	$APIEXT	7:37	7:37 F						
11	TT@TYP@P	$APIEXT	7:37 F	7:37						
0	TT@TYP@S	$APIEXT	7:37	7:37 F						
20	TT@USB@N	$APIEXT	4:39	4:39 F						

```
 10   TT@USB@P  $APTEXT      4:39 F     4:39
  0   TT@USB@S  $APTEXT      4:39       4:39 F

      SYMBOL QUALIFIER IS CRTDEM

  0   $CONST                 5:24 D     5:24 F     5:38 D     5:38 F     5:38       6: 4 D     6: 4 F     6: 4
                             9:23 D     9:23 F    10:31 D    10:31 F    11:18 D    11:18 F    11:18      13:32 D
                            13:32 F    13:34 D    13:34 F    16:26 D    16:26 F    17:32 D    17:32 F    17:33 D
                            17:33 F    18: 5 D    18: 5 F    18: 5      18:28 D    18:28 F    18:28      18:41 D
                            18:41 F    19: 8 D    19: 8 F    19:10 D    19:10 F    19:10
445   $REGORG                2:15 D     2:15 F     2:16       2:16 D     2:17       2:17 D     2:18       2:18 D
                             2:19       2:19 D     2:20       2:20 D     2:20 F     2:21       2:21 D     2:22
                             2:22 D     2:23       2:23 D     2:24       2:24 D     2:25       2:25 D     2:26
                             2:26 D     2:26 F     2:27       2:27 D     2:28       2:28 D     2:29       2:29 D
                             2:30       2:30 D     2:31       2:31 D     2:31 F     2:32       2:32 D     2:33
                             2:33 D     2:34       2:34 D     2:35       2:35 D     2:36       2:36 D     2:37
                             2:37 D     2:38       2:38 D     2:39       2:39 D     2:40       2:40 D     2:41
                             2:41 D     2:42       2:42 D     3: 1       3: 1 D     3: 2       3: 2 D     3: 2 F
  0   %NP                    1: 4       2:31 D
430   %P                     1: 4       2:31 D     2:31
410   %S1                    2:15       2:16 D     4:37       4:38       4:39       5: 4       5: 8       5:16
                             5:23       5:24       5:38       5:39       5:40       6: 2       6: 4       7:37
                             8: 8       9:12       9:17       9:20       9:23       9:29       9:30       9:32
                             9:34       9:35      10:10      10:16      10:17      10:31      10:37      11:18
                            11:41      11:42      12;31      12:34      12:37      13:32      13:33      13:34
                            15:29      15:30      15:34      16:26      16:40      17:13      17:32      17:33
                            17:40      18: 5      18:26      18:27      18:28      18:33      18:36      18:37
                            18:38      18:39      18:41      19: 5      19: 8      19:14      19:32      19:38
                            19:40
411   %S2                    2:15       2:17 D
412   %S3                    2:15       2:18 D
413   %S4                    2:15       2:19 D
414   %S5                    2:15       2:20 D
415   %T1                    2:21 D     9:32      10:37      12:37      16:15      16:20      16:24      19: 5
                            19:38      19:40
416   %T2                    2:22 D    16:15      16:20      16:24      19:38      19:40
417   %T3                    2:23 D    16:15      16:20      16:24
420   %T4                    2:24 D
421   %T5                    2:25 D
422   %T6                    2:26 D
423   %W1                    2:27 D     5:18       5:21
424   %W2                    2:28 D
```

425	%W3	2:29 D							
426	%W4	2:30 D							
427	%W5	2:31 D							
433	BFR	2:35 D	5:39	5:42	6: 1	6: 2	10:10	10:12	10:37
		11: 9	12:31	12:33	13:33	13:37	14:10	14:12	14:34
		17:40	17:42	18:26	18:27	19: 5	19:40		
430	CHN	2:32 D	4: 9	4:21	4:23	4:29	5: 4	5:25	9:31
		10: 3	11: 6	11: 9	14: 1	14:35	15:38	17:13	17:14
		17:16	17:34	18: 1	18:20	19:15			
432	CHR	2:34 D	4:21	5:16	5:23	6:20	6:22	6:27	6:28
		6:34	6:37	7: 5	7:10	7:11	7:16	7:17	7:22
		8: 6	8: 9	8:33	10:16	10:23	10:24	10:30	10:37
		11: 9	11:15	11:16	11:22	11:41	12: 5	12: 7	12:12
		12:23	12:29	12:30	12:37	12:41	13: 4	13: 5	13: 8
		13: 9	13:14	13:15	13:22	13:23	13:29	15:34	16:13
		16:18	16:22	17:19	19: 5	19:33	19:36	19:38	19:40
440	GSV	2:40 D	13:42	14:30					
437	LST	2:39 D	4:38	8:12	13:31	15:30	16:30	16:31	16:39
		17:31	19:36	20: 1					
435	MAX	2:37 D	5:40	6: 3	8: 8	11:40	12:34	19:32	19:41
1544	MSGDN	10:32	11:23	17:29 D					
1633	MSGNXT	15: 3	15: 8	15:18	16:41	18:11	18:19 D		
6	NEXT	4: 5 D	5:27	10: 5	15:40				
436	NXT	2:38 D	4:37	6:28	6:36	6:39	7:17	8: 8	8:11
		8:12	10:37	12:17	12:37	13:30	15:29	16:28	16:30
		17:30	19: 5	19:32	19:37	19:38	19:40	19:41	20: 1
1524	OUTCHAR	8:41	10:25	11:17	11:28	13: 1	13:13	13:24	16:14
		16:19	16:23	17:11 D	19: 6				
2024	PUTCHAR	6:21	7:12	7:23	12:42	13:12	19:30 D		
1352	RUBOUT	12: 6	12:13	12:18	16:10 D				
441	SC0	2:41 D	5: 8	5:10	5:34	6:14	7:28	7:42	8: 1
		8:20	8:32	9:12	9:14	10: 9	10:17	10:23	10:30
		11: 7	11: 8	11: 9	11:39	11:42	12:15	12:30	12:39
		13:35	13:37	14: 7	14: 9	14:13	14:18	14:22	14:35
		14:37	15:17	16:12	16:16	16:17	16:21	16:27	17:34
		17:36	18: 2	18:20	18:21				
442	SC1	2:42 D	7:37	7:42	9:15	9:16	9:17	9:19	9:20
		9:23	9:29	9:30	9:34	9:35	9:36	14:19	18:21
		18:23	18:26	18:33	18:36	18:38			
443	SC2	3: 1 D	9:20	9:22	9:29	9:32	14:20	18:33	18:35
		18:36	18:37	18:38	18:39	19: 9	19:12		
444	SC3	3: 2 D	9:30	9:32	14: 6	14:21			

431	TBL	2:33	D	4:30	4:31	4:37	4:38	4:39	5: 8	5:12
		5:16		5:38	5:39	5:40	6: 2	6: 4	7:37	9:12
		9:17		9:25	10:10	10:16	10:17	10:31	11:18	11:41
		11:42		12:31	12:34	13:32	13:33	13:34	13:35	15:29
		15:30		15:34	16:15	16:20	16:24	16:26	16:40	17:32
		17:33		17:40	18: 5	18:27	18:28	18:37	18:39	18:41
		19: 8		19: 9	19:10	19:14				
434	USB	2:36	D	4:39	6: 1	12:33	12:37	13:37	14:34	19:14
		19:38								

```
<macro>                    OVERLAY INDD29                                    INDD29.4

                           COMMENT   'Copyright (C) Cray Research, Inc., 1984'      OVERLAY  1
```

```
                    ***********************************************************************  INDD29.5
                    *                                                                     *  INDD29.6
                    * NAME: INDD29                                                        *  INDD29.7
                    *                                                                     *  INDD29.8
                    * PURPOSE: INITIALIZE DD29/19 FOR USE BY THE CURRENT IOP. SET UP      *  INDD29.9
                    *          TABLES, INTERRUPT ADDRESSES AND RUN MINIMAL RELIABILITY    *  INDD29.10
                    *          CHECKS ON CHANNEL AND DEVICE.                              *  INDD29.11
                    *                                                                     *  INDD29.12
                    * ENTRY: LOCAL ADDRESS OF AMAP.                                       *  INDD29.13
                    *                                                                     *  INDD29.14
                    ***********************************************************************  INDD29.15
                                                                                             INDD29.16
                             BLOCK MESSAGE                                                   INDD29.17
                    ***********************************************************************  INDD29.18
                                                                                             INDD29.19
1434                MESS0    *                                                               INDD29.20
1434                         'CHANNEL XX CHANNEL TIMEOUT'                                    INDD29.21
1452                MESS1    *                                                               INDD29.22
1452                         'CHANNEL XX CHANNEL ERROR'                                      INDD29.23
1467                MESS2    *                                                               INDD29.24
1467                         'CHANNEL XX CHANNEL DATA ERROR'                                 INDD29.25
1506                MESS3    *                                                               INDD29.26
1506                         'CHANNEL XX STATUS REGISTER ERROR'                              INDD29.27
1527                MESS4    *                                                               INDD29.28
1527                         'CHANNEL XX ADDRESS REGISTER ERROR'                             INDD29.29
1550                MESS5    *                                                               INDD29.30
1550                         'CHANNEL XX DEVICE SELECT ERROR'                                INDD29.31
1570                MESS6    *                                                               INDD29.32
1570                         'EXPECTED:          RECEIVED:        '                          INDD29.33
                                                                                             INDD29.34
                 12 EXPOFF   EQUALS   D'10                                                   INDD29.35
                 34 RECOFF   EQUALS   D'28                                                   INDD29.36
                  4 CHANOFF  EQUALS   4                                                      INDD29.37
                764 PASSES   EQUALS   D'500                                                  INDD29.38
                 21 NCHNS    EQUALS   D'16+1                                                 INDD29.39
                 41 NBUFS    EQUALS   D'32+1                                                 INDD29.40
                 50 MSGLNG   EQUALS   D'40                                                   INDD29.41
                                                                                             INDD29.42
                154 $$X      SET      RA@LEN@P*RA$NUMS                                       INDD29.43
                154 MTBLE    EQUALS   $$X                                                    INDD29.44
                 22 $$X      SET      RA@LEN@P*RA$NUM                                        INDD29.45
                 22 STBLE    EQUALS   $$X                                                    INDD29.46
```

```
                                                                                                INDD29.47
                                ***************************************************************** INDD29.48
                                        BLOCK *                                                 INDD29.49
                                                                                                INDD29.50
                                        EXT    IDID                                             INDD29.51
                                        EXT    DCCB                                             INDD29.52
                                        EXT    EITB                                             INDD29.53
                                        EXT    %MYID                                            INDD29.54
                                        EXT    IDKTOUT                                          INDD29.55
                                                                                                INDD29.56
6     <macro>                           REGDEFS ,(AMAP),(CHTBLE,BUFTBLE,CHENT,BUFENT,ERR,SCO,RANUM,  INDD29.57
                                                 RALEN,MEM,SIZE,PASS,TX0,TX1,TX2,DATAO,DATAI)   INDD29.58
                410     %S1     EQUALS   $REGORG                                                $REG    3
                411     %S2     EQUALS   $REGORG                                                $REG    3
                412     %S3     EQUALS   $REGORG                                                $REG    3
                413     %S4     EQUALS   $REGORG                                                $REG    3
                414     %S5     EQUALS   $REGORG                                                $REG    3
                415     %T1     EQUALS   $REGORG                                                $REG    3
                416     %T2     EQUALS   $REGORG                                                $REG    3
                417     %T3     EQUALS   $REGORG                                                $REG    3
                420     %T4     EQUALS   $REGORG                                                $REG    3
                421     %T5     EQUALS   $REGORG                                                $REG    3
                422     %T6     EQUALS   $REGORG                                                $REG    3
                423     %W1     EQUALS   $REGORG                                                $REG    3
                424     %W2     EQUALS   $REGORG                                                $REG    3
                425     %W3     EQUALS   $REGORG                                                $REG    3
                426     %W4     EQUALS   $REGORG                                                $REG    3
                427     %W5     EQUALS   $REGORG                                                $REG    3
                430     AMAP    EQUALS   $REGORG                                                $REG    3
                431     CHTBLE  EQUALS   $REGORG                                                $REG    3
                432     BUFTBLE EQUALS   $REGORG                                                $REG    3
                433     CHENT   EQUALS   $REGORG                                                $REG    3
                434     BUFENT  EQUALS   $REGORG                                                $REG    3
                435     ERR     EQUALS   $REGORG                                                $REG    3
                436     SCO     EQUALS   $REGORG                                                $REG    3
                437     RANUM   EQUALS   $REGORG                                                $REG    3
                440     RALEN   EQUALS   $REGORG                                                $REG    3
                441     MEM     EQUALS   $REGORG                                                $REG    3
                442     SIZE    EQUALS   $REGORG                                                $REG    3
                443     PASS    EQUALS   $REGORG                                                $REG    3
                444     TX0     EQUALS   $REGORG                                                $REG    3
                445     TX1     EQUALS   $REGORG                                                $REG    3
```

```
                446     TX2     EQUALS    $REGORG                                                  $REG    3
                447     DATAO   EQUALS    $REGORG                                                  $REG    3
                450     DATAI   EQUALS    $REGORG                                                  $REG    3
                                                                                              INDD29.59
                                *                                                             INDD29.60
                                * ALLOCATE TABLES FOR CHANNEL TEST                            INDD29.61
                                *                                                             INDD29.62
                                                                                              INDD29.63
  6  010021  024442            R!SIZE = NCHNS                .CHANNEL TABLE                  INDD29.64
 10  072673                    R = GMEM                      .ALLOCATE                       INDD29.65
 11  020441  024431            R!CHTBLE = R!MEM              .SAVE                           INDD29.66
 13  010041  024442            R!SIZE = NBUFS                .BUFFER ADDRESS TABLE           INDD29.67
 15  072666                    R = GMEM                      .ALLOCATE                       INDD29.68
 16  020441  024432            R!BUFTBLE = R!MEM             .SAVE                           INDD29.69
 20  010000  024447            R!DATAO = 0                   .CLEAR DATA REGS                INDD29.70
 22  010000  024450            R!DATAI = 0                   .                               INDD29.71
                                                                                              INDD29.72
                                *                                                             INDD29.73
                                * LOCATE ALL DD-19/29 CHANNELS                                INDD29.74
                                * - BUILD DCB                                                 INDD29.75
                                * - ENTER INTERRUPT ADDRESS                                   INDD29.76
                                * - ENTER IN CHANNEL TEST TABLE                               INDD29.77
                                *                                                             INDD29.78
                                                                                              INDD29.79
 24  020431  024433            R!CHENT = R!CHTBLE            .CHANNEL TEST TABLE             INDD29.80
 26  020430  012006  012005    R!%W1 = R!AMAP + OV@ENT@P + 5 + R!%MYID  .FIND THIS IOPS AREA INDD29.81
     022000  024423
 33  030423  022430  024423    R!%W1 = (R!%W1) + R!AMAP      .                               INDD29.82
 36   <macro>                  $PUNTIF (A = R!AMAP)          .IF THIS IOP NOT CONFIGURED     INDD29.83
 41  010006  025423            R!%W1 = R!%W1 + 6             .POINTER TO CHANNEL INFO        INDD29.84
 43  030423  022430  024423    R!%W1 = (R!%W1) + R!AMAP      .                               INDD29.85
 46  030423  024424            R!%W2 = (R!%W1)               .LAST CHANNEL NUMBER            INDD29.86
 50  010006  024425            R!%W3 = 6                     .FIRST CHANNEL NUMBER           INDD29.87
 52   <macro>                  $UNTIL (R!%W3 = R!%W2)        .CHECK ALL CHANNELS             INDD29.88
 55  026423                      R!%W1 = R!%W1 + 1           .NEXT ENTRY                     INDD29.89
 56  030423  011777  024426      R!%W4 = (R!%W1) & 777       .CHANNEL TYPE                   INDD29.90
 61   <macro>                    $IF (R!%W4 = CH$D1) OR (R!%W4 = CH$D2) .DD19/29             INDD29.91
                                                                          .>>>>> 0                $IF    1
 70  072651                        R = SETUP                 .SET UP CHANNEL                 INDD29.92
 71   <macro>                    $ELSEIF (R!%W4 = CH$D1S) OR (R!%W4 = CH$D2S) .STRIPED       INDD29.93
101  072640                        R = SETUP                 .SET UP FOR STRIPING            INDD29.94
102   <macro>                    $ENDIF                                                      INDD29.95
```

```
                                                                                .<<<<< 0          $EN      1
102   026425                        R!%W3 = R!%W3 + 1          .NEXT CHANNEL                    INDD29.96
103    <macro>                    $ENDTIL                                                         INDD29.97
                                                                                                   INDD29.98
                                  *                                                                INDD29.99
                                  * ALLOCATE BUFFERS FOR CHANNEL TEST                             INDD29.100
                                  * - 2 512 PARCEL BUFFERS PER CHANNEL                            INDD29.101
                                  *                                                                INDD29.102
                                                                                                   INDD29.103
104   020431  024423              R!%W1 = R!CHTBLE           .CHANNEL TABLE                       INDD29.104
106   020432  024424              R!%W2 = R!BUFTBLE          .BUFFER TABLE                        INDD29.105
110   030423  024425              R!%W3 = (R!%W1)            .FIRST CHANNEL                       INDD29.106
112   010000  024426              R!%W4 = 0                  .BUFFER ADDRESS                      INDD29.107
114    <macro>                    $UNTIL (R!%W3 = 0)         .FOR EACH CHANNEL                    INDD29.108
116    <macro>                      $IF (0 = R!%W4 & 3777)   .IF NO MORE SPACE                    INDD29.109
                                                                                .>>>>> 0          $IF      1
126    <macro>                        BGET R!%W4             .GET ANOTHER BUFFER                  INDD29.110
136    <macro>                        $PUNTIF (A # 0)        .IF NONE                             INDD29.111
140    <macro>                      $ENDIF                                                        INDD29.112
                                                                                .<<<<< 0          $ENDIF   1
140   020426  034424                (R!%W2) = R!%W4            .ENTER ADDRESS                       INDD29.113
142   026424                        R!%W2 = R!%W2 + 1          .READ BUFFER                         INDD29.114
143   014000 /001000  025426        R!%W4 = R!%W4 + D'512      .ADDRESS                             INDD29.115
146   020426  034424                (R!%W2) = R!%W4            .ENTER                               INDD29.116
150   026423                        R!%W1 = R!%W1 + 1          .NEXT CHANNEL                        INDD29.117
151   026424                        R!%W2 = R!%W2 + 1          .NEXT BUFFER                         INDD29.118
152   030423  024425                R!%W3 = (R!%W1)            .                                    INDD29.119
154   014000 /001000  025426        R!%W4 = R!%W4 + D'512      .NEXT ADDRESS                        INDD29.120
157    <macro>                    $ENDTIL                                                         INDD29.121
                                                                                                   INDD29.122
                                  *                                                                INDD29.123
                                  * RUN BUFFER ECHO CHANNEL TEST                                   INDD29.124
                                  *                                                                INDD29.125
                                                                                                   INDD29.126
                                  * FILL WRITE BUFFERS WITH REAL TIME CLOCK VALUES                 INDD29.127
                                                                                                   INDD29.128
160   010764  024443              R!PASS = PASSES            .INITIALIZE PASS COUNT               INDD29.129
162                       ECHO    *                                                                INDD29.130
162   020431  024433              R!CHENT = R!CHTBLE         .FIRST ENTRY IN CHANNEL TABLE        INDD29.131
164   020432  024434              R!BUFENT = R!BUFTBLE       .FIRST ENTRY IN BUFFER TABLE         INDD29.132
166    <macro>                    $UNTIL ((R!CHENT) = 0)     .FOR EACH CHANNEL                    INDD29.133
170    <macro>                      $IF (0 = (R!CHENT) & 100000)  .IF CHANNEL ACTIVE              INDD29.134
```

```
                                                                    .>>>>> 0          $IF      1
200   030434  024423                R!%W1 = (R!BUFENT)        .WRITE BUFFER                INDD29.135
202   020434  012001  024410        R!%W2 = (R!BUFENT + 1)    .READ BUFFER                 INDD29.136
      030410  024424
207   014000 /001000  024425        R!%W3 = D'512             .LENGTH                      INDD29.137
212    <macro>                      $UNTIL (R!%W3 = 0)        .INITIALIZE BUFFERS          INDD29.138
214   150004                          RTC:10                  .CURRENT CLOCK               INDD29.139
215   034423                          (R!%W1) = A             .ENTER                       INDD29.140
216   010000  034424                  (R!%W2) = 0             .CLEAR                       INDD29.141
220   026423                          R!%W1 = R!%W1 + 1       .NEXT                        INDD29.142
221   026424                          R!%W2 = R!%W2 + 1       .                            INDD29.143
222   027425                          R!%W3 = R!%W3 - 1       .                            INDD29.144
223    <macro>                      $ENDTIL                                                INDD29.145
224    <macro>                    $ENDIF                                                   INDD29.146
                                                                    .<<<<< 0          $ENDIF   1
224   026433                       R!CHENT = R!CHENT + 1       .NEXT CHANNEL                INDD29.147
225   010002  025434               R!BUFENT = R!BUFENT + 2     .NEXT BUFFERS                INDD29.148
227    <macro>                   $ENDTIL                                                   INDD29.149
                                 *                                                          INDD29.150
                                                                                            INDD29.151
                                 * WRITE ALL THE CHANNEL BUFFERS                            INDD29.152
                                                                                            INDD29.153
230   020431  024433             R!CHENT = R!CHTBLE            .FIRST ENTRY IN CHANNEL TABLE INDD29.154
232   020432  024434             R!BUFENT = R!BUFTBLE          .FIRST ENTRY IN BUFFER TABLE  INDD29.155
234    <macro>                   $UNTIL ((R!CHENT) = 0)        .ALL                         INDD29.156
236    <macro>                     $IF (0 = (R!CHENT) & 100000) .IF ENABLED                 INDD29.157
                                                                    .>>>>> 0          $IF      1
246   030433  054000                B = (R!CHENT)             .CHANNEL NUMBER              INDD29.158
250   030434                        A = (R!BUFENT)            .BUFFER ADDRESS              INDD29.159
251   174000                        IOB:14                    .ENTER                       INDD29.160
252   163000                        IOB:3                     .START WRITE                 INDD29.161
253    <macro>                    $ENDIF                                                   INDD29.162
                                                                    .<<<<< 0          $ENDIF   1
253   026433                       R!CHENT = R!CHENT + 1       .NEXT CHANNEL                INDD29.163
254   010002  025434               R!BUFENT = R!BUFENT + 2     .NEXT BUFFER                 INDD29.164
256    <macro>                   $ENDTIL                                                   INDD29.165
                                                                                            INDD29.166
                                 * READ ALL CHANNELS BACK                                   INDD29.167
                                                                                            INDD29.168
257   020431  024433             R!CHENT = R!CHTBLE            .FIRST CHANNEL               INDD29.169
261   020432  012001  024434     R!BUFENT = R!BUFTBLE + 1      .FIRST BUFFER                INDD29.170
264    <macro>                   $UNTIL ((R!CHENT) = 0)        .ALL CHANNELS                INDD29.171
```

```
266    <macro>                          $IF (0 = (R!CHEN     100000) .IF ENABLED                    INDD29.172
                                                                                  .>>>>> 0            $IF      1
276    030433  054000                     B = (R!CHENT)              .CHANNEL NUMBER                INDD29.173
300    077000 /001164                     R = WAITDN                 .WAIT CHANNEL DONE             INDD29.174
302    <macro>                            $IF (R!ERR = 0)            .IF NO ERROR                   INDD29.175
                                                                                  .>>>>> 1            $IF      1
                                                                                                     INDD29.176
                                            * VALIDATE CURRENT DATA ADDRESS.                         INDD29.177
                                            * NOTE: AFTER A WRITE, DATA ADDRESS MAY BE ADVANCED      INDD29.178
                                            *       4 PARCELS MORE THAN EXPECTED DUE TO TIMING.      INDD29.179
                                                                                                     INDD29.180
305    170000                              IOB:10                     .READ CURRENT ADDRESS          INDD29.181
306    024450                              R!DATAI = A                .SAVE                          INDD29.182
307    030434  024447                      R!DATAO = (R!BUFENT)       .EXPECTED ADDRESS              INDD29.183
311    <macro>                             $IF (R!DATAI = R!DATAO + 4) .ACCEPT TIMING ANOMALY        INDD29.184
                                                                                  .>>>>> 2            $IF      1
320    020447  024450                        R!DATAI = R!DATAO        .                              INDD29.185
322    <macro>                             $ENDIF                                                   INDD29.186
                                                                                  .<<<<< 2            $ENDIF   1
322    <macro>                             $IF (R!DATAI # R!DATAO) .MISCOMPARE                      INDD29.187
                                                                                  .>>>>> 2            $IF      1
326    014000 /001527  024435                R!ERR = MESS4            .SEND ERROR MESSAGE            INDD29.188
331    077000 /001207                        R = REPORT               .                              INDD29.189
333    014000 /100000  035433                (R!CHENT) = (R!CHENT) + 100000  .DISABLE FURTHER TEST  INDD29.190
336    <macro>                             $ELSE                                                    INDD29.191
337    030434                                A = (R!BUFENT)           .ADDRESS                       INDD29.192
340    174000                                IOB:14                   .ENTER                         INDD29.193
341    162000                                IOB:2                    .START READ                    INDD29.194
342    <macro>                             $ENDIF                                                   INDD29.195
                                                                                  .<<<<< 2            $ENDIF   1
342    <macro>                            $ELSE                                                     INDD29.196
343    077000 /001207                      R = REPORT                 .REPORT ERROR                  INDD29.197
345    014000 /100000  035433              (R!CHENT) = (R!CHENT) + 100000 .DISABLE CHANNEL           INDD29.198
350    <macro>                            $ENDIF                                                    INDD29.199
                                                                                  .<<<<< 1            $ENDIF   1
350    <macro>                            $ENDIF                                                    INDD29.200
                                                                                  .<<<<< 0            $ENDIF   1
350    026433                             R!CHENT = R!CHENT + 1      .NEXT CHANNEL                  INDD29.201
351    010002  025434                     R!BUFENT = R!BUFENT + 2    .NEXT BUFFER                   INDD29.202
353    <macro>                           $ENDTIL                                                    INDD29.203
                                                                                                     INDD29.204
                                         * COMPARE DATA                                              INDD29.205
```

```
                                                                                                INDD29.206
354   020431  024433                R!CHENT = R!CHTBLE            .FIRST CHANNEL                INDD29.207
356   020432  024434                R!BUFENT = R!BUFTBLE          .FIRST BUFFER                 INDD29.208
360    <macro>                      $UNTIL ((R!CHENT) = 0)        .ALL CHANNELS                 INDD29.209
362    <macro>                        $IF (0 = (R!CHENT) & 100000) .IF ENABLED                  INDD29.210
                                                                                  .>>>>> 0          $IF      1
372   030433  054000                   B = (R!CHENT)              .CHANNEL                      INDD29.211
374   077000 /001164                   R = WAITDN                 .WAIT TIL DONE                INDD29.212
376    <macro>                         $IF (R!ERR = 0)            .OK                           INDD29.213
                                                                                  .>>>>> 1          $IF      1
                                                                                                INDD29.214
                                         * VALIDATE DATA ADDRESS REGISTER                       INDD29.215
                                                                                                INDD29.216
401   170000                            IOB:10                     .READ ADDRESS                 INDD29.217
402   024450                            R!DATAI = A                .SAVE                         INDD29.218
403   020434  012001  024410            R!DATAO = (R!BUFENT + 1) + D'512 .EXPECTED ADDRESS       INDD29.219
      030410  016000 /001000
      024447
412    <macro>                          $IF (R!DATAI # R!DATAO) .IF MISCOMPARE                  INDD29.220
                                                                                  .>>>>> 2          $IF      1
416   014000 /001527  024435             R!ERR = MESS4              .REPORT ERROR                 INDD29.221
421   077000 /001207                     R = REPORT                 .                             INDD29.222
423   014000 /100000  035433             (R!CHENT) = (R!CHENT) + 100000 .DISABLE CHANNEL          INDD29.223
426    <macro>                          $ELSE                                                    INDD29.224
427   030434  024423                     R!%W1 = (R!BUFENT)         .OUTPUT BUFFER                INDD29.225
431   020434  012001  024410             R!%W2 = (R!BUFENT + 1) .INPUT BUFFER                     INDD29.226
      030410  024424
436   014000 /001000  024425             R!%W3 = D'512              .LENGTH                       INDD29.227
441    <macro>                           $UNTIL (R!%W3 = 0)         .EACH PARCEL                  INDD29.228
443   030423  024447                       R!DATAO = (R!%W1)        .DATA OUT                     INDD29.229
445   030424  024450                       R!DATAI = (R!%W2)        .DATA IN                      INDD29.230
447    <macro>                             $IF (R!DATAI # R!DATAO) .MISMATCH                      INDD29.231
                                                                                  .>>>>> 3          $IF      1
453   014000 /001467  024435                 R!ERR = MESS2          .ERROR CODE                   INDD29.232
456   077000 /001207                         R = REPORT             .REPORT ERROR                 INDD29.233
460   030433  015000 /100000                 (R!CHENT) = (R!CHENT) & 100000 .DISABLE CHANNEL       INDD29.234
      034433
464   010000  024425                         R!%W3 = 0              .END LOOP                     INDD29.235
466    <macro>                             $ELSE                                                  INDD29.236
467   027425                                 R!%W3 = R!%W3 - 1 .NEXT                              INDD29.237
470    <macro>                             $ENDIF                                                 INDD29.238
                                                                                  .<<<<< 3          $ENDIF   1
```

```
470   <macro>                                  $ENDTIL                                                    INDD29.23
471   <macro>                                  $ENDIF                                                     INDD29.240
                                                                                   .<<<<< 2                $ENDIF    1
471   <macro>                                  $ELSE                                                      INDD29.241
472   077000 /001207                             R = REPORT                .REPORT ERROR                  INDD29.242
474   014000 /100000  035433                     (R!CHENT) = (R!CHENT) + 100000 .DISABLE CHANNEL          INDD29.243
477   <macro>                                  $ENDIF                                                     INDD29.244
                                                                                   .<<<<< 1                $ENDIF    1
477   <macro>                                $ENDIF                                                       INDD29.245
                                                                                   .<<<<< 0                $ENDIF    1
477   026433                                 R!CHENT = R!CHENT + 1       .NEXT CHANNEL                   INDD29.246
500   010002  025434                         R!BUFENT = R!BUFENT + 2     .NEXT BUFFER                    INDD29.247
502   <macro>                              $ENDTIL                                                       INDD29.248
503   010000  024447                       R!DATAO = 0                   .CLEAR DATA REGS                INDD29.249
505   010000  024450                       R!DATAI = 0                   .                               INDD29.250
507   027443                               R!PASS = R!PASS - 1           .DECREMENT PASS COUNT           INDD29.251
510   107326                               P = ECHO, A # 0               .LOOP                           INDD29.252
                                                                                                          INDD29.253
                                           *                                                              INDD29.254
                                           * CHECK STATUS REGISTERS                                      INDD29.255
                                           *                                                              INDD29.256
                                                                                                          INDD29.257
511   010764  024443                       R!PASS = PASSES                                                INDD29.258
513                             REGTEST    *                                                              INDD29.259
513   020431  024433                       R!CHENT = R!CHTBLE            .FIRST CHANNEL                  INDD29.260
515   <macro>                              $UNTIL ((R!CHENT) = 0)        .ALL CHANNELS                   INDD29.261
517   <macro>                                $IF (0 = (R!CHENT) & 100000) .IF ENABLED                     INDD29.262
                                                                                   .>>>>> 0                $IF       1
527   030433  054000                           B = (R!CHENT)             .CHANNEL NUMBER                 INDD29.263
531   150004                                   RTC:10                    .REAL TIME CLOCK                INDD29.264
532   015000 /177774  024447                   R!DATAO = A & 177774      .OUTPUT VALUE                   INDD29.265
535   175000                                   IOB:15                    .DIAGNOSTIC ECHO                INDD29.266
536   171000                                   IOB:11                    .READ BACK                      INDD29.267
537   024450                                   R!DATAI = A               .SAVE                           INDD29.268
540   <macro>                                  $IF (R!DATAI # R!DATAO)   .ERROR                          INDD29.269
                                                                                   .>>>>> 1                $IF       1
544   014000 /001506  024435                     R!ERR = MESS3           .CODE                           INDD29.270
547   077000 /001207                             R = REPORT              .                               INDD29.271
551   014000 /100000  035433                     (R!CHENT) = (R!CHENT) + 100000 .DISABLE                  INDD29.272
554   <macro>                                  $ENDIF                                                     INDD29.273
                                                                                   .<<<<< 1                $ENDIF    1
554   <macro>                                $ENDIF                                                       INDD29.274
```

```
                                                                                .<<<<< 0          $ENDIF   1
554   026433                          R!CHENT = R!CHENT + 1       .NEXT CHANNEL                  INDD29.275
555    <macro>                        $ENDTIL                                                    INDD29.276
556   027443                          R!PASS = R!PASS - 1         .DECREMENT PASS COUNT          INDD29.277
557   107044                          P = REGTEST, A # 0          .LOOP                          INDD29.278
560   010000  024447                  R!DATAO = 0                 .CLEAR DATA REGS               INDD29.279
562   010000  024450                  R!DATAI = 0                 .                              INDD29.280
                                                                                                   INDD29.281
                                                                                                   INDD29.282
                                      *                                                            INDD29.283
                                      * SELECT DEVICES                                            INDD29.284
                                      *                                                            INDD29.285
                                                                                                   INDD29.286
564   020431  024433                  R!CHENT = R!CHTBLE          .FIRST CHANNEL                 INDD29.287
566    <macro>                        $UNTIL ((R!CHENT) = 0)      .ALL CHANNELS                  INDD29.288
570   030433  054000                    B = (R!CHENT)             .CHANNEL                       INDD29.289
572   160000                            IOB:0                     .CLEAR                         INDD29.290
573   000000                            PASS                      .                              INDD29.291
574   014000 /001000                    A = 1000                  .SELECT                        INDD29.292
576   161000                            IOB:1                     .                              INDD29.293
577   077000 /001164                    R = WAITDN                .WAIT FOR DONE                 INDD29.294
601    <macro>                          $IF (R!ERR # 0)           .IF ERROR                      INDD29.295
                                                                                .>>>>> 0             $IF      1
604   014000 /001550  024435              R!ERR = MESS5           .SEND MESSAGE                  INDD29.296
607   077000 /001207                      R = REPORT              .                              INDD29.297
611    <macro>                          $ENDIF                                                     INDD29.298
                                                                                .<<<<< 0          $ENDIF   1
611   160000                            IOB:0                     .CLEAR INTERRUPT               INDD29.299
612   050000  013020  016000            R!%W1 = B - 20 + DCCB     .FIND CHANNEL TABLE            INDD29.300
     /000000  024423
617   030423  024423                    R!%W1 = (R!%W1)           .                              INDD29.301
621    <macro>                          STORE 1,DB@SEL,R!%W1      .SET SELECT FLAG               INDD29.302
631   167000                            IOB:7                     .ENABLE INTERRUPTS             INDD29.303
632   026433                            R!CHENT = R!CHENT + 1     .NEXT CHANNEL                  INDD29.304
633    <macro>                        $ENDTIL                                                    INDD29.305
                                                                                                   INDD29.306
                                      *                                                            INDD29.307
                                      * RELEASE MEMORY                                            INDD29.308
                                      *                                                            INDD29.309
                                                                                                   INDD29.310
634    <macro>                        RELMEM R!CHTBLE             .CHANNEL TEST TABLE            INDD29.311
644   020432  024434                  R!BUFENT = R!BUFTBLE        .BUFFER TABLE                  INDD29.312
```

```
646    <macro>                          $UNTIL ((R!BUFENT)     )          .                                    INDD29.31.
650   030434  024423                      R!%W1 = (R!BUFENT)               .ADDRESS                             INDD29.314
652    <macro>                            BRET R!%W1                       .RELEASE                             INDD29.315
662   010004  025434                      R!BUFENT = R!BUFENT + 4          .NEXT                                INDD29.316
664    <macro>                          $ENDTIL                                                                   INDD29.317
665    <macro>                          RELMEM R!BUFTBLE                 .RELEASE TABLE                         INDD29.318
                                                                                                                   INDD29.319
675    <macro>                          RETURN                                                                    INDD29.320
                                                                                                                   INDD29.321
                                 ****************************** SUBROUTINES ****************************           INDD29.322
                                                                                                                   INDD29.323
                                        *                                                                          INDD29.324
                                        * ALLOCATE MEMORY                                                         INDD29.325
                                        *                                                                          INDD29.326
                                                                                                                   INDD29.327
703                              GMEM   *                                                                          INDD29.328
703    <macro>                          GETMEM R!SIZE,R!MEM,R!TX0,R!TX1,R!TX2 .REQUEST SPACE                     INDD29.329
723    <macro>                          $PUNTIF (A # 0)                  .IF NONE                               INDD29.330
725    <macro>                          CLEAR START=R!MEM,COUNT=R!SIZE  .CLEAR IT                               INDD29.331
740   001000                            EXIT                                                                      INDD29.332
                                                                                                                   INDD29.333
                                        *                                                                          INDD29.334
                                        * SETUP DISK CHANNEL                                                     INDD29.335
                                        *                                                                          INDD29.336
                                                                                                                   INDD29.337
741                              SETUP  *                                                                          INDD29.338
741   020425  034433                    (R!CHENT) = R!%W3                .ENTER IN CHANNEL TABLE                INDD29.339
743   026433                            R!CHENT = R!CHENT + 1            .NEXT                                  INDD29.340
744    <macro>                          $IF (R!%W4 = CH$D1) OR (R!%W4 = CH$D2) .SINGLE DEVICE                   INDD29.341
                                                                                                .>>>>> 0              $IF      1
753   010022  024440                      R!RALEN = STBLE                  .LENGTH OF READ AHEAD                  INDD29.342
755   010003  024437                      R!RANUM = RA$NUM                 .NUMBER OF READ-AHEADS                 INDD29.343
757    <macro>                          $ELSE                                                                     INDD29.344
760   010154  024440                      R!RALEN = MTBLE                  .STRIPED                               INDD29.345
762   010022  024437                      R!RANUM = RA$NUMS                .NUMBER OF READ AHEADS                 INDD29.346
764    <macro>                          $ENDIF                                                                    INDD29.347
                                                                                                .<<<<< 0              $ENDIF   1
764   020440  012047  024442            R!SIZE = R!RALEN + DD@LEN@P      .TOTAL LENGTH OF TABLE                 INDD29.348
767   014000 /042102  024444            R!TX0 = 'DB'                     .TABLE NAME                            INDD29.349
772   014000 /040040  024445            R!TX1 = '@ '                     .                                      INDD29.350
775   020425  004003  012060            R!TX2 = R!%W3 > 3 + 'O'R < D'8  .CONVERT TO ASCII                       INDD29.351
      005010  024446
```

```
1002   020425  011007  012060          R!TX2 = R!%W3 & 7 + 'O'R + R!TX2 .                          INDD29.352
       022446  024446
1007   073104                          R = GMEM                        .ALLOCATE                   INDD29.353
                                                                                                    INDD29.354
                                       * SET SYSTEM POINTERS FOR DEVICE                            INDD29.355
                                                                                                    INDD29.356
1010   020425  013020  016000          R!SCO = R!%W3 - 20 + DCCB       .LOOK UP TABLE              INDD29.357
      /000000  024436
1015   020441  034436                  (R!SCO) = R!MEM                 .ENTER                      INDD29.358
1017   020425  016000 /000000          (R!%W3 + EITB) = IDID           .INTERRUPT HANDLER          INDD29.359
       024410  014000 /000000
       034410
                                                                                                          .
                                                                                                    INDD29.360
                                       * INITIALIZE TABLE                                          INDD29.361
                                                                                                    INDD29.362
1026    <macro>                        STORE 3777,DB@CYL,R!MEM         .FORCE INITIAL SEEK         INDD29.363
1037    <macro>                        STORE 37,DB@HED,R!MEM           .                           INDD29.364
1047    <macro>                        PUT R!%W3,DB@CHN,R!MEM          .ENTER CHANNEL NUMBER       INDD29.365
1062    <macro>                        $IF (R!%W4 = CH$D1),OR,(R!%W4 = CH$D1S)                     INDD29.366
                                                                                  .>>>>> 0                $IF      1
1071   010001  024426                    R!%W4 = DSK$DD19              .DD-19                      INDD29.367
1073    <macro>                        $ELSE                                                       INDD29.368
1074   010002  024426                    R!%W4 = DSK$DD29              .DD-29                      INDD29.369
1076    <macro>                        $ENDIF                                                      INDD29.370
                                                                                  .<<<<< 0                $ENDIF   1
1076    <macro>                        PUT R!%W4,DB@TYP,R!MEM          .ENTER CHANNEL TYPE         INDD29.371
1112    <macro>                        PUT R!RANUM,DB@RAK,R!MEM        .ENTER                      INDD29.372
1117    <macro>                        ADDRESS R!SCO,DD@LEN,R!MEM      .START OF READ-AHEAD AREA   INDD29.373
1122    <macro>                        PUT R!SCO,DB@INF,R!MEM          .ENTER FIRST                INDD29.374
1127    <macro>                        PUT R!SCO,DB@IN,R!MEM           .ENTER IN                   INDD29.375
1134    <macro>                        PUT R!SCO,DB@OUT,R!MEM          .ENTER OUT                  INDD29.376
1141   020441  022442  013006          R!SCO = R!MEM + R!SIZE - RA@LEN@P .LAST READ-AHEAD          INDD29.377
       024436
1145    <macro>                        PUT R!SCO,DB@UNF,R!MEM                                      INDD29.378
1152    <macro>                        ADDRESS R!SCO,DB@TMO,R!MEM      .TIMEOUT ENTRY              INDD29.379
1155    <macro>                        STORE IDKTOUT,TMR@RT,R!SCO      .ENTER HANDLER ADDRESS      INDD29.380
1163   001000                          EXIT                                                        INDD29.381
                                                                                                    INDD29.382
                                       *                                                           INDD29.383
                                       * WAIT FOR CHANNEL DONE                                     INDD29.384
                                       *                                                           INDD29.385
                                                                                                    INDD29.386
```

```
1163                                WAITDN   *                                                          INDD29.38
1164   010000                                A = 0                          .TIMER                     INDD29.386
1165    <macro>                              $UNTIL (IOB = DN)                                         INDD29.389
1167   013001                                  A = A - 1                   .COUNT DOWN                INDD29.390
1170   102007                                  P = TIMEOUT, A = 0          .IF TIMED OUT              INDD29.391
1171    <macro>                              $ENDTIL                                                   INDD29.392
1172   043000  101010                        P = CHANERR, IOB = BZ       .IF ERROR                  INDD29.393
1174   010000  024435                        R!ERR = 0                   .RETURN STATUS             INDD29.394
1176   001000                                EXIT                                                      INDD29.395
                                                                                                        INDD29.396
                                             * CHANNEL TIMEOUT                                         INDD29.397
                                                                                                        INDD29.398
1177                                TIMEOUT  *                                                          INDD29.399
1177   014000 /001434  024435                R!ERR = MESS0               .CODE                      INDD29.400
1202   001000                                EXIT                                                      INDD29.401
                                                                                                        INDD29.402
                                             * CHANNEL ERROR                                           INDD29.403
                                                                                                        INDD29.404
1203                                CHANERR  *                                                          INDD29.405
1203   014000 /001452  024435                R!ERR = MESS1               .CODE                      INDD29.406
1206   001000                                EXIT                                                      INDD29.407
                                                                                                        INDD29.408
                                             *                                                         INDD29.409
                                             * SEND MESSAGE TO CONSOLE                                 INDD29.410
                                             *                                                         INDD29.411
                                                                                                        INDD29.412
1207                                REPORT   *                                                          INDD29.413
1207   050000  024436                        R!SCO = B                   .SAVE B                    INDD29.414
1211   010050  024442                        R!SIZE = MSGLNG             .ALLOCATE MEMORY           INDD29.415
1213   073310                                R = GMEM                    .                          INDD29.416
1214    <macro>                              COPY R!ERR+R!%B,R!MEM,MSGLNG .COPY                     INDD29.417
1233   030433  011777  024423                R!%W1 = (R!CHENT) & 777     .CHANNEL                   INDD29.418
1236   020423  004003  012060                R!%W2 = R!%W1 > 3 + 'O'R < D'8 .CONVERT TO ASCII       INDD29.419
       005010  024424
1243   020423  011007  012060                R!%W2 = R!%W1 & 7 + 'O'R + R!%W2 .                     INDD29.420
       022424  024424
1250   020441  012004  024410                (R!MEM + CHANOFF) = R!%W2   .ENTER IN MESSAGE          INDD29.421
       020424  034410
1255    <macro>                              MSG R!MEM                   .SEND                      INDD29.422
                                                                                                        INDD29.423
                                             * REPORT EXPECTED AND RECEIVED DATA                       INDD29.424
                                                                                                        INDD29.425
```

```
1266    <macro>                          $IF (R!DATAO # R!DATAI)        .DATA MISCOMPARE                    INDD29.426
                                                                                          .>>>>> 0              $IF      1
1272    <macro>                            COPY R!%B+MESS6,R!MEM,MSGLNG .ERROR MESSAGE                      INDD29.427
1312    020441  012050  013001            R!%W1 = R!MEM + MSGLNG - 1   .USE FOR SCRATCH PARCEL             INDD29.428
        024423
1316    020447  034423                     (R!%W1) = R!DATAO            .EXPECTED DATA                      INDD29.429
1320    <macro>                            CALL BTO,(R!%W1,0,D'16,R!MEM,EXPOFF,2,6)                         INDD29.430
1350    020441  012050  013001             R!%W1 = R!MEM + MSGLNG - 1   .RECEIVED DATA                      INDD29.431
        024423
1354    020450  034423                     (R!%W1) = R!DATAI            .                                   INDD29.432
1356    <macro>                            CALL BTO,(R!%W1,0,D'16,R!MEM,RECOFF,2,6)                         INDD29.433
1406    <macro>                            MSG R!MEM                    .SEND                               INDD29.434
1417    <macro>                          $ENDIF                                                             INDD29.435
                                                                                          .<<<<< 0              $ENDIF   1
1417    <macro>                          RELMEM R!MEM                   .RELEASE                            INDD29.436
1427    020436  054000                   B = R!SC0                      .RESTORE B                          INDD29.437
1431    010000  024435                   R!ERR = 0                      .CLEAR ERROR                        INDD29.438
1433    001000                           EXIT                                                               INDD29.439
                                                                                                            INDD29.440
                                         END                                                                INDD29.441
```

0X	%B		1: 4 D	7: 4 B	B	7:32 B	8: 8 B	8:22 B	8:35 B	9: 5 B	
			9:38 B	10:21 B	10:25 B	13:31	14: 3				
0X	%EX		5:17 D	5:17	10:41 D	10:41	11: 3 D	11: 3	11: 6 D	11: 6	
			11: 8 D	11: 8	11:17 D	11:17	13:39 D	13:39	14: 7 D	14: 7	
			14:11 D	14:11	14:12 D	14:12	14:15 D	14:15			
10	%GBLNUM	$APTEXT	3:12 F	3:12							
400	%GBLREG	$APTEXT	3:12								
0X	%MYID		3: 8 D	4:26							
0X	%PUNT		4:29 D	4:29	5:18 D	5:18	11:18 D	11:18			
1	CH$D1	$APTEXT	4:38	11:30	12:20						
101	CH$D1S	$APTEXT	4:40	12:20							
2	CH$D2	$APTEXT	4:38	11:30							
102	CH$D2S	$APTEXT	4:40								
777	DB@CHN@M	$APTEXT	12:18								
11	DB@CHN@N	$APTEXT	12:18 F								
7	DB@CHN@P	$APTEXT	12:18 F	12:18							
7	DB@CHN@S	$APTEXT	12:18 F								
177000	DB@CHN@X	$APTEXT	12:18								
3777	DB@CYL@M	$APTEXT	12:16 F								
13	DB@CYL@N	$APTEXT	12:16 F	12:16							
1	DB@CYL@P	$APTEXT	12:16 F	12:16							
0	DB@CYL@S	$APTEXT	12:16 F	12:16							
37	DB@HED@M	$APTEXT	12:17 F								
5	DB@HED@N	$APTEXT	12:17 F								
1	DB@HED@P	$APTEXT	12:17 F	12:17							
13	DB@HED@S	$APTEXT	12:17 F								
177740	DB@HED@X	$APTEXT	12:17								
20	DB@IN@N	$APTEXT	12:30 F								
44	DB@IN@P	$APTEXT	12:30 F	12:30							
0	DB@IN@S	$APTEXT	12:30 F								
20	DB@INF@N	$APTEXT	12:29 F								
12	DB@INF@P	$APTEXT	12:29 F	12:29							
0	DB@INF@S	$APTEXT	12:29 F								
20	DB@OUT@N	$APTEXT	12:31 F								
45	DB@OUT@P	$APTEXT	12:31 F	12:31							
0	DB@OUT@S	$APTEXT	12:31 F								
20	DB@RAK@N	$APTEXT	12:27 F								
10	DB@RAK@P	$APTEXT	12:27 F	12:27							
0	DB@RAK@S	$APTEXT	12:27 F								
3	DB@SEI@M	$APTEXT	10:32 F								
2	DB@SEI@N	$APTEXT	10:32 F	10:32							
7	DB@SEI@P	$APTEXT	10:32 F	10:32							

```
 0   DB@SEL@S   $APTEXT     10:32 F   10:32
40   DB@TMO@P   $APTEXT     12:35
37   DB@TYP@M   $APTEXT     12:26
 5   DB@TYP@N   $APTEXT     12:26 F   12:26
 7   DB@TYP@P   $APTEXT     12:26 F   12:26
 2   DB@TYP@S   $APTEXT     12:26 F   12:26
20   DB@UNF@N   $APTEXT     12:34 F
46   DB@UNF@P   $APTEXT     12:34 F   12:34
 0   DB@UNF@S   $APTEXT     12:34 F
 0X  DCCB                    3: 6 D   10:29      12: 7
47   DD@LEN@P   $APTEXT     11:38     12:28
 1   DSK$DD19   $APTEXT     12:21
 2   DSK$DD29   $APTEXT     12:23
 0   EITB                    3: 7 D   12:10
32   F$BGET     $APTEXT      5:17
33   F$BRET     $APTEXT     11: 3
50   F$CALL     $APTEXT     14: 7     14:11
30   F$GMEM     $APTEXT     11:17
20   F$MESG     $APTEXT     13:39     14:12
52   F$RETURN   $APTEXT     11: 8
31   F$RMEM     $APTEXT     10:41     11: 6      14:15
 0X  IDID                    3: 5 D   12:11
 0X  IDKTOUT                 3: 9 D   12:36
 0X  O$BTO                  14: 7 D   14: 7      14:11 D   14:11
 0X  O$INDD29                1: 4 D    1: 4
 6   OV@ENT@P   $APTEXT      4:26
17   OV@NUM@N   $APTEXT      1: 4
 1   OV@TYP@N   $APTEXT      1: 4
 0   PT$HUH     $APTEXT      4:29      5:18      11:18
 3   RA$NUM     $APTEXT      2:41     11:32
22   RA$NUMS    $APTEXT      2:39     11:35
 6   RA@LEN@P   $APTEXT      2:39      2:41      12:32
 4   RTC        $APTEXT      6: 7      9:30
11   SM@FST@N   $APTEXT      1: 4
 7   SM@NUM@N   $APTEXT      1: 4
20   TMR@RT@N   $APTEXT     12:36 F
 2   TMR@RT@P   $APTEXT     12:36 F   12:36
 0   TMR@RT@S   $APTEXT     12:36 F

     SYMBOL QUALIFIER IS INDD29

22   $$X                     2:39 D    2:40       2:41 D    2:42
```

0X	$CONST	10:32 D	10:32 F	[illegible]	12:16 D	12:16 F	12:16	12:17 D	12:17 F
		12:17	12:36 D	12:36 F	12:36				
451	$REGORG	3:12 D	3:12 F	3:13	3:13 D	3:14	3:14 D	3:15	3:15 D
		3:16	3:16 D	3:17	3:17 D	3:17 F	3:18	3:18 D	3:19
		3:19 D	3:20	3:20 D	3:21	3:21 D	3:22	3:22 D	3:23
		3:23 D	3:23 F	3:24	3:24 D	3:25	3:25 D	3:26	3:26 D
		3:27	3:27 D	3:28	3:28 D	3:28 F	3:29	3:29 D	3:29 F
		3:30	3:30 D	3:31	3:31 D	3:32	3:32 D	3:33	3:33 D
		3:34	3:34 D	3:35	3:35 D	3:36	3:36 D	3:37	3:37 D
		3:38	3:38 D	3:39	3:39 D	3:40	3:40 D	3:41	3:41 D
		3:42	3:42 D	4: 1	4: 1 D	4: 2	4: 2 D	4: 3	4: 3 D
		4: 3 F	5:17	10:41	11: 3	11: 6	11: 8	11:17	13:39
		13:39 F	14: 7	14: 7 F	14:11	14:11 F	14:12	14:12 F	14:15
1	%NP	1: 4	3:29 D						
430	%P	1: 4	3:28 D	3:29	13:39 F	13:39	14: 7 F	14: 7	14:11 F
		14:11	14:12 F	14:12					
410	%S1	3:12	3:13 D	5:16	6: 1	6: 3	6:27	7: 2	7:16
		8: 6	8:16	8:26	9:28	10:32	12:10	12:11	12:16
		12:17	12:18	12:26	12:27	12:29	12:30	12:31	12:34
		12:36	13:37	13:38					
411	%S2	3:12	3:14 D						
412	%S3	3:12	3:15 D						
413	%S4	3:12	3:16 D						
414	%S5	3:12	3:17 D						
415	%T1	3:18 D	11:19	13:31	14: 3				
416	%T2	3:19 D	11:19	13:31	14: 3				
417	%T3	3:20 D	13:31	14: 3					
420	%T4	3:21 D							
421	%T5	3:22 D							
422	%T6	3:23 D							
423	%W1	3:24 D	4:27	4:28	4:30	4:31	4:32	4:35	4:36
		5:10	5:12	5:25	5:27	6: 2	6: 8	6:10	8:25
		8:30	10:30	10:31	10:32	11: 2	11: 3	13:32	13:33
		13:35	14: 4	14: 6	14: 7	14: 8	14:10	14:11	
424	%W2	3:25 D	4:32	4:34	5:11	5:21	5:22	5:24	5:26
		6: 4	6: 9	6:11	8:27	8:31	13:34	13:35	13:36
		13:37							
425	%W3	3:26 D	4:33	4:34	5: 2	5:12	5:14	5:27	6: 5
		6: 6	6:12	8:28	8:29	8:38	8:40	11:27	11:41
		12: 1	12: 7	12:10	12:18				
426	%W4	3:27 D	4:36	4:38	4:40	5:13	5:16	5:17	5:21
		5:23	5:24	5:28	11:30	12:20	12:21	12:23	12:26

427	%W5	3:28 D							
430	AMAP	3:29 D	4:26	4:28	4:29	4:31			
434	BUFENT	3:33 D	5:40	6: 2	6: 3	6:17	6:24	6:29	6:35
		6:41	7:14	7:26	7:39	8: 3	8:16	8:25	8:26
		9:12	10:42	11: 1	11: 2	11: 4			
432	BUFTBLE	3:31 D	4:14	5:11	5:40	6:24	6:41	8: 3	10:42
		11: 6							
1203	CHANERR	13: 7	13:19 D						
4	CHANOFF	2:33 D	13:37						
433	CHENT	3:32 D	4:25	5:39	5:41	6: 1	6:16	6:23	6:25
		6:27	6:28	6:34	6:40	6:42	7: 2	7: 3	7:24
		7:33	7:38	8: 2	8: 4	8: 6	8: 7	8:23	8:36
		9: 6	9:11	9:25	9:26	9:28	9:29	9:39	10: 2
		10:14	10:15	10:16	10:34	11:27	11:28	13:32	
431	CHTBLE	3:30 D	4:11	4:25	5:10	5:39	6:23	6:40	8: 2
		9:25	10:14	10:41					
450	DATAI	4: 3 D	4:16	7:13	7:16	7:17	7:21	8:15	8:20
		8:31	8:33	9:15	9:34	9:36	10: 7	14: 2	14:10
447	DATAO	4: 2 D	4:15	7:14	7:16	7:17	7:21	8:17	8:20
		8:30	8:33	9:14	9:31	9:36	10: 6	14: 2	14: 6
162	ECHO	5:38 D	9:17						
435	ERR	3:34 D	7: 6	7:22	8:10	8:21	8:34	9:37	10:23
		10:24	13: 8	13:14	13:20	13:31	14:17		
12	EXPOFF	2:31 D	14: 7						
703	GMEM	4:10	4:13	11:16 D	12: 3	13:30			
441	MEM	3:38 D	4:11	4:14	11:17	11:19	12: 9	12:16	12:17
		12:18	12:26	12:27	12:28	12:29	12:30	12:31	12:32
		12:34	12:35	13:31	13:37	13:39	14: 3	14: 4	14: 7
		14: 8	14:11	14:12	14:15				
1434	MESS0	2:16 D	13:14						
1452	MESS1	2:18 D	13:20						
1467	MESS2	2:20 D	8:34						
1506	MESS3	2:22 D	9:37						
1527	MESS4	2:24 D	7:22	8:21					
1550	MESS5	2:26 D	10:24						
1570	MESS6	2:28 D	14: 3						
50	MSGLNG	2:37 D	13:29	13:31	14: 3	14: 4	14: 8		
154	MTBLE	2:40 D	11:34						
41	NBUFS	2:36 D	4:12						
21	NCHNS	2:35 D	4: 9						
443	PASS	3:40 D	5:37	9:16	9:23	10: 4			
764	PASSES	2:34 D	5:37	9:23					

440	RALEN	3:37 D	11:31	[illegible]	11:38				
437	RANUM	3:36 D	11:32	11:35	12:27				
34	RECOFF	2:32 D	14:11						
513	REGTEST	9:24 D	10: 5						
1207	REPORT	7:23	7:32	8:22	8:35	9: 5	9:38	10:25	13:27 D
436	SCO	3:35 D	12: 8	12: 9	12:28	12:29	12:30	12:31	12:32
		12:34	12:35	12:36	13:28	14:16			
741	SETUP	4:39	4:41	11:26 D					
442	SIZE	3:39 D	4: 9	4:12	11:17	11:19	11:38	12:32	13:29
22	STBLE	2:42 D	11:31						
1177	TIMEOUT	13: 5	13:13 D						
444	TX0	3:41 D	11:17	11:39					
445	TX1	3:42 D	11:17	11:40					
446	TX2	4: 1 D	11:17	11:42	12: 1	12: 2			
1164	WAITDN	7: 4	8: 8	10:21	13: 1 D				

```
0     <macro>                    OVERLAY   INDD39                                          INDD39.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'              OVERLAY  1
```

```
                    ***************************    ****************************************  INDD39.5
                    *                                                                     *  INDD39.6
                    *        INDD39     Initialize DD39 disk channels for use by the      *  INDD39.7
                    *                   current IOP.  Sets up control tables, interrupt    *  INDD39.8
                    *                   addresses and runs minimal reliability checks      *  INDD39.9
                    *                   on the channel and device.                        *  INDD39.10
                    *                                                                     *  INDD39.11
                    *        Entry      R!AMAP = AMAP base address                        *  INDD39.12
                    *                                                                     *  INDD39.13
                    ***********************************************************************  INDD39.14
                                                                                             INDD39.15
6   <macro>                     REGDEFS    ,(AMAP),(CHAN,NEXT,LAST,TYPE,TBLE,COUNT,MCB,TOUT,NUM,  INDD39.16
                    ,BUF,ERR,BUFU,BUFL,MOSU,MOSL,LEN,GEN,STAT,CON,RETRY,SIZE,NE,DPT,DCB,         INDD39.17
                    ,UNIT)                                                                       INDD39.18
                410 %S1        EQUALS     $REGORG                                          $REG    3
                411 %S2        EQUALS     $REGORG                                          $REG    3
                412 %S3        EQUALS     $REGORG                                          $REG    3
                413 %S4        EQUALS     $REGORG                                          $REG    3
                414 %S5        EQUALS     $REGORG                                          $REG    3
                415 %T1        EQUALS     $REGORG                                          $REG    3
                416 %T2        EQUALS     $REGORG                                          $REG    3
                417 %T3        EQUALS     $REGORG                                          $REG    3
                420 %T4        EQUALS     $REGORG                                          $REG    3
                421 %T5        EQUALS     $REGORG                                          $REG    3
                422 %T6        EQUALS     $REGORG                                          $REG    3
                423 %W1        EQUALS     $REGORG                                          $REG    3
                424 %W2        EQUALS     $REGORG                                          $REG    3
                425 %W3        EQUALS     $REGORG                                          $REG    3
                426 %W4        EQUALS     $REGORG                                          $REG    3
                427 %W5        EQUALS     $REGORG                                          $REG    3
                430 AMAP       EQUALS     $REGORG                                          $REG    3
                431 CHAN       EQUALS     $REGORG                                          $REG    3
                432 NEXT       EQUALS     $REGORG                                          $REG    3
                433 LAST       EQUALS     $REGORG                                          $REG    3
                434 TYPE       EQUALS     $REGORG                                          $REG    3
                435 TBLE       EQUALS     $REGORG                                          $REG    3
                436 COUNT      EQUALS     $REGORG                                          $REG    3
                437 MCB        EQUALS     $REGORG                                          $REG    3
                440 TOUT       EQUALS     $REGORG                                          $REG    3
                441 NUM        EQUALS     $REGORG                                          $REG    3
                442 BUF        EQUALS     $REGORG                                          $REG    3
                443 ERR        EQUALS     $REGORG                                          $REG    3
```

```
444     BUFU      EQUALS    $REGORG                                                    $REG     3
445     BUFL      EQUALS    $REGORG                                                    $REG     3
446     MOSU      EQUALS    $REGORG                                                    $REG     3
447     MOSL      EQUALS    $REGORG                                                    $REG     3
450     LEN       EQUALS    $REGORG                                                    $REG     3
451     GEN       EQUALS    $REGORG                                                    $REG     3
452     STAT      EQUALS    $REGORG                                                    $REG     3
453     CON       EQUALS    $REGORG                                                    $REG     3
454     RETRY     EQUALS    $REGORG                                                    $REG     3
455     SIZE      EQUALS    $REGORG                                                    $REG     3
456     NE        EQUALS    $REGORG                                                    $REG     3
457     DPI       EQUALS    $REGORG                                                    $REG     3
460     DCB       EQUALS    $REGORG                                                    $REG     3
461     UNIT      EQUALS    $REGORG                                                    $REG     3
                                                                                   INDD39.19
                  EXT       %DCBQ                                                    INDD39.20
                  EXT       %DD49                                                    INDD39.21
                  EXT       %MEMIO                                                   INDD39.22
                  EXT       %MYID                                                    INDD39.23
                  EXT       D4TOUT                                                   INDD39.24
                  EXT       DCCB                                                     INDD39.25
                  EXT       DD49                                                     INDD39.26
                  EXT       EITB                                                     INDD39.27
                  EXT       TMCB                                                     INDD39.28
                                                                                     INDD39.29
 14     LB@SZ     EQUALS    LB@LE*LB$DD39        .Local buffer table size            INDD39.30
  4     CHNOFF    EQUALS    4                    .Offset for channel in msg          INDD39.31
 10     UNTOFF    EQUALS    D'8                  .Offset for unit in msg             INDD39.32
 44     MSGLEN    EQUALS    D'36                 .Msg length                         INDD39.33
  2     CNVRTLEN  EQUALS    2                    .Conversion table length            INDD39.34
                                                                                     INDD39.35
  1     RTR$CLR   EQUALS    1                    .Max retries for channel clear      INDD39.36
  5     RTR$RDY   EQUALS    5                    .Max retries for ready status       INDD39.37
  5     RTR$SLC   EQUALS    5                    .Max retries for unit select        INDD39.38
  2     RTR$RES   EQUALS    2                    .Max retries for reset              INDD39.39
  5     RTR$STA   EQUALS    5                    .Max retries for getting status     INDD39.40
```

```
                                                                                                INDD39.42
                                 *        Set up one Device Parameter Table (DPT) to serve all channels.   INDD39.43
                                                                                                INDD39.44
   6   010011  024450                      R!LEN = DP@LT                  .Table length         INDD39.45
  10   014000 /042120  024423              R!%W1 = 'DP'                   .ASCII header         INDD39.46
  13   014000 /052040  024424              R!%W2 = 'T '                                         INDD39.47
  16   014000 /031471  024425              R!%W3 = '39'                                         INDD39.48
  21   072550                              R = GETBUF                     .Get a buffer         INDD39.49
  22   020442  024457                      R!DPT = R!BUF                  .Save address         INDD39.50
  24   020000  016000 /002265              R!%W1 = R!%B + D3CNVRT         .Spiral offsets       INDD39.51
       024423
  30    <macro>                            COPY R!%W1,R!DPT+DP@SPI@P,CNVRTLEN .Copy             INDD39.52
  47    <macro>                            STORE I@SPIRD3,DP@SPF,R!DPT    .Spiral enable flag   INDD39.53
  57    <macro>                            STORE I@INDXD3,DP@IDX,R!DPT    .Spiral index value   INDD39.54
  67    <macro>                            STORE D3$SPT,DP@DSE,R!DPT      .Default sectors/track INDD39.55
  77    <macro>                            STORE D3$CPD,DP@DCY,R!DPT      .Default cylinders/disk INDD39.56
 105    <macro>                            STORE D3$SPTA,DP@ASE,R!DPT     .Absolute sectors/track INDD39.57
 115    <macro>                            STORE D3$CPDA,DP@ACY,R!DPT     .Absolute cylinder/disk INDD39.58
                                                                                                INDD39.59
                                 *        Set up Disk Control Blocks (DCB) and initialize each channel.    INDD39.60
                                                                                                INDD39.61
 123   014000 /000000  024435              R!TBLE = EITB                  .Interrupt jump table INDD39.62
 126   020430  012006  012005              R!%W1 = R!AMAP + OV@ENT@P + 5 + R!%MYID .Pointer to IOP info INDD39.63
       022000  024423
 133   030423  022430  024423              R!%W1 = (R!%W1) + R!AMAP       .Info for this IOP    INDD39.64
 136   010006  025423                      R!%W1 = R!%W1 + 6              .Channel pointer      INDD39.65
 140   030423  022430  024432              R!NEXT = (R!%W1) + R!AMAP      .Channel info         INDD39.66
 143   030432  024433                      R!LAST = (R!NEXT)              .Last channel to search INDD39.67
 145   010006  024431                      R!CHAN = 6                     .First channel to search INDD39.68
 147   010000  024436                      R!COUNT = 0                    .Initialize           INDD39.69
 151   024437                              R!MCB = A                                            INDD39.70
 152    <macro>                            $UNTIL (R!CHAN = R!LAST)       .Until all searched   INDD39.71
 155   026432                               R!NEXT = R!NEXT + 1          .Next channel         INDD39.72
 156   030432  011777  024434               R!TYPE = (R!NEXT) & O'777    .Channel type         INDD39.73
 161    <macro>                             $IF (R!TYPE = CH$D3),OR,(R!TYPE = CH$D3S) .If DD39 INDD39.74
                                                                                .>>>>> 0          $IF      1
 170   026436                                 R!COUNT = R!COUNT + 1      .Count                INDD39.75
 171   020000  112435                         R = RESIDENT, R!%DD49 = 0 .Read in interrupt overlay INDD39.76
 173   020435  022431  024423                 R!%W1 = R!TBLE + R!CHAN    .Interrupt table pointer INDD39.77
 176   020000  016000 /000000                 (R!%W1) = R!%DD49 + DD49   .Set interrupt entrance address INDD39.78
       034423
 202   020000  016000 /000000                 R!TOUT = R!%DD49 + D4TOUT .Timeout routine        INDD39.79
```

```
        024440
206     077000 /001361                  R = SETDCB                  .Set up the DCB                     INDD39.80
210     072516                          R = CHINIT                  .Initialize the disk channel        INDD39.81
211      <macro>                      $ENDIF                                                            INDD39.82
                                                                                        .<<<<< 0            $ENDIF    1
211     026431                       R!CHAN = R!CHAN + 1            .Increment                          INDD39.83
212      <macro>                    $ENDTIL                                                             INDD39.84
                                                                                                         INDD39.85
                                *    Set up the read-ahead/write-behind areas for all channels.          INDD39.86
                                *    Always initialize pointers in the DCB for the MOS buffer            INDD39.87
                                *    area.  Additionally, if a high speed path to a particular           INDD39.88
                                *    memory exists only on some other IOP, set up pointers in            INDD39.89
                                *    the MCB (in MOS) so that the other IOP may have access              INDD39.90
                                *    to them.  The same set of MOS buffers is used for both              INDD39.91
                                *    the local path and the remote path.                                 INDD39.92
                                                                                                         INDD39.93
213     010000  024442              R!BUF = 0                      .Assume MCB table not needed        INDD39.94
215     014000 /000000  024423      R!%W1 = TMCB                   .Target memory control block        INDD39.95
220     020423  012000  024410      R!%W2 = (R!%W1 + TM$CMEM)      .Check central memory               INDD39.96
        030410  024424
225      <macro>                    $PUNTIF (A = 0),CODE=PT$BPATH .Path must exist                     INDD39.97
227      <macro>                    GET A,TM@IOP,R!%W2             .IOP with high speed path           INDD39.98
233     023000  102003  010001      R!BUF = 1, A # R!%MYID         .If path is remote                  INDD39.99
        024442
237     020423  012002  024410      R!%W2 = (R!%W1 + TM$SSD)       .Check SSD memory                   INDD39.100
        030410  024424
244      <macro>                    $IF (A # 0)                    .If a backdoor exists               INDD39.101
                                                                                        .>>>>> 0            $IF       1
246      <macro>                      GET A,TM@IOP,R!%W2           .IOP with high speed path           INDD39.102
252     023000  102003  010001        R!BUF = 1, A # R!%MYID       .If path is remote                  INDD39.103
        024442
256      <macro>                    $ENDIF                                                              INDD39.104
                                                                                        .<<<<< 0            $ENDIF    1
                                                                                                         INDD39.105
256      <macro>                    $IF (R!BUF # 0)                .If MCB is needed                   INDD39.106
                                                                                        .>>>>> 0            $IF       1
261     020437  012777  004011        R!%W1 = R!MCB + O'777 > D'9 .Number of buffers needed            INDD39.107
        024423
265      <macro>                      MGET R!MOSU,R!MOSL,NUM=R!%W1 .Allocate                           INDD39.108
313      <macro>                      $PUNTIF (A # 0),CODE=PT$NOMOS .Not enough MOS                    INDD39.109
315     010014  024450                R!LEN = CB@SZ@P              .Local buffer size                  INDD39.110
317     072252                        R = GETBUF                   .Get working storage                INDD39.111
```

```
 326   <macro>                    $ENDIF                                                    INDD39.11.
                                                                   .<<<<< 0                     $ENL     1
 320   010000  024441             R!NUM = 0                                                 INDD39.113
 322   <macro>                    $UNTIL (R!COUNT = 0),OR,(R!NUM = O'20) .Scan all DCB's    INDD39.114
 327   020441  016000 /000000       R!DCB = (R!NUM + DCCB)       .Next                      INDD39.115
       024410  030410  024460
 335   <macro>                      $IF (A # 0)                  .If something configured   INDD39.116
                                                                   .>>>>> 0                     $IF      1
 337   <macro>                        GET A,DK@TYP,R!DCB         .Device type               INDD39.117
 345   <macro>                        $IF (A = DSK$DD39)         .If DD39                   INDD39.118
                                                                   .>>>>> 1                     $IF      1
 350   <macro>                          GET A,DK@SEL,R!DCB       .Flag indicating striped disk INDD39.119
 355   <macro>                          $IF (A # 0)              .If striped                INDD39.120
                                                                   .>>>>> 2                     $IF      1
 357   010031  024456                     R!NE = MB$DD39S        .Number of buffers         INDD39.121
 361   <macro>                          $ELSE                                               INDD39.123
 362   010031  024456                     R!NE = MB$DD39         .Normal device             INDD39.124
 364   <macro>                            STORE 1,DK@SEL,R!DCB   .Set released            R13739AG.1
 374   <macro>                          $ENDIF                                              INDD39.125
                                                                   .<<<<< 2                     $ENDIF   1
 374   <macro>                          MGET R!BUFU,R!BUFL,NUM=R!NE .Allocate buffers       INDD39.126
 422   <macro>                          $PUNTIF (A # 0),CODE=PT$NOMOS .Not enough MOS       INDD39.127
 424   <macro>                          $IF (R!BUF # 0)          .If MCB is needed          INDD39.128
                                                                   .>>>>> 2                     $IF      1
 427   <macro>                            PUT R!DCB,DD@DCB,R!BUF .Save DCB address          INDD39.129
 434   010002  005002  022442             R!%W1 = CB@LH < 2 + R!BUF .Entry base             INDD39.130
       024423
 440   <macro>                            PUT R!NE,CB@NUM,R!%W1 .Save number of buffers     INDD39.131
 445   <macro>                            PUT R!BUFU,CB@BM0,R!%W1 .Base address of buffers  INDD39.132
 452   <macro>                            PUT R!BUFL,CB@BM1,R!%W1 .Lower                    INDD39.133
 457   <macro>                            PUT R!MOSU,DK@MC0,R!DCB .MCB address              INDD39.134
 464   <macro>                            PUT R!MOSL,DK@MC1,R!DCB                           INDD39.135
 471   <macro>                            MOSW R!MOSU,R!MOSL,R!BUF,CB@SZ .Write to MOS      INDD39.136
 511   <macro>                            DADD R!MOSU,R!MOSL,CB@SZ .Next MCB                INDD39.137
 515   <macro>                          $ENDIF                                              INDD39.138
                                                                   .<<<<< 2                     $ENDIF   1
 515   <macro>                          PUT R!NE,DK@CNT,R!DCB    .Number of buffers         INDD39.139
 530   <macro>                          PUT R!BUFU,DK@BF0,R!DCB .Base address of buffers   INDD39.140
 535   <macro>                          PUT R!BUFL,DK@BF1,R!DCB                             INDD39.141
 542   027436                           R!COUNT = R!COUNT - 1    .Decrement channel count   INDD39.142
 543   <macro>                        $ENDIF                                                INDD39.143
                                                                   .<<<<< 1                     $ENDIF   1
```

```
543    <macro>          $ENDIF                                                    INDD39.144
                                                                  .<<<<< 0                $ENDIF    1
543   026441            R!NUM = R!NUM + 1          .Next DCB                      INDD39.145
544    <macro>        $ENDTIL                                                     INDD39.146
545    <macro>        $PUNTIF (R!COUNT # 0),CODE=PT$DD49 .If something amiss      INDD39.147
550    <macro>        $IF (R!BUF # 0)            .If MCB needed                   INDD39.148
                                                                  .>>>>> 0                $IF       1
553    <macro>          RELMEM R!BUF             .Release local working storage   INDD39.149
563    <macro>        $ENDIF                                                      INDD39.150
                                                                  .<<<<< 0                $ENDIF    1
563    <macro>        RETURN                                                      INDD39.151
```

```
                                 ************************    ****************************************   INDD39.15
                                 *        SUBROUTINES                                             *   INDD39.154
                                 ************************************************************************   INDD39.155
                                                                                                         INDD39.156
                                 ************************************************************************   INDD39.157
                                 *                                                                      *   INDD39.158
                                 *        GETBUF:    Allocate local memory buffer and clear to zeros.   *   INDD39.159
                                 *                                                                      *   INDD39.160
                                 *        Entry:     R!LEN = Size needed (parcels)                      *   INDD39.161
                                 *                   R!%W1 = ID to store in header                      *   INDD39.162
                                 *                   R!%W2 = "                  "                       *   INDD39.163
                                 *                   R!%W3 = "                  "                       *   INDD39.164
                                 *                                                                      *   INDD39.165
                                 ************************************************************************   INDD39.166
                                                                                                         INDD39.167
571                              GETBUF   *                                                              INDD39.168
571     <macro>                           GETMEM R!LEN,R!BUF,R!%W1,R!%W2,R!%W3 .Allocate memory          INDD39.169
611     <macro>                           $PUNTIF (A # 0),CODE=PT$LOCAL .If not available                INDD39.170
613     <macro>                           CLEAR START=R!BUF,COUNT=R!LEN .Clear                           INDD39.171
626    001000                             EXIT                                                           INDD39.172
                                                                                                         INDD39.173
                                 ************************************************************************   INDD39.174
                                 *                                                                      *   INDD39.175
                                 *        RESIDENT: Establish resident code for DD39, DD49 interrupt    *   INDD39.176
                                 *                  handler.                                            *   INDD39.177
                                 *                                                                      *   INDD39.178
                                 ************************************************************************   INDD39.179
                                                                                                         INDD39.180
627                              RESIDENT *                                                              INDD39.181
627     <macro>                           FIND DD49,R!MOSU,R!MOSL,R!LEN .Locate overlay                  INDD39.182
646     <macro>                           $PUNTIF (A # 0),CODE=PT$NOLAY .Overlay not found               INDD39.183
650    020450  005002  024450             R!LEN = R!LEN < 2             .Size in parcels                 INDD39.184
653    024455                             R!SIZE = A                    .Save overlay size               INDD39.185
654    020450  012012  012006             R!LEN = R!LEN + MEM@LT + DCB@LT .Add MEMIO and DCB queues      INDD39.186
       024450
660    014000 /042103  024423             R!%W1 = 'DC'                  .ASCII header                    INDD39.187
663    014000 /041121  024424             R!%W2 = 'BQ'                                                   INDD39.188
666    010000  024425                     R!%W3 = 0                                                      INDD39.189
670    073077                             R = GETBUF                    .Get local memory                INDD39.190
671    020442  024000                     R!%DCBQ = R!BUF               .DCB queue                       INDD39.191
673    020000  012006  024000             R!%MEMIO = R!%DCBQ + DCB@LT   .MEMIO request queues            INDD39.192
676    020000  012012  012003             R!%DD49 = R!%MEMIO + MEM@LT + 3 & O'177774 .Word boundary      INDD39.193
```

```
        015000 /177774  024000
  704    <macro>                          MOSR R!MOSU,R!MOSL,R!%DD49,R!SIZE>2,NOWAIT .Read in          INDD39.194
  725    001000                           EXIT                                                         INDD39.195
                                                                                                       INDD39.196
                                  ********************************************************************INDD39.197
                                  *                                                                  *INDD39.198
                                  *        CHINIT:    Initialize the channel and perform minimal     *INDD39.199
                                  *                   reliability tests on the device.               *INDD39.200
                                  *                                                                  *INDD39.201
                                  *        Entry:     R!CHAN = Channel number                        *INDD39.202
                                  *                   R!UNIT = Units per channel                     *INDD39.203
                                  *                                                                  *INDD39.204
                                  ********************************************************************INDD39.205
                                                                                                       INDD39.206
  726                             CHINIT   *                                                           INDD39.207
  726    020431  054000                    B = R!CHAN                                                  INDD39.208
  730    <macro>                           $UNTIL (R!UNIT = 0)          .Initialize all units          INDD39.209
  733    027461                              R!UNIT = R!UNIT - 1                                       INDD39.210
                                                                                                       INDD39.211
                                  *        Select the drive.                                           INDD39.212
                                                                                                       INDD39.213
  734    010000  024454                      R!RETRY = 0                                               INDD39.214
  736                             SELECT   *                                                           INDD39.215
  736    077000 /001123                      R = READY                    .Make sure channel is ready   INDD39.216
  740    020443  127000 /001121              P = FATAL, R!ERR # 0         .If error                     INDD39.217
  743    077000 /001161                      R = CLEAR                    .Clear the channel            INDD39.218
  745    020443  127000 /001121              P = FATAL, R!ERR # 0         .If error                     INDD39.219
  750    014000 /010000  022461              A = O'10000 + R!UNIT                                       INDD39.220
  753    161000                              IOB: 1                       .Select unit                  INDD39.221
  754    077000 /001330                      R = WAITDN                   .Wait for DN                  INDD39.222
  756    <macro>                             $IF (R!ERR # 0),OR,(R!CON # R!UNIT < D'8 + 3) .If error   INDD39.223
                                                                                        .>>>>> 0          $IF      1
  771    <macro>                               $IF (R!RETRY < RTR$SLC)    .If more retries              INDD39.224
                                                                                        .>>>>> 1          $IF      1
  776    077000 /001161                        R = CLEAR                  .Clear channel                INDD39.225
 1000    020443  127000 /001121                P = FATAL, R!ERR # 0       .If error                     INDD39.226
 1003    014000 /140000                        A = O'140000                                             INDD39.227
 1005    161000                                IOB: 1                     .Clear faults                 INDD39.228
 1006    072322                                R = WAITDN                 .Wait for DN                  INDD39.229
 1007    026454                                R!RETRY = R!RETRY + 1      .Count up                     INDD39.230
 1010    071052                                P = SELECT                 .Try again                    INDD39.231
 1011    <macro>                             $ENDIF                                                    INDD39.232
```

```
                                                                                .<<<<< 1          $ENL     1
1011   020443  103004  014000          R!ERR = MSG3, R!ERR = 0   .Error code                  INDD39.233
      /002114  024443
1016   070103                          P = FATAL                 .Tell the world               INDD39.234
1017    <macro>                       $ENDIF                                                  INDD39.235
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                INDD39.236
                              *       Analyze general status.  Reset any errors.                INDD39.237
                                                                                                INDD39.238
1017   173000                          IOB: 13                   .Get status register 1        INDD39.239
1020   024451                          R!GEN = A                 .Save (general status)        INDD39.240
1021                          STATUS  *                                                       INDD39.241
1021    <macro>                       $IF (R!GEN # O'1400)      .If error                     INDD39.242
                                                                                .>>>>> 0          $IF      1
1026   020451  011400  024410          P = CAT, 0 = R!GEN & O'400 .If unit ready not set       INDD39.243
       010000  023410  102061
1034   010003  024452                  R!STAT = 3                                              INDD39.244
1036   072145                          R = SELSTAT               .Get selected status 3        INDD39.245
1037   020443  103670                  P = MSGOUT, R!ERR # 0     .If error                     INDD39.246
1041   020452  011400  024410          P = CAT, 0 # R!STAT & O'400 .If DE sequence check       INDD39.247
       010000  023410  103046
1047   072202                          R = RESET                 .Reset the drive              INDD39.248
1050   020443  113657                  R = MSGOUT, R!ERR # 0     .If error                     INDD39.249
1052   020443  102005                  P = RELEASE, R!ERR = 0    .If no error                  INDD39.250
1054   020443  017000 /002214          P = NXTUNIT, R!ERR # MSG6 .If don't want to release     INDD39.251
       103033
1060    <macro>                       $ENDIF                                                  INDD39.252
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                INDD39.253
                              *       Release the drive.                                        INDD39.254
                                                                                                INDD39.255
1060                          RELEASE *                                                       INDD39.256
1060   072101                          R = CLEAR                 .Clear the channel            INDD39.257
1061   020443  103037                  P = FATAL, R!ERR # 0      .If error                     INDD39.258
1063   014000 /170000  022461          A = O'170000 + R!UNIT                                   INDD39.259
1066   161000                          IOB: 1                    .Release unit                 INDD39.260
1067   072241                          R = WAITDN                .Wait for DN                  INDD39.261
1070    <macro>                       $IF (R!ERR # 0),OR,(R!CON # R!UNIT < D'8 + 3) .If error INDD39.262
                                                                                .>>>>> 0          $IF      1
1102   072147                          R = RESET                 .Reset the drive              INDD39.263
1103   020443  103004  014000          R!ERR = MSG7, R!ERR = 0   .Error code                   INDD39.264
      /002241  024443
```

```
1110   070011                                P = FATAL                 .Tell the world                    INDD39.265
1111    <macro>                            $ENDIF                                                        INDD39.266
                                                                                   .<<<<< 0                $ENDIF    1
1111   072050                              R = CLEAR                 .Clear channel                     INDD39.267
1112                             NXTUNIT   *                                                             INDD39.268
1112    <macro>                            $ENDTIL                                                       INDD39.269
1113   001000                              EXIT                                                          INDD39.270
                                                                                                         INDD39.271
1114                             CAT       *                                                             INDD39.272
1114   014000 /002140  024443              R!ERR = MSG4              .Error code                        INDD39.273
1117   072611                              R = MSGOUT                .Tell the world                    INDD39.274
1120   071040                              P = RELEASE               .Release the drive                 INDD39.275
1121                             FATAL     *                                                             INDD39.276
1121   072607                              R = MSGOUT                .Tell the world                    INDD39.277
1122   071010                              P = NXTUNIT               .Try next unit                     INDD39.278
                                                                                                         INDD39.279
                                 ***********************************************************************INDD39.280
                                 *                                                                     *INDD39.281
                                 *         READY:    Get channel ready status.                           *INDD39.282
                                 *                                                                     *INDD39.283
                                 *         Exit:     R!ERR = error code                                  *INDD39.284
                                 *                                                                     *INDD39.285
                                 ***********************************************************************INDD39.286
                                                                                                         INDD39.287
1123                             READY     *                                                             INDD39.288
1123   010000  024424                      R!%W2 = 0                                                     INDD39.289
1125                             READYA    *                                                             INDD39.290
1125   072034                              R = CLEAR                 .Clear the channel                 INDD39.291
1126   020443  102002  001000              EXIT, R!ERR # 0           .If error                          INDD39.292
1131   010000                              A = 0                                                         INDD39.293
1132   177000                              IOB: 17                   .Get ready status                  INDD39.294
1133   072175                              R = WAITDN                .Wait for DN                       INDD39.295
1134    <macro>                            $IF (R!ERR # 0),OR,(0 = R!CON & 1) .If not good              INDD39.296
                                                                                   .>>>>> 0                $IF       1
1145    <macro>                              $IF (R!%W2 < RTR$RDY)   .If more retries                   INDD39.297
                                                                                   .>>>>> 1                $IF       1
1151   026424                                R!%W2 = R!%W2 + 1       .Count up                          INDD39.298
1152   071025                                P = READYA              .Try again                         INDD39.299
1153    <macro>                              $ENDIF                                                     INDD39.300
                                                                                   .<<<<< 1                $ENDIF    1
1153   020443  103004  014000                R!ERR = MSG2, R!ERR = 0 .Error code                        INDD39.301
      /002072  024443
```

```
1160    <macro>                         $ENDIF                                                  INDD39.302
                                                                          .<<<<< 0                $ENDIF    1
1160    001000                          EXIT                                                    INDD39.303
                                                                                                INDD39.304
                                 ********************************************************       INDD39.305
                                 *                                                      *       INDD39.306
                                 *       CLEAR:    Clear the DD39 disk channel.         *       INDD39.307
                                 *                                                      *       INDD39.308
                                 *       Exit:     R!ERR = error code                   *       INDD39.309
                                 *                                                      *       INDD39.310
                                 ********************************************************       INDD39.311
                                                                                                INDD39.312
1161                             CLEAR   *                                                      INDD39.313
1161    010000  024423                   R!%W1 = 0                                              INDD39.314
1163                             CLEARA  *                                                      INDD39.315
1163    160000                           IOB: 0                     .Clear channel              INDD39.316
1164    010002  025427                   R!%W5 = R!%W5 + 2          .Delay 6 C.P.'s             INDD39.317
1166    <macro>                          $IF (IOB = DN),OR,(IOB = BZ)  .If not cleared          INDD39.318
                                                                          .>>>>> 0                $IF       1
1173    020423  013001  101160             P = CHNERR, R!%W1 >= RTR$CLR .Channel error          INDD39.319
1176    026423                             R!%W1 = R!%W1 + 1          .Count up                 INDD39.320
1177    071014                             P = CLEARA                 .Try again                INDD39.321
1200    <macro>                          $ENDIF                                                 INDD39.322
                                                                          .<<<<< 0                $ENDIF    1
1200    010000  024443                   R!ERR = 0                                              INDD39.323
1202    001000                           EXIT                                                   INDD39.324
                                                                                                INDD39.325
                                 ********************************************************       INDD39.326
                                 *                                                      *       INDD39.327
                                 *       SELSTAT:  Get selected status.                 *       INDD39.328
                                 *                                                      *       INDD39.329
                                 *       Entry:    R!STAT = status number desired       *       INDD39.330
                                 *                                                      *       INDD39.331
                                 *       Exit:     R!ERR = error code                   *       INDD39.332
                                 *                 R!STAT = selected status, if obtainable *    INDD39.333
                                 *                                                      *       INDD39.334
                                 ********************************************************       INDD39.335
                                                                                                INDD39.336
1203                             SELSTAT *                                                      INDD39.337
1203    010000  024425                   R!%W3 = 0                                              INDD39.338
1205                             SELSTATA *                                                     INDD39.339
1205    073024                           R = CLEAR                  .Clear channel              INDD39.340
```

```
1206  020443  102002  001000          EXIT, R!ERR # 0                 .If error                          INDD39.341
1211  020452  005010  012003          R!%W4 = R!STAT < D'8 + 3        .Save for channel echo             INDD39.342
      024426
1215  020452  016000 /070000          A = R!STAT + O'70000                                               INDD39.343
1220  161000                          IOB: 1                          .Get selected status               INDD39.344
1221  072107                          R = WAITDN                      .Wait for DN                       INDD39.345
1222   <macro>                        $IF (R!ERR = 0),AND,(R!CON = R!%W4) .If OK                         INDD39.346
                                                                                        .>>>>> 0              $IF      1
1231  173000                            IOB: 13                       .Read in                           INDD39.347
1232  024452                            R!STAT = A                    .Save                              INDD39.348
1233   <macro>                        $ELSEIF (R!%W3 < RTR$STA)       .If more retries                   INDD39.349
1240  026425                            R!%W3 = R!%W3 + 1             .Count up                          INDD39.350
1241  071034                            P = SELSTATA                  .Try again                         INDD39.351
1242   <macro>                        $ELSE                                                              INDD39.352
1243  020443  103004  014000            R!ERR = MSG5, R!ERR = 0       .If error code not set yet         INDD39.353
     /002166  024443
1250   <macro>                        $ENDIF                                                             INDD39.354
                                                                                        .<<<<< 0              $ENDIF   1
1250  001000                          EXIT                                                               INDD39.355
                                                                                                         INDD39.356
                                 ***********************************************************************  INDD39.357
                                 *                                                                     *  INDD39.358
                                 *        RESET:    Reset the DD39 disk.                                 *  INDD39.359
                                 *                                                                     *  INDD39.360
                                 *        Exit:     R!ERR = error code                                   *  INDD39.361
                                 *                                                                     *  INDD39.362
                                 ***********************************************************************  INDD39.363
                                                                                                         INDD39.364
1251                             RESET    *                                                             INDD39.365
1251  010000  024454                      R!RETRY = 0                                                    INDD39.366
1253                             RESETA   *                                                             INDD39.367
1253  073072                              R = CLEAR                     .Clear channel                     INDD39.368
1254  020443  102002  001000              EXIT, R!ERR # 0               .If error                          INDD39.369
1257  014000 /130000                      A = O'130000                                                   INDD39.370
1261  161000                              IOB: 1                        .Reset                             INDD39.371
1262   <macro>                            PAUSE D'20                    .Wait for completion               INDD39.372
1273  020431  054000                      B = R!CHAN                    .Set B                             INDD39.373
1275  072033                              R = WAITDN                    .Check for DN                      INDD39.374
1276   <macro>                            $IF (R!ERR = 0),AND,(R!CON = 3) .If OK                           INDD39.375
                                                                                        .>>>>> 0              $IF      1
1305  173000                              IOB: 13                       .Get general status                INDD39.376
1306  024451                              R!GEN = A                     .Save                              INDD39.377
```

```
1307   017000 /001400  106130          P = CLEAR, A = 0     0      .If OK                          INDD39.378
1312    <macro>                      $ENDIF                                                        INDD39.379
                                                                                  .<<<<< 0              $ENDIF   1
1312    <macro>                      $IF (R!RETRY < RTR$RES)      .If more retries                 INDD39.380
                                                                                  .>>>>> 0              $IF      1
1316   026454                          R!RETRY = R!RETRY + 1        .Count up                       INDD39.381
1317   071044                          P = RESETA                   .Try again                      INDD39.382
1320    <macro>                      $ENDIF                                                        INDD39.383
                                                                                  .<<<<< 0              $ENDIF   1
1320   020443  017000 /002027        R!ERR = MSG6, R!ERR # MSG0   .If not timeout                 INDD39.384
       102004  014000 /002214
       024443                                                                                        .
1327   001000                        EXIT                                                          INDD39.385
                                                                                                     INDD39.386
                                 ***********************************************************************INDD39.387
                                 *                                                                   *INDD39.388
                                 *        WAITDN:   Wait for DN flag on DD39 disk channel.           *INDD39.389
                                 *                                                                   *INDD39.390
                                 *        Exit:     R!ERR = error code                               *INDD39.391
                                 *                  R!CON = Status register 0 (controller status)    *INDD39.392
                                 *                                                                   *INDD39.393
                                 ***********************************************************************INDD39.394
                                                                                                     INDD39.395
1330                             WAITDN     *                                                        INDD39.396
1330   010000                        A = 0                        .Initialize                     INDD39.397
1331    <macro>                      $UNTIL (IOB = DN)            .Until DN sets                  INDD39.398
1333   013001                          A = A - 1                                                    INDD39.399
1334   102014                          P = TIMEOUT, A = 0           .If timeout                     INDD39.400
1335    <macro>                      $ENDTIL                                                       INDD39.401
1336   166000                        IOB: 6                       .Clear interrupt                INDD39.402
1337   010002  025427                R!%W5 = R!%W5 + 2            .Delay                          INDD39.403
1341   172000                        IOB: 12                      .Status register 0              INDD39.404
1342   024453                        R!CON = A                    .Save                           INDD39.405
1343   043000  101011                P = CHNERR, IOB = BZ         .If channel error               INDD39.406
1345   010000  024443                R!ERR = 0                                                    INDD39.407
1347   001000                        EXIT                                                          INDD39.408
                                                                                                     INDD39.409
1350                             TIMEOUT    *                                                        INDD39.410
1350   166000                        IOB: 6                       .Clear interrupt                INDD39.411
1351   014000 /002027  024443        R!ERR = MSG0                 .Error code                     INDD39.412
1354   001000                        EXIT                                                          INDD39.413
1355                             CHNERR     *                                                        INDD39.414
```

```
1355   014000 /002051  024443          R!ERR = MSG1                    .Error code                    INDD39.415
1360   001000                          EXIT                                                           INDD39.416
                                                                                                      INDD39.417
                                 ***********************************************************************INDD39.418
                                 *                                                                     *INDD39.419
                                 *        SETDCB:   Set up the Disk Control Block (DCB) for this       *INDD39.420
                                 *                  channel.                                           *INDD39.421
                                 *                                                                     *INDD39.422
                                 *        Entry:    R!CHAN = Channel number                            *INDD39.423
                                 *                                                                     *INDD39.424
                                 ***********************************************************************INDD39.425
                                                                                                      INDD39.426
1361                             SETDCB   *                                                           INDD39.427
1361   010066  012014  024450          R!LEN = DK@LT + LB@SZ           .Size needed                   INDD39.428
1364   014000 /042113  024423          R!%W1 = 'DK'                    .Table name                    INDD39.429
1367   014000 /040040  024424          R!%W2 = '@ '                                                   INDD39.430
1372   020431  004003  012060          R!%W3 = R!CHAN > 3 + 'O'R < D'8 .Convert ch.# to ASCII         INDD39.431
       005010  024425
1377   020431  011007  012060          R!%W3 = R!CHAN & 7 + 'O'R + R!%W3                              INDD39.432
       022425  024425
1404   073613                          R = GETBUF                      .Allocate local memory         INDD39.433
1405   020431  013020  016000          R!%W1 = R!CHAN - O'20 + DCCB    .Channel table cell            INDD39.434
      /000000  024423
1412   020442  034423                  (R!%W1) = R!BUF                 .Save pointer to DCB           INDD39.435
1414    <macro>                        STORE DSK$DD39,DK@TYP,R!BUF     .Channel type                  INDD39.437
1425    <macro>                        PUT R!CHAN,DK@CHN,R!BUF         .Channel number                INDD39.438
1440    <macro>                        STORE O'177777,DK@CYL,R!BUF     .Force initial seek            INDD39.439
1446    <macro>                        STORE O'37,DK@HED,R!BUF         .Force initial head select     INDD39.440
1457   010003  024461                  R!UNIT = D3$UPC                 .Units/channel                 INDD39.441
1461    <macro>                        PUT R!UNIT,DK@UPC,R!BUF         .Units/channel                 INDD39.442
1474    <macro>                        STORE D3$CPD,DK@CPD,R!BUF       .Cylinders/disk                INDD39.443
1502    <macro>                        STORE D3$TPC,DK@TPC,R!BUF       .Tracks/cylinder               INDD39.444
1513    <macro>                        STORE D3$SPT,DK@SPT,R!BUF       .Sectors/track                 INDD39.445
1523    <macro>                        STORE I@SPIRD3,DK@SPF,R!BUF     .Spiral enable flag            INDD39.446
1534    <macro>                        PUT R!DPT,DK@DPT,R!BUF          .Device parameter table        INDD39.447
1541   010066  022442  024423          R!%W1 = DK@LT + R!BUF           .First LMA entry               INDD39.448
1544   020423  012014  013006          R!%W2 = R!%W1 + LB@SZ - LB@LE   .Last LMA entry                INDD39.449
       024424
1550    <macro>                        PUT R!%W1,DK@FLB,R!BUF          .Save first                    INDD39.450
1555    <macro>                        PUT R!%W1,DK@CLB,R!BUF          .Initialize current            INDD39.451
1562    <macro>                        PUT R!%W2,DK@LLB,R!BUF          .Save last                     INDD39.452
1567    <macro>                        STORE LB$DD39,DK@NLB,R!BUF      .Number of LMA entries         INDD39.453
```

```
1571    <macro>                          $UNTIL (R!%W1 > R!          .Do all entries                    INDD39.454
1604    <macro>                            PUT R!BUF,LB@DCB,R!%W1    .Set DCB address                   INDD39.455
1611    010006  025423                     R!%W1 = R!%W1 + LB@LE     .Next                              INDD39.456
1613    <macro>                          $ENDTIL                                                        INDD39.457
1614    <macro>                          ADDRESS R!%W1,DK@TMO,R!BUF  .Timout table                        INDD39.458
1617    <macro>                          PUT R!TOUT,TMR@RT,R!%W1     .Timout routine address              INDD39.459
1624    <macro>                          $IF (R!TYPE = CH$D3S)       .If striped disk                     INDD39.460
                                                                                        .>>>>> 0          $IF      1
1630    010030  024424                     R!%W2 = RA$DD39S          .Read ahead count                  INDD39.461
1632    010030  024425                     R!%W3 = WB$DD39S          .Write behind count                INDD39.462
1634    <macro>                            STORE 1,DK@SEL,R!BUF      .Temp. flag for initialization     INDD39.463
1644    <macro>                          $ELSE                                                          INDD39.464
1645    010030  024424                     R!%W2 = RA$DD39                                              INDD39.465
1647    010030  024425                     R!%W3 = WB$DD39                                              INDD39.466
1651    <macro>                          $ENDIF                                                         INDD39.467
                                                                                        .<<<<< 0          $ENDIF   1
1651    <macro>                          PUT R!%W2,DK@RCC,R!BUF      .Store in DCB                      INDD39.468
1664    <macro>                          PUT R!%W3,DK@WCC,R!BUF                                         INDD39.469
1677    <macro>                          PUT R!%W2,DK@DRC,R!BUF      .Set default counts                INDD39.470
1712    <macro>                          PUT R!%W3,DK@DWC,R!BUF                                         INDD39.471
1725    010003  025437                   R!MCB = R!MCB + CB@SZ       .Running sum for MCB space         INDD39.472
1727    001000                           EXIT                                                           INDD39.473
                                                                                                        INDD39.474
                                 ***********************************************************************  INDD39.475
                                 *                                                                     *  INDD39.476
                                 *        MSGOUT:    Send error message to the console.                 *  INDD39.477
                                 *                                                                     *  INDD39.478
                                 *        Entry:     R!CHAN = Channel number                            *  INDD39.479
                                 *                   R!UNIT = Unit number                               *  INDD39.480
                                 *                   R!ERR = Error message code                         *  INDD39.481
                                 *                                                                     *  INDD39.482
                                 ***********************************************************************  INDD39.483
                                                                                                        INDD39.484
1730                             MSGOUT   *                                                             INDD39.485
1730    010044  024450                   R!LEN = MSGLEN              .Size                              INDD39.486
1732    077000 /000571                   R = GETBUF                  .Allocate local memory             INDD39.487
1734    <macro>                          COPY R!ERR+R!%B,R!BUF,R!LEN .Move msg to buffer                INDD39.488
1753    020431  004003  012060           R!%W1 = R!CHAN > 3 + '0'R < D'8 .Convert channel to ASCII      INDD39.489
        005010  024423
1760    020431  011007  012060           R!%W1 = R!CHAN & 7 + '0'R + R!%W1                              INDD39.490
        022423  024423
1765    020442  012004  024410           (R!BUF + CHNOFF) = R!%W1    .Store in msg                      INDD39.491
```

```
        020423  034410
1772    020461  016000 /020060        R!%W2 = R!UNIT + ' 0'R        .Convert unit to ASCII        INDD39.492
        024424
1776    020442  012010  024410        (R!BUF + UNTOFF) = R!%W2      .Store in msg                 INDD39.493
        020424  034410
2003     <macro>                      MSG R!BUI                     .Send to console              INDD39.494
2014     <macro>                      RELMEM R!BUF                  .Release local memory         INDD39.495
2024    020431  054000                B = R!CHAN                                                  INDD39.496
2026    001000                        EXIT                                                        INDD39.497
```

```
                        *        Message data.                                          INDD39.49
                                                                                        INDD39.500
2027                    MSG0     *                                                      INDD39.501
2027                             'CHANNEL XX UNIT XX CHANNEL TIMEOUT'                   INDD39.502
2051                    MSG1     *                                                      INDD39.503
2051                             'CHANNEL XX UNIT XX CHANNEL ERROR'                     INDD39.504
2072                    MSG2     *                                                      INDD39.505
2072                             'CHANNEL XX UNIT XX DEVICE NOT READY'                  INDD39.506
2114                    MSG3     *                                                      INDD39.507
2114                             'CHANNEL XX UNIT XX DEVICE SELECT ERROR'               INDD39.508
2140                    MSG4     *                                                      INDD39.509
2140                             'CHANNEL XX UNIT XX CATASTROPHIC DRIVE ERROR'          INDD39.510
2166                    MSG5     *                                                      INDD39.511
2166                             'CHANNEL XX UNIT XX UNABLE TO OBTAIN STATUS'           INDD39.512
2214                    MSG6     *                                                      INDD39.513
2214                             'CHANNEL XX UNIT XX UNABLE TO RESET DEVICE'            INDD39.514
2241                    MSG7     *                                                      INDD39.515
2241                             'CHANNEL XX UNIT XX DEVICE RELEASE ERROR'              INDD39.516
                                                                                        INDD39.517
                        *        Spiral conversion table.                               INDD39.518
                                                                                        INDD39.519
2265                    D3CNVRT  *                                                      INDD39.520
2265   000000                    0                                                      INDD39.521
2266   000014                    D'12                                                   INDD39.522
                                 END                                                    INDD39.523
```

0X	%ACTIVE		5:39 D	5:39	6:21 D	6:21				
0X	%B		1: 4 D	4:10	5: 2 B	9:17 B	9:24 B	9:25 B	9:26 B	9:27 B
			9:30 B	9:32 B	9:34 B	9:35 B	9:36 B	16:36 B	16:37	
0X	%DCBQ		3:16 D	8:40	8:41					
0X	%DD49		3:17 D	4:38	4:40	4:42	9: 1	9: 2		
0X	%EX		5:39 D	5:39	6:21 D	6:21	6:33 D	6:33	7: 8 D	7: 8
			7:11 D	7:11	8:17 D	8:17	8:30 D	8:30	9: 2 D	9: 2
			13:36 D	13:36	17: 6 D	17: 6	17: 7 D	17: 7		
10	%GBLNUM	$APTEXT	2:14 F	2:14						
400	%GBLREG	$APTEXT	2:14							
0X	%MEMIO		3:18 D	8:41	8:42					
0X	%MYID		3:19 D	4:23	5:23	5:30				
0X	%PUNT		5:21 D	5:21	5:40 D	5:40	6:22 D	6:22	7: 5 D	7: 5
			8:18 D	8:18	8:31 D	8:31				
20	AD@P1@N	$APTEXT	5:39	5:39 F	6:21	6:21 F				
14	AD@P1@P	$APTEXT	5:39 F	5:39	6:21 F	6:21				
0	AD@P1@S	$APTEXT	5:39	5:39 F	6:21	6:21 F				
20	AD@P2@N	$APTEXT	5:39	5:39 F	6:21	6:21 F				
15	AD@P2@P	$APTEXT	5:39 F	5:39	6:21 F	6:21				
0	AD@P2@S	$APTEXT	5:39	5:39 F	6:21	6:21 F				
20	AD@RC@N	$APTEXT	5:39	5:39 F	6:21	6:21 F				
13	AD@RC@P	$APTEXT	5:39 F	5:39	6:21 F	6:21				
0	AD@RC@S	$APTEXT	5:39	5:39 F	6:21	6:21 F				
20	CB@BMO@N	$APTEXT	6:29 F							
2	CB@BMO@P	$APTEXT	6:29 F	6:29						
0	CB@BMO@S	$APTEXT	6:29 F							
20	CB@BM1@N	$APTEXT	6:30 F							
3	CB@BM1@P	$APTEXT	6:30 F	6:30						
0	CB@BM1@S	$APTEXT	6:30 F							
2	CB@LH	$APTEXT	6:26							
20	CB@NUM@N	$APTEXT	6:28 F							
1	CB@NUM@P	$APTEXT	6:28 F	6:28						
0	CB@NUM@S	$APTEXT	6:28 F							
3	CB@SZ	$APTEXT	6:33	6:34	16:21					
14	CB@SZ@P	$APTEXT	5:41							
4	CH$D3	$APTEXT	4:36							
104	CH$D3S	$APTEXT	4:36	16: 8						
1510	D3$CPD	$APTEXT	4:16	15:31						
1512	D3$CPDA	$APTEXT	4:18							
30	D3$SPT	$APTEXT	4:15	15:33						
31	D3$SPTA	$APTEXT	4:17							
5	D3$TPC	$APTEXT	15:32							

3	D3$UPC	$APTEXT	15:29			
0X	D4TOUT		3:20	D	4:42	
6	DCB@LT	$APTEXT	8:34		8:41	
0X	DCCB		3:21	D	6: 5	15:22
0X	DD49		3:22	D	4:40	
20	DD@DCB@N	$APTEXT	6:25	F		
2	DD@DCB@P	$APTEXT	6:25	F	6:25	
0	DD@DCB@S	$APTEXT	6:25	F		
20	DK@BFO@N	$APTEXT	6:38	F		
63	DK@BFO@P	$APTEXT	6:38	F	6:38	
0	DK@BFO@S	$APTEXT	6:38	F		
20	DK@BF1@N	$APTEXT	6:39	F		
64	DK@BF1@P	$APTEXT	6:39	F	6:39	
0	DK@BF1@S	$APTEXT	6:39	F		
777	DK@CHN@M	$APTEXT	15:26			
11	DK@CHN@N	$APTEXT	15:26	F		
7	DK@CHN@P	$APTEXT	15:26	F	15:26	
7	DK@CHN@S	$APTEXT	15:26	F		
177000	DK@CHN@X	$APTEXT	15:26			
20	DK@CLB@N	$APTEXT	15:40	F		
13	DK@CLB@P	$APTEXT	15:40	F	15:40	
0	DK@CLB@S	$APTEXT	15:40	F		
377	DK@CNT@M	$APTEXT	6:37			
10	DK@CNT@N	$APTEXT	6:37	F		
16	DK@CNT@P	$APTEXT	6:37	F	6:37	
10	DK@CNT@S	$APTEXT	6:37	F		
177400	DK@CNT@X	$APTEXT	6:37			
20	DK@CPD@N	$APTEXT	15:31	F		
57	DK@CPD@P	$APTEXT	15:31	F	15:31	
0	DK@CPD@S	$APTEXT	15:31	F		
20	DK@CYL@N	$APTEXT	15:27	F		
3	DK@CYL@P	$APTEXT	15:27	F	15:27	
0	DK@CYL@S	$APTEXT	15:27	F		
20	DK@DPI@N	$APTEXT	15:35	F		
61	DK@DPI@P	$APTEXT	15:35	F	15:35	
0	DK@DPI@S	$APTEXT	15:35	F		
377	DK@DRC@M	$APTEXT	16:19			
10	DK@DRC@N	$APTEXT	16:19	F	16:19	
54	DK@DRC@P	$APTEXT	16:19	F	16:19	
0	DK@DRC@S	$APTEXT	16:19	F	16:19	
377	DK@DWC@M	$APTEXT	16:20			
10	DK@DWC@N	$APTEXT	16:20	F	16:20	

55	DK@DWC@P	$APTEXT	16:20	F	16:20						
0	DK@DWC@S	$APTEXT	16:20	F	16:20						
20	DK@FLB@N	$APTEXT	15:39	F							
11	DK@FLB@P	$APTEXT	15:39	F	15:39						
0	DK@FLB@S	$APTEXT	15:39	F							
37	DK@HED@M	$APTEXT	15:28	F							
5	DK@HED@N	$APTEXT	15:28	F	15:28						
4	DK@HED@P	$APTEXT	15:28	F	15:28						
2	DK@HED@S	$APTEXT	15:28	F	15:28						
20	DK@LLB@N	$APTEXT	15:41	F							
12	DK@LLB@P	$APTEXT	15:41	F	15:41						
0	DK@LLB@S	$APTEXT	15:41	F							
66	DK@LJ	$APTEXT	15:14		15:36						
20	DK@MCO@N	$APTEXT	6:31	F							
32	DK@MCO@P	$APTEXT	6:31	F	6:31						
0	DK@MCO@S	$APTEXT	6:31	F							
20	DK@MC1@N	$APTEXT	6:32	F							
33	DK@MC1@P	$APTEXT	6:32	F	6:32						
0	DK@MC1@S	$APTEXT	6:32	F							
17	DK@NLB@M	$APTEXT	15:42	F							
4	DK@NLB@N	$APTEXT	15:42	F	15:42						
56	DK@NLB@P	$APTEXT	15:42	F	15:42						
0	DK@NLB@S	$APTEXT	15:42	F	15:42						
377	DK@RCC@M	$APTEXT	16:17								
10	DK@RCC@N	$APTEXT	16:17	F							
54	DK@RCC@P	$APTEXT	16:17	F	16:17						
10	DK@RCC@S	$APTEXT	16:17	F							
177400	DK@RCC@X	$APTEXT	16:17								
1	DK@SEL@M	$APTEXT	6:18	F	16:11	F					
1	DK@SEL@N	$APTEXT	6:12		6:12	F	6:18	F	6:18	16:11	F 16:11
7	DK@SEL@P	$APTEXT	6:12	F	6:12		6:18	F	6:18	16:11	F 16:11
0	DK@SEL@S	$APTEXT	6:12		6:12	F	6:18	F	6:18	16:11	F 16:11
1	DK@SPF@M	$APTEXT	15:34	F							
1	DK@SPF@N	$APTEXT	15:34	F	15:34						
2	DK@SPF@P	$APTEXT	15:34	F	15:34						
7	DK@SPF@S	$APTEXT	15:34	F	15:34						
777	DK@SPI@M	$APTEXT	15:33	F							
11	DK@SPI@N	$APTEXT	15:33	F							
60	DK@SPI@P	$APTEXT	15:33	F	15:33						
7	DK@SPI@S	$APTEXT	15:33	F							
177000	DK@SPI@X	$APTEXT	15:33								
44	DK@TMO@P	$APTEXT	16: 5								

37	DK@IPC@M	$APTEXT	15:32	F						
5	DK@IPC@N	$APTEXT	15:32	F	15:32					
60	DK@IPC@P	$APTEXT	15:32	F	15:32					
2	DK@IPC@S	$APTEXT	15:32	F	15:32					
37	DK@IYP@M	$APTEXT	6: 9		15:25	F				
5	DK@IYP@N	$APTEXT	6: 9		6: 9	F	15:25	F	15:25	
7	DK@IYP@P	$APTEXT	6: 9	F	6: 9		15:25	F	15:25	
2	DK@IYP@S	$APTEXT	6: 9		6: 9	F	15:25	F	15:25	
3	DK@UPC@M	$APTEXT	15:30							
2	DK@UPC@N	$APTEXT	15:30	F	15:30					
60	DK@UPC@P	$APTEXT	15:30	F	15:30					
0	DK@UPC@S	$APTEXT	15:30	F	15:30					
377	DK@WCC@M	$APTEXT	16:18							
10	DK@WCC@N	$APTEXT	16:18	F						
55	DK@WCC@P	$APTEXT	16:18	F	16:18					
10	DK@WCC@S	$APTEXT	16:18	F						
177400	DK@WCC@X	$APTEXT	16:18							
20	DP@ACY@N	$APTEXT	4:18	F						
10	DP@ACY@P	$APTEXT	4:18	F	4:18					
0	DP@ACY@S	$APTEXT	4:18	F						
777	DP@ASE@M	$APTEXT	4:17	F						
11	DP@ASE@N	$APTEXT	4:17	F						
7	DP@ASE@P	$APTEXT	4:17	F	4:17					
7	DP@ASE@S	$APTEXT	4:17	F						
177000	DP@ASE@X	$APTEXT	4:17							
20	DP@DCY@N	$APTEXT	4:16	F						
6	DP@DCY@P	$APTEXT	4:16	F	4:16					
0	DP@DCY@S	$APTEXT	4:16	F						
777	DP@DSE@M	$APTEXT	4:15	F						
11	DP@DSE@N	$APTEXT	4:15	F						
5	DP@DSE@P	$APTEXT	4:15	F	4:15					
7	DP@DSE@S	$APTEXT	4:15	F						
177000	DP@DSE@X	$APTEXT	4:15							
17	DP@IDX@M	$APTEXT	4:14	F						
4	DP@IDX@N	$APTEXT	4:14	F						
4	DP@IDX@P	$APTEXT	4:14	F	4:14					
14	DP@IDX@S	$APTEXT	4:14	F						
177760	DP@IDX@X	$APTEXT	4:14							
11	DP@LT	$APTEXT	4: 4							
1	DP@SPI@M	$APTEXT	4:13	F						
1	DP@SPI@N	$APTEXT	4:13	F	4:13					
4	DP@SPI@P	$APTEXT	4:13	F	4:13					

0	DP@SPI@S	$APTEXT	4:13	F	4:13			
0	DP@SPI@P	$APTEXT	4:12					
4	DSK$DD39	$APTEXT	6:11		15:25			
0X	EITB		3:23	D	4:22			
53	F$FIND	$APTEXT	8:30					
30	F$GMEM	$APTEXT	8:17					
20	F$MESG	$APTEXT	17: 6					
35	F$MGET	$APTEXT	5:39		6:21			
46	F$MOSR	$APTEXT	9: 2					
47	F$MOSW	$APTEXT	6:33					
7	F$PAUSE	$APTEXT	13:36					
52	F$RETURN	$APTEXT	7:11					
31	F$RMEM	$APTEXT	7: 8		17: 7			
1	FS$NWAIT	$APTEXT	9: 2					
0	FS$WAIT	$APTEXT	6:33					
1	I@INDXD3	$APTEXT	4:14					
1	I@SPIRD3	$APTEXT	4:13		15:34			
2	LB$DD39	$APTEXT	3:26		15:42			
20	LB@DCB@N	$APTEXT	16: 2	F				
5	LB@DCB@P	$APTEXT	16: 2	F	16: 2			
0	LB@DCB@S	$APTEXT	16: 2	F				
6	LB@LE	$APTEXT	3:26		15:37		16: 3	
31	MB$DD39	$APTEXT	6:17					
31	MB$DD39S	$APTEXT	6:15					
12	MEM@LT	$APTEXT	8:34		8:42			
0X	O$DD49		8:30	D	8:30			
0X	O$INDD39		1: 4	D	1: 4			
6	OV@ENT@P	$APTEXT	4:23					
17	OV@NUM@N	$APTEXT	1: 4					
1	OV@TYP@N	$APTEXT	1: 4					
112	PT$BPATH	$APTEXT	5:21					
50	PT$DD49	$APTEXT	7: 5					
14	PT$LOCAL	$APTEXT	8:18					
12	PT$NOLAY	$APTEXT	8:31					
15	PT$NOMOS	$APTEXT	5:40		6:22			
30	RA$DD39	$APTEXT	16:13					
30	RA$DD39S	$APTEXT	16: 9					
11	SM@FST@N	$APTEXT	1: 4					
7	SM@NUM@N	$APTEXT	1: 4					
0	TM$CMEM	$APTEXT	5:19					
2	TM$SSD	$APTEXT	5:25					
20	TM@IOP@N	$APTEXT	5:22		5:22	F	5:29	5:29 F

```
  5   IM@IOP@P  $APTEXT     5:22 F    5:22        J F    5:29
  0   IM@IOP@S  $APTEXT     5:22      5:22 F    5:29     5:29 F
 0X   IMCB                  3:24 D    5:18
 20   IMR@RI@N  $APTEXT    16: 6 F
  2   IMR@RI@P  $APTEXT    16: 6 F   16: 6
  0   IMR@RI@S  $APTEXT    16: 6 F
 30   WB$DD39   $APTEXT    16:14
 30   WB$DD39S  $APTEXT    16:10

      SYMBOL QUALIFIER IS INDD39

  1   $CONST                4:13 D    4:13 F    4:13      4:14 D    4:14 F    4:14      4:15 D    4:15 F
                            4:15      4:16 D    4:16 F    4:16      4:17 D    4:17 F    4:17      4:18 D
                            4:18 F    4:18      6:18 D    6:18 F    6:18     15:25 D   15:25 F   15:25
                           15:27 D   15:27 F   15:27     15:28 D   15:28 F   15:28     15:31 D   15:31 F
                           15:31     15:32 D   15:32 F   15:32     15:33 D   15:33 F   15:33     15:34 D
                           15:34 F   15:34     15:42 D   15:42 F   15:42     16:11 D   16:11 F   16:11
462   $REGORG               2:14 D    2:14 F    2:15      2:15 D    2:16      2:16 D    2:17      2:17 D
                            2:18      2:18 D    2:19      2:19 D    2:19 F    2:20      2:20 D    2:21
                            2:21 D    2:22      2:22 D    2:23      2:23 D    2:24      2:24 D    2:25
                            2:25 D    2:25 F    2:26      2:26 D    2:27      2:27 D    2:28      2:28 D
                            2:29      2:29 D    2:30      2:30 D    2:30 F    2:31      2:31 D    2:31 F
                            2:32      2:32 D    2:33      2:33 D    2:34      2:34 D    2:35      2:35 D
                            2:36      2:36 D    2:37      2:37 D    2:38      2:38 D    2:39      2:39 D
                            2:40      2:40 D    2:41      2:41 D    2:42      2:42 D    3: 1      3: 1 D
                            3: 2      3: 2 D    3: 3      3: 3 D    3: 4      3: 4 D    3: 5      3: 5 D
                            3: 6      3: 6 D    3: 7      3: 7 D    3: 8      3: 8 D    3: 9      3: 9 D
                           3:10      3:10 D    3:11      3:11 D    3:12      3:12 D    3:13      3:13 D
                           3:14      3:14 D    3:14 F    5:39      6:21      6:33      7: 8      7:11
                           8:17      8:30      9: 2     13:36     13:36 F   17: 6     17: 6 F   17: 7
  1   %NP                  1: 4      2:31 D
430   %P                   1: 4      2:30 D    2:31     13:36 F   13:36     17: 6 F   17: 6
410   %S1                  2:14      2:15 D    4:13      4:14      4:15      4:16      4:17      4:18
                           5:19      5:22      5:25      5:29      5:39      6: 5      6: 6      6: 9
                           6:12      6:18      6:21      6:25      6:28      6:29      6:30      6:31
                           6:32      6:37      6:38      6:39      9:32     10:15     10:16     10:20
                          10:21     10:39     11:34     15:25     15:26     15:27     15:28     15:30
                          15:31     15:32     15:33     15:34     15:35     15:39     15:40     15:41
                          15:42     16: 2     16: 6     16:11     16:17     16:18     16:19     16:20
                          16:42     17: 1     17: 4     17: 5
411   %S2                  2:14      2:16 D
412   %S3                  2:14      2:17 D
```

413	%S4	2:14	2:18 D						
414	%S5	2:14	2:19 D						
415	%T1	2:20 D	4:12	8:19	16:37				
416	%T2	2:21 D	4:12	8:19	16:37				
417	%T3	2:22 D	4:12	16:37					
420	%T4	2:23 D							
421	%T5	2:24 D							
422	%T6	2:25 D							
423	%W1	2:26 D	4: 5	4:10	4:12	4:24	4:25	4:26	4:27
		4:39	4:40	5:18	5:19	5:25	5:37	5:39	6:26
		6:28	6:29	6:30	8:17	8:36	12:14	12:20	12:21
		15:15	15:23	15:24	15:36	15:37	15:39	15:40	16: 1
		16: 2	16: 3	16: 5	16: 6	16:39	16:40	16:41	16:42
424	%W2	2:27 D	4: 6	5:20	5:22	5:26	5:29	8:17	8:37
		11:26	11:36	11:37	15:16	15:37	15:41	16: 1	16: 9
		16:13	16:17	16:19	17: 2	17: 4			
425	%W3	2:28 D	4: 7	8:17	8:38	12:40	13:11	13:12	15:18
		15:19	15:20	16:10	16:14	16:18	16:20		
426	%W4	2:29 D	13: 2	13: 8					
427	%W5	2:30 D	12:17	14:31					
430	AMAP	2:31 D	4:23	4:25	4:27				
442	BUF	2:41 D	4: 9	5:17	5:23	5:30	5:36	6:24	6:25
		6:26	6:33	7: 7	7: 8	8:17	8:19	8:40	15:24
		15:25	15:26	15:27	15:28	15:30	15:31	15:32	15:33
		15:34	15:35	15:36	15:39	15:40	15:41	15:42	16: 2
		16: 5	16:11	16:17	16:18	16:19	16:20	16:37	16:42
		17: 4	17: 6	17: 7					
445	BUFL	3: 2 D	6:21	6:30	6:39				
444	BUFU	3: 1 D	6:21	6:29	6:38				
1114	CAT	10:16	10:21	11: 9 D					
431	CHAN	2:32 D	4:29	4:32	4:39	5: 6	9:16	13:37	15:17
		15:19	15:22	15:26	16:38	16:40	17: 8		
726	CHINIT	5: 3	9:15 D						
1355	CHNERR	12:20	14:34	14:42 D					
4	CHNOFF	3:27 D	16:42						
1161	CLEAR	9:26	9:35	10:33	11: 4	11:28	12:13 D	12:42	13:32
		14: 1							
1163	CLEARA	12:15 D	12:22						
2	CNVRTLEN	3:30 D	4:12						
453	CON	3: 8 D	9:32	10:39	11:34	13: 8	13:40	14:33	
436	COUNT	2:37 D	4:30	4:37	6: 4	6:40	7: 5		
2265	D3CNVRT	4:10	18:22 D						

460	DCB	3:13 D	6: 6	[illegible]	6:12	6:18	6:25	6:31	6:32
		6:37	6:38	6:39					
457	DPI	3:12 D	4: 9	4:12	4:13	4:14	4:15	4:16	4:17
		4:18	15:35						
443	ERR	2:42 D	9:25	9:27	9:32	9:36	10: 2	10: 3	10:19
		10:23	10:24	10:25	10:34	10:39	10:41	10:42	11:10
		11:29	11:34	11:41	11:42	12:25	13: 1	13: 8	13:15
		13:16	13:33	13:40	14:10	14:11	14:35	14:40	15: 1
		16:37							
1121	FATAL	9:25	9:27	9:36	10: 4	10:34	11: 1	11:13 D	
451	GEN	3: 6 D	10:11	10:14	10:15	13:42			
571	GETBUF	4: 8	5:42	8:16 D	8:39	15:21	16:36		
433	LAST,	2:34 D	4:28	4:32					
14	LB@SZ	3:26 D	15:14	15:37					
450	LEN	3: 5 D	4: 4	5:41	8:17	8:19	8:30	8:32	8:34
		15:14	16:35	16:37					
437	MCB	2:38 D	4:31	5:37	16:21				
447	MOSL	3: 4 D	5:39	6:32	6:33	6:34	8:30	9: 2	
446	MOSU	3: 3 D	5:39	6:31	6:33	6:34	8:30	9: 2	
2027	MSG0	14:10	14:40	18: 3 D					
2051	MSG1	15: 1	18: 5 D						
2072	MSG2	11:41	18: 7 D						
2114	MSG3	10: 2	18: 9 D						
2140	MSG4	11:10	18:11 D						
2166	MSG5	13:15	18:13 D						
2214	MSG6	10:25	14:11	18:15 D					
2241	MSG7	10:41	18:17 D						
44	MSGLEN	3:29 D	16:35						
1730	MSGOUT	10:19	10:23	11:11	11:14	16:34 D			
456	NE	3:11 D	6:15	6:17	6:21	6:28	6:37		
432	NEXT	2:33 D	4:27	4:28	4:33	4:34			
441	NUM	2:40 D	6: 3	6: 4	6: 5	7: 3			
1112	NXTUNIT	10:25	11: 5 D	11:15					
1123	READY	9:24	11:25 D						
1125	READYA	11:27 D	11:38						
1060	RELEASE	10:24	10:32 D	11:12					
1251	RESET	10:22	10:40	13:29 D					
1253	RESETA	13:31 D	14: 7						
627	RESIDENT	4:38	8:29 D						
454	RETRY	3: 9 D	9:22	9:34	9:40	13:30	14: 5	14: 6	
1	RTR$CLR	3:32 D	12:20						
5	RTR$RDY	3:33 D	11:36						

2	RTR$RES	3:35 D	14: 5						
5	RTR$SLC	3:34 D	9:34						
5	RTR$STA	3:36 D	13:11						
736	SELECT	9:23 D	9:41						
1203	SELSTAT	10:18	12:39 D						
1205	SELSTATA	12:41 D	13:13						
1361	SETDCB	5: 2	15:13 D						
455	SIZE	3:10 D	8:33	9: 2					
452	STAT	3: 7 D	10:17	10:20	13: 2	13: 4	13:10		
1021	STATUS	10:12 D							
435	TBLE	2:36 D	4:22	4:39					
1350	TIMEOUT	14:28	14:38 D						
440	TOUT	2:39 D	4:42	16: 6					
434	TYPE	2:35 D	4:34	4:36	16: 8				
461	UNIT	3:14 D	9:17	9:18	9:28	9:32	10:35	10:39	15:29
		15:30	17: 2						
10	UNTOFF	3:28 D	17: 4						
1330	WAITDN	9:30	9:39	10:37	11:32	13: 6	13:38	14:24 D	

```
0     <macro>                    OVERLAY   INDD49                                          INDD49.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ********************************************************************  INDD49.5
                    *                                                                  *  INDD49.6
                    *        INDD49     Initialize DD49 disk channels for use by the   *  INDD49.7
                    *                   current IOP.  Sets up control tables, interrupt *  INDD49.8
                    *                   addresses and runs minimal reliability checks  *  INDD49.9
                    *                   on the channel and device.                     *  INDD49.10
                    *                                                                  *  INDD49.11
                    *        Entry      R!AMAP = AMAP base address                     *  INDD49.12
                    *                                                                  *  INDD49.13
                    ********************************************************************  INDD49.14
                                                                                          INDD49.15
6   <macro>                     REGDEFS    ,(AMAP),(CHAN,NEXT,LAST,TYPE,TBLE,COUNT,MCB,TOUT,NUM,  INDD49.16
                    ,BUF,ERR,BUFU,BUFL,MOSU,MOSL,LEN,GEN,STAT,CON,RETRY,SIZE,NE,DPT,DCB)  INDD49.17
                410 %S1        EQUALS     $REGORG                                          $REG    3
                411 %S2        EQUALS     $REGORG                                          $REG    3
                412 %S3        EQUALS     $REGORG                                          $REG    3
                413 %S4        EQUALS     $REGORG                                          $REG    3
                414 %S5        EQUALS     $REGORG                                          $REG    3
                415 %T1        EQUALS     $REGORG                                          $REG    3
                416 %T2        EQUALS     $REGORG                                          $REG    3
                417 %T3        EQUALS     $REGORG                                          $REG    3
                420 %T4        EQUALS     $REGORG                                          $REG    3
                421 %T5        EQUALS     $REGORG                                          $REG    3
                422 %T6        EQUALS     $REGORG                                          $REG    3
                423 %W1        EQUALS     $REGORG                                          $REG    3
                424 %W2        EQUALS     $REGORG                                          $REG    3
                425 %W3        EQUALS     $REGORG                                          $REG    3
                426 %W4        EQUALS     $REGORG                                          $REG    3
                427 %W5        EQUALS     $REGORG                                          $REG    3
                430 AMAP       EQUALS     $REGORG                                          $REG    3
                431 CHAN       EQUALS     $REGORG                                          $REG    3
                432 NEXT       EQUALS     $REGORG                                          $REG    3
                433 LAST       EQUALS     $REGORG                                          $REG    3
                434 TYPE       EQUALS     $REGORG                                          $REG    3
                435 TBLE       EQUALS     $REGORG                                          $REG    3
                436 COUNT      EQUALS     $REGORG                                          $REG    3
                437 MCB        EQUALS     $REGORG                                          $REG    3
                440 TOUT       EQUALS     $REGORG                                          $REG    3
                441 NUM        EQUALS     $REGORG                                          $REG    3
                442 BUF        EQUALS     $REGORG                                          $REG    3
                443 ERR        EQUALS     $REGORG                                          $REG    3
                444 BUFU       EQUALS     $REGORG                                          $REG    3
```

```
445     BUFL     EQUALS   $REGORG                                          $REG    3
446     MOSU     EQUALS   $REGORG                                          $REG    3
447     MOSL     EQUALS   $REGORG                                          $REG    3
450     LEN      EQUALS   $REGORG                                          $REG    3
451     GEN      EQUALS   $REGORG                                          $REG    3
452     STAT     EQUALS   $REGORG                                          $REG    3
453     CON      EQUALS   $REGORG                                          $REG    3
454     RETRY    EQUALS   $REGORG                                          $REG    3
455     SIZE     EQUALS   $REGORG                                          $REG    3
456     NE       EQUALS   $REGORG                                          $REG    3
457     DPT      EQUALS   $REGORG                                          $REG    3
460     DCB      EQUALS   $REGORG                                          $REG    3
                                                                      INDD49.18
                 EXT      %DCBQ                                        INDD49.19
                 EXT      %DD49                                        INDD49.20
                 EXT      %MEMIO                                       INDD49.21
                 EXT      %MYID                                        INDD49.22
                 EXT      D4TOUT                                       INDD49.23
                 EXT      DCCB                                         INDD49.24
                 EXT      DD49                                         INDD49.25
                 EXT      EITB                                         INDD49.26
                 EXT      TMCB                                         INDD49.27
                                                                      INDD49.28
 22      LB@SZ    EQUALS   LB@LE*LB$DD49      .Local buffer table size  INDD49.29
  4      CHNOFF   EQUALS   4                  .Offset for channel in msg INDD49.30
 40      MSGLEN   EQUALS   D'32               .Msg length               INDD49.31
  4      CNVRTLEN EQUALS   4                  .Conversion table length  INDD49.32
                                                                      INDD49.33
  1      RTR$CLR  EQUALS   1                  .Max retries for channel clear INDD49.34
  5      RTR$RDY  EQUALS   5                  .Max retries for ready status  INDD49.35
  5      RTR$SLC  EQUALS   5                  .Max retries for unit select   INDD49.36
  2      RTR$RES  EQUALS   2                  .Max retries for reset         INDD49.37
  5      RTR$STA  EQUALS   5                  .Max retries for getting status INDD49.38
```

```
                                                                                INDD49.40
                                *       Set up one Device Parameter Table (DPT) to serve all channels.   INDD49.41
                                                                                INDD49.42
  6   010011  024450                    R!LEN = DP@LT                 .Table length                    INDD49.43
 10   014000 /042120  024423            R!%W1 = 'DP'                  .ASCII header                    INDD49.44
 13   014000 /052040  024424            R!%W2 = 'T '                                                   INDD49.45
 16   014000 /032071  024425            R!%W3 = '49'                                                   INDD49.46
 21   072674                            R = GETBUF                    .Get a buffer                    INDD49.47
 22   020442  024457                    R!DPT = R!BUF                 .Save address                    INDD49.48
 24   020000  016000 /002614            R!%W1 = R!%B + D4CNVRT        .Spiral offsets                  INDD49.49
      024423
 30    <macro>                          COPY R!%W1,R!DPT+DP@SPI@P,CNVRTLEN .Copy                       INDD49.50
 47    <macro>                          STORE I@SPIRD4,DP@SPF,R!DPT   .Spiral enable flag              INDD49.51
 57    <macro>                          STORE I@INDXD4,DP@IDX,R!DPT   .Spiral index value              INDD49.52
 67    <macro>                          STORE D4$SPT,DP@DSE,R!DPT     .Default sectors/track           INDD49.53
 77    <macro>                          STORE D4$CPD,DP@DCY,R!DPT     .Default cylinders/disk          INDD49.54
105    <macro>                          STORE D4$SPTA,DP@ASE,R!DPT    .Absolute sectors/track          INDD49.55
115    <macro>                          STORE D4$CPDA,DP@ACY,R!DPT    .Absolute cylinder/disk          INDD49.56
                                                                                INDD49.57
                                *       Set up a Disk Control Block (DCB) for each channel.              INDD49.58
                                                                                INDD49.59
123   014000 /000000  024435            R!TBLE = EITB                 .Interrupt jump table            INDD49.60
126   020430  012006  012005            R!%W1 = R!AMAP + OV@ENT@P + 5 + R!%MYID .Pointer to IOP info   INDD49.61
      022000  024423
133   030423  022430  024423            R!%W1 = (R!%W1) + R!AMAP      .Info for this IOP               INDD49.62
136   010006  025423                    R!%W1 = R!%W1 + 6             .Channel pointer                 INDD49.63
140   030423  022430  024432            R!NEXT = (R!%W1) + R!AMAP     .Channel info                    INDD49.64
143   030432  024433                    R!LAST = (R!NEXT)             .Last channel to search          INDD49.65
145   010006  024431                    R!CHAN = 6                    .First channel to search         INDD49.66
147   010000  024436                    R!COUNT = 0                   .Initialize                      INDD49.67
151   024437                            R!MCB = A                                                      INDD49.68
152    <macro>                          $UNTIL (R!CHAN = R!LAST)      .Until all searched              INDD49.69
155   026432                              R!NEXT = R!NEXT + 1         .Next channel                    INDD49.70
156   030432  011777  024434              R!TYPE = (R!NEXT) & O'777   .Channel type                    INDD49.71
161    <macro>                            $IF (R!TYPE = CH$D4),OR,(R!TYPE = CH$D4S) .If DD49           INDD49.72
                                                                                .>>>>> 0          $IF      1
170   026436                                R!COUNT = R!COUNT + 1     .Count                           INDD49.73
171   020000  112561                        R = RESIDENT, R!%DD49 = 0 .Read in interrupt overlay       INDD49.74
173   020435  022431  024423                R!%W1 = R!TBLE + R!CHAN   .Interrupt table pointer         INDD49.75
176   020000  016000 /000000                (R!%W1) = R!%DD49 + DD49  .Set interrupt entrance address  INDD49.76
      034423
202   077000 /001715                        R = SETDCB                .Set up the DCB                  INDD49.77
```

```
204     <macro>                         $ENDIF                                                          INDD49.78
                                                                                  .<<<<< 0                $ENDIF    1
204     026431                          R!CHAN   R!CHAN + 1          .Increment                         INDD49.79
205     <macro>                         $ENDIF                                                          INDD49.80
                                                                                                        INDD49.81
                                *       Set up the read-ahead/write-behind areas for all channels.      INDD49.82
                                *       Always initialize pointers in the DCB for the MOS buffer        INDD49.83
                                *       area.  Additionally, if a high speed path to a particular       INDD49.84
                                *       memory exists only on some other IOP, set up pointers in        INDD49.85
                                *       the MCB (in MOS) so that the other IOP may have access          INDD49.86
                                                                                                        INDD49.87

                                                                                                        INDD49.96
                                IF (A # 0)                   .If there is a backdoor                    INDD49.97

                                                                                                        INDD49.100
                                                                                                        INDD49.101
                                SPUNIT (A # 0),CODE PISNOMOS .Not enough MOS                            INDD49.105
                                R!LEN    CB@SZ/@P            .Local buffer size                         INDD49.106
                                R    GETBUF                  .Get working storage                       INDD49.107
                                $ENDIF                                                                  INDD49.108
                                                                                  .<<<<< 0                $ENDIF    1
313     010000  024441          R!NUM = 0                                                               INDD49.109
```

```
315    <macro>                          $UNTIL (R!COUNT = 0),OR,(R!NUM = O'20) .Scan all DCB's          INDD49.110
322    020441  016000 /000000             R!DCB = (R!NUM + DCCB)       .Next                          INDD49.111
       024410  030410  024460
330    <macro>                            $IF (A # 0)                  .If something configured       INDD49.112
                                                                                        .>>>>> 0          $IF      1
332    <macro>                              GET A,DK@TYP,R!DCB         .Device type                   INDD49.113
340    <macro>                              $IF (A = DSK$DD49)         .If DD49                       INDD49.114
                                                                                        .>>>>> 1          $IF      1
343    <macro>                                GET A,DK@SEL,R!DCB       .Flag indicating striped disk  INDD49.115
350    <macro>                                $IF (A # 0)              .If striped                    INDD49.116
                                                                                        .>>>>> 2          $IF      1
352    010053  024456                           R!NE = MB$DD49S        .Number of buffers             INDD49.117
354    <macro>                                  STORE 0,DK@SEL,R!DCB   .Clear flag                    INDD49.118
365    <macro>                                $ELSE                                                   INDD49.119
366    010053  024456                           R!NE = MB$DD49         .Normal device                 INDD49.120
370    <macro>                                $ENDIF                                                  INDD49.121
                                                                                        .<<<<< 2          $ENDIF   1
370    <macro>                                MGET R!BUFU,R!BUFL,NUM=R!NE .Allocate buffers           INDD49.122
416    <macro>                                $PUNTIF (A # 0),CODE=PT$NOMOS .Not enough MOS           INDD49.123
420    <macro>                                $IF (R!BUF # 0)          .If MCB is needed              INDD49.124
                                                                                        .>>>>> 2          $IF      1
423    <macro>                                  PUT R!DCB,DD@DCB,R!BUF .Save DCB address              INDD49.125
430    010002  005002  022442                   R!%W1 = CB@LH < 2 + R!BUF .Entry base                 INDD49.126
       024423
434    <macro>                                  PUT R!NE,CB@NUM,R!%W1 .Save number of buffers         INDD49.127
441    <macro>                                  PUT R!BUFU,CB@BM0,R!%W1 .Base address of buffers      INDD49.128
446    <macro>                                  PUT R!BUFL,CB@BM1,R!%W1 .Lower                        INDD49.129
453    <macro>                                  PUT R!MOSU,DK@MC0,R!DCB .MCB address                  INDD49.130
460    <macro>                                  PUT R!MOSL,DK@MC1,R!DCB                               INDD49.131
465    <macro>                                  MOSW R!MOSU,R!MOSL,R!BUF,CB@SZ .Write to MOS          INDD49.132
505    <macro>                                  DADD R!MOSU,R!MOSL,CB@SZ .Next MCB                    INDD49.133
511    <macro>                                $ENDIF                                                  INDD49.134
                                                                                        .<<<<< 2          $ENDIF   1
511    <macro>                                PUT R!NE,DK@CNT,R!DCB    .Number of buffers            INDD49.135
524    <macro>                                PUT R!BUFU,DK@BF0,R!DCB .Base address of buffers       INDD49.136
531    <macro>                                PUT R!BUFL,DK@BF1,R!DCB                                INDD49.137
536    027436                                 R!COUNT = R!COUNT - 1    .Decrement channel count      INDD49.138
537    <macro>                              $ENDIF                                                    INDD49.139
                                                                                        .<<<<< 1          $ENDIF   1
537    <macro>                            $ENDIF                                                      INDD49.140
                                                                                        .<<<<< 0          $ENDIF   1
537    026441                             R!NUM = R!NUM + 1            .Next DCB                      INDD49.141
```

```
540    <macro>                    $ENDTIL                                                    INDD49.142
541    <macro>                    $PUNTIF (R!COUNT # 0),CODE=PT$DD49 .If something amiss     INDD49.143
544    <macro>                    $IF (R!BUF # 0)                .If MCB needed              INDD49.144
                                                                                .>>>>> 0          $IF      1
547    <macro>                      RELMEM R!BUF                 .Release local working storage INDD49.145
557    <macro>                    $ENDIF                                                     INDD49.146
                                                                                .<<<<< 0          $ENDIF   1
                                                                                               INDD49.147
                            *      Initialize each channel.                                    INDD49.148
                                                                                               INDD49.149
557   010000  024441              R!NUM = 0                                                   INDD49.150
561    <macro>                    $UNTIL (R!NUM = O'20)          .Scan all channels          INDD49.151
564   020441  016000 /000000        R!DCB = (R!NUM + DCCB)       .DCB pointer                INDD49.152
      024410  030410  024460
572    <macro>                      $IF (A # 0)                  .If something configured    INDD49.153
                                                                                .>>>>> 0          $IF      1
574    <macro>                        GET A,DK@TYP,R!DCB         .Device type                INDD49.154
602    <macro>                        $IF (A = DSK$DD49)         .If DD49                    INDD49.155
                                                                                .>>>>> 1          $IF      1
605    <macro>                          GET R!CHAN,DK@CHN,R!DCB .Channel                     INDD49.156
613   077000 /001056                    R = CHINIT               .Start initialization       INDD49.157
615    <macro>                        $ENDIF                                                 INDD49.158
                                                                                .<<<<< 1          $ENDIF   1
615    <macro>                      $ENDIF                                                   INDD49.159
                                                                                .<<<<< 0          $ENDIF   1
615   026441                        R!NUM = R!NUM + 1            .Next channel               INDD49.160
616    <macro>                    $ENDTIL                                                    INDD49.161
617    <macro>                    PAUSE D'40                     .Wait for recalibration     INDD49.162
630   010000  024441              R!NUM = 0                                                   INDD49.163
632    <macro>                    $UNTIL (R!NUM = O'20)          .Scan all channels          INDD49.164
635   020441  016000 /000000        R!DCB = (R!NUM + DCCB)       .DCB pointer                INDD49.165
      024410  030410  024460
643    <macro>                      $IF (A # 0)                  .If something configured    INDD49.166
                                                                                .>>>>> 0          $IF      1
645    <macro>                        GET A,DK@TYP,R!DCB         .Device type                INDD49.167
653    <macro>                        $IF (A = DSK$DD49)         .If DD49                    INDD49.168
                                                                                .>>>>> 1          $IF      1
656    <macro>                          GET A,DK@SEL,R!DCB       .Flag indicating error      INDD49.169
663    <macro>                          $IF (A = 0)              .If no error                INDD49.170
                                                                                .>>>>> 2          $IF      1
665    <macro>                            GET R!CHAN,DK@CHN,R!DCB .Channel                   INDD49.171
673   077000 /001257                      R = CHEND              .End initialization         INDD49.172
```

```
675    <macro>                    $ENDIF                                                    INDD49.173
                                                                        .<<<<< 2                $ENDIF    1
675    <macro>                    STORE 1,DK@SEL,R!DCB     .Set released                  R13739AH.1
705    <macro>                  $ENDIF                                                      INDD49.175
                                                                        .<<<<< 1                $ENDIF    1
705    <macro>                $ENDIF                                                        INDD49.176
                                                                        .<<<<< 0                $ENDIF    1
705   026441                  R!NUM = R!NUM + 1         .Next channel                       INDD49.177
706    <macro>              $ENDTIL                                                         INDD49.178
707    <macro>              RETURN                                                          INDD49.179
```

```
                                        ***************************  ****************************************  INDD49.181
                                        *        SUBROUTINES                                            *  INDD49.182
                                        ************************************************************************  INDD49.183
                                                                                                              INDD49.184
                                        ************************************************************************  INDD49.185
                                        *                                                                *  INDD49.186
                                        *        GETBUF:   Allocate local memory buffer and clear to zeros.  *  INDD49.187
                                        *                                                                *  INDD49.188
                                        *        Entry:    R!LEN = Size needed (parcels)                 *  INDD49.189
                                        *                  R!%W1 = ID to store in header                 *  INDD49.190
                                        *                  R!%W2 = "                   "                 *  INDD49.191
                                        *                  R!%W3 = "                   "                 *  INDD49.192
                                        *                                                                *  INDD49.193
                                        ************************************************************************  INDD49.194
                                                                                                              INDD49.195
 715                                    GETBUF   *                                                          INDD49.196
 715    <macro>                                  GETMEM R!LEN,R!BUF,R!%W1,R!%W2,R!%W3 .Allocate memory       INDD49.197
 735    <macro>                                  $PUNTIF (A # 0),CODE=PT$LOCAL .If not available            INDD49.198
 737    <macro>                                  CLEAR START=R!BUF,COUNT=R!LEN .Clear                       INDD49.199
 752    001000                                   EXIT                                                       INDD49.200
                                                                                                              INDD49.201
                                        ************************************************************************  INDD49.202
                                        *                                                                *  INDD49.203
                                        *        RESIDENT: Establish resident code for DD49 interrupt    *  INDD49.204
                                        *                  handler.                                      *  INDD49.205
                                        *                                                                *  INDD49.206
                                        ************************************************************************  INDD49.207
                                                                                                              INDD49.208
 753                                    RESIDENT *                                                          INDD49.209
 753    <macro>                                  FIND DD49,R!MOSU,R!MOSL,R!LEN .Locate overlay              INDD49.210
 772    <macro>                                  $PUNTIF (A # 0),CODE=PT$NOLAY .Overlay not found           INDD49.211
 774    020450  005002  024450                   R!LEN = R!LEN < 2             .Size in parcels             INDD49.212
 777    024455                                   R!SIZE = A                    .Save overlay size           INDD49.213
1000    020450  012012  012006                   R!LEN = R!LEN + MEM@LT + DCB@LT .Add MEMIO and DCB queues INDD49.214
        024450
1004    014000 /042103  024423                   R!%W1 = 'DC'                  .ASCII header                INDD49.215
1007    014000 /041121  024424                   R!%W2 = 'BQ'                                               INDD49.216
1012    010000  024425                           R!%W3 = 0                                                  INDD49.217
1014    073077                                   R = GETBUF                    .Get local memory            INDD49.218
1015    020442  024000                           R!%DCBQ = R!BUF               .DCB queue                   INDD49.219
1017    020000  012006  024000                   R!%MEMIO = R!%DCBQ + DCB@LT   .MEMIO request queues        INDD49.220
1022    020000  012012  012003                   R!%DD49 = R!%MEMIO + MEM@LT + 3 & O'177774 .Word boundary  INDD49.221
```

```
        015000 /177774  024000
1030     <macro>                          MOSR R!MOSU,R!MOSL,R!%DD49,R!SIZE>2,NOWAIT .Read in          INDD49.222
1051    020000  016000 /000000            R!TOUT = R!%DD49 + D4TOUT      .Timout routine               INDD49.223
        024440
1055    001000                            EXIT                                                         INDD49.224
                                                                                                       INDD49.225
                                 ******************************************************************** INDD49.226
                                 *                                                                  * INDD49.227
                                 *        CHINIT:    Initialize the channel and perform minimal     * INDD49.228
                                 *                   reliability tests on the device.               * INDD49.229
                                 *                                                                  * INDD49.230
                                 *        Entry:     R!CHAN = Channel number                        * INDD49.231
                                 *                                                                  * INDD49.232
                                 ******************************************************************** INDD49.233
                                                                                                       INDD49.234
1056                             CHINIT   *                                                           INDD49.235
1056    020431  054000                    B = R!CHAN                                                   INDD49.236
                                                                                                       INDD49.237
                                 *        Select the drive.                                            INDD49.238
                                                                                                       INDD49.239
1060    010000  024454                    R!RETRY = 0                                                  INDD49.240
1062                             SELECT   *                                                           INDD49.241
1062    072332                            R = READY                      .Make sure channel is ready   INDD49.242
1063    020443  127000 /002260            P = MSGOUT, R!ERR # 0          .If error                     INDD49.243
1066    072364                            R = CLEAR                      .Clear the channel            INDD49.244
1067    020443  127000 /002260            P = MSGOUT, R!ERR # 0          .If error                     INDD49.245
1072    014000 /012524                    A = O'12524                                                  INDD49.246
1074    161000                            IOB: 1                         .Select unit 0                INDD49.247
1075    072567                            R = WAITDN                     .Wait for DN                  INDD49.248
1076     <macro>                          $IF (R!ERR # 0),OR,(R!CON # O'52003) .If error              INDD49.249
                                                                                   .>>>>> 0              $IF      1
1105     <macro>                            $IF (R!RETRY < RTR$SLC)      .If more retries              INDD49.250
                                                                                   .>>>>> 1              $IF      1
1111    026454                               R!RETRY = R!RETRY + 1      .Count up                     INDD49.251
1112    071030                               P = SELECT                 .Try again                    INDD49.252
1113     <macro>                            $ENDIF                                                     INDD49.253
                                                                                   .<<<<< 1              $ENDIF   1
1113    020443  103004  014000               R!ERR = MSG3, R!ERR = 0    .Error code                   INDD49.254
       /002427  024443
1120    075000 /002260                       P = MSGOUT                 .Tell the world               INDD49.255
1122     <macro>                          $ENDIF                                                     INDD49.256
                                                                                   .<<<<< 0              $ENDIF   1
```

```
                                                                                                INDD49.25.
                                 *        Analyze general status.  Reset any errors.            INDD49.258
                                                                                                INDD49.259
1122  010000  024454                      R!RETRY = 0                                           INDD49.260
1124  173000                               IOB: 13                    .Get status register 1     INDD49.261
1125  024451                               R!GEN = A                  .Save (general status)     INDD49.262
1126                             STATUS    *                                                     INDD49.263
1126   <macro>                             $IF (R!GEN # 0)            .If error                  INDD49.264
                                                                                   .>>>>> 0          $IF      1
1131   <macro>                               $IF (0 # R!GEN & O'20000)  .If seq op in progress   INDD49.265
                                                                                   .>>>>> 1          $IF      1
1141  020454  013074  101237                 P = SEQOP, R!RETRY >= D'60  .If out of retries     INDD49.266
1144  026454                                 R!RETRY = R!RETRY + 1      .Count up               INDD49.267
1145  072305                                 R = CLEAR                  .Clear channel          INDD49.268
1146  020443  127000 /002260                P = MSGOUT, R!ERR # 0      .If error               INDD49.269
1151   <macro>                               PAUSE D'10                 .Delay one second       INDD49.270
1162  020431  054000                         B = R!CHAN                 .Set B                  INDD49.271
1164  072310                                 R = GENERAL                .Obtain general status  INDD49.272
1165  020443  127000 /002260                P = MSGOUT, R!ERR # 0      .If error               INDD49.273
1170  026454                                 R!RETRY = R!RETRY + 1      .Count up               INDD49.274
1171  071043                                 P = STATUS                 .Test again             INDD49.275
1172   <macro>                               $ELSEIF (0 # R!GEN & O'40000) .If drive fault      INDD49.276
1203  010007  024452                         R!STAT = 7                                         INDD49.277
1205  072330                                 R = SELSTAT                .Get selected status 7  INDD49.278
1206  020443  127000 /002260                P = MSGOUT, R!ERR # 0      .If error               INDD49.279
1211  020452  015000 /174711                P = CAT, O'140111 # R!STAT & O'174711 .Catastrophic error  INDD49.280
      024410  014000 /140111
      023410  103166
1221  010010  024452                         R!STAT = D'8                                       INDD49.281
1223  072312                                 R = SELSTAT                .Get selected status 8  INDD49.282
1224  020443  127000 /002260                P = MSGOUT, R!ERR # 0      .If error               INDD49.283
1227  020452  103156                         P = CAT, R!STAT # 0        .Catastrophic error     INDD49.284
1231   <macro>                               $ENDIF                                             INDD49.285
                                                                                   .<<<<< 1          $ENDIF   1
1231  072352                                 R = RESET                  .Reset the drive        INDD49.286
1232   <macro>                               $IF (R!ERR # 0)            .If error               INDD49.287
                                                                                   .>>>>> 1          $IF      1
1235  077000 /002260                         R = MSGOUT                 .Broadcast error        INDD49.288
1237  020443  017000 /002532                EXIT, R!ERR # MSG7         .If don't want to release INDD49.289
      102002  001000
1244  070105                                 P = RELEASE                                        INDD49.290
1245   <macro>                               $ENDIF                                             INDD49.291
```

```
                                                                                .<<<<< 1          $ENDIF   1
1245    <macro>                         $ENDIF                                                  INDD49.292
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                INDD49.293
                                *       Recalibrate the drive.  Issue the command, then exit    INDD49.294
                                *       to wait for completion.  Cleanup after this command     INDD49.295
                                *       is performed in CHEND.                                  INDD49.296
                                                                                                INDD49.297
1245    072205                          R = CLEAR                   .Clear BZ and DN            INDD49.298
1246    020443  127000 /002260          P = MSGOUT, R!ERR # 0       .If error                   INDD49.299
1251    014000 /004003                  A = O'4003                  .Servo calibration          INDD49.300
1253    016000 /110000                  A = A + O'110000            .Diagnostic select command  INDD49.301
1255    161000                          IOB: 1                      .Issue                      INDD49.302
1256    001000                          EXIT                                                    INDD49.303
                                                                                                INDD49.304
                                ***********************************************************     INDD49.305
                                *                                                               INDD49.306
                                *       CHEND:    Finish up channel initialization following drive *  INDD49.307
                                *                 recalibration.                                *  INDD49.308
                                *                                                               *  INDD49.309
                                *       Entry:    R!CHAN = Channel number                       *  INDD49.310
                                *                                                               *  INDD49.311
                                ***********************************************************     INDD49.312
                                                                                                INDD49.313
1257                            CHEND   *                                                       INDD49.314
1257    020431  054000                  B = R!CHAN                  .Channel                    INDD49.315
1261    072403                          R = WAITDN                  .Check for DN               INDD49.316
1262    <macro>                         $IF (R!ERR # 0),OR,(R!CON # O'1401) .If error           INDD49.317
                                                                                .>>>>> 0          $IF      1
1271    020443  103004  014000            R!ERR = MSG9, R!ERR = 0   .Set error code             INDD49.318
       /002573  024443
1276    077000 /002260                    R = MSGOUT                .Display msg                INDD49.319
1300    072303                            R = RESET                 .Reset the drive            INDD49.320
1301    <macro>                           $IF (R!ERR # 0)           .If error                   INDD49.321
                                                                                .>>>>> 1          $IF      1
1304    077000 /002260                      R = MSGOUT              .Broadcast error            INDD49.322
1306    020443  017000 /002532              EXIT, R!ERR # MSG7      .If don't want to release   INDD49.323
        102002  001000
1313    <macro>                           $ENDIF                                                INDD49.324
                                                                                .<<<<< 1          $ENDIF   1
1313    070036                            P = RELEASE                                           INDD49.325
1314    <macro>                         $ENDIF                                                  INDD49.326
```

```
                                                                                .<<<<< 0          $EN     1
1314   010017  024452                R!STAT = D'15                 .Selected status 15          INDD49.327
1316   072217                        R = SELSTAT                   .Get it                      INDD49.328
1317   020443  127000 /002260        P = MSGOUT, R!ERR # 0         .If error                    INDD49.329
1322    <macro>                      $IF (3 # R!STAT > D'8)        .Unsuccessful calibration    INDD49.330
                                                                                .>>>>> 0          $IF     1
1331   014000 /002573  024443         R!ERR = MSG9                                               INDD49.331
1334   077000 /002260                 R = MSGOUT                    .Display msg                 INDD49.332
1336   072245                         R = RESET                     .Reset the drive             INDD49.333
1337    <macro>                       $IF (R!ERR # 0)               .If error                    INDD49.334
                                                                                .>>>>> 1          $IF     1
1342   077000 /002260                   R = MSGOUT                  .Broadcast error             INDD49.335
1344   020443  017000 /002532           EXIT, R!ERR # MSG7          .If don't want to release    INDD49.336
       102002  001000
1351    <macro>                       $ENDIF                                                     INDD49.337
                                                                                .<<<<< 1          $ENDIF  1
1351    <macro>                      $ENDIF                                                      INDD49.338
                                                                                .<<<<< 0          $ENDIF  1
                                                                                                   INDD49.339
                            *        Release the drive.                                          INDD49.340
                                                                                                   INDD49.341
1351                        RELEASE  *                                                           INDD49.342
1351   072101                        R = CLEAR                     .Clear the channel           INDD49.343
1352   020443  127000 /002260        P = MSGOUT, R!ERR # 0         .If error                    INDD49.344
1355   014000 /172524                A = O'172524                                               INDD49.345
1357   161000                        IOB: 1                        .Release unit                INDD49.346
1360   072304                        R = WAITDN                    .Wait for DN                 INDD49.347
1361    <macro>                      $IF (R!ERR # 0),OR,(R!CON # O'52003) .If error            INDD49.348
                                                                                .>>>>> 0          $IF     1
1370   072213                         R = RESET                     .Reset the drive             INDD49.349
1371   020443  103004  014000         R!ERR = MSG8, R!ERR = 0       .Error code                  INDD49.350
      /002553  024443
1376   075000 /002260                 P = MSGOUT                    .Tell the world              INDD49.351
1400    <macro>                      $ENDIF                                                      INDD49.352
                                                                                .<<<<< 0          $ENDIF  1
1400   072052                        R = CLEAR                     .Clear channel               INDD49.353
1401   001000                        EXIT                                                       INDD49.354
                                                                                                   INDD49.355
1402                        SEQOP    *                                                           INDD49.356
1402   014000 /002447  024443        R!ERR = MSG4                  .Error code                  INDD49.357
1405   070004                        P = ERRMSG                    .Tell the world              INDD49.358
1406                        CAT      *                                                           INDD49.359
```

```
1406   014000 /002466  024443          R!ERR = MSG5                    .Error code                          INDD49.360
1411                           ERRMSG  *                                                                    INDD49.361
1411   077000 /002260                  R = MSGOUT                      .Tell the world                      INDD49.362
1413   071042                          P = RELEASE                     .Release the drive                   INDD49.363
                                                                                                            INDD49.364
                               ***********************************************************************  INDD49.365
                               *                                                                     *  INDD49.366
                               *       READY:    Get channel ready status.                           *  INDD49.367
                               *                                                                     *  INDD49.368
                               *       Exit:     R!ERR = error code                                  *  INDD49.369
                               *                                                                     *  INDD49.370
                               ***********************************************************************  INDD49.371
                                                                                                            INDD49.372
1414                           READY   *                                                                    INDD49.373
1414   010000  024424                  R!%W2 = 0                                                            INDD49.374
1416                           READYA  *                                                                    INDD49.375
1416   072034                          R = CLEAR                       .Clear the channel                   INDD49.376
1417   020443  102002  001000          EXIT, R!ERR # 0                 .If error                            INDD49.377
1422   010000                          A = 0                                                                INDD49.378
1423   177000                          IOB: 17                         .Get ready status                    INDD49.379
1424   072240                          R = WAITDN                      .Wait for DN                         INDD49.380
1425    <macro>                        $IF (R!ERR # 0),OR,(0 = R!CON & 1) .If not good                      INDD49.381
                                                                                       .>>>>> 0                $IF      1
1436    <macro>                          $IF (R!%W2 < RTR$RDY)         .If more retries                   INDD49.382
                                                                                       .>>>>> 1                $IF      1
1442   026424                             R!%W2 = R!%W2 + 1            .Count up                          INDD49.383
1443   071025                             P = READYA                   .Try again                         INDD49.384
1444    <macro>                          $ENDIF                                                             INDD49.385
                                                                                       .<<<<< 1                $ENDIF   1
1444   020443  103004  014000            R!ERR = MSG2, R!ERR = 0      .Error code                        INDD49.386
      /002411  024443
1451    <macro>                        $ENDIF                                                             INDD49.387
                                                                                       .<<<<< 0                $ENDIF   1
1451   001000                          EXIT                                                               INDD49.388
                                                                                                            INDD49.389
                               ***********************************************************************  INDD49.390
                               *                                                                     *  INDD49.391
                               *       CLEAR:    Clear the DD49 disk channel.                        *  INDD49.392
                               *                                                                     *  INDD49.393
                               *       Exit:     R!ERR = error code                                  *  INDD49.394
                               *                                                                     *  INDD49.395
                               ***********************************************************************  INDD49.396
```

```
                                                                                                INDD49.397
1452                            CLEAR     *                                                     INDD49.398
1452   010000  024423                     R!%W1 = 0                                             INDD49.399
1454                            CLEARA    *                                                     INDD49.400
1454   160000                             IOB: 0                        .Clear channel          INDD49.401
1455   010002  025427                     R!%W5 = R!%W5 + 2             .Delay 6 C.P.'s         INDD49.402
1457    <macro>                           $IF (IOB = DN),OR,(IOB = BZ)  .If not cleared         INDD49.403
                                                                                  .>>>>> 0          $IF      1
1464   020423  013001  101223               P = CHNERR, R!%W1 >= RTR$CLR .Channel error         INDD49.404
1467   026423                               R!%W1 = R!%W1 + 1            .Count up              INDD49.405
1470   071014                               P = CLEARA                   .Try again             INDD49.406
1471    <macro>                           $ENDIF                                                INDD49.407
                                                                                  .<<<<< 0          $ENDIF   1
1471   010000  024443                     R!ERR = 0                                             INDD49.408
1473   001000                             EXIT                                                  INDD49.409
                                                                                                INDD49.410
                                ***********************************************************     INDD49.411
                                *                                                         *     INDD49.412
                                *          GENERAL:  Get general status.                  *     INDD49.413
                                *                                                         *     INDD49.414
                                *          Exit:     R!ERR = error code                   *     INDD49.415
                                *                    R!GEN = general status, if obtainable *     INDD49.416
                                *                                                         *     INDD49.417
                                ***********************************************************     INDD49.418
                                                                                                INDD49.419
1474                            GENERAL   *                                                     INDD49.420
1474   010000  024425                     R!%W3 = 0                                             INDD49.421
1476                            GENERALA  *                                                     INDD49.422
1476   073024                             R = CLEAR                     .Clear channel          INDD49.423
1477   020443  102002  001000             EXIT, R!ERR # 0               .If error               INDD49.424
1502   014000 /100000                     A = O'100000                                          INDD49.425
1504   161000                             IOB: 1                        .Get general status     INDD49.426
1505   072157                             R = WAITDN                    .Wait for DN            INDD49.427
1506    <macro>                           $IF (R!ERR = 0),AND,(R!CON = 3) .If OK                INDD49.428
                                                                                  .>>>>> 0          $IF      1
1515   173000                               IOB: 13                     .Read in                INDD49.429
1516   024451                               R!GEN = A                   .Save                   INDD49.430
1517    <macro>                           $ELSEIF (R!%W3 < RTR$STA)     .If more retries        INDD49.431
1524   026425                               R!%W3 = R!%W3 + 1           .Count up               INDD49.432
1525   071027                               P = GENERALA                .Try again              INDD49.433
1526    <macro>                           $ELSE                                                 INDD49.434
1527   020443  103004  014000               R!ERR = MSG6, R!ERR = 0     .If error code not set yet INDD49.435
```

```
        /002510  024443
1534     <macro>                        $ENDIF                                                    INDD49.436
                                                                                  .<<<<< 0              $ENDIF    1
1534    001000                          EXIT                                                      INDD49.437
                                                                                                  INDD49.438
                                 ******************************************************************INDD49.439
                                 *                                                                *INDD49.440
                                 *       SELSTAT:  Get selected status.                           *INDD49.441
                                 *                                                                *INDD49.442
                                 *       Entry:    R!STAT = status number desired                 *INDD49.443
                                 *                                                                *INDD49.444
                                 *       Exit:     R!ERR = error code                             *INDD49.445
                                 *                 R!STAT = selected status, if obtainable        *INDD49.446
                                 *                                                                *INDD49.447
                                 ******************************************************************INDD49.448
                                                                                                  INDD49.449
1535                             SELSTAT  *                                                       INDD49.450
1535    010000  024425                    R!%W3 = 0                                               INDD49.451
1537                             SELSTATA *                                                       INDD49.452
1537    073065                            R - CLEAR                    .Clear channel             INDD49.453
1540    020443  102002  001000            EXIT, R!ERR # 0              .If error                  INDD49.454
1543    020452  005010  012003            R!%W4 = R!STAT < D'8 + 3     .Save for channel echo     INDD49.455
        024426
1547    020452  016000 /070000            A = R!STAT + O'70000                                    INDD49.456
1552    161000                            IOB: 1                       .Get selected status       INDD49.457
1553    072111                            R = WAITDN                   .Wait for DN               INDD49.458
1554     <macro>                          $IF (R!ERR = 0),AND,(R!CON = R!%W4) .If OK              INDD49.459
                                                                                  .>>>>> 0              $IF       1
1563    173000                              IOB: 13                    .Read in                   INDD49.460
1564    024452                              R!STAT = A                 .Save                      INDD49.461
1565     <macro>                          $ELSEIF (R!%W3 < RTR$STA)    .If more retries           INDD49.462
1572    026425                              R!%W3 = R!%W3 + 1          .Count up                  INDD49.463
1573    071034                              P = SELSTATA               .Try again                 INDD49.464
1574     <macro>                          $ELSE                                                   INDD49.465
1575    020443  103004  014000             R!ERR = MSG6, R!ERR = 0    .If error code not set yet INDD49.466
        /002510  024443
1602     <macro>                          $ENDIF                                                  INDD49.467
                                                                                  .<<<<< 0              $ENDIF    1
1602    001000                          EXIT                                                      INDD49.468
                                                                                                  INDD49.469
                                 ******************************************************************INDD49.470
                                 *                                                                *INDD49.471
```

```
                                *        RESET:    Reset tl    49 disk.                                  *   INDD49.472
                                *                                                                      *   INDD49.473
                                *        Exit:     R!ERR = error code                                    *   INDD49.474
                                *                                                                      *   INDD49.475
                                ************************************************************************   INDD49.476
                                                                                                           INDD49.477
1603                            RESET    *                                                                 INDD49.478
1603   010000  024454                    R!RETRY = 0                                                       INDD49.479
1605                            RESETA   *                                                                 INDD49.480
1605   073133                            R = CLEAR                        .Clear channel                   INDD49.481
1606   020443  102002  001000            EXIT, R!ERR # 0                  .If error                        INDD49.482
1611   014000 /130000                    A = O'130000                                                      INDD49.483
1613   161000                            IOB: 1                           .Reset                           INDD49.484
1614    <macro>                          PAUSE D'20                       .Wait for completion             INDD49.485
1625   020431  054000                    B = R!CHAN                       .Set B                           INDD49.486
1627   072035                            R = WAITDN                       .Check for DN                    INDD49.487
1630    <macro>                          $IF (R!ERR = MSG1),AND,(R!CON = O'203) .If OK                     INDD49.488
                                                                                        .>>>>> 0                $IF      1
1641   173000                              IOB: 13                        .Get general status              INDD49.489
1642   024451                              R!GEN = A                      .Save                            INDD49.490
1643   106171                              P = CLEAR, A = 0               .If OK                           INDD49.491
1644    <macro>                          $ENDIF                                                            INDD49.492
                                                                                        .<<<<< 0                $ENDIF   1
1644    <macro>                          $IF (R!RETRY < RTR$RES)          .If more retries                 INDD49.493
                                                                                        .>>>>> 0                $IF      1
1650   026454                              R!RETRY = R!RETRY + 1          .Count up                        INDD49.494
1651   071044                              P = RESETA                     .Try again                       INDD49.495
1652    <macro>                          $ENDIF                                                            INDD49.496
                                                                                        .<<<<< 0                $ENDIF   1
1652   020443  102036                    P = CHNERR, R!ERR = 0            .If channel error                INDD49.497
1654   020443  017000 /002356            R!ERR = MSG7, R!ERR # MSG0       .If not timeout                  INDD49.498
       102004  014000 /002532
       024443
1663   001000                            EXIT                                                              INDD49.499
                                                                                                           INDD49.500
                                ************************************************************************   INDD49.501
                                *                                                                      *   INDD49.502
                                *        WAITDN:   Wait for DN flag on DD49 disk channel.              *   INDD49.503
                                *                                                                      *   INDD49.504
                                *        Exit:     R!ERR = error code                                    *   INDD49.505
                                *                  R!CON = Status register 0 (controller status)         *   INDD49.506
                                *                                                                      *   INDD49.507
```

```
                                 ***********************************************************************  INDD49.508
                                                                                                         INDD49.509
1664                             WAITDN     *                                                            INDD49.510
1664   010000                             A = 0                          .Initialize                      INDD49.511
1665    <macro>                           $UNTIL (IOB = DN)              .Until DN sets                   INDD49.512
1667   013001                               A = A - 1                                                     INDD49.513
1670   102014                               P = TIMEOUT, A = 0           .If timeout                      INDD49.514
1671    <macro>                           $ENDTIL                                                        INDD49.515
1672   166000                             IOB: 6                         .Clear interrupt                 INDD49.516
1673   010002  025427                     R!%W5 = R!%W5 + 2              .Delay                           INDD49.517
1675   172000                             IOB: 12                        .Status register 0               INDD49.518
1676   024453                             R!CON = A                      .Save                            INDD49.519
1677   043000  101011                     P = CHNERR, IOB = BZ           .If channel error                INDD49.520
1701   010000  024443                     R!ERR = 0                                                       INDD49.521
1703   001000                             EXIT                                                            INDD49.522
                                                                                                         INDD49.523
1704                             TIMEOUT    *                                                            INDD49.524
1704   166000                             IOB: 6                         .Clear interrupt                 INDD49.525
1705   014000 /002356  024443             R!ERR = MSG0                   .Error code                      INDD49.526
1710   001000                             EXIT                                                            INDD49.527
1711                             CHNERR     *                                                            INDD49.528
1711   014000 /002374  024443             R!ERR = MSG1                   .Error code                      INDD49.529
1714   001000                             EXIT                                                            INDD49.530
                                                                                                         INDD49.531
                                 ***********************************************************************  INDD49.532
                                 *                                                                     *  INDD49.533
                                 *         SETDCB:    Set up the Disk Control Block (DCB) for this       *  INDD49.534
                                 *                    channel.                                           *  INDD49.535
                                 *                                                                       *  INDD49.536
                                 *         Entry:     R!CHAN = Channel number                            *  INDD49.537
                                 *                                                                       *  INDD49.538
                                 ***********************************************************************  INDD49.539
                                                                                                         INDD49.540
1715                             SETDCB     *                                                            INDD49.541
1715   010066  012022  024450             R!LEN = DK@LT + LB@SZ          .Size needed                     INDD49.542
1720   014000 /042113  024423             R!%W1 = 'DK'                   .Table name                      INDD49.543
1723   014000 /040040  024424             R!%W2 = '@ '                                                    INDD49.544
1726   020431  004003  012060             R!%W3 = R!CHAN > 3 + '0'R < D'8 .Convert ch.# to ASCII          INDD49.545
       005010  024425
1733   020431  011007  012060             R!%W3 = R!CHAN & 7 + '0'R + R!%W3                               INDD49.546
       022425  024425
1740   077000 /000715                     R = GETBUF                     .Allocate local memory           INDD49.547
```

```
1742   020431  013020  016000          R!%W1 = R!CHAN - O      DCCB  .Channel table cell                INDD49.548
      /000000  024423
1747   020442  034423                  (R!%W1) = R!BUF               .Save pointer to DCB               INDD49.549
1751    <macro>                        STORE DSK$DD49,DK@TYP,R!BUF   .Channel type                      INDD49.551
1762    <macro>                        STORE D4$UPC,DK@UPC,R!BUF     .Units/channel                     INDD49.552
1772    <macro>                        STORE D4$CPD,DK@CPD,R!BUF     .Cylinders/disk                    INDD49.553
2000    <macro>                        STORE D4$TPC,DK@TPC,R!BUF     .Tracks/cylinder                   INDD49.554
2011    <macro>                        STORE D4$SPT,DK@SPT,R!BUF     .Sectors/track                     INDD49.555
2021    <macro>                        STORE I@SPIRD4,DK@SPF,R!BUF   .Spiral enable flag                INDD49.556
2032    <macro>                        PUT R!DPT,DK@DPT,R!BUF        .Device parameter table            INDD49.557
2037    <macro>                        PUT R!CHAN,DK@CHN,R!BUF       .Channel number                    INDD49.558
2052    <macro>                        STORE O'177777,DK@CYL,R!BUF   .Force initial seek                INDD49.559
2060    <macro>                        STORE O'37,DK@HED,R!BUF       .Force initial head select         INDD49.560
2071   010066  022442  024423          R!%W1 = DK@LT + R!BUF         .First LMA entry                   INDD49.561
2074   020423  012022  013006          R!%W2 = R!%W1 + LB@SZ - LB@LE .Last LMA entry                    INDD49.562
       024424
2100    <macro>                        PUT R!%W1,DK@FLB,R!BUF        .Save first                        INDD49.563
2105    <macro>                        PUT R!%W1,DK@CLB,R!BUF        .Initialize current                INDD49.564
2112    <macro>                        PUT R!%W2,DK@LLB,R!BUF        .Save last                         INDD49.565
2117    <macro>                        STORE LB$DD49,DK@NLB,R!BUF    .Number of LMA entries             INDD49.566
2127    <macro>                        $UNTIL (R!%W1 > R!%W2)        .Do all entries                    INDD49.567
2134    <macro>                          PUT R!BUF,LB@DCB,R!%W1      .Set DCB address                   INDD49.568
2141   010006  025423                    R!%W1 = R!%W1 + LB@LE       .Next                              INDD49.569
2143    <macro>                        $ENDTIL                                                          INDD49.570
2144    <macro>                        ADDRESS R!%W1,DK@TMO,R!BUF    .Timout table                      INDD49.571
2147    <macro>                        PUT R!TOUT,TMR@RT,R!%W1       .Timout routine address            INDD49.572
2154    <macro>                        $IF (R!TYPE = CH$D4S)         .If striped disk                   INDD49.573
                                                                                        .>>>>> 0          $IF      1
2160   010052  024424                    R!%W2 = RA$DD49S            .Read ahead count                  INDD49.574
2162   010052  024425                    R!%W3 = WB$DD49S            .Write behind count                INDD49.575
2164    <macro>                          STORE 1,DK@SEL,R!BUF        .Temp. flag for initialization     INDD49.576
2174    <macro>                        $ELSE                                                            INDD49.577
2175   010052  024424                    R!%W2 = RA$DD49                                                INDD49.578
2177   010052  024425                    R!%W3 = WB$DD49                                                INDD49.579
2201    <macro>                        $ENDIF                                                           INDD49.580
                                                                                        .<<<<< 0          $ENDIF   1
2201    <macro>                        PUT R!%W2,DK@RCC,R!BUF        .Store in DCB                      INDD49.581
2214    <macro>                        PUT R!%W3,DK@WCC,R!BUF                                           INDD49.582
2227    <macro>                        PUT R!%W2,DK@DRC,R!BUF        .Set default counts                INDD49.583
2242    <macro>                        PUT R!%W3,DK@DWC,R!BUF                                           INDD49.584
2255   010003  025437                  R!MCB = R!MCB + CB@SZ         .Running sum for MCB space         INDD49.585
2257   001000                          EXIT                                                             INDD49.586
```

```
                                                                                INDD49.587
                                 ********************************************************************  INDD49.588
                                 *                                                                  *  INDD49.589
                                 *        MSGOUT:    Send error message to the console.             *  INDD49.590
                                 *                                                                  *  INDD49.591
                                 *        Entry:     R!CHAN = Channel number                        *  INDD49.592
                                 *                   R!ERR = Error message code                     *  INDD49.593
                                 *                   R!DCB = DCB address                            *  INDD49.594
                                 *                                                                  *  INDD49.595
                                 ********************************************************************  INDD49.596
                                                                                                       INDD49.597
2260                             MSGOUT   *                                                            INDD49.598
2260   010040  024450                     R!LEN = MSGLEN                 .Size                         INDD49.599
2262   077000 /000715                     R = GETBUF                     .Allocate local memory        INDD49.600
2264    <macro>                           COPY R!ERR+R!%B,R!BUF,R!LEN    .Move msg to buffer           INDD49.601
2303   020431  004003  012060             R!%W1 = R!CHAN > 3 + 'O'R < D'8 .Convert channel to ASCII    INDD49.602
       005010  024423
2310   020431  011007  012060             R!%W1 = R!CHAN & 7 + 'O'R + R!%W1                             INDD49.603
       022423  024423
2315   020442  012004  024410             (R!BUF + CHNOFF) = R!%W1       .Store in msg                 INDD49.604
       020423  034410
2322    <macro>                           MSG R!BUF                      .Send to console              INDD49.605
2333    <macro>                           RELMEM R!BUF                   .Release local memory         INDD49.606
2343   020431  054000                     B = R!CHAN                                                   INDD49.607
2345    <macro>                           STORE 1,DK@SEL,R!DCB           .Set error indicator          INDD49.608
2355   001000                             EXIT                                                         INDD49.609
```

```
                        *        Message data.                                    INDD49.611
                                                                                  INDD49.612
2356                    MSG0     *                                                INDD49.613
2356                             'CHANNEL XX CHANNEL TIMEOUT'                     INDD49.614
2374                    MSG1     *                                                INDD49.615
2374                             'CHANNEL XX CHANNEL ERROR'                       INDD49.616
2411                    MSG2     *                                                INDD49.617
2411                             'CHANNEL XX DEVICE NOT READY'                    INDD49.618
2427                    MSG3     *                                                INDD49.619
2427                             'CHANNEL XX DEVICE SELECT ERROR'                 INDD49.620
2447                    MSG4     *                                                INDD49.621
2447                             'CHANNEL XX SEQ OP IN PROGRESS'                  INDD49.622
2466                    MSG5     *                                                INDD49.623
2466                             'CHANNEL XX CATASTROPHIC DRIVE ERROR'            INDD49.624
2510                    MSG6     *                                                INDD49.625
2510                             'CHANNEL XX UNABLE TO OBTAIN STATUS'             INDD49.626
2532                    MSG7     *                                                INDD49.627
2532                             'CHANNEL XX UNABLE TO RESET DEVICE'              INDD49.628
2553                    MSG8     *                                                INDD49.629
2553                             'CHANNEL XX DEVICE RELEASE ERROR'                INDD49.630
2573                    MSG9     *                                                INDD49.631
2573                             'CHANNEL XX UNABLE TO RECALIBRATE'               INDD49.632
                                                                                  INDD49.633
                        *        Spiral conversion table.                         INDD49.634
                                                                                  INDD49.635
2614                    D4CNVRT  *                                                INDD49.636
2614   000000                    0                                                INDD49.637
2615   000012                    D'10                                             INDD49.638
2616   000025                    D'21                                             INDD49.639
2617   000037                    D'31                                             INDD49.640
                                 END                                              INDD49.641
```

0X	%ACTIVE		5:36 D	5:36	6:18 D	6:18				
0X	%B		1: 4 D	4:10	4:42 B	7:21 B	7:42 B	10:24 B	10:26 B	10:40 B
			11:15 B	11:19 B	11:25 B	11:31 B	11:38 B	12:10 B	12:32 B	12:36 B
			13: 4 B	13: 8 B	13:12 B	13:24 B	13:33 B	14: 3 B	18:42 B	20:14 B
			20:15							
0X	%DCBQ		3:14 D	9:40	9:41					
0X	%DD49		3:15 D	4:38	4:40	10: 1	10: 2	10: 3		
0X	%EX		5:36 D	5:36	6:18 D	6:18	6:30 D	6:30	7: 5 D	7: 5
			7:28 D	7:28	8:10 D	8:10	9:17 D	9:17	9:30 D	9:30
			10: 2 D	10: 2	11:16 D	11:16	17:14 D	17:14	20:22 D	20:22
			20:23 D	20:23						
10	%GBLNUM	$APTEXT	2:13 F	2:13						
400	%GBLREG	$APTEXT	2:13							
0X	%MEMIO		3:16 D	9:41	9:42					
0X	%MYID		3:17 D	4:23	5:20	5:27				
0X	%PUNT		5:18 D	5:18	5:37 D	5:37	6:19 D	6:19	7: 2 D	7: 2
			9:18 D	9:18	9:31 D	9:31				
20	AD@P1@N	$APTEXT	5:36	5:36 F	6:18	6:18 F				
14	AD@P1@P	$APTEXT	5:36 F	5:36	6:18 F	6:18				
0	AD@P1@S	$APTEXT	5:36	5:36 F	6:18	6:18 F				
20	AD@P2@N	$APTEXT	5:36	5:36 F	6:18	6:18 F				
15	AD@P2@P	$APTEXT	5:36 F	5:36	6:18 F	6:18				
0	AD@P2@S	$APTEXT	5:36	5:36 F	6:18	6:18 F				
20	AD@RC@N	$APTEXT	5:36	5:36 F	6:18	6:18 F				
13	AD@RC@P	$APTEXT	5:36 F	5:36	6:18 F	6:18				
0	AD@RC@S	$APTEXT	5:36	5:36 F	6:18	6:18 F				
20	CB@BM0@N	$APTEXT	6:26 F							
2	CB@BM0@P	$APTEXT	6:26 F	6:26						
0	CB@BM0@S	$APTEXT	6:26 F							
20	CB@BM1@N	$APTEXT	6:27 F							
3	CB@BM1@P	$APTEXT	6:27 F	6:27						
0	CB@BM1@S	$APTEXT	6:27 F							
2	CB@LH	$APTEXT	6:23							
20	CB@NUM@N	$APTEXT	6:25 F							
1	CB@NUM@P	$APTEXT	6:25 F	6:25						
0	CB@NUM@S	$APTEXT	6:25 F							
3	CB@SZ	$APTEXT	6:30	6:31	19:41					
14	CB@SZ@P	$APTEXT	5:38							
3	CH$D4	$APTEXT	4:36							
103	CH$D4S	$APTEXT	4:36	19:28						
1566	D4$CPD	$APTEXT	4:16	19: 6						
1572	D4$CPDA	$APTEXT	4:18							

```
    52   D4$SPT     $APTEXT      4:15      19: 8
    54   D4$SPTA    $APTEXT      4:17
    10   D4$TPC     $APTEXT     19: 7
     1   D4$UPC     $APTEXT     19: 5
    0X   D4TOUT                 3:18 D    10: 3
     6   DCB@LT     $APTEXT      9:34      9:41
    0X   DCCB                   3:19 D     6: 2      7:13      7:31      19: 1
    0X   DD49                   3:20 D     4:40
    20   DD@DCB@N   $APTEXT      6:22 F
     2   DD@DCB@P   $APTEXT      6:22 F     6:22
     0   DD@DCB@S   $APTEXT      6:22 F
    20   DK@BFO@N   $APTEXT      6:35 F
    63   DK@BFO@P   $APTEXT      6:35 F     6:35
     0   DK@BFO@S   $APTEXT      6:35 F
    20   DK@BF1@N   $APTEXT      6:36 F
    64   DK@BF1@P   $APTEXT      6:36 F     6:36
     0   DK@BF1@S   $APTEXT      6:36 F
   777   DK@CHN@M   $APTEXT      7:20       7:41     19:11
    11   DK@CHN@N   $APTEXT      7:20       7:41     19:11 F
     7   DK@CHN@P   $APTEXT      7:20 F     7:20      7:41 F    7:41      19:11 F   19:11
     7   DK@CHN@S   $APTEXT      7:20       7:20 F    7:41      7:41 F    19:11 F
177000   DK@CHN@X   $APTEXT     19:11
    20   DK@CLB@N   $APTEXT     19:18 F
    13   DK@CLB@P   $APTEXT     19:18 F    19:18
     0   DK@CLB@S   $APTEXT     19:18 F
   377   DK@CNT@M   $APTEXT      6:34
    10   DK@CNT@N   $APTEXT      6:34 F
    16   DK@CNT@P   $APTEXT      6:34 F     6:34
    10   DK@CNT@S   $APTEXT      6:34 F
177400   DK@CNT@X   $APTEXT      6:34
    20   DK@CPD@N   $APTEXT     19: 6 F
    57   DK@CPD@P   $APTEXT     19: 6 F    19: 6
     0   DK@CPD@S   $APTEXT     19: 6 F
    20   DK@CYL@N   $APTEXT     19:12 F
     3   DK@CYL@P   $APTEXT     19:12 F    19:12
     0   DK@CYL@S   $APTEXT     19:12 F
    20   DK@DPT@N   $APTEXT     19:10 F
    61   DK@DPT@P   $APTEXT     19:10 F    19:10
     0   DK@DPT@S   $APTEXT     19:10 F
   377   DK@DRC@M   $APTEXT     19:39
    10   DK@DRC@N   $APTEXT     19:39 F    19:39
    54   DK@DRC@P   $APTEXT     19:39 F    19:39
```

0	DK@DRC@S	$APTEXT	19:39	F	19:39											
377	DK@DWC@M	$APTEXT	19:40													
10	DK@DWC@N	$APTEXT	19:40	F	19:40											
55	DK@DWC@P	$APTEXT	19:40	F	19:40											
0	DK@DWC@S	$APTEXT	19:40	F	19:40											
20	DK@FLB@N	$APTEXT	19:17	F												
11	DK@FLB@P	$APTEXT	19:17	F	19:17											
0	DK@FLB@S	$APTEXT	19:17	F												
37	DK@HED@M	$APTEXT	19:13	F												
5	DK@HED@N	$APTEXT	19:13	F	19:13											
4	DK@HED@P	$APTEXT	19:13	F	19:13											
2	DK@HED@S	$APTEXT	19:13	F	19:13											
20	DK@LLB@N	$APTEXT	19:19	F												
12	DK@LLB@P	$APTEXT	19:19	F	19:19											
0	DK@LLB@S	$APTEXT	19:19	F												
66	DK@LT	$APTEXT	18:35		19:14											
20	DK@MC0@N	$APTEXT	6:28	F												
32	DK@MC0@P	$APTEXT	6:28	F	6:28											
0	DK@MC0@S	$APTEXT	6:28	F												
20	DK@MC1@N	$APTEXT	6:29	F												
33	DK@MC1@P	$APTEXT	6:29	F	6:29											
0	DK@MC1@S	$APTEXT	6:29	F												
17	DK@NLB@M	$APTEXT	19:20	F												
4	DK@NLB@N	$APTEXT	19:20	F	19:20											
56	DK@NLB@P	$APTEXT	19:20	F	19:20											
0	DK@NLB@S	$APTEXT	19:20	F	19:20											
377	DK@RCC@M	$APTEXT	19:37													
10	DK@RCC@N	$APTEXT	19:37	F												
54	DK@RCC@P	$APTEXT	19:37	F	19:37											
10	DK@RCC@S	$APTEXT	19:37	F												
177400	DK@RCC@X	$APTEXT	19:37													
1	DK@SEL@M	$APTEXT	6:13	F	8: 3	F	19:31	F	20:25	F						
1	DK@SEL@N	$APTEXT	6: 9		6: 9	F	6:13	F	6:13		7:38		7:38	F	8: 3	F	8: 3
			19:31	F	19:31		20:25	F	20:25							
7	DK@SEL@P	$APTEXT	6: 9	F	6: 9		6:13	F	6:13		7:38	F	7:38		8: 3	F	8: 3
			19:31	F	19:31		20:25	F	20:25							
0	DK@SEL@S	$APTEXT	6: 9		6: 9	F	6:13	F	6:13		7:38		7:38	F	8: 3	F	8: 3
			19:31	F	19:31		20:25	F	20:25							
1	DK@SPF@M	$APTEXT	19: 9	F												
1	DK@SPF@N	$APTEXT	19: 9	F	19: 9											
2	DK@SPF@P	$APTEXT	19: 9	F	19: 9											
7	DK@SPF@S	$APTEXT	19: 9	F	19: 9											

777	DK@SPI@M	$APTEXT	19: 8 F								
11	DK@SPI@N	$APTEXT	19: 8 F								
60	DK@SPI@P	$APTEXT	19: 8 F	19: 8							
7	DK@SPI@S	$APTEXT	19: 8 F								
177000	DK@SPI@X	$APTEXT	19: 8								
44	DK@TMO@P	$APTEXT	19:25								
37	DK@TPC@M	$APTEXT	19: 7 F								
5	DK@TPC@N	$APTEXT	19: 7 F	19: 7							
60	DK@TPC@P	$APTEXT	19: 7 F	19: 7							
2	DK@TPC@S	$APTEXT	19: 7 F	19: 7							
37	DK@TYP@M	$APTEXT	6: 6	7:17	7:35	19: 4 F					
5	DK@TYP@N	$APTEXT	6: 6	6: 6 F	7:17	7:17 F	7:35	7:35 F	19: 4 F	19: 4	
7	DK@TYP@P	$APTEXT	6: 6 F	6: 6	7:17 F	7:17	7:35 F	7:35	19: 4 F	19: 4	
2	DK@TYP@S	$APTEXT	6: 6	6: 6 F	7:17	7:17 F	7:35	7:35 F	19: 4 F	19: 4	
3	DK@UPC@M	$APTEXT	19: 5 F								
2	DK@UPC@N	$APTEXT	19: 5 F	19: 5							
60	DK@UPC@P	$APTEXT	19: 5 F	19: 5							
0	DK@UPC@S	$APTEXT	19: 5 F	19: 5							
377	DK@WCC@M	$APTEXT	19:38								
10	DK@WCC@N	$APTEXT	19:38 F								
55	DK@WCC@P	$APTEXT	19:38 F	19:38							
10	DK@WCC@S	$APTEXT	19:38 F								
177400	DK@WCC@X	$APTEXT	19:38								
20	DP@ACY@N	$APTEXT	4:18 F								
10	DP@ACY@P	$APTEXT	4:18 F	4:18							
0	DP@ACY@S	$APTEXT	4:18 F								
777	DP@ASE@M	$APTEXT	4:17 F								
11	DP@ASE@N	$APTEXT	4:17 F								
7	DP@ASE@P	$APTEXT	4:17 F	4:17							
7	DP@ASE@S	$APTEXT	4:17 F								
177000	DP@ASE@X	$APTEXT	4:17								
20	DP@DCY@N	$APTEXT	4:16 F								
6	DP@DCY@P	$APTEXT	4:16 F	4:16							
0	DP@DCY@S	$APTEXT	4:16 F								
777	DP@DSE@M	$APTEXT	4:15 F								
11	DP@DSE@N	$APTEXT	4:15 F								
5	DP@DSE@P	$APTEXT	4:15 F	4:15							
7	DP@DSE@S	$APTEXT	4:15 F								
177000	DP@DSE@X	$APTEXT	4:15								
17	DP@IDX@M	$APTEXT	4:14 F								
4	DP@IDX@N	$APTEXT	4:14 F								
4	DP@IDX@P	$APTEXT	4:14 F	4:14							

14	DP@IDX@S	$APIEXT	4:14	F			
177760	DP@IDX@X	$APIEXT	4:14				
11	DP@LT	$APIEXT	4: 4				
1	DP@SPI@M	$APIEXT	4:13	F			
1	DP@SPI@N	$APIEXT	4:13	F	4:13		
4	DP@SPI@P	$APIEXT	4:13	F	4:13		
0	DP@SPI@S	$APIEXT	4:13	F	4:13		
0	DP@SPI@P	$APIEXT	4:12				
3	DSK$DD49	$APIEXT	6: 8		7:19	7:37	19: 4
0X	EITB		3:21	D	4:22		
53	F$FIND	$APIEXT	9:30				
30	F$GMEM	$APIEXT	9:17				
20	F$MESG	$APIEXT	20:22				
35	F$MGET	$APIEXT	5:36		6:18		
46	F$MOSR	$APIEXT	10: 2				
47	F$MOSW	$APIEXT	6:30				
7	F$PAUSE	$APIEXT	7:28		11:16	17:14	
52	F$RETURN	$APIEXT	8:10				
31	F$RMEM	$APIEXT	7: 5		20:23		
1	FS$NWAIT	$APIEXT	10: 2				
0	FS$WAIT	$APIEXT	6:30				
3	I@INDXD4	$APIEXT	4:14				
1	I@SPIRD4	$APIEXT	4:13		19: 9		
3	LB$DD49	$APIEXT	3:24		19:20		
20	LB@DCB@N	$APIEXT	19:22	F			
5	LB@DCB@P	$APIEXT	19:22	F	19:22		
0	LB@DCB@S	$APIEXT	19:22	F			
6	LB@LE	$APIEXT	3:24		19:15	19:23	
53	MB$DD49	$APIEXT	6:15				
53	MB$DD49S	$APIEXT	6:12				
12	MEM@LT	$APIEXT	9:34		9:42		
0X	O$DD49		9:30	D	9:30		
0X	O$INDD49		1: 4	D	1: 4		
6	OV@ENT@P	$APIEXT	4:23				
17	OV@NUM@N	$APIEXT	1: 4				
1	OV@TYP@N	$APIEXT	1: 4				
112	PT$BPATH	$APIEXT	5:18				
50	PT$DD49	$APIEXT	7: 2				
14	PT$LOCAL	$APIEXT	9:18				
12	PT$NOLAY	$APIEXT	9:31				
15	PT$NOMOS	$APIEXT	5:37		6:19		
52	RA$DD49	$APIEXT	19:33				

52	RA$DD49S	$APTEXT	19:29			
11	SM@LIST@N	$APTEXT	1: 4			
7	SM@NUM@N	$APTEXT	1: 4			
0	TM$CMEM	$APTEXT	5:16			
2	TM$SSD	$APTEXT	5:22			
20	TM@IOP@N	$APTEXT	5:19	5:19 F	5:26	5:26 F
5	TM@IOP@P	$APTEXT	5:19 F	5:19	5:26 F	5:26
0	TM@IOP@S	$APTEXT	5:19	5:19 F	5:26	5:26 F
0X	TMCB		3:22 D	5:15		
20	TMR@RT@N	$APTEXT	19:26 F			
2	TMR@RT@P	$APTEXT	19:26 F	19:26		
0	TMR@RT@S	$APTEXT	19:26 F			
52	WB$DD49	$APTEXT	19:34			
52	WB$DD49S	$APTEXT	19:30			

SYMBOL QUALIFIER IS INDD49

1	$CONST	4:13 D	4:13 F	4:13	4:14 D	4:14 F	4:14	4:15 D	4:15 F
		4:15	4:16 D	4:16 F	4:16	4:17 D	4:17 F	4:17	4:18 D
		4:18 F	4:18	6:13 D	6:13 F	6:13	8: 3 D	8: 3 F	8: 3
		19: 4 D	19: 4 F	19: 4	19: 5 D	19: 5 F	19: 5	19: 6 D	19: 6 F
		19: 6	19: 7 D	19: 7 F	19: 7	19: 8 D	19: 8 F	19: 8	19: 9 D
		19: 9 F	19: 9	19:12 D	19:12 F	19:12	19:13 D	19:13 F	19:13
		19:20 D	19:20 F	19:20	19:31 D	19:31 F	19:31	20:25 D	20:25 F
		20:25							
461	$REGORG	2:13 D	2:13 F	2:14	2:14 D	2:15	2:15 D	2:16	2:16 D
		2:17	2:17 D	2:18	2:18 D	2:18 F	2:19	2:19 D	2:20
		2:20 D	2:21	2:21 D	2:22	2:22 D	2:23	2:23 D	2:24
		2:24 D	2:24 F	2:25	2:25 D	2:26	2:26 D	2:27	2:27 D
		2:28	2:28 D	2:29	2:29 D	2:29 F	2:30	2:30 D	2:30 F
		2:31	2:31 D	2:32	2:32 D	2:33	2:33 D	2:34	2:34 D
		2:35	2:35 D	2:36	2:36 D	2:37	2:37 D	2:38	2:38 D
		2:39	2:39 D	2:40	2:40 D	2:41	2:41 D	2:42	2:42 D
		3: 1	3: 1 D	3: 2	3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D
		3: 5	3: 5 D	3: 6	3: 6 D	3: 7	3: 7 D	3: 8	3: 8 D
		3: 9	3: 9 D	3:10	3:10 D	3:11	3:11 D	3:12	3:12 D
		3:12 F	5:36	6:18	6:30	7: 5	7:28	7:28 F	8:10
		9:17	9:30	10: 2	11:16	11:16 F	17:14	17:14 F	20:22
		20:22 F	20:23						
1	%NP	1: 4	2:30 D						
430	%P	1: 4	2:29 D	2:30	7:28 F	7:28	11:16 F	11:16	17:14 F
		17:14	20:22 F	20:22					

410	%S1	2:13	2:14 D	4:13	4:14	4:15	4:16	4:17	4:18
		5:16	5:19	5:22	5:26	5:36	6: 2	6: 3	6: 6
		6: 9	6:13	6:18	6:22	6:25	6:26	6:27	6:28
		6:29	6:34	6:35	6:36	7:13	7:14	7:17	7:20
		7:31	7:32	7:35	7:38	7:41	8: 3	11:11	11:22
		11:26	11:27	13: 6	14:23	19: 4	19: 5	19: 6	19: 7
		19: 8	19: 9	19:10	19:11	19:12	19:13	19:17	19:18
		19:19	19:20	19:22	19:26	19:31	19:37	19:38	19:39
		19:40	20:20	20:21	20:25				
411	%S2	2:13	2:15 D						
412	%S3	2:13	2:16 D						
413	%S4	2:13	2:17 D						
414	%S5	2:13	2:18 D						
415	%T1	2:19 D	4:12	9:19	20:15				
416	%T2	2:20 D	4:12	9:19	20:15				
417	%T3	2:21 D	4:12	20:15					
420	%T4	2:22 D							
421	%T5	2:23 D							
422	%T6	2:24 D							
423	%W1	2:25 D	4: 5	4:10	4:12	4:24	4:25	4:26	4:27
		4:39	4:40	5:15	5:16	5:22	5:34	5:36	6:23
		6:25	6:26	6:27	9:17	9:36	15: 3	15: 9	15:10
		18:36	19: 2	19: 3	19:14	19:15	19:17	19:18	19:21
		19:22	19:23	19:25	19:26	20:17	20:18	20:19	20:20
424	%W2	2:26 D	4: 6	5:17	5:19	5:23	5:26	9:17	9:37
		14:15	14:25	14:26	18:37	19:15	19:19	19:21	19:29
		19:33	19:37	19:39					
425	%W3	2:27 D	4: 7	9:17	9:38	15:27	15:38	15:39	16:18
		16:31	16:32	18:39	18:40	18:41	19:30	19:34	19:38
		19:40							
426	%W4	2:28 D	16:22	16:28					
427	%W5	2:29 D	15: 6	18:10					
430	AMAP	2:30 D	4:23	4:25	4:27				
442	BUF	2:40 D	4: 9	5:14	5:20	5:27	5:33	6:21	6:22
		6:23	6:30	7: 4	7: 5	9:17	9:19	9:40	19: 3
		19: 4	19: 5	19: 6	19: 7	19: 8	19: 9	19:10	19:11
		19:12	19:13	19:14	19:17	19:18	19:19	19:20	19:22
		19:25	19:31	19:37	19:38	19:39	19:40	20:15	20:20
		20:22	20:23						
445	BUFL	3: 1 D	6:18	6:27	6:36				
444	BUFU	2:42 D	6:18	6:26	6:35				
1406	CAT	11:28	11:32	13:42 D					

431	CHAN	2:31 D	4:29		4:39	5: 3	7:20	7:41	10:17
		11:17	12:26	17:15	18:38	18:40	19: 1	19:11	20:16
		20:18	20:24						
1257	CHEND	7:42	12:25 D						
1056	CHENTT	7:21	10:16 D						
1711	CHNERR	15: 9	17:30	18:13	18:21 D				
4	CHNOFF	3:25 D	20:20						
1452	CLEAR	10:25	11:14	12: 9	13:23	13:36	14:17	15: 2 D	15:29
		16:20	17:10	17:21					
1454	CLEARA	15: 4 D	15:11						
4	CNVRTLEN	3:27 D	4:12						
453	CON	3: 7 D	10:31	12:29	13:29	14:23	15:35	16:28	17:18
		18:12							
436	COUNT	2:36 D	4:30	4:37	6: 1	6:37	7: 2		
2614	D4CNVRT	4:10	21:26 D						
460	DCB	3:12 D	6: 3	6: 6	6: 9	6:13	6:22	6:28	6:29
		6:34	6:35	6:36	7:14	7:17	7:20	7:32	7:35
		7:38	7:41	8: 3	20:25				
457	DPT	3:11 D	4: 9	4:12	4:13	4:14	4:15	4:16	4:17
		4:18	19:10						
443	ERR	2:41 D	10:24	10:26	10:31	10:38	10:39	11:15	11:19
		11:25	11:31	11:37	11:39	12:10	12:29	12:30	12:31
		12:35	12:37	13: 4	13: 7	13:11	13:13	13:24	13:29
		13:31	13:32	13:40	14: 1	14:18	14:23	14:30	14:31
		15:14	15:30	15:35	15:42	16: 1	16:21	16:28	16:35
		16:36	17:11	17:18	17:30	17:31	17:32	18:14	18:19
		18:22	20:15						
1411	ERRMSG	13:41	14: 2 D						
451	GEN	3: 5 D	11: 6	11: 9	11:11	11:22	15:37	17:20	
1474	GENERAL	11:18	15:26 D						
1476	GENERALA	15:28 D	15:40						
715	GETBUF	4: 8	5:39	9:16 D	9:39	18:42	20:14		
433	LAST	2:33 D	4:28	4:32					
22	LB@SZ	3:24 D	18:35	19:15					
450	LEN	3: 4 D	4: 4	5:38	9:17	9:19	9:30	9:32	9:34
		18:35	20:13	20:15					
437	MCB	2:37 D	4:31	5:34	19:41				
447	MOSI	3: 3 D	5:36	6:29	6:30	6:31	9:30	10: 2	
446	MOSU	3: 2 D	5:36	6:28	6:30	6:31	9:30	10: 2	
2356	MSG0	17:31	18:19	21: 3 D					
2374	MSG1	17:18	18:22	21: 5 D					
2411	MSG2	14:30	21: 7 D						

2427	MSG3	10:38	21: 9 D						
2447	MSG4	13:40	21:11 D						
2466	MSG5	14: 1	21:13 D						
2510	MSG6	15:42	16:35	21:15 D					
2532	MSG7	11:39	12:37	13:13	17:32	21:17 D			
2553	MSG8	13:31	21:19 D						
2573	MSG9	12:30	13: 7	21:21 D					
40	MSGLEN	3:26 D	20:13						
2260	MSGOUT	10:24	10:26	10:40	11:15	11:19	11:25	11:31	11:38
		12:10	12:32	12:36	13: 4	13: 8	13:12	13:24	13:33
		14: 3	20:12 D						
456	NE	3:10 D	6:12	6:15	6:18	6:25	6:34		
432	NEXT	2:32 D	4:27	4:28	4:33	4:34			
441	NUM	2:39 D	5:42	6: 1	6: 2	6:42	7:11	7:12	7:13
		7:26	7:29	7:30	7:31	8: 8			
1414	READY	10:23	14:14 D						
1416	READYA	14:16 D	14:27						
1351	RELEASE	11:41	12:41	13:22 D	14: 4				
1603	RESET	11:35	12:33	13: 9	13:30	17: 7 D			
1605	RESETA	17: 9 D	17:27						
753	RESIDENT	4:38	9:29 D						
454	RETRY	3: 8 D	10:21	10:33	10:34	11: 4	11:12	11:13	11:20
		17: 8	17:25	17:26					
1	RTR$CLR	3:29 D	15: 9						
5	RTR$RDY	3:30 D	14:25						
2	RTR$RES	3:32 D	17:25						
5	RTR$SLC	3:31 D	10:33						
5	RTR$STA	3:33 D	15:38	16:31					
1062	SELECT	10:22 D	10:35						
1535	SELSTAT	11:24	11:30	13: 3	16:17 D				
1537	SELSTATA	16:19 D	16:33						
1402	SEQOP	11:12	13:39 D						
1715	SETDCB	4:42	18:34 D						
455	SIZE	3: 9 D	9:33	10: 2					
452	STAT	3: 6 D	11:23	11:26	11:29	11:3[illegible]	13: 2	13: 6	16:22
		16:24	16:30						
1126	STATUS	11: 7 D	11:21						
435	TBLE	2:35 D	4:22	4:39					
1704	TIMEOUT	18: 7	18:17 D						
440	TOUT	2:38 D	10: 3	19:26					
434	TYPE	2:34 D	4:34	4:36	19:28				
1664	WAITDN	10:29	12:27	13:27	14:21	15:33	16:26	[illegible]:16	18: 3 D

```
0     <macro>                    OVERLAY    LISTO                                          LISTO.4

                                 COMMENT    'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
6    <macro>                    REGDEFS ,(PA,PB),(ST,WD,HO,SZ,LN,IM,OV,DS,DT,DU,SU,YA,YB,YC,YD    LISTO.5
                      ,,SP,MO,MP,DV)                                                                LISTO.6
                410   %S1     EQUALS   $REGORG                                                  $REG    3
                411   %S2     EQUALS   $REGORG                                                  $REG    3
                412   %S3     EQUALS   $REGORG                                                  $REG    3
                413   %S4     EQUALS   $REGORG                                                  $REG    3
                414   %S5     EQUALS   $REGORG                                                  $REG    3
                415   %T1     EQUALS   $REGORG                                                  $REG    3
                416   %T2     EQUALS   $REGORG                                                  $REG    3
                417   %T3     EQUALS   $REGORG                                                  $REG    3
                420   %T4     EQUALS   $REGORG                                                  $REG    3
                421   %T5     EQUALS   $REGORG                                                  $REG    3
                422   %T6     EQUALS   $REGORG                                                  $REG    3
                423   %W1     EQUALS   $REGORG                                                  $REG    3
                424   %W2     EQUALS   $REGORG                                                  $REG    3
                425   %W3     EQUALS   $REGORG                                                  $REG    3
                426   %W4     EQUALS   $REGORG                                                  $REG    3
                427   %W5     EQUALS   $REGORG                                                  $REG    3
                430   PA      EQUALS   $REGORG                                                  $REG    3
                431   PB      EQUALS   $REGORG                                                  $REG    3
                432   ST      EQUALS   $REGORG                                                  $REG    3
                433   WD      EQUALS   $REGORG                                                  $REG    3
                434   HO      EQUALS   $REGORG                                                  $REG    3
                435   SZ      EQUALS   $REGORG                                                  $REG    3
                436   LN      EQUALS   $REGORG                                                  $REG    3
                437   IM      EQUALS   $REGORG                                                  $REG    3
                440   OV      EQUALS   $REGORG                                                  $REG    3
                441   DS      EQUALS   $REGORG                                                  $REG    3
                442   DT      EQUALS   $REGORG                                                  $REG    3
                443   DU      EQUALS   $REGORG                                                  $REG    3
                444   SU      EQUALS   $REGORG                                                  $REG    3
                445   YA      EQUALS   $REGORG                                                  $REG    3
                446   YB      EQUALS   $REGORG                                                  $REG    3
                447   YC      EQUALS   $REGORG                                                  $REG    3
                450   YD      EQUALS   $REGORG                                                  $REG    3
                451   SP      EQUALS   $REGORG                                                  $REG    3
                452   MO      EQUALS   $REGORG                                                  $REG    3
                453   MP      EQUALS   $REGORG                                                  $REG    3
                454   DV      EQUALS   $REGORG                                                  $REG    3

                              EXT      O$MAX                                                    LISTO.8
```

```
                                                                                LISTO.9
                                 ************************************************************   LISTO.10
                                 *                                                          *   LISTO.11
                                 * LISTO:  Utility                                          *   LISTO.12
                                 *  Overlay to list Kernel overlays, number, size, and Buffer *   LISTO.13
                                 *  Memory address.  Sent to expander channel printer       *   LISTO.14
                                 *                                                          *   LISTO.15
                                 ************************************************************   LISTO.16
                                                                                LISTO.17
                     102     P$BSIZ   EQUALS    D'66          .Printer buffer size          LISTO.18
                      26     P$ISIZ   EQUALS    D'22          .Image size                   LISTO.19
                       1     P$OVNUM  EQUALS    1             .Destination of Ovl num       LISTO.20
                       5     P$NAME   EQUALS    D'5           .Name place                   LISTO.21
                      11     P$SIZE   EQUALS    D'9           .Ov'l size                    LISTO.22
                      15     P$MOSU   EQUALS    D'13          .MOS upper                    LISTO.23
                      21     P$MOSL   EQUALS    D'17          .MOS lower                    LISTO.24
                                                                                LISTO.25
  6   <macro>                        OPEN TYPE=PRINT,MODE=ALPHA,DCT=DV  .OPEN PRINTER      LISTO.26
 46   103274                         P = ERRX, A # R$CMPT          .IF ERROR ON OPEN       LISTO.27
                                                                                LISTO.28
 47   <macro>                        GETMEM   D'72,PA          .Get a line plus 4 parcels  LISTO.29
 67   103225                         P = RETX, A # 0               .If not available       LISTO.30
                                                                                LISTO.31
                                     * Move header info into buffer                     LISTO.32
 70   020430  024434                 HO = PA                   .Destination of data        LISTO.33
 72   014000 /000572  022000         ST = LHDER + R!%B         .Start of move              LISTO.34
      024432
 76   010026  024436                 LN = P$ISIZ               .Length of move             LISTO.35
100   072374                         R = MOVE                  .Move the data              LISTO.36
101   020430  012026  024434         HO = PA + P$ISIZ          .Next image                 LISTO.37
104   072370                         R = MOVE                                              LISTO.38
105   020430  012026  012026         HO = PA + P$ISIZ + P$ISIZ  .Third                     LISTO.39
      024434
111   072363                         R = MOVE                                              LISTO.40
112   010000  024433                 WD = 0                    .Some blank space at top    LISTO.41
114   072401                         R = PRINT                                             LISTO.42
115   072400                         R = PRINT                                             LISTO.43
116   010102  024433                 WD = P$BSIZ               .Size of print line in parcels LISTO.44
120   072375                         R = PRINT                 .Print header               LISTO.45
                                                                                LISTO.46
                                     *  Process overlays to O$MAX                          LISTO.47
121   010000  024440                 OV = 0                    .Start at zero              LISTO.48
```

```
123                                 NXTLINE  *                                                          LISTO.49
123   014000 /020040  024451                 SP = '  'H                 .Space fill buffer             LISTO.50
126   020430  012102  012003                 PB = PA + P$BSIZ + 3       .Size of print buffer + MOS rd buffer  LISTO.51
      024431
132                                 SPCLR    *                                                          LISTO.52
132   020451  034431                         (PB) = SP                  .Store spaces                  LISTO.53
134   027431                                 PB = PB - 1                .Done?                         LISTO.54
135   020431  023430  107005                 P = SPCLR, PB # PA         .                              LISTO.55
                                                                                                       LISTO.56
                                             * PB is equal to PA                                       LISTO.57
140   010000  024437                         IM = 0                     .Image number                  LISTO.58
142                                 NXTIMAG  *                                                          LISTO.59
142    <macro>                               FIND   OV,MO,MP,SZ,TYPE=NUMBER  .Find overlay, MOS addr, size  LISTO.60
160   020431  012001  024441                 DS = PB + P$OVNUM          .Destination of overlay number LISTO.61
163   020440  024442                         DT = OV                    .                              LISTO.62
165   072210                                 R = DSUP                   .Convert, suppress leading '0' LISTO.63
166   020431  012011  024441                 DS = PB + P$SIZE           .Place for size in image       LISTO.64
171   020435  005002  024442                 DT = SZ < 2                .Size in parcels (retrned in words)  LISTO.65
174   072201                                 R = DSUP                   .Convert, suppress leading '0' LISTO.66
175   020431  012015  024441                 DS = PB + P$MOSU           .MOS upper address             LISTO.67
200   020452  024442                         DT = MO                    .MOS upper                     LISTO.68
202   072173                                 R = DSUP                                                  LISTO.69
203   020431  012021  024441                 DS = PB + P$MOSL           .MOS lower                     LISTO.70
206   020453  024442                         DT = MP                    . address                      LISTO.71
210   072170                                 R = CVRT                   .Convert to ascii              LISTO.72
                                                                                                       LISTO.73
                                             * Read in the overlay name                                LISTO.74
211    <macro>                               MOSR   MO,MP,PA+D'68,1     .Read  4 parcels               LISTO.75
232   020431  012005  024441                 DS = PB + P$NAME           .Place for name                LISTO.76
235   020430  012104  024442                 DT = PA + D'68             .Pick up address               LISTO.77
240   010004  024443                         DU = 4                     .Max name size                 LISTO.78
242                                 MNOTDN   *                                                          LISTO.79
242   030442  034441                         (DS) = (DT)                .Move 2 characters             LISTO.80
244   026441                                 DS = DS + 1                .Next 2                        LISTO.81
245   026442                                 DT = DT + 1                                               LISTO.82
246   027443                                 DU = DU - 1                                               LISTO.83
247   030442  102003                         P = MDNDN, (DT) = 0        .If zeroes, done               LISTO.84
251   020443  107010                         P = MNOTDN, DU # 0         .                              LISTO.85
253                                 MDNDN    *                                                          LISTO.86
                                             *  Strip zero character; make space                       LISTO.87
253   027441                                 DS = DS - 1                .Last char.                    LISTO.88
254   030441  011177  024443                 DU = (DS) & 177            .                              LISTO.89
```

```
257   020443  103003                  P = EVEN, DU # 0          .                          LISTO.90
261   010040  035441                  (DS) = (DS) + 40          .Add space char.           LISTO.91
263                           EVEN    *                                                    LISTO.92
                                                                                           LISTO.93
                                                                                           LISTO.94
                                      *  Image done, see about more                        LISTO.95
263   026437                          IM = IM + 1               .Next image                LISTO.96
264   026440                          OV = OV + 1               .Next overlay              LISTO.97
265   020440  017000 /000000          P = ALLDN, OV = O$MAX     .Done?                     LISTO.98
      102013
271   020437  013003  102004          P = NOTNXT, IM = 3        .No, print line?           LISTO.99
274   010026  025431                  PB = PB + P$ISIZ          .Next image home           LISTO.100
276   071134                          P = NXTIMAG               .                          LISTO.101
277                           NOTNXT  *                                                    LISTO.102
277   072216                          R = PRINT                 .Print line                LISTO.103
300   010000  024437                  IM = 0                    .First image               LISTO.104
302   071157                          P = NXTLINE               .Next line                 LISTO.105
                                                                                           LISTO.106
303                           ALLDN   *                                                    LISTO.107
303   072212                          R = PRINT                 .Print a line              LISTO.108
                                                                                           LISTO.109
                                                                                           LISTO.110
304                           RETMEM  *                                                    LISTO.111
304    <macro>                        RELMEM   PA               .Release free memory       LISTO.112
314                           RETX    *                                                    LISTO.113
314    <macro>                        CLOSE DCT=DV                   .CLOSE PRINTER      LISTO.114
334                           RETY    *                                                    LISTO.115
334    <macro>                        RETURN                                               LISTO.116
                                                                                           LISTO.117
                                                                                           LISTO.118
342                           ERRX    *                                                    LISTO.119
342    <macro>                        GETMEM  D'72,PA           .Get memory                LISTO.120
362   107046                          P = RETX, A # 0           .Allocated?                LISTO.121
363   072164                          R = ERRM                  .Error message             LISTO.122
364    <macro>                        RELMEM   PA               .Put back memory           LISTO.123
374   071040                          P = RETY                  .Return                    LISTO.124
                                                                                           LISTO.125
                                      *  Subroutines                                       LISTO.126
                                                                                           LISTO.127
                                      * DSUP: convert octal to ascii, suppress leading '0' s  LISTO.128
                                      * CVRT: convert octal to ascii                        LISTO.129
                                      *      Entry:  DS = Destination, DT = data to convert LISTO.130
```

0X	%ACTIVE		3:18 D	3:18						
0X	%B		1: 4 D	3:26	7:25					
0X	%EX		3:18 D	3:18	3:21 D	3:21	4:13 D	4:13	4:28 D	4:28
			5:24 D	5:24	5:26 D	5:26	5:28 D	5:28	5:32 D	5:32
			5:35 D	5:35	7:16 D	7:16	7:30 D	7:30		
10	%GBLNUM	$APTEXT	2: 2 F	2: 2						
400	%GBLREG	$APTEXT	2: 2							
50	F$CALL	$APTEXT	3:18	5:26	7:16					
6	F$CLSRW	$APTEXT	5:26							
53	F$FIND	$APTEXT	4:13							
30	F$GMEM	$APTEXT	3:21	5:32						
20	F$MESG	$APTEXT	7:30							
46	F$MOSR	$APTEXT	4:28							
52	F$RETURN	$APTEXT	5:28							
31	F$RMEM	$APTEXT	5:24	5:35						
3	F$WRT	$APTEXT	7:16							
0	FSSWAIT	$APTEXT	4:28							
0X	O$LISTO		1: 4 D	1: 4						
0X	O$MAX		2:42 D	5: 9						
0X	O$XCLS		5:26 D	5:26						
0X	O$XOPN		3:18 D	3:18						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
0	R$CMPT	$APTEXT	3:19	7:17						
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
14	SM@REG@P	$APTEXT	3:18							
0	XD$ALPHA	$APTEXT	3:18							
177777	XD$ANY	$APTEXT	3:18							
50122	XD$PRINT	$APTEXT	3:18							
20	XD@OVL@N	$APTEXT	7:16	7:16 F						
3	XD@OVL@P	$APTEXT	7:16 F	7:16						
0	XD@OVL@S	$APTEXT	7:16	7:16 F						

SYMBOL QUALIFIER IS LISTO

40	$$RO		3:18 D	3:18						
455	$REGORG		2: 2 D	2: 2 F	2: 3	2: 3 D	2: 4	2: 4 D	2: 5	2: 5 D
			2: 6	2: 6 D	2: 7	2: 7 D	2: 7 F	2: 8	2: 8 D	2: 9
			2: 9 D	2:10	2:10 D	2:11	2:11 D	2:12	2:12 D	2:13
			2:13 D	2:13 F	2:14	2:14 D	2:15	2:15 D	2:16	2:16 D
			2:17	2:17 D	2:18	2:18 D	2:18 F	2:19	2:19 D	2:20

		2:20 D	2:20 F	2	2:21 D	2:22	2:22 D	2:23	2:23 D
		2:24	2:24 D	2:25	2:25 D	2:26	2:26 D	2:27	2:27 D
		2:28	2:28 D	2:29	2:29 D	2:30	2:30 D	2:31	2:31 D
		2:32	2:32 D	2:33	2:33 D	2:34	2:34 D	2:35	2:35 D
		2:36	2:36 D	2:37	2:37 D	2:38	2:38 D	2:39	2:39 D
		2:39 F	3:18	3:18 F	3:21	4:13	4:28	5:24	5:26
		5:26 F	5:28	5:32	5:35	7:16	7:16 F	7:30	7:30 F
2	%NP	1: 4	2:20 D						
430	%P	1: 4	2:18 D	2:20	3:18 F	3:18	5:26 F	5:26	7:16 F
		7:16	7:30 F	7:30					
410	%S1	2: 2	2: 3 D	7:16					
411	%S2	2: 2	2: 4 D						
412	%S3	2: 2	2: 5 D						
413	%S4	2: 2	2: 6 D						
414	%S5	2: 2	2: 7 D						
415	%T1	2: 8 D	3:18	5:26	7:16				
416	%T2	2: 9 D	3:18						
417	%T3	2:10 D	3:18						
420	%T4	2:11 D	3:18						
421	%T5	2:12 D							
422	%T6	2:13 D							
423	%W1	2:14 D							
424	%W2	2:15 D							
425	%W3	2:16 D							
426	%W4	2:17 D							
427	%W5	2:18 D							
303	ALLDN	5: 9	5:19 D						
400	CVRT	4:25	6: 5 D						
402	CVRTI	6: 4	6: 6 D						
441	DS	2:28 D	4:14	4:17	4:20	4:23	4:29	4:33	4:34
		4:41	4:42	5: 2	6:24	6:27	6:30	6:31	6:33
		6:34	6:36						
375	DSUP	4:16	4:19	4:22	6: 2 D				
442	DT	2:29 D	4:15	4:18	4:21	4:24	4:30	4:33	4:35
		4:37	6: 7	6: 8	6:11	6:12	6:13	6:16	6:17
		6:18							
443	DU	2:30 D	4:31	4:36	4:38	4:42	5: 1		
454	DV	2:39 D	3:18 F	3:18	5:26	7:16			
547	ERRM	5:34	7:18	7:24 D					
620	ERRMSG	7:25	7:35 D						
342	ERRX	3:19	5:31 D						
263	EVEN	5: 1	5: 3 D						

434	HO	2:23 D	3:25	3:30	3:32	7: 2	7:27		
437	IM	2:26 D	4:11	5: 7	5:11	5:16			
572	LHDER	3:26	7:34 D						
436	LN	2:25 D	3:28	6:42	7: 3	7:28			
253	MDNDN	4:37	4:39 D						
242	MNOTDN	4:32 D	4:38						
452	MO	2:37 D	4:13	4:21	4:28				
474	MOVE	3:29	3:31	3:34	6:41 D	7:29			
505	MOVER	7: 4 D	7: 9						
453	MP	2:38 D	4:13	4:24	4:28				
277	NOTNXT	5:11	5:14 D						
142	NXTIMAG	4:12 D	5:13						
123	NXTLINE	4: 1 D	5:17						
440	OV	2:27 D	3:42	4:13	4:15	5: 8	5: 9		
102	P$BSIZ	3:10 D	3:38	4: 3					
26	P$ISIZ	3:11 D	3:28	3:30	3:32	5:12			
21	P$MOSL	3:16 D	4:23						
15	P$MOSU	3:15 D	4:20						
5	P$NAME	3:13 D	4:29						
1	P$OVNUM	3:12 D	4:14						
11	P$SIZE	3:14 D	4:17						
430	PA	2:19 D	3:21	3:25	3:30	3:32	4: 3	4: 8	4:28
		4:30	5:24	5:32	5:35	7:16	7:27	7:30	
431	PB	2:20 D	4: 3	4: 6	4: 7	4: 8	4:14	4:17	4:20
		4:23	4:29	5:12					
515	PRINT	3:36	3:37	3:39	5:15	5:20	7:15 D		
304	RETMEM	5:23 D	7:19						
314	RETX	3:22	5:25 D	5:33					
334	RETY	5:27 D	5:36						
451	SP	2:36 D	4: 2	4: 6					
132	SPCLR	4: 5 D	4: 8						
432	ST	2:21 D	3:26	7: 1	7:25				
463	STOR	6:21	6:22	6:29 D					
466	STOR1	6:25	6:32 D						
471	STOR2	6:28	6:35 D						
444	SU	2:31 D	6: 3	6: 5	6:21				
435	SZ	2:24 D	4:13	4:18					
433	WD	2:22 D	3:35	3:38	7:16				
445	YA	2:32 D	6: 7	6: 8	6:12	6:13	6:17	6:18	7: 1
		7: 5	7: 6						
446	YB	2:33 D	6:19	6:22	6:30	7: 2	7: 5	7: 7	
447	YC	2:34 D	6:14	6:25	6:33	7: 3	7: 8	7: 9	

450 YD 2:35 D 6: 9 c

```
0     <macro>                    OVERLAY   LISTP                                          LISTP.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'            OVERLAY  1
```

```
<macro>                    REGDEFS ,(PA),(LN,     D,SR,SC,SD,PC,LM,DV)                    LISTP.5
410      %S1       EQUALS    $REGORG                                                        $REG    3
411      %S2       EQUALS    $REGORG                                                        $REG    3
412      %S3       EQUALS    $REGORG                                                        $REG    3
413      %S4       EQUALS    $REGORG                                                        $REG    3
414      %S5       EQUALS    $REGORG                                                        $REG    3
415      %T1       EQUALS    $REGORG                                                        $REG    3
416      %T2       EQUALS    $REGORG                                                        $REG    3
417      %T3       EQUALS    $REGORG                                                        $REG    3
420      %T4       EQUALS    $REGORG                                                        $REG    3
421      %T5       EQUALS    $REGORG                                                        $REG    3
422      %T6       EQUALS    $REGORG                                                        $REG    3
423      %W1       EQUALS    $REGORG                                                        $REG    3
424      %W2       EQUALS    $REGORG                                                        $REG    3
425      %W3       EQUALS    $REGORG                                                        $REG    3
426      %W4       EQUALS    $REGORG                                                        $REG    3
427      %W5       EQUALS    $REGORG                                                        $REG    3
430      PA        EQUALS    $REGORG                                                        $REG    3
431      LN        EQUALS    $REGORG                                                        $REG    3
432      HO        EQUALS    $REGORG                                                        $REG    3
433      WD        EQUALS    $REGORG                                                        $REG    3
434      SR        EQUALS    $REGORG                                                        $REG    3
435      SC        EQUALS    $REGORG                                                        $REG    3
436      SD        EQUALS    $REGORG                                                        $REG    3
437      PC        EQUALS    $REGORG                                                        $REG    3
440      LM        EQUALS    $REGORG                                                        $REG    3
441      DV        EQUALS    $REGORG                                                        $REG    3
         ******************************************                                      LISTP.6
         *                                                                                LISTP.7
         *   LISTP: Utility                                                               LISTP.8
         *                                                                                LISTP.9
         *   Overlay to list all the kernel halt codes on the expander                    LISTP.10
         *   channel printer                                                              LISTP.11
         *                                                                                LISTP.12
         ******************************************                                      LISTP.13
                                                                                          LISTP.14
                                                                                          LISTP.15
 6   <macro>                   OPEN TYPE=PRINT,MODE=ALPHA,DCT=DV    .OPEN PRINTER        LISTP.16
46   103314                    P = NOTAVL, A # R$CMPT          .IF NOT AVAILABLE          LISTP.17
47   <macro>                   GETMEM D'32, PA            .Get a line of memory + 4 parcels LISTP.18
67   103242                    P = CLOSE, A # 0                .If no memory available    LISTP.19
70   010000  034430            (PA) = 0                   .Clear control parcel            LISTP.20
```

```
                                  *        move title into buffer memory                                   LISTP.21
                                                                                                          LISTP.22
                                                                                                          LISTP.23
  72   010000  024433                      WD = 0                                                         LISTP.24
  74   072152                              R = PRINT                 .Do some spacing at top              LISTP.25
  75   072151                              R = PRINT                                                      LISTP.26
  76   020430  012001  024432              HO = PA + 1                                                    LISTP.27
 101   014000 /000441  022000              LM = TITLE + R!%B         .Source of title                     LISTP.28
       024440
 105   072111                              R = MOVE                  .Move to buffer memory               LISTP.29
 106   020431  024433                      WD = LN                   .Set print line length               LISTP.30
 110   072136                              R = PRINT                 .Print the title                     LISTP.31
 111   010000  024433                      WD = 0                                                         LISTP.32
 113   072133                              R = PRINT                 .Skip down a line                    LISTP.33
                                                                                                          LISTP.34
                                  *        move header into buffer memory                                  LISTP.35
                                                                                                          LISTP.36
 114   020430  012001  024432              HO = PA + 1               .Destination of data                 LISTP.37
 117   014000 /000470  022000              LM = HEADER + R!%B        .Source of data                      LISTP.38
       024440
 123   072073                              R = MOVE                  .Move the data                       LISTP.39
 124   020431  024433                      WD = LN                   .Set length of print line            LISTP.40
 126   072120                              R = PRINT                 .Print the line                      LISTP.41
                                                                                                          LISTP.42
                                  *        Print all codes and text from table                             LISTP.43
                                                                                                          LISTP.44
 127   014000 /000477  024434              R!SR = PTABLE                  .Offset of table in overlay     LISTP.45
 132   020000  022434  024423              R!%W1 = R!%B + R!SR            .Absolute address of table      LISTP.46
 135   030423  024437                      R!PC = (R!%W1)                 .First punt code                LISTP.47
                                                                                                          LISTP.48
 137    <macro>                            $UNTIL (PC = PT$LAST)          .All printed                    LISTP.49
 142   020430  012001  024432              HO = PA + 1               .Reset destination pointer          LISTP.50
 145   026434                              SR = SR + 1               .Point to local text address        LISTP.51
 146   020000  022434  024423              R!%W1 = R!%B + R!SR                                           LISTP.52
 151   020000  032423  024440              R!LM = R!%B + (R!%W1)          .Set source pointer             LISTP.53
                                  *                                  .First 2 digits to ascii             LISTP.54
 154   020437  011700  005002              R!%W1 = R!PC & O'700 < 2                                      LISTP.55
       024423
 160   020437  011070  004003              R!%W2 = R!PC & O'70 > 3                                       LISTP.56
       024424
 164   020423  022424  016000              (R!HO) = R!%W1 + R!%W2 + '00'H                                LISTP.57
      /030060  034432
```

```
17.   026432                                HO = HO + 1                 .Account for first parcel          LISTP.58
172   020437  011007  005010               (HO) = PC & 7<10 + '0 'H .Convert third digit to ascii         LISTP.59
      016000 /030040  034432
200   026432                                HO = HO + 1                 .Account for second parcel         LISTP.60
201   072015                                R = MOVE                    .Move to destination               LISTP.61
202   020431  012002  024433                WD = LN + 2                 .Set length of print line          LISTP.62
205   072041                                R = PRINT                   .Print the line                    LISTP.63
                                                                                                          LISTP.64
206   026434                                SR = SR + 1                      .Next table entry             LISTP.65
207   020000  022434  024423                R!%W1 = R!%B + R!SR                                            LISTP.66
212   030423  024437                        R!PC = (R!%W1)                   .Next punt code               LISTP.67
214    <macro>                              $ENDTIL                                                        LISTP.68
215   070104                                P = RETMEM                       .All done                     LISTP.69
                                                                                                          LISTP.70
                                   *        MOVE: Move data from LM to HO until null parcel                LISTP.71
                                                                                                          LISTP.72
216                                MOVE     *                                                              LISTP.73
216   020440  024435                        SC = LM                     .Source                            LISTP.74
220   020432  024436                        SD = HO                     .Destination                       LISTP.75
222   010000  024431                        LN = 0                      .Initialize parcel count           LISTP.76
224                                MOVEM    *                                                              LISTP.77
224   030435  103002  001000                EXIT, (SC) = 0              .Exit if this parcel is null       LISTP.78
227   030435  034436                        (SD) = (SC)                 .Move a parcel                     LISTP.79
231   026431                                LN = LN + 1                 .Keep track of parcels moved       LISTP.80
232   030435  011177  024410                P = INCR, 0 # (SC) & 177 .Add ascii space on null byte         LISTP.81
      010000  023410  103004
240   010040  035436                        (SD) = (SD) + 040                                              LISTP.82
242   001000                                EXIT                                                           LISTP.83
243   026436                               INCR      SD = SD + 1                 .Increment dest parcel address LISTP.84
244   026435                                SC = SC + 1                 .Increment source parcel addr      LISTP.85
245   071021                                P = MOVEM                                                      LISTP.86
                                                                                                          LISTP.87
                                   *        PRINT: Print line on PRO.  If WD = 0, print a blank line else  LISTP.88
                                   *        WD has buffer size.                                            LISTP.89
                                                                                                          LISTP.90
246                                PRINT    *                                                              LISTP.91
246    <macro>                              WRITE DCT=DV,BUF=PA,LNG=WD       .WRITE 1 LINE                 LISTP.92
274   103002  001000                        EXIT, A = R$CMPT                 .IF NO ERROR                  LISTP.93
276   072002                                R = PRIERR                  .Send message to CRT               LISTP.94
277   070022                                P = RETMEM                  .Return memory                     LISTP.95
                                                                                                          LISTP.96
                                   *        PRIERR: put out error message                                  LISTP.97
```

```
                                                                                                LISTP.98
300                         PRTERR   *                                                          LISTP.99
300   014000 /000415  022000         LM = ERRMSG + R!%B            .Error message source         LISTP.100
      024440
304   020430  024432                 HO = PA                  .Destination                     LISTP.101
306   073070                         R = MOVE                 .Move to buffer memory           LISTP.102
307    <macro>                       MSG PA                   .Send to CRT                     LISTP.103
320   001000                         EXIT                     .Return                          LISTP.104
                                                                                                LISTP.105
                            *        RETMEM: return acquired memory                              LISTP.106
                                                                                                LISTP.107
321                         RETMEM   *                                                          LISTP.108
321    <macro>                       RELMEM PA                .Release memory                  LISTP.109
331                         CLOSE    *                                                          LISTP.110
331    <macro>                       $IF (DV # 0)                  .If device allocated          LISTP.111
                                                                                  .>>>>> 0          $IF      1
334    <macro>                         CLOSE DCT=DV                .Close printer                LISTP.112
354    <macro>                       $ENDIF                                                     LISTP.113
                                                                                  .<<<<< 0          $ENDIF   1
354    <macro>                       RETURN                   .Beat feet outa here             LISTP.114
                                                                                                LISTP.115
362                         NOTAVL   *                                                          LISTP.116
362    <macro>                       GETMEM    D'32,R!PA           .Allocate a buffer            LISTP.117
402   107051                         P = CLOSE, A # 0              .If no memory available       LISTP.118
403   073103                         R = PRTERR                                                  LISTP.119
404    <macro>                       RELMEM PA                .Return memory                   LISTP.120
414   071063                         P = CLOSE                                                   LISTP.121
                                                                                                LISTP.122
                                                                                                LISTP.123
415                         ERRMSG   'PRINTER ERROR OR PRINTER UNAVAILABLE'                      LISTP.124
440   000000                         0                                                           LISTP.125
441                         TITLE    '                    KERNEL HALT (PUNT) CODES'              LISTP.126
470                         HEADER   'CODE MEANING'                                              LISTP.127
                                                                                                LISTP.128
                            *        The following table is structured in parcels with the first LISTP.129
                            *        parcel containing PUNT code as defined in AP$TEXT.  The    LISTP.130
                            *        next contiguous parcel contains the local text address.  The LISTP.131
                            *        last entry in the table must always be PT$LAST.            LISTP.132
                                                                                                LISTP.133
477   000000  000677        PTABLE   PDATA     PT$HUH,HUH                                        LISTP.134
501   000001  000724                 PDATA     PT$MEMER,MEMER                                    LISTP.135
503   000002  000747                 PDATA     PT$START,START                                    LISTP.136
```

```
50   000003  001001   PDATA   PI$MOS,I                    LISTP.13
507  000004  001025   PDATA   PI$HISP,HISP                LISTP.138
511  000005  001053   PDATA   PI$CRAY,CRAY                LISTP.139
513  000006  001075   PDATA   PI$DISK,DISK                LISTP.140
515  000007  001123   PDATA   PI$EXECO,EXECO              LISTP.141
517  000010  001146   PDATA   PI$CLOBO,CLOBO              LISTP.142
521  000011  001173   PDATA   PI$ACCUM,ACCUM              LISTP.143
523  000012  001230   PDATA   PI$NOLAY,NOLAY              LISTP.144
525  000013  001244   PDATA   PI$STACK,STACK              LISTP.145
527  000014  001267   PDATA   PI$LOCAL,LOCAL              LISTP.146
531  000015  001311   PDATA   PI$NOMOS,NOMOS              LISTP.147
533  000016  001336   PDATA   PI$MAP,MAP                  LISTP.148
535  000017  001360   PDATA   PI$CONFG,CONFG              LISTP.149
537  000020  001403   PDATA   PI$MSGCH,MSGCH              LISTP.150
541  000021  001432   PDATA   PI$SMDSZ,SMDSZ              LISTP.151
543  000022  001461   PDATA   PI$BADAD,BADAD              LISTP.152
545  000023  001500   PDATA   PI$BADPR,BADPR              LISTP.153
547  000024  001525   PDATA   PI$KILL,KILL                LISTP.154
551  000025  001545   PDATA   PI$LOSP,LOSP                LISTP.155
553  000026  001573   PDATA   PI$BLOCK,BLOCK              LISTP.156
555  000040  001612   PDATA   PI$BMCHE,BMCHE              LISTP.157
557  000041  001635   PDATA   PI$BMDE,BMDE                LISTP.158
561  000042  001656   PDATA   PI$BMSIO,BMSIO              LISTP.159
563  000044  001674   PDATA   PI$BMDAT,BMDAT              LISTP.160
565  000050  001714   PDATA   PI$DD49,DD49                LISTP.161
567  000051  001731   PDATA   PI$NODBG,NODBG              LISTP.162
571  000052  001750   PDATA   PI$BMRQ,BMRQ                LISTP.163
573  000053  001773   PDATA   PI$HSLA,HSLA                LISTP.164
575  000054  002023   PDATA   PI$IOLEN,IOLEN              LISTP.165
577  000055  002042   PDATA   PI$NOHSP,NOHSP              LISTP.166
601  000056  002070   PDATA   PI$NACT,NACT                LISTP.167
603  000057  002114   PDATA   PI$BMDAL,BMDAL              LISTP.168
605  000060  002136   PDATA   PI$BDEMC,BDEMC              LISTP.169
607  000061  002150   PDATA   PI$BKSR,BKSR                LISTP.170
611  000062  002171   PDATA   PI$IKSR,IKSR                LISTP.171
613  000063  002211   PDATA   PI$BKSRP,BKSRP              LISTP.172
615  000064  002234   PDATA   PI$BDEV,BDEV                LISTP.173
617  000065  002255   PDATA   PI$BIOP,BIOP                LISTP.174
621  000066  002306   PDATA   PI$QFULL,QFULL              LISTP.175
623  000067  002321   PDATA   PI$IOADD,IOADD              LISTP.176
625  000070  002341   PDATA   PI$SMER,SMER                LISTP.177
627  000071  002347   PDATA   PI$LMSP,LMSP                LISTP.178
```

```
 631   000072  002366                PDATA    PT$BOVL,BOVL                                    LISTP.179
 633   000073  002406                PDATA    PT$LMCER,LMCER                                  LISTP.180
 635   000074  002425                PDATA    PT$LMREL,LMREL                                  LISTP.181
 637   000075  002442                PDATA    PT$BDIOP,BDIOP                                  LISTP.182
 641   000076  002454                PDATA    PT$UXINT,UXINT                                  LISTP.183
 643   000077  002474                PDATA    PT$DISKE,DISKE                                  LISTP.184
 645   000100  002502                PDATA    PT$AMAP,AMAP                                    LISTP.185
 647   000101  002531                PDATA    PT$OMAX,OMAX                                    LISTP.186
 651   000102  002563                PDATA    PT$INIT,INIT                                    LISTP.187
 653   000103  002600                PDATA    PT$MOSC,MOSC                                    LISTP.188
 655   000104  002615                PDATA    PT$OVLSZ,OVLSZ                                  LISTP.189
 657   000105  002635                PDATA    PT$EOF,EOF                                      LISTP.190
 661   000106  002655                PDATA    PT$XSOV,XSOV                                    LISTP.191
 663   000107  002666                PDATA    PT$DEV,DEV                                      LISTP.192
 665   000110  002706                PDATA    PT$SSD,SSD                                      LISTP.193
 667   000111  002725                PDATA    PT$NOSSD,NOSSD                                  LISTP.194
 671   000112  002747                PDATA    PT$BPATH,BPATH                                  LISTP.195
 673   000113  002776                PDATA    PT$BADTM,BADTM                                  LISTP.196
 675   000116  000000                PDATA    PT$LAST,0          .All entries must precede this LISTP.197
                                                                                             LISTP.198
 677                          HUH      ' No error code specified on $PUNTIF macro'           LISTP.199
 724                          MEMER    ' Local memory error (always HARDWARE)'               LISTP.200
 747                          START    ' Buffer memory error on deadstart (always HARDWARE)' LISTP.201
1001                          MOS      ' Buffer memory error (always HARDWARE)'              LISTP.202
1025                          HISP     ' High-speed channel error (always HARDWARE)'         LISTP.203
1053                          CRAY     ' Invalid message received from CPU'                  LISTP.204
1075                          DISK     ' Invalid parameter in disk request from CPU'         LISTP.205
1123                          EXEC0    ' Program was executing at location 0'                LISTP.206
1146                          CLOB0    ' Local memory location 0 was overwritten'            LISTP.207
1173                          ACCUM    ' Undefined message received on IOP communication channel' LISTP.208
1230                          NOLAY    ' Overlay does not exist'                             LISTP.209
1244                          STACK    ' Station stack overflow or underflow'                LISTP.210
1267                          LOCAL    ' Local memory buffer not available'                  LISTP.211
1311                          NOMOS    ' Buffer memory disk buffer not available'            LISTP.212
1336                          MAP      ' Invalid local buffer release call'                  LISTP.213
1360                          CONFG    ' Buffer memory incorrectly configured'               LISTP.214
1403                          MSGCH    ' IOP message channels incorrectly configured'        LISTP.215
1432                          SMDSZ    ' SMOD is too large for area on buffer memory'        LISTP.216
1461                          BADAD    ' Invalid local memory address'                       LISTP.217
1500                          BADPR    ' Illegal interrupt program sequence code'            LISTP.218
1525                          KILL     ' Stop request received from CPU'                     LISTP.219
1545                          LOSP     ' Low-speed channel error (always HARDWARE)'          LISTP.220
```

```
1575   BLOCK   ' Block number val    ion trap'                                  LISTP.221
1612   BMCHE   ' Block Mux interrupt processor error'                          LISTP.222
1635   BMDE    ' Bad CRW address in device table'                              LISTP.223
1656   BMSIO   ' Block mux start I/O error'                                    LISTP.224
1674   BMDAT   ' Block Mux configuration error'                                LISTP.225
1714   DD49    ' DD49 disk software error'                                     LISTP.226
1731   NODBG   ' The debugger was not loaded'                                  LISTP.227
1750   BMRQ    ' Bad buffer memory allocation request'                         LISTP.228
1773   HSLA    ' Bad local memory address on hi speed I/O call'                LISTP.229
2023   IOLEN   ' Invalid I/O length specified'                                 LISTP.230
2042   NOHSP   ' No hi speed channel configured for request'                   LISTP.231
2070   NACT    ' Illegal activity activation requested'                        LISTP.232
2114   BMDAL   ' Buffer memory DAL queue exhausted'                            LISTP.233
2136   BDEMC   ' Illegal Demon call'                                           LISTP.234
2150   BKSR    ' Undefined kernel service request'                             LISTP.235
2171   IKSR    ' Illegal kernel service request'                               LISTP.236
2211   BKSRP   ' Bad kernel service request parameter'                         LISTP.237
2234   BDEV    ' Requested device not configured'                              LISTP.238
2255   BIOP    ' Illegal IOP requested on kernel service request'              LISTP.239
2306   QFULL   ' Requested queue full'                                         LISTP.240
2321   IOADD   ' Illegal I/O address specified'                                LISTP.241
2341   SMER    ' SMOD error'                                                   LISTP.242
2347   LMSP    ' Local memory space exhausted'                                 LISTP.243
2366   BOVL    ' Illegal overlay load requested'                               LISTP.244
2406   LMCER   ' Corrupted local memory chain'                                 LISTP.245
2425   LMREL   ' Bad local memory release'                                     LISTP.246
2442   BDIOP   ' Bad I/O parameter'                                            LISTP.247
2454   UXINT   ' Unexpected interrupt received'                                LISTP.248
2474   DISKE   ' Disk error'                                                   LISTP.249
2502   AMAP    ' AMAP not available for system initialization'                 LISTP.250
2531   OMAX    ' Illegal overlay number read during initialization'            LISTP.251
2563   INIT    ' IOP initialization error'                                     LISTP.252
2600   MOSC    ' MOS configuration error'                                      LISTP.253
2615   OVLSZ   ' Overlay too large for loading'                                LISTP.254
2635   EOF     ' Premature tape EOF encountered'                               LISTP.255
2655   XSOV    ' Exit stack fault'                                             LISTP.256
2666   DEV     ' Illegal device type configured'                               LISTP.257
2706   SSD     ' SSD high speed channel error'                                 LISTP.258
2725   NOSSD   ' No SSD backdoor channel configured'                           LISTP.259
2747   BPATH   ' High speed path to target memory not defined'                 LISTP.260
2776   BADTM   ' Illegal target memory type'                                   LISTP.261
       END                                                                     LISTP.262
```

0X	%ACTIVE		2:38 D	2:38						
0X	%B		1: 4 D	3: 8	3:19	3:28	3:34	3:35	4:10	5: 3
0X	%EX		2:38 D	2:38	2:40 D	2:40	4:37 D	4:37	5: 7 D	5: 7
			5:13 D	5:13	5:17 D	5:17	5:20 D	5:20	5:23 D	5:23
			5:26 D	5:26						
10	%GBLNUM	$APTEXT	2: 1 F	2: 1						
400	%GBLREG	$APTEXT	2: 1							
50	F$CALL	$APTEXT	2:38	4:37	5:17					
6	F$CLSRW	$APTEXT	5:17							
30	F$GMEM	$APTEXT	2:40	5:23						
20	F$MESG	$APTEXT	5: 7							
52	F$RETURN	$APTEXT	5:20							
31	F$RMEM	$APTEXT	5:13	5:26						
3	F$WRT	$APTEXT	4:37							
0X	O$LISTP		1: 4 D	1: 4						
0X	O$XCLS		5:17 D	5:17						
0X	O$XOPN		2:38 D	2:38						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
11	PT$ACCUM	$APTEXT	6: 7 D							
100	PT$AMAP	$APTEXT	7: 7 D							
22	PT$BADAD	$APTEXT	6:16 D							
23	PT$BADPR	$APTEXT	6:17 D							
113	PT$BADTM	$APTEXT	7:18 D							
60	PT$BDEMC	$APTEXT	6:33 D							
64	PT$BDEV	$APTEXT	6:37 D							
75	PT$BDIOP	$APTEXT	7: 4 D							
65	PT$BIOP	$APTEXT	6:38 D							
61	PT$BKSR	$APTEXT	6:34 D							
63	PT$BKSRP	$APTEXT	6:36 D							
26	PT$BLOCK	$APTEXT	6:20 D							
40	PT$BMCHE	$APTEXT	6:21 D							
57	PT$BMDAL	$APTEXT	6:32 D							
44	PT$BMDAT	$APTEXT	6:24 D							
41	PT$BMDE	$APTEXT	6:22 D							
52	PT$BMRQ	$APTEXT	6:27 D							
42	PT$BMSTO	$APTEXT	6:23 D							
72	PT$BOVL	$APTEXT	7: 1 D							
112	PT$BPATH	$APTEXT	7:17 D							
10	PT$CLOBO	$APTEXT	6: 6 D							
17	PT$CONFG	$APTEXT	6:13 D							
5	PT$CRAY	$APTEXT	6: 3 D							

50	PT$DD49	$APTEXT	6:25 D	
107	PT$DEV	$APTEXT	7:14 D	
6	PT$DISK	$APTEXT	6: 4 D	
77	PT$DISKE	$APTEXT	7: 6 D	
105	PT$EOF	$APTEXT	7:12 D	
7	PT$EXECO	$APTEXT	6: 5 D	
4	PT$HISP	$APTEXT	6: 2 D	
53	PT$HSLA	$APTEXT	6:28 D	
0	PT$HUH	$APTEXT	5:40 D	
62	PT$IKSR	$APTEXT	6:35 D	
102	PT$INIT	$APTEXT	7: 9 D	
67	PT$IOADD	$APTEXT	6:40 D	
54	PT$IOLEN	$APTEXT	6:29 D	
24	PT$KILL	$APTEXT	6:18 D	
116	PT$LAST	$APTEXT	3:31	7:19 D
73	PT$LMCER	$APTEXT	7: 2 D	
74	PT$LMREL	$APTEXT	7: 3 D	
71	PT$LMSP	$APTEXT	6:42 D	
14	PT$LOCAL	$APTEXT	6:10 D	
25	PT$LOSP	$APTEXT	6:19 D	
16	PT$MAP	$APTEXT	6:12 D	
1	PT$MEMER	$APTEXT	5:41 D	
3	PT$MOS	$APTEXT	6: 1 D	
103	PT$MOSC	$APTEXT	7:10 D	
20	PT$MSGCH	$APTEXT	6:14 D	
56	PT$NACT	$APTEXT	6:31 D	
51	PT$NODBG	$APTEXT	6:26 D	
55	PT$NOHSP	$APTEXT	6:30 D	
12	PT$NOLAY	$APTEXT	6: 8 D	
15	PT$NOMOS	$APTEXT	6:11 D	
111	PT$NOSSD	$APTEXT	7:16 D	
101	PT$OMAX	$APTEXT	7: 8 D	
104	PT$OVLSZ	$APTEXT	7:11 D	
66	PT$QFULL	$APTEXT	6:39 D	
21	PT$SMDSZ	$APTEXT	6:15 D	
70	PT$SMER	$APTEXT	6:41 D	
110	PT$SSD	$APTEXT	7:15 D	
13	PT$STACK	$APTEXT	6: 9 D	
2	PT$START	$APTEXT	5:42 D	
76	PT$UXINT	$APTEXT	7: 5 D	
106	PT$XSOV	$APTEXT	7:13 D	
0	R$CMPT	$APTEXT	2:39	4:38

```
    11   SM@FSI@N  $APTEXT     1: 4
     7   SM@NUM@N  $APTEXT     1: 4
    14   SM@REG@P  $APTEXT     2:38
     0   XD$ALPHA  $APTEXT     2:38
177777   XD$ANY    $APTEXT     2:38
 50122   XD$PRINT  $APTEXT     2:38
    20   XD@OVL@N  $APTEXT     4:37      4:37 F
     3   XD@OVL@P  $APTEXT     4:37 F    4:37
     0   XD@OVL@S  $APTEXT     4:37      4:37 F

         SYMBOL QUALIFIER IS LISTP

    25   $$RO                  2:38 D    2:38
   442   $REGORG               2: 1 D    2: 1 F    2: 2      2: 2 D    2: 3      2: 3 D    2: 4      2: 4 D
                              2: 5      2: 5 D    2: 6      2: 6 D    2: 6 F    2: 7      2: 7 D    2: 8
                              2: 8 D    2: 9      2: 9 D    2:10      2:10 D    2:11      2:11 D    2:12
                              2:12 D    2:12 F    2:13      2:13 D    2:14      2:14 D    2:15      2:15 D
                              2:16      2:16 D    2:17      2:17 D    2:17 F    2:18      2:18 D    2:18 F
                              2:19      2:19 D    2:20      2:20 D    2:21      2:21 D    2:22      2:22 D
                              2:23      2:23 D    2:24      2:24 D    2:25      2:25 D    2:26      2:26 D
                              2:27      2:27 D    2:27 F    2:38      2:38 F    2:40      4:37      4:37 F
                              5: 7      5: 7 F    5:13      5:17      5:17 F    5:20      5:23      5:26
     1   %NP                   1: 4      2:18 D
   430   %P                    1: 4      2:17 D    2:18      2:38 F    2:38      4:37 F    4:37      5: 7 F
                              5: 7      5:17 F    5:17
   410   %S1                   2: 1      2: 2 D    4:25      4:26      4:37
   411   %S2                   2: 1      2: 3 D
   412   %S3                   2: 1      2: 4 D
   413   %S4                   2: 1      2: 5 D
   414   %S5                   2: 1      2: 6 D
   415   %T1                   2: 7 D    2:38      4:37      5:17
   416   %T2                   2: 8 D    2:38
   417   %T3                   2: 9 D    2:38
   420   %T4                   2:10 D    2:38
   421   %T5                   2:11 D
   422   %T6                   2:12 D
   423   %W1                   2:13 D    3:28      3:29      3:34      3:35      3:37      3:41      4:10
                              4:11
   424   %W2                   2:14 D    3:39      3:41
   425   %W3                   2:15 D
   426   %W4                   2:16 D
   427   %W5                   2:17 D
```

1173	ACCUM	6: 7 D	7:30 D						
2502	AMAP	7: 7 D	8:30 D						
1461	BADAD	6:16 D	7:39 D						
1500	BADPR	6:17 D	7:40 D						
2776	BADTM	7:18 D	8:41 D						
2136	BDEMC	6:33 D	8:14 D						
2234	BDEV	6:37 D	8:18 D						
2442	BDIOP	7: 4 D	8:27 D						
2255	BIOP	6:38 D	8:19 D						
2150	BKSR	6:34 D	8:15 D						
2211	BKSRP	6:36 D	8:17 D						
1573	BLOCK	6:20 D	8: 1 D						
1612	BMCHE	6:21 D	8: 2 D						
2114	BMDAL	6:32 D	8:13 D						
1674	BMDAT	6:24 D	8: 5 D						
1635	BMDE	6:22 D	8: 3 D						
1750	BMRQ	6:27 D	8: 8 D						
1656	BMSIO	6:23 D	8: 4 D						
2366	BOVL	7: 1 D	8:24 D						
2747	BPATH	7:17 D	8:40 D						
1146	CLOBO	6: 6 D	7:29 D						
331	CLOSE	2:41	5:14 D	5:24	5:27				
1360	CONFG	6:13 D	7:36 D						
1053	CRAY	6: 3 D	7:26 D						
1714	DD49	6:25 D	8: 6 D						
2666	DEV	7:14 D	8:37 D						
1075	DISK	6: 4 D	7:27 D						
2474	DISKE	7: 6 D	8:29 D						
441	DV	2:27 D	2:38 F	2:38	4:37	5:16	5:17		
2635	EOF	7:12 D	8:35 D						
415	ERRMSG	5: 3	5:30 D						
1123	EXECO	6: 5 D	7:28 D						
470	HEADER	3:19	5:33 D						
1025	HISP	6: 2 D	7:25 D						
432	HO	2:20 D	3: 7	3:18	3:32	3:42	4: 1	4: 3	4: 4
		4:19	5: 5						
1773	HSLA	6:28 D	8: 9 D						
677	HUH	5:40 D	7:21 D						
2171	IKSR	6:35 D	8:16 D						
243	INCR	4:26	4:29 D						
2563	INIT	7: 9 D	8:32 D						
2321	IOADD	6:40 D	8:21 D						

2023	IOLEN	6:29 D	8:10 D						
1525	KILL	6:18 D	7:41 D						
440	LM	2:26 D	3: 8	3:19	3:35	4:18	5: 3		
2406	LMCER	7: 2 D	8:25 D						
2425	LMREL	7: 3 D	8:26 D						
2347	LMSP	6:42 D	8:23 D						
431	LN	2:19 D	3:11	3:22	4: 6	4:20	4:24		
1267	LOCAL	6:10 D	7:33 D						
1545	LOSP	6:19 D	7:42 D						
1336	MAP	6:12 D	7:35 D						
724	MEMER	5:41 D	7:22 D						
1001	MOS	6: 1 D	7:24 D						
2600	MOSC.	7:10 D	8:33 D						
216	MOVE	3:10	3:21	4: 5	4:17 D	5: 6			
224	MOVEM	4:21 D	4:31						
1403	MSGCH	6:14 D	7:37 D						
2070	NACT	6:31 D	8:12 D						
1731	NODBG	6:26 D	8: 7 D						
2042	NOHSP	6:30 D	8:11 D						
1230	NOLAY	6: 8 D	7:31 D						
1311	NOMOS	6:11 D	7:34 D						
2725	NOSSD	7:16 D	8:39 D						
362	NOTAVL	2:39	5:22 D						
2531	OMAX	7: 8 D	8:31 D						
2615	OVLSZ	7:11 D	8:34 D						
430	PA	2:18 D	2:40	2:42	3: 7	3:18	3:32	4:37	5: 5
		5: 7	5:13	5:23	5:26				
437	PC	2:25 D	3:29	3:31	3:37	3:39	4: 2	4:11	
246	PRINT	3: 5	3: 6	3:12	3:14	3:23	4: 7	4:36 D	
300	PRTERR	4:39	5: 2 D	5:25					
477	PTABLE	3:27	5:40 D						
2306	QFULL	6:39 D	8:20 D						
321	RETMEM	4:13	4:40	5:12 D					
435	SC	2:23 D	4:18	4:22	4:23	4:25	4:30		
436	SD	2:24 D	4:19	4:23	4:27	4:29			
1432	SMDSZ	6:15 D	7:38 D						
2341	SMER	6:41 D	8:22 D						
434	SR	2:22 D	3:27	3:28	3:33	3:34	4: 9	4:10	
2706	SSD	7:15 D	8:38 D						
1244	STACK	6: 9 D	7:32 D						
747	START	5:42 D	7:23 D						
441	TITLE	3: 8	5:32 D						

2454	UXINI	7: 5 D	8:28 D					
433	WD	2:21 D	3: 4	3:11	3:13	3:22	4: 6	4:37
2655	XSOV	7:13 D	8:36 D					

```
0     <macro>                    OVERLAY   OVLNUM,TYPE=DATA,EXTN=NO                    OVLNUM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                                 ************************************************** OVLNUM.5
                                 *                                                * OVLNUM.6
                                 *        OVNM      Define overlay numbers        * OVLNUM.7
                                 *                                                * OVLNUM.8
                                 * ovl    OVNM      (text),TYPE=type              * OVLNUM.9
                                 *                                                * OVLNUM.10
                                 *        text      Text describing the function of the overlay.  This  * OVLNUM.11
                                 *                  information is printed as part of the LISTO output. * OVLNUM.12
                                 *                                                * OVLNUM.13
                                 *        type      Overlay type.  APML overlay if not specified.  If   * OVLNUM.14
                                 *                  CAL, the overlay number must match the correspond-  * OVLNUM.15
                                 *                  ing, C$ovl number defined in $APTEXT.               * OVLNUM.16
                                 *                                                * OVLNUM.17
                                 ****************************************************************** OVLNUM.18
                                                                                                     OVLNUM.19
                                          MACRO                                                      OVLNUM.20
<prototype>                      OVL      OVNM      TEXT,TYPE=,NULL=                                 OVLNUM.21
                                          LOCAL     $OVN,$COVN                                       OVLNUM.22
<definition>                     $OVN     $CONCAT   (O$,OVL),D'8                                     OVLNUM.23
<definition>                     "$OVN"   DEFINE    (TEXT),N_ULL=NULL                                OVLNUM.24
<definition>                              IFC       !_TYPE_!,EQ,!CAL!,2                              OVLNUM.25
<definition>                     $COVN    $CONCAT   (C$,OVL),D'8                                     OVLNUM.26
<definition>                              ERRIF     "$OVN",NE,"$COVN"   .Overlay number mismatch     OVLNUM.27
                                 OVNM     ENDM                                                       OVLNUM.28
```

```
                                                                        OVLNUM.30
                    ********************************************************  OVLNUM.31
                    *                                                    *  OVLNUM.32
                    *       Overlay number definitions                   *  OVLNUM.33
                    *                                                    *  OVLNUM.34
                    ********************************************************  OVLNUM.35

                    ********************************************************  OVLNUM.36
                    *                                                    *  OVLNUM.37
                    *       $COVL     CAL overlays                       *  OVLNUM.38
                    *                                                    *  OVLNUM.39
                    ********************************************************  OVLNUM.40
                                                                            OVLNUM.41
                    *       Since CAL does not allow externals in an absolute assembly,  OVLNUM.42
                    *       the CAL overlay numbers are also defined via C$ovl equates  OVLNUM.43
                    *       in $APTEXT.  The O$ovl value must correspond to the C$ovl  OVLNUM.44
                    *       value.                                       OVLNUM.45
                                                                            OVLNUM.46
  6   <macro>       SDMPA     OVNM      (Dump executing exchange packages),TYPE=CAL  OVLNUM.47
 14   <macro>       SDMPB     OVNM      (Dump Cray registers),TYPE=CAL  OVLNUM.48
 20   <macro>       SDMPC     OVNM      (Dump cluster registers),TYPE=CAL  OVLNUM.49
 24   <macro>       SDMPD     OVNM      (Dump SSD from a Cray 1/S),TYPE=CAL  OVLNUM.50
 30   <macro>       SDMPE     OVNM      (Dump SSD from a Cray X-MP),TYPE=CAL  OVLNUM.51
 34   <macro>       MFCHK     OVNM      (Mainframe confidence test),TYPE=CAL  OVLNUM.52

                    ********************************************************  OVLNUM.54
                    *                                                    *  OVLNUM.55
                    *       $KOVL     Kernel overlays                    *  OVLNUM.56
                    *                                                    *  OVLNUM.57
                    ********************************************************  OVLNUM.58
                                                                            OVLNUM.59
 40   <macro>       AMAP      OVNM      (Configuration map)  OVLNUM.60
 44   <macro>       ADEM      OVNM      (Cray interface demon)  OVLNUM.61
 50   <macro>       AMSG      OVNM      (Inter-IOP communications demon)  OVLNUM.62
 54   <macro>       BEGIN     OVNM      (IOS initialization)  OVLNUM.63
 60   <macro>       BTD       OVNM      (Binary to decimal ASCII conversion)  OVLNUM.64
 64   <macro>       BTH       OVNM      (Binary to hex ASCII conversion)  OVLNUM.65
 70   <macro>       BTO       OVNM      (Binary to octal ASCII conversion)  OVLNUM.66
 74   <macro>       CALL      OVNM      (Kernel command processor)  OVLNUM.67
100   <macro>       CDEM      OVNM      (Cray interface demon)  OVLNUM.68
104   <macro>       CLKSNC    OVNM      (Synchronize day clock)  OVLNUM.69
```

110	<macro>	CLOCK	OVNM	(Clock d	OVLNUM.70
114	<macro>	CONFIG	OVNM	(Configuration display)	OVLNUM.71
120	<macro>	CRAY	OVNM	(Cray interface initialization)	OVLNUM.72
124	<macro>	CRTDEM	OVNM	(CRT demon)	OVLNUM.73
130	<macro>	DISKIO	OVNM	(Internal disk I/O)	OVLNUM.74
134	<macro>	DIVIDE	OVNM	(32 bit integer divide)	OVLNUM.75
140	<macro>	DKDMP	OVNM	(Disk dump)	OVLNUM.76
144	<macro>	DKIOEX	OVNM	(Internal disk I/O)	OVLNUM.77
150	<macro>	DKLOOK	OVNM	(Verify disk channel)	OVLNUM.78
154	<macro>	DTB	OVNM	(ASCII decimal to binary conversion)	OVLNUM.79
160	<macro>	ERR	OVNM	(Error channel data display)	OVLNUM.80
164	<macro>	ERRDMP	OVNM	(Error channel data printout)	OVLNUM.81
170	<macro>	FIND	OVNM	(FIND overlay by name)	OVLNUM.82
174	<macro>	GETEXT	OVNM	(Get message text)	OVLNUM.83
200	<macro>	INFORM	OVNM	(Operator message output)	OVLNUM.84
204	<macro>	INDD29	OVNM	(Initialize DD19/29 devices)	OVLNUM.85
210	<macro>	INDD39	OVNM	(Initialize DD39 devices)	OVLNUM.86
214	<macro>	INDD40	OVNM	(Initialize DD40 devices)	R13739AJ.1
220	<macro>	INDD49	OVNM	(Initialize DD49 devices)	OVLNUM.87
224	<macro>	INDD50	OVNM	(Initialize DD50 devices)	OVLNUM.88
230	<macro>	LISTO	OVNM	(Overlay information print out)	OVLNUM.89
234	<macro>	LISTP	OVNM	(Punt code print out)	OVLNUM.90
240	<macro>	MFCLR	OVNM	(Write a pattern into CRAY central memory)	OVLNUM.91
244	<macro>	MFINIT	OVNM	(Mainframe confidence monitor)	OVLNUM.92
250	<macro>	MSGHND	OVNM	(File/device handler message output)	OVLNUM.93
254	<macro>	MULTIPLY	OVNM	(Integer multiply)	OVLNUM.94
260	<macro>	NOBEAT	OVNM	(Warning of unresponsive IOP)	OVLNUM.95
264	<macro>	OPSYS	OVNM	(Define Cray operating system)	OVLNUM.96
270	<macro>	OTB	OVNM	(ASCII octal to binary conversion)	OVLNUM.97
274	<macro>	OUTCALL	OVNM	(Call overlay in another IOP)	OVLNUM.98
300	<macro>	OVLNUM	OVNM	(Overlay numbers)	OVLNUM.99
304	<macro>	PATCH	OVNM	(Patch overlays)	OVLNUM.100
310	<macro>	SCRUB	OVNM	(Refresh memory)	OVLNUM.101
314	<macro>	SETIME	OVNM	(Set internal or hardware clock)	OVLNUM.102
320	<macro>	STATS	OVNM	(Statistics collection)	OVLNUM.103
324	<macro>	SYSS	OVNM	(IOS initialization)	OVLNUM.104
330	<macro>	SYSTEXT	OVNM	(Error message text)	OVLNUM.105
334	<macro>	TCOM	OVNM	(Get trace buffer)	OVLNUM.106
340	<macro>	TDUMP	OVNM	(Trace dump print out)	OVLNUM.107
344	<macro>	TIME	OVNM	(Query/set time and date)	OVLNUM.108
350	<macro>	TRACE	OVNM	(Trace dump/control)	OVLNUM.109
354	<macro>	TRACK	OVNM	(Trace enable/disable)	OVLNUM.110

```
360   <macro>   TSTASH   OVNM   (Get ASCII time or date)                          OVLNUM.111
364   <macro>   USURP    OVNM   (Allocate CRT)                                    OVLNUM.112
370   <macro>   WATCH    OVNM   (Display time and date)                           OVLNUM.113

                ***********************************************************      OVLNUM.115
                *                                                         *      OVLNUM.116
                *        $DSK      Disk driving overlays                  *      OVLNUM.117
                *                                                         *      OVLNUM.118
                ***********************************************************      OVLNUM.119
                                                                                 OVLNUM.120
374   <macro>   ACOM     OVNM   (DD29 inter-IOP message handler)                  OVLNUM.121
400   <macro>   BADDAT   OVNM   (DD29 error recovery, accept bad data)           OVLNUM.122
404   <macro>   D3ECC    OVNM   (DD39 error correction code)                      OVLNUM.123
410   <macro>   D3ERR    OVNM   (DD39 error recovery, main overlay)               OVLNUM.124
414   <macro>   D3IOR    OVNM   (DD39 read/write error recovery)                  OVLNUM.125
420   <macro>   D3LOG    OVNM   (DD39 error logging)                              OVLNUM.126
424   <macro>   D3MSG    OVNM   (DD39 operator messages)                          OVLNUM.127
430   <macro>   D3RLR    OVNM   (DD39 release error recovery)                     OVLNUM.128
434   <macro>   D3SKR    OVNM   (DD39 seek error recovery)                        OVLNUM.129
440   <macro>   D3SLR    OVNM   (DD39 select error recovery)                      OVLNUM.130
444   <macro>   D40ECC   OVNM   (DD40 error correction code)                     R13739AJ.2
450   <macro>   D40ERR   OVNM   (DD40 error recovery, main overlay)              R13739AJ.3
454   <macro>   D40IOR   OVNM   (DD40 read/write error recovery)                 R13739AJ.4
460   <macro>   D4DEM    OVNM   (DD49, DD39 disk demon)                           OVLNUM.131
464   <macro>   D4ECC    OVNM   (DD49 error correction code)                      OVLNUM.132
470   <macro>   D4ERR    OVNM   (DD49 error recovery, main overlay)               OVLNUM.133
474   <macro>   D4IOR    OVNM   (DD49 read/write error recovery)                  OVLNUM.134
500   <macro>   D4LOG    OVNM   (DD49 error logging)                              OVLNUM.135
504   <macro>   D4MSG    OVNM   (DD49 operator messages)                          OVLNUM.136
510   <macro>   D4RES    OVNM   (DD49, DD39 reset overlay, error recovery)        OVLNUM.137
514   <macro>   D4RLR    OVNM   (DD49 unit release error recovery)                OVLNUM.138
520   <macro>   D4SKR    OVNM   (DD49 seek error recovery)                        OVLNUM.139
524   <macro>   D4SLR    OVNM   (DD49 unit select error recovery)                 OVLNUM.140
530   <macro>   D4STAT   OVNM   (DD49, DD39 select status, error recovery)        OVLNUM.141
534   <macro>   DD49     OVNM   (DD49, DD39 interrupt handler)                    OVLNUM.142
540   <macro>   DDIAG    OVNM   (Online diagnostic special function processor)    OVLNUM.143
544   <macro>   DISK     OVNM   (DD29 disk demon)                                 OVLNUM.144
550   <macro>   ERRECK   OVNM   (DD29 error recovery, main overlay)               OVLNUM.145
554   <macro>   FIRECODE OVNM   (DD29 error correction code)                      OVLNUM.146
560   <macro>   ICOM     OVNM   (DD49, DD39 inter-IOP message handler)            OVLNUM.147
564   <macro>   REPORT   OVNM   (DD29 error logging)                              OVLNUM.148
```

```
57    <macro>   TRANSFR  OVNM   (DD49,    buffer memory I/O demon)                OVLNUM.14

                ***********************************************************************  OVLNUM.151
                *                                                                     *  OVLNUM.152
                *       $BMX     Block Mux channel overlays                           *  OVLNUM.153
                *                                                                     *  OVLNUM.154
                ***********************************************************************  OVLNUM.155
                                                                                        OVLNUM.156
574   <macro>   BMXAIO   OVNM   (Auxiliary BMX functions)                             OVLNUM.157
600   <macro>   BMXCON   OVNM   (Configure channels and devices)                      OVLNUM.158
604   <macro>   BMXCPU   OVNM   (Configure from Cray map)                             OVLNUM.159
610   <macro>   BMXDEM   OVNM   (BMX channel demon)                                   OVLNUM.160
614   <macro>   BMXOPE   OVNM   (Open/close operations)                               OVLNUM.161
620   <macro>   BMXSIO   OVNM   (Start I/O operation)                                 OVLNUM.162
624   <macro>   BMXTPO   OVNM   (Open BMX tape device)                                OVLNUM.163

                ***********************************************************************  OVLNUM.165
                *                                                                     *  OVLNUM.166
                *       $TAPE    Block Mux tape handling overlays                     *  OVLNUM.167
                *                                                                     *  OVLNUM.168
                ***********************************************************************  OVLNUM.169
                                                                                        OVLNUM.170
630   <macro>   BCOM     OVNM   (Inter-IOP message demon)                             OVLNUM.171
634   <macro>   BUFMAN   OVNM   (MOS buffer manager)                                  OVLNUM.172
640   <macro>   BYPASS   OVNM   (Data-only request handle)                            OVLNUM.173
644   <macro>   CONMAN   OVNM   (Get configuration map from Cray)                     OVLNUM.174
650   <macro>   DSCGET   OVNM   (Create/destroy DSC table)                            OVLNUM.175
654   <macro>   TAPDIS   OVNM   (Discard DSC data)                                    OVLNUM.176
660   <macro>   TAPEIO   OVNM   (Tape data functions handler)                         OVLNUM.177
664   <macro>   TAPEND   OVNM   (End of read data handling)                           OVLNUM.178
670   <macro>   TAPERR   OVNM   (Tape error recovery handler)                         OVLNUM.179
674   <macro>   TAPFUN   OVNM   (Issue non-data function)                             OVLNUM.180
700   <macro>   TAPMOV   OVNM   (Handle tape control functions)                       OVLNUM.181
704   <macro>   TCART    OVNM   (Error handler for 3480 drives)                       OVLNUM.182
710   <macro>   TCTELL   OVNM   (Display error messages for 3480 drives)              OVLNUM.183
714   <macro>   TDEM     OVNM   (MOS <-> tape demon)                                  OVLNUM.184
720   <macro>   TDEM0    OVNM   (MOS <-> tape demon; read reverse only)               OVLNUM.185
724   <macro>   TDEM1    OVNM   (MOS <-> Cray demon)                                  OVLNUM.186
730   <macro>   TERROR   OVNM   (Tape error handler)                                  OVLNUM.187
734   <macro>   TEX      OVNM   (Tape executive)                                      OVLNUM.188
```

```
  740   <macro>   TRBOC    OVNM   (Bus out check)                                OVLNUM.189
  744   <macro>   TRCER    OVNM   (Channel error)                                OVLNUM.190
  750   <macro>   TRCLN    OVNM   (Issue tape cleaner sequence)                  OVLNUM.191
  754   <macro>   TRDCK    OVNM   (Data check)                                   OVLNUM.192
  760   <macro>   TRDCKO   OVNM   (Read recovery data manager)                   OVLNUM.193
  764   <macro>   TREQC    OVNM   (Equipment check)                              OVLNUM.194
  770   <macro>   TRIDB    OVNM   (ID burst error)                               OVLNUM.195
  774   <macro>   TRINR    OVNM   (Intervention required)                        OVLNUM.196
 1000   <macro>   TRORN    OVNM   (Overrun)                                      OVLNUM.197
 1004   <macro>   TRRDB    OVNM   (Read reverse block)                           OVLNUM.198
 1010   <macro>   TRRDF    OVNM   (Read forward block)                           OVLNUM.199
 1014   <macro>   TRREDO   OVNM   (Read reverse data sorter)                     OVLNUM.200
 1020   <macro>   TRSET    OVNM   (Reset position)                               OVLNUM.201
 1024   <macro>   TRTELL   OVNM   (Display error message)                        OVLNUM.202
 1030   <macro>   TRWRT    OVNM   (Write block)                                  OVLNUM.203

                  ************************************************************* OVLNUM.205
                  *                                                           * OVLNUM.206
                  *        $SDMP     System dump and IOS restart overlays      * OVLNUM.207
                  *                                                           * OVLNUM.208
                  ************************************************************* OVLNUM.209
                                                                                OVLNUM.210
 1034   <macro>   RSTRT    OVNM   (IOS restart utility)                          OVLNUM.211
 1040   <macro>   SDMP0    OVNM   (System dump parameter processor)              OVLNUM.212
 1044   <macro>   SDMP1    OVNM   (MIOP driver)                                  OVLNUM.213
 1050   <macro>   SDMP2    OVNM   (BIOP driver)                                  OVLNUM.214
 1054   <macro>   SDMP3    OVNM   (DD29 disk driver)                             OVLNUM.215
 1060   <macro>   SDMP3A   OVNM   (DD49 disk driver)                             OVLNUM.216
 1064   <macro>   SDMP4    OVNM   (Dump Cray memory and registers)               OVLNUM.217
 1070   <macro>   SDMP5    OVNM   (Dump MOS)                                     OVLNUM.218
 1074   <macro>   SDMP6    OVNM   (Dump IOP0)                                    OVLNUM.219
 1100   <macro>   SDMP7    OVNM   (Dump IOP1)                                    OVLNUM.220
 1104   <macro>   SDMP8    OVNM   (Dump IOP2 and IOP3)                           OVLNUM.221
 1110   <macro>   SDMP9    OVNM   (Dump SSD)                                     OVLNUM.222
 1114   <macro>   SDMP10   OVNM   (Dump Cluster registers)                       OVLNUM.223

                  ************************************************************* OVLNUM.225
                  *                                                           * OVLNUM.226
                  *        $DIAG     Diagnostic overlays                       * OVLNUM.227
                  *                                                           * OVLNUM.228
```

```
                                 ***************************  ****************************************  OVLNUM.229
                                                                                                      OVLNUM.230
1120    <macro>                  CHNTES   OVNM      (Cray 50 Mbyte channel test)                     OVLNUM.231
1124    <macro>                  CPSPIN   OVNM      (CPTEST initialization)                         OVLNUM.232
1130    <macro>                  CPTEST   OVNM      (Block mux tape test)                           OVLNUM.233
1134    <macro>                  ECHOCP   OVNM      (Mainframe low speed channel test)              OVLNUM.234
1140    <macro>                  HSPTES   OVNM      (Mainframe memory test)                         OVLNUM.235
1144    <macro>                  MOSTES   OVNM      (Buffer memory test)                            OVLNUM.236
1150    <macro>                  SSDTES   OVNM      (SSD memory test)                               OVLNUM.237
1154    <macro>                  STOP     OVNM      (Terminate an executing test)                   OVLNUM.238
1160    <macro>                  XDK      OVNM      (Expander disk test)                            OVLNUM.239
1164    <macro>                  XMT      OVNM      (Expander tape test)                            OVLNUM.240
1170    <macro>                  XPR      OVNM      (Expander printer test)                         OVLNUM.241

                                 ***********************************************************************  OVLNUM.243
                                 *                                                                     *  OVLNUM.244
                                 *        $EXP      Expander channel overlays                          *  OVLNUM.245
                                 *                                                                     *  OVLNUM.246
                                 ***********************************************************************  OVLNUM.247
                                                                                                      OVLNUM.248
1174    <macro>                  XCLOCK   OVNM      (Expander clock driver)                         OVLNUM.249
1200    <macro>                  XCLS     OVNM      (Expander device close)                         OVLNUM.250
1204    <macro>                  XDISK    OVNM      (Expander disk driver)                          OVLNUM.251
1210    <macro>                  XDISKA   OVNM      (Ampex disk error handler)                      OVLNUM.252
1214    <macro>                  XDISKB   OVNM      (CDC disk error handler)                        OVLNUM.253
1220    <macro>                  XOPN     OVNM      (Expander device open)                          OVLNUM.254
1224    <macro>                  XPRINT   OVNM      (Expander printer driver)                       OVLNUM.255
1230    <macro>                  XPRNTA   OVNM      (Gould/Calcomp printer error handler)           OVLNUM.256
1234    <macro>                  XPRNTB   OVNM      (Versatec printer error handler)                OVLNUM.257
1240    <macro>                  XTAPE    OVNM      (Expander tape driver)                          OVLNUM.258
1244    <macro>                  XTAPEA   OVNM      (Data General/Kennedy open/close driver)        OVLNUM.259
1250    <macro>                  XTAPEB   OVNM      (Data General/Kennedy error handler)            OVLNUM.260
1254    <macro>                  XTAPEC   OVNM      (Data General/Kennedy error handler)            OVLNUM.261
1260    <macro>                  XTAPED   OVNM      (Data General/Kennedy volume switch handler)    OVLNUM.262

                                 ***********************************************************************  OVLNUM.264
                                 *                                                                     *  OVLNUM.265
                                 *        $FILE     Editor and file system overlays                    *  OVLNUM.266
                                 *                                                                     *  OVLNUM.267
                                 ***********************************************************************  OVLNUM.268
```

```
                                                                                OVLNUM.269
1264   <macro>   CLEAR    OVNM   (Clear directory utility)                      OVLNUM.270
1270   <macro>   CLEARO   OVNM   (Clear disk directory)                         OVLNUM.271
1274   <macro>   COPY     OVNM   (Copy file utility)                            OVLNUM.272
1300   <macro>   COPY0    OVNM   (Copy tape to disk)                            OVLNUM.273
1304   <macro>   COPY1    OVNM   (Copy tape to MOS)                             OVLNUM.274
1310   <macro>   COPY2    OVNM   (Copy disk to tape)                            OVLNUM.275
1314   <macro>   COPY3    OVNM   (Copy disk to MOS)                             OVLNUM.276
1320   <macro>   COPY4    OVNM   (Copy MOS to tape)                             OVLNUM.277
1324   <macro>   COPY5    OVNM   (Copy MOS to disk)                             OVLNUM.278
1330   <macro>   DDUMP    OVNM   (Directory dump utility)                       OVLNUM.279
1334   <macro>   DDUMP0   OVNM   (Directory dump disk to tape)                  OVLNUM.280
1340   <macro>   DEF      OVNM   (Set default disk parameters)                  OVLNUM.281
1344   <macro>   DELET0   OVNM   (Delete disk file)                             OVLNUM.282
1350   <macro>   DELETE   OVNM   (Delete file utility)                          OVLNUM.283
1354   <macro>   DLOAD    OVNM   (Directory load utility)                       OVLNUM.284
1360   <macro>   DLOAD0   OVNM   (Directory load tape to disk)                  OVLNUM.285
1364   <macro>   DSTAT    OVNM   (Directory status display)                     OVLNUM.286
1370   <macro>   DSTAT0   OVNM   (Disk directory status)                        OVLNUM.287
1374   <macro>   EDDELE   OVNM   (Editor delete line routine)                   OVLNUM.288
1400   <macro>   EDINST   OVNM   (Editor insert line routine)                   OVLNUM.289
1404   <macro>   EDIT     OVNM   (Editor initialization)                        OVLNUM.290
1410   <macro>   EDIT0    OVNM   (Editor driver)                                OVLNUM.291
1414   <macro>   EDIT1    OVNM   (Editor file saver)                            OVLNUM.292
1420   <macro>   EDPRNT   OVNM   (Editor print routine)                         OVLNUM.293
1424   <macro>   EDREPL   OVNM   (Editor replace line routine)                  OVLNUM.294
1430   <macro>   EDTYPE   OVNM   (Editor display line routine)                  OVLNUM.295
1434   <macro>   FDUMP    OVNM   (File dump utility)                            OVLNUM.296
1440   <macro>   FDUMP0   OVNM   (File dump disk to tape)                       OVLNUM.297
1444   <macro>   FLAW     OVNM   (Flaw disk utility)                            OVLNUM.298
1450   <macro>   FLOAD    OVNM   (File load utility)                            OVLNUM.299
1454   <macro>   FLOAD0   OVNM   (File load tape to disk)                       OVLNUM.300
1460   <macro>   FORMAT   OVNM   (Disk format utility)                          OVLNUM.301
1464   <macro>   FORMT0   OVNM   (Disk format routine)                          OVLNUM.302
1470   <macro>   FSTAT    OVNM   (File status utility)                          OVLNUM.303
1474   <macro>   FSTAT0   OVNM   (Disk file status)                             OVLNUM.304
1500   <macro>   INIT     OVNM   (Initialize utility)                           OVLNUM.305
1504   <macro>   INIT0    OVNM   (Initialize disk)                              OVLNUM.306
1510   <macro>   LINGET   OVNM   (Editor line locating)                         OVLNUM.307
1514   <macro>   LINPUT   OVNM   (Editor file update)                           OVLNUM.308
1520   <macro>   PLOT     OVNM   (Plot file utility)                            R15159AD.1
1524   <macro>   PLOT0    OVNM   (Plot disk file routine)                       R15159AD.2
```

```
1530   <macro>   PLOT1    OVNM   (Plot t     ile routine)                R15159AD.3
1534   <macro>   PROC     OVNM   (Procedure call utility)                OVLNUM.310
1540   <macro>   PROC0    OVNM   (Procedure call disk file)             OVLNUM.311
1544   <macro>   PRTAPE   OVNM   (Print unblocked tape file)            OVLNUM.312
1550   <macro>   RENAME   OVNM   (Rename file utility)                  OVLNUM.313
1554   <macro>   RENAM0   OVNM   (Rename disk file)                     OVLNUM.314
1560   <macro>   START    OVNM   (Cray deadstart utility)               OVLNUM.315
1564   <macro>   START0   OVNM   (Cray deadstart driver)               OVLNUM.316
1570   <macro>   START1   OVNM   (Cray deadstart from MOS)             OVLNUM.317
1574   <macro>   START2   OVNM   (Cray deadstart from tape)            OVLNUM.318
1600   <macro>   START3   OVNM   (Cray deadstart from disk)            OVLNUM.319
1604   <macro>   UBTAPE   OVNM   (Print blocked tape file)             OVLNUM.320
1610   <macro>   XFMACC   OVNM   (Access expander disk file)           OVLNUM.321
1614   <macro>   XFMCLS   OVNM   (File close)                          OVLNUM.322
1620   <macro>   XFMCRE   OVNM   (File creation)                       OVLNUM.323
1624   <macro>   XFMDEL   OVNM   (File deletion)                       OVLNUM.324
1630   <macro>   XFMDIR   OVNM   (Directory search)                    OVLNUM.325
1634   <macro>   XFMDST   OVNM   (Directory statistics)                OVLNUM.326
1640   <macro>   XFMFIL   OVNM   (File search)                         OVLNUM.327
1644   <macro>   XFMFST   OVNM   (File statistics)                     OVLNUM.328
1650   <macro>   XFMFLW   OVNM   (Flaw routine)                        OVLNUM.329
1654   <macro>   XFMFND   OVNM   (Directory search)                    OVLNUM.330
1660   <macro>   XFMGET   OVNM   (File extention)                      OVLNUM.331
1664   <macro>   XFMIO    OVNM   (Expander disk I/O)                   OVLNUM.332
1670   <macro>   XFMNIT   OVNM   (Disk initialization)                 OVLNUM.333
1674   <macro>   XFMPUT   OVNM   (File space release)                  OVLNUM.334
1700   <macro>   XFMSTR   OVNM   (Update file descriptor)              OVLNUM.335

                 ***********************************************************  OVLNUM.337
                 *                                                         *  OVLNUM.338
                 *        $CONC     Concentrator overlays                  *  OVLNUM.339
                 *                                                         *  OVLNUM.340
                 ***********************************************************  OVLNUM.341
                                                                             OVLNUM.342
1704   <macro>   CONC     OVNM   (Concentrator initialization)          OVLNUM.343
1710   <macro>   CONCERR  OVNM   (Handle concentrator errors)           OVLNUM.344
1714   <macro>   CONCID   OVNM   (Concentrator ID handler)              OVLNUM.345
1720   <macro>   CONCIO   OVNM   (Concentrator activity)                OVLNUM.346
1724   <macro>   ENDCONC  OVNM   (Terminate the concentrator)           OVLNUM.347
1730   <macro>   ENTRID   OVNM   (Enter ID table)                       OVLNUM.348
1734   <macro>   REMVID   OVNM   (Remove entry from ID table)           OVLNUM.349
```

```
                                 ********************************************************************  OVLNUM.350
                                 *                                                                  *  OVLNUM.351
                                 *        $FEI      FEI Logical Path Connection                     *  OVLNUM.352
                                 *                                                                  *  OVLNUM.353
                                 ********************************************************************  OVLNUM.354
                                                                                                       OVLNUM.355
1740    <macro>                  FEIMSG   OVNM     (FEI N-Packet Message Activity)                   OVLNUM.356
1744    <macro>                  FEIR     OVNM     (FEI N-Packet Read Activity)                      OVLNUM.357
1750    <macro>                  FEIW     OVNM     (FEI N-Packet Write Activity)                     OVLNUM.358

                                 ********************************************************************  OVLNUM.360
                                 *                                                                  *  OVLNUM.361
                                 *        $STAT     Station overlays                                *  OVLNUM.362
                                 *                                                                  *  OVLNUM.363
                                 ********************************************************************  OVLNUM.364
                                                                                                       OVLNUM.365
1754    <macro>                  ACQTRM   OVNM      (Acquire response processing)                    OVLNUM.366
1760    <macro>                  ACQUIRE  OVNM      (Acquire response processing)                    OVLNUM.367
1764    <macro>                  AMPEX    OVNM      (Conversion for AMPEX console)                   OVLNUM.368
1770    <macro>                  BABEL    OVNM      (STMSG display)                                  OVLNUM.369
1774    <macro>                  BARDAT   OVNM      (Monitor display descriptors)                    OVLNUM.370
2000    <macro>                  BLOCK    OVNM      (Dataset blocking)                               OVLNUM.371
2004    <macro>                  BMAGET   OVNM      (Get MOS I/O statistics)                         OVLNUM.372
2010    <macro>                  BMGET    OVNM      (Get Mos biffer statistics)                      OVLNUM.373
2014    <macro>                  BXDIS    OVNM      (Display BMX transfers)                          OVLNUM.374
2020    <macro>                  BXSET    OVNM      (Get BMX statistics)                             OVLNUM.375
2024    <macro>                  CFINIT   OVNM      (Command file initialization)                    OVLNUM.376
2030    <macro>                  CFREAD   OVNM      (Command file read)                              OVLNUM.377
2034    <macro>                  CLI      OVNM      (Command language processor)                     OVLNUM.378
2040    <macro>                  CLINIT   OVNM      (Command language initialization)                OVLNUM.379
2044    <macro>                  CLUSTR   OVNM      (Cluster register display)                       OVLNUM.380
2050    <macro>                  CMGET    OVNM      (Get Cray memory statistics)                     OVLNUM.381
2054    <macro>                  CNCDIS   OVNM      (Concentrator monitor)                           OVLNUM.382
2060    <macro>                  COMBO    OVNM      (Debug discriptor validation)                    OVLNUM.383
2064    <macro>                  COMM01   OVNM      (Station commands)                               OVLNUM.384
2070    <macro>                  COMM02   OVNM      (Station commands)                               OVLNUM.385
2074    <macro>                  COMM03   OVNM      (Station commands)                               OVLNUM.386
2100    <macro>                  COMM04   OVNM      (Station commands)                               OVLNUM.387
2104    <macro>                  COMM05   OVNM      (Station commands)                               OVLNUM.388
2110    <macro>                  COMM06   OVNM      (Station commands)                               OVLNUM.389
2114    <macro>                  COMM07   OVNM      (Station commands)                               OVLNUM.390
```

2120	<macro>	COMM08	OVNM	(Station commands)	OVLNUM.39
2124	<macro>	COMM09	OVNM	(Station commands)	OVLNUM.392
2130	<macro>	COMM10	OVNM	(Station commands)	OVLNUM.393
2134	<macro>	COMM11	OVNM	(Station commands)	OVLNUM.394
2140	<macro>	COMM12	OVNM	(Station commands)	OVLNUM.395
2144	<macro>	COMM13	OVNM	(Station commands)	OVLNUM.396
2150	<macro>	COMM14	OVNM	(Station commands)	OVLNUM.397
2154	<macro>	COMM15	OVNM	(Station commands)	OVLNUM.398
2160	<macro>	COMM16	OVNM	(Station commands)	OVLNUM.399
2164	<macro>	CONSL	OVNM	(Console driver)	OVLNUM.400
2170	<macro>	CPUGET	OVNM	(Get Cray time statistics)	OVLNUM.401
2174	<macro>	CRAYIO	OVNM	(Cray message interface)	OVLNUM.402
2200	<macro>	DBGET	OVNM	(Get disk buffer statistics)	OVLNUM.403
2204	<macro>	DECOD2	OVNM	(Parameter scanning)	OVLNUM.404
2210	<macro>	DECODE	OVNM	(Parameter scanning)	OVLNUM.405
2214	<macro>	DELMSG	OVNM	(Delete station messages)	OVLNUM.406
2220	<macro>	DESCRIBE	OVNM	(Command descriptions)	OVLNUM.407
2224	<macro>	DEVDAT	OVNM	(Display expander device status)	OVLNUM.408
2230	<macro>	DISP01	OVNM	(Station displays)	OVLNUM.409
2234	<macro>	DISP02	OVNM	(Station displays)	OVLNUM.410
2240	<macro>	DISPLAY	OVNM	(Display processor)	OVLNUM.411
2244	<macro>	DKDIS	OVNM	(Display disk activity graph)	OVLNUM.412
2250	<macro>	DKERR	OVNM	(Display disk errors)	OVLNUM.413
2254	<macro>	DKERR1	OVNM	(Set DKERR MOS buffer address)	OVLNUM.414
2260	<macro>	DKSET	OVNM	(Get disk I/O statistics)	OVLNUM.415
2264	<macro>	DKSET0	OVNM	(Get disk I/O statistics)	OVLNUM.416
2270	<macro>	DKSTAT	OVNM	(DISK display)	OVLNUM.417
2274	<macro>	DSPL	OVNM	(CPU formatted displays)	OVLNUM.418
2300	<macro>	ERRDIS	OVNM	(Error log display)	OVLNUM.419
2304	<macro>	ERROR	OVNM	(Error message formatter)	OVLNUM.420
2310	<macro>	F80	OVNM	(80 column rotated font)	OVLNUM.421
2314	<macro>	F132	OVNM	(132 column rotated font)	OVLNUM.422
2320	<macro>	FMGET	OVNM	(Get free memory statistics)	OVLNUM.423
2324	<macro>	GRAPH	OVNM	(Bar graph generator)	OVLNUM.424
2330	<macro>	GRCDIS	OVNM	(Generic resource cumulatives)	OVLNUM.425
2334	<macro>	GRPDIS	OVNM	(Generic resource profile)	OVLNUM.426
2340	<macro>	GRUDIS	OVNM	(Generic resource utilization)	OVLNUM.427
2344	<macro>	HDRPAG	OVNM	(Listing header page generation)	OVLNUM.428
2350	<macro>	HELP	OVNM	(Help facility)	OVLNUM.429
2354	<macro>	HELP2	OVNM	(Help facility)	OVLNUM.430
2360	<macro>	HELPD	OVNM	(Help facility data)	OVLNUM.431
2364	<macro>	HPDATA	OVNM	(Listing header page data)	OVLNUM.432

```
2370    <macro>    HSPGET   OVNM    (Get high speed transfer statistics)    OVLNUM.433
2374    <macro>    ICONSL   OVNM    (Console initialization)                OVLNUM.434
2400    <macro>    IDEBUG   OVNM    (Debug displays)                        OVLNUM.435
2404    <macro>    IDLGET   OVNM    (Get idle time statistics)             OVLNUM.436
2410    <macro>    IDRCT    OVNM    (DEBUG display)                        OVLNUM.437
2414    <macro>    IFRMT    OVNM    (Debug display)                        OVLNUM.438
2420    <macro>    KEYBD    OVNM    (Station keyboard input)               OVLNUM.439
2424    <macro>    LCP      OVNM    (LCP generation and validation)        OVLNUM.440
2430    <macro>    LINK     OVNM    (LINK display)                         OVLNUM.441
2434    <macro>    LOGON    OVNM    (Station logon)                        OVLNUM.442
2440    <macro>    MESSAGE  OVNM    (Text message formatting)              OVLNUM.443
2444    <macro>    MSTAT    OVNM    (STORAGE display)                      OVLNUM.444
2450    <macro>    MSTDIS   OVNM    (Mass storage Graph)                   OVLNUM.445
2454    <macro>    NCNDIS   OVNM    (NSC CONC display)                     OVLNUM.446
2460    <macro>    NEWDIS   OVNM    (Display type change)                  OVLNUM.447
2464    <macro>    NSCLP    OVNM    (NSC/FEI logical path display)         OVLNUM.448
2470    <macro>    NSCNET   OVNM    (NSC configuration display)            OVLNUM.449
2474    <macro>    NSCSFP   OVNM    (NSC network statistics display)       OVLNUM.450
2500    <macro>    OBIT     OVNM    (Print file termination notification)  OVLNUM.451
2504    <macro>    OFRMT    OVNM    (Debug display)                        OVLNUM.452
2510    <macro>    ONLINE   OVNM    (Station logon)                        OVLNUM.453
2514    <macro>    POST     OVNM    (Operator message I/O interface)       OVLNUM.454
2520    <macro>    PROTINIT OVNM    (Protocol task initialization)         OVLNUM.455
2524    <macro>    PROTOCOL OVNM    (Cray communications task)             OVLNUM.456
2530    <macro>    QUEUE    OVNM    (Input dataset queuing)                OVLNUM.457
2534    <macro>    READ     OVNM    (CLI read)                             OVLNUM.458
2540    <macro>    SNAP     OVNM    (SNAP command)                         OVLNUM.459
2544    <macro>    SOROC    OVNM    (SOROC translator)                     OVLNUM.460
2550    <macro>    SSDGRA   OVNM    (SSD Channel Graph)                    OVLNUM.461
2554    <macro>    STADIS   OVNM    (STATION display)                      OVLNUM.462
2560    <macro>    STAGEIN  OVNM    (Initialize staging to Cray)           OVLNUM.463
2564    <macro>    STAGEOUT OVNM    (Initialize staging from Cray)         OVLNUM.464
2570    <macro>    STATCL   OVNM    (STATCLASS display)                    OVLNUM.465
2574    <macro>    STATINIT OVNM    (Station initiation)                   OVLNUM.466
2600    <macro>    STATION  OVNM    (Console initiation)                   OVLNUM.467
2604    <macro>    STIO     OVNM    (Dataset staging)                      OVLNUM.468
2610    <macro>    STMSG    OVNM    (Process received station message)     OVLNUM.469
2614    <macro>    STPLOT   OVNM    (Staging to plotter)                   OVLNUM.470
2620    <macro>    STREAMS  OVNM    (Stream control)                       OVLNUM.471
2624    <macro>    STRSTAT  OVNM    (STRSTAT display)                      OVLNUM.472
2630    <macro>    STSGET   OVNM    (Get system statistics)                OVLNUM.473
2634    <macro>    STTAPI   OVNM    (Stage to tape)                        OVLNUM.474
```

```
264     <macro>     STTAPO   OVNM     (Stage  tape)                          OVLNUM.47
2644    <macro>     STUBPR   OVNM     (Stage to printer)                     OVLNUM.476
2650    <macro>     STXDKI   OVNM     (Stage from disk)                      OVLNUM.477
2654    <macro>     STXDKO   OVNM     (Stage to disk)                        OVLNUM.478
2660    <macro>     SUMDAY   OVNM     (SUMMARY display)                      OVLNUM.479
2664    <macro>     SUMHOW   OVNM     (Summary control routine)              OVLNUM.480
2670    <macro>     SUMTIME  OVNM     (Summary tallying)                     OVLNUM.481
2674    <macro>     SWAP     OVNM     (Swap Space Statistics)                OVLNUM.482
2700    <macro>     SYNTAX   OVNM     (Command syntax descriptors)           OVLNUM.483
2704    <macro>     SYSTAT   OVNM     (STATUS display)                       OVLNUM.484
2710    <macro>     TAPEC    OVNM     (TAPE display)                         OVLNUM.485
2714    <macro>     TEC455   OVNM     (Conversion for TEC console)           OVLNUM.486
2720    <macro>     TJOB     OVNM     (TJOB display)                         OVLNUM.487
2724    <macro>     TKSTAT   OVNM     (STP display)                          OVLNUM.488
2730    <macro>     TURNPG   OVNM     (Rotate print page)                    OVLNUM.489
2734    <macro>     UNBLK    OVNM     (Dataset deblocking)                   OVLNUM.490
2740    <macro>     UPDATE   OVNM     (Update stream state)                  OVLNUM.491
2744    <macro>     XFRMT    OVNM     (Debug display)                        OVLNUM.492
2750    <macro>     XJSTAT   OVNM     (JSTAT display)                        OVLNUM.493
2754    <macro>     XMPXP    OVNM     (Debug display)                        OVLNUM.494
2760    <macro>     XMP4XP   OVNM    (Debug display for XMP-4)               OVLNUM.495

                    ***********************************************************  OVLNUM.497
                    *                                                         *  OVLNUM.498
                    *        $INTER    Interactive station overlays           *  OVLNUM.499
                    *                                                         *  OVLNUM.500
                    ***********************************************************  OVLNUM.501
                                                                               OVLNUM.502
2764    <macro>     IACMD    OVNM     (Command processor)                    OVLNUM.503
2770    <macro>     IACON    OVNM     (Console initialization)               OVLNUM.504
2774    <macro>     IACON1   OVNM     (Console input)                        OVLNUM.505
3000    <macro>     IAFUNC   OVNM     (Concentrator functions)               OVLNUM.506
3004    <macro>     IAIOP    OVNM     (Interactive station)                  OVLNUM.507
3010    <macro>     IAIOP1   OVNM     (Interactive concentrator)             OVLNUM.508
3014    <macro>     IAMSG    OVNM     (Message distibutor)                   OVLNUM.509
3020    <macro>     IAOUT    OVNM     (Console output)                       OVLNUM.510

                    ***********************************************************  OVLNUM.512
                    *                                                         *  OVLNUM.513
                    *        $NSC      NSC concentrator overlays              *  OVLNUM.514
```

```
                                 *                                                      *  OVLNUM.515
                                 ************************************************************  OVLNUM.516
                                                                                          OVLNUM.517
3024    <macro>                  FNSC     OVNM     (NSC N-packet interface)              OVLNUM.518
3030    <macro>                  NIDEND   OVNM     (NSC concentrator termination)        OVLNUM.519
3034    <macro>                  NSC      OVNM     (NSC initialization)                  OVLNUM.520
3040    <macro>                  NSCEND   OVNM     (NSC termination)                     OVLNUM.521
3044    <macro>                  NSCID    OVNM     (NSC logical ID processor)            OVLNUM.522
3050    <macro>                  NSCIO    OVNM     (NSC I/O control)                     OVLNUM.523
3054    <macro>                  NSCMSG   OVNM     (NSC message handler)               R16552CA.1
3060    <macro>                  NSCRW    OVNM     (NSC R/W Handler)                     OVLNUM.524
3064    <macro>                  SCPIO    OVNM     (SCP Protocol Handler)                OVLNUM.525
3070    <macro>                  TERMNSC  OVNM     (N-packet interface termination)      OVLNUM.529
                                                                                          OVLNUM.530
                                 ************************************************************  OVLNUM.531
                                 *                                                      *  OVLNUM.532
                                 *        $UCHN    User channel overlays                 *  OVLNUM.533
                                 *                                                      *  OVLNUM.534
                                 ************************************************************  OVLNUM.535
                                                                                          OVLNUM.536
3074    <macro>                  UCSHL    OVNM     (User channel Shell routine)          OVLNUM.537
3100    <macro>                  UCOPN    OVNM     (User channel Open routine)           OVLNUM.538
3104    <macro>                  UCCLS    OVNM     (User channel Close routine)          OVLNUM.539
3110    <macro>                  UCRD     OVNM     (User channel Read routine)           OVLNUM.540
3114    <macro>                  UCWRT    OVNM     (User channel Write routine)          OVLNUM.541
3120    <macro>                  UCDRV    OVNM     (User channel Driver routine)         OVLNUM.542
3124    <macro>                  UCXFR    OVNM     (User channel Data Transfer)          OVLNUM.543
                                                                                          OVLNUM.544
                                 *        User channel drivers                            OVLNUM.545
                                                                                          OVLNUM.546
3130    <macro>                  ISPDRV   OVNM     (ISP channel driver)                  OVLNUM.547
3134    <macro>                  SL2DRV   OVNM     (Superlink FEI driver)                OVLNUM.548
3140    <macro>                  UCDRV1   OVNM     (User channel dummy driver 1)         OVLNUM.549
3144    <macro>                  UCDRV2   OVNM     (User channel dummy driver 2)         OVLNUM.550
3150    <macro>                  UCDRV3   OVNM     (User channel dummy driver 3)         OVLNUM.551
3154    <macro>                  UCDRV4   OVNM     (User channel dummy driver 4)         OVLNUM.552
3160    <macro>                  UCDRV5   OVNM     (User channel dummy driver 5)         OVLNUM.553
3164    <macro>                  UCDRV6   OVNM     (User channel dummy driver 6)         OVLNUM.554
3170    <macro>                  UCDRV7   OVNM     (User channel dummy driver 7)         OVLNUM.555
3174    <macro>                  UCDRV8   OVNM     (User channel dummy driver 8)         OVLNUM.556
3200    <macro>                  UCDRV9   OVNM     (User channel dummy driver 9)         OVLNUM.557
                                                                                          OVLNUM.558
```

```
                                 ***************************************************************  OVLNUM.559
                                 *                                                              *  OVLNUM.560
                                 *        $UNIX     Unix overlays                               *  OVLNUM.561
                                 *                                                              *  OVLNUM.562
                                 ****************************************************************  OVLNUM.563
                                                                                                   OVLNUM.564
3204    <macro>                  ASD       OVNM      (ASD console driver initiator)             R17260AB.1
3210    <macro>                  ASDINP    OVNM      (ASD console input driver)                 R17260AB.2
3214    <macro>                  ASDOUT    OVNM      (ASD console output driver)                R17260AB.3
3220    <macro>                  FSCOPY    OVNM      (Copy expander pack partition)              OVLNUM.565
3224    <macro>                  PART      OVNM      (Expander disk partition allocation)        OVLNUM.566
3230    <macro>                  TTKEY     OVNM      (TTY keyboard driver)                       OVLNUM.567
3234    <macro>                  TTY       OVNM      (TTY kernel command)                        OVLNUM.568
3240    <macro>                  TTYIN     OVNM      (TTY input, IOS to Cray)                    OVLNUM.569
3244    <macro>                  TTYNIT    OVNM      (T-packet NSC driver initialization)        OVLNUM.570
3250    <macro>                  TTYNSC    OVNM      (T-packet NSC driver)                       OVLNUM.571
3254    <macro>                  TTYOUT    OVNM      (TTY output, Cray to IOS)                   OVLNUM.572
3260    <macro>                  UDISK     OVNM      (Disk driver)                               OVLNUM.573
3264    <macro>                  UFILE     OVNM      (Expander disk file i/o)                    OVLNUM.574
3270    <macro>                  UPRINT    OVNM      (Printer driver)                            OVLNUM.575
3274    <macro>                  UPRT1     OVNM      (Print output)                              OVLNUM.576
3300    <macro>                  UPRT2     OVNM      (Rotated print output)                      OVLNUM.577
3304    <macro>                  UPRT3     OVNM      (Plot output)                               OVLNUM.578
3310    <macro>                  UTAPE     OVNM      (Tape driver)                               OVLNUM.579
                                                                                                   OVLNUM.580
3314    <macro>                  MAX       OVNM      (),NULL=OK          .Maximum overlay number OVLNUM.581

                                           ERRIF     O$MAX,GT,OVLTBL     .Overlay table overflow OVLNUM.583
```

```
                    ********************************************************************  OVLNUM.585
                    *                                                                  *  OVLNUM.586
                    *        COMPARS   Define command parameters                       *  OVLNUM.587
                    *                                                                  *  OVLNUM.588
                    * ovl    COMPARS   (cmd)                                           *  OVLNUM.589
                    *                                                                  *  OVLNUM.590
                    *        ovl       Overlay (COMMnn) which processes the list of com-*  OVLNUM.591
                    *                  mands.  CV$cmd is defined for each entry in the *  OVLNUM.592
                    *                  list.                                           *  OVLNUM.593
                    *                                                                  *  OVLNUM.594
                    *        cmd       List of commands handled by the overlay.  Each  *  OVLNUM.595
                    *                  command must be truncated to five characters if *  OVLNUM.596
                    *                  necessary.  A unique value CP$cmd is assigned to *  OVLNUM.597
                    *                  each entry in the list (The first entry is assigned *  OVLNUM.598
                    *                  a value of zero).  Null entries are ignored.    *  OVLNUM.599
                    *                                                                  *  OVLNUM.600
                    *        The station overlay CLI uses the value CV$cmd to call the *  OVLNUM.601
                    *        overlay which processes a particular command.  The overlays *  OVLNUM.602
                    *        (COMMnn) use the values CP$cmd to determine which command is *  OVLNUM.603
                    *        being processed if it handles more than one.              *  OVLNUM.604
                    *                                                                  *  OVLNUM.605
                    ********************************************************************  OVLNUM.606
                                                                                          OVLNUM.607
                             MACRO                                                        OVLNUM.608
<prototype>         OVL      COMPARS   CMD                                               OVLNUM.609
<definition>                 QUAL                                                        OVLNUM.610
<definition>                 NEXT      -1                                                OVLNUM.611
<definition>        $ECHO    ECHO      PAR=(CMD)                                         OVLNUM.612
<definition>                 IFC       !_PAR_!,NE,,4                                     OVLNUM.613
<definition>                 ENTRY     CP$_PAR                                           OVLNUM.614
<definition>                 ENTRY     CV$_PAR                                           OVLNUM.615
<definition>        CP$_PAR  NEXT                                                        OVLNUM.616
<definition>        CV$_PAR  EQUALS    O$_OVL                                            OVLNUM.617
<definition>        $ECHO    ENDDUP                                                      OVLNUM.618
<definition>                 QUAL      *                                                 OVLNUM.619
                    COMPARS  ENDM                                                        OVLNUM.620


3320  <macro>       COMM01   COMPARS   (CLEAR,DEBUG,ERROR,LINK,SCROL,STATI,STATU,STORA,  OVLNUM.622
                    ,                  PERIO,MONIT,BABEL,STATC,TAPE,TJOB,DKSTA,SUMMA,STP, OVLNUM.623
```

```
                                                XJST,STK    LP,RSTAT,CNC,DSPL,LPATH,SWAP,DKERR)        OVLNUM.624
                                                                                                       OVLNUM.625
3320    <macro>         COMM02    COMPARS   (END,LOGOF,LOGON,IACON)                                   OVLNUM.626
                                                                                                       OVLNUM.627
3320    <macro>         COMM03    COMPARS   (PLUS,MINUS,GREAT,LESS,INC,DEC)                          OVLNUM.628
                                                                                                       OVLNUM.629
3320    <macro>         COMM04    COMPARS   (COMME,CONSO,DELAY,POLL,REFRE,SET,SNAP,STAGE,PAUSE)      OVLNUM.630
                                                                                                       OVLNUM.631
3320    <macro>         COMM05    COMPARS   (ESIGN)                                                  OVLNUM.632
                                                                                                       OVLNUM.633
3320    <macro>         COMM06    COMPARS   (DROP,KILL,RECOV,RELEA,RERUN,RESUM,SHUTD,SUSPE)          OVLNUM.634
                                                                                                       OVLNUM.635
3320    <macro>         COMM07    COMPARS   (CHANN,DISCO,LIMIT,RESTO,ROUTE,STREA,SWEEP,SWITC)        OVLNUM.636
                                                                                                       OVLNUM.637
3320    <macro>         COMM08    COMPARS   (CLASS,ENTER,FLUSH,OPERA)                                OVLNUM.638
                                                                                                       OVLNUM.639
3320    <macro>         COMM09    COMPARS   (ASSIG,MODE,ALTER,DISPL,BREAK,REMOV,RUN,STOP,INITI)      OVLNUM.640
                                                                                                       OVLNUM.641
3320    <macro>         COMM10    COMPARS   (MESSA,REPLY,CONFI,MSG)                                  OVLNUM.642
                                                                                                       OVLNUM.643
3320    <macro>         COMM11    COMPARS   (DATAS)                                                  OVLNUM.644
                                                                                                       OVLNUM.645
3320    <macro>         COMM12    COMPARS   (JOB)                                                    OVLNUM.646
                                                                                                       OVLNUM.647
3320    <macro>         COMM13    COMPARS   (SAVE,SUBMI)                                             OVLNUM.648
                                                                                                       OVLNUM.649
3320    <macro>         COMM14    COMPARS   (ATSIG,DEFAU)                                            OVLNUM.650
                                                                                                       OVLNUM.651
3320    <macro>         COMM15    COMPARS   (NSC)                                                    OVLNUM.652
                                                                                                       OVLNUM.653
3320    <macro>         COMM16    COMPARS   (TLINK,UNLOA)                                            OVLNUM.654
```

```
                    ***********************************************************************  OVLNUM.656
                    *                                                                     *  OVLNUM.657
                    *        DISPARS   Define display parameters                          *  OVLNUM.658
                    *                                                                     *  OVLNUM.659
                    * ovl    DISPARS   (disp)                                             *  OVLNUM.660
                    *                                                                     *  OVLNUM.661
                    *        ovl       Overlay (DISPnn) which controls generation of the  *  OVLNUM.662
                    *                  displays listed.  The value DV$disp is defined for *  OVLNUM.663
                    *                  each entry in the list.                            *  OVLNUM.664
                    *                                                                     *  OVLNUM.665
                    *        disp      List of displays handled by the overlay.  Each     *  OVLNUM.666
                    *                  entry is limited to five characters.  A unique     *  OVLNUM.667
                    *                  value DP$disp is assigned to each entry in the     *  OVLNUM.668
                    *                  list (The first entry is assigned a value of zero). *  OVLNUM.669
                    *                  Null entries are ignored.                          *  OVLNUM.670
                    *                                                                     *  OVLNUM.671
                    *        The station command overlays (COMMnn) use the values DV$disp *  OVLNUM.672
                    *        and DP$disp to activate a particular display.  The display   *  OVLNUM.673
                    *        overlay DISPLAY uses the value DV$disp value to call the     *  OVLNUM.674
                    *        overlay (DISPnn) which controls the display generation.      *  OVLNUM.675
                    *        The overlay DISPnn uses the value DP$disp to select which    *  OVLNUM.676
                    *        display is being processed.                                  *  OVLNUM.677
                    *                                                                     *  OVLNUM.678
                    ***********************************************************************  OVLNUM.679
                                                                                             OVLNUM.680
                             MACRO                                                           OVLNUM.681
<prototype>         OVL      DISPARS   DISP                                                  OVLNUM.682
<definition>                 QUAL                                                            OVLNUM.683
<definition>                 NEXT      -1                                                    OVLNUM.684
<definition>        $ECHO    ECHO      PAR=(DISP)                                            OVLNUM.685
<definition>                 IFC       !_PAR_!,NE,,4                                         OVLNUM.686
<definition>                 ENTRY     DP$_PAR                                               OVLNUM.687
<definition>                 ENTRY     DV$_PAR                                               OVLNUM.688
<definition>        DP$_PAR  NEXT                                                            OVLNUM.689
<definition>        DV$_PAR  EQUALS    O$_OVL                                                OVLNUM.690
<definition>        $ECHO    ENDDUP                                                          OVLNUM.691
<definition>                 QUAL      *                                                     OVLNUM.692
                    DISPARS  ENDM                                                            OVLNUM.693
```

```
3320   <macro>   DISP01   DISPARS   (DEBUG,[illegible]R,LINK,STATI,STATU,STORA,MONIT,BABEL,      OVLNUM.69
                 ,                   STATC,TAPE,TJOB,DKSTA,SUMMA,STP,XJST,STRS,HELP,          OVLNUM.696
                 ,                   RSTAT,CNC,NSC,DSPL,NSCLP,SWAP,DKERR)                     OVLNUM.697
                                                                                              OVLNUM.698
3320   <macro>   DISP02   DISPARS   (SPLIT)                                                   OVLNUM.699

                          END                                                                 OVLNUM.701
```

1	$NEXT		18: 1 D	18: 1	18: 3 D	18: 3	18: 5 D	18: 5	18: 7 D	18: 7
			18: 9 D	18: 9	18:11 D	18:11	18:13 D	18:13	18:15 D	18:15
			18:17 D	18:17	18:19 D	18:19	18:21 D	18:21	18:23 D	18:23
			18:25 D	18:25	18:27 D	18:27	18:29 D	18:29	18:31 D	18:31
			20: 3 D	20: 3	20: 5 D	20: 5				
5	C$MECHK	$APTEXT	3:24 F							
0	C$SDMPA	$APTEXT	3:19 F							
1	C$SDMPB	$APTEXT	3:20 F							
2	C$SDMPC	$APTEXT	3:21 F							
3	C$SDMPD	$APTEXT	3:22 F							
4	C$SDMPE	$APTEXT	3:23 F							
2	CP$ALTER		18:17 E	18:17 D						
0	CP$ASSIG		18:17 E	18:17 D						
0	CP$ATSIG		18:27 E	18:27 D						
12	CP$BABEL		18: 1 E	18: 1 D						
4	CP$BREAK		18:17 E	18:17 D						
0	CP$CHANN		18:13 E	18:13 D						
0	CP$CLASS		18:15 E	18:15 D						
0	CP$CLEAR		18: 1 E	18: 1 D						
25	CP$CNC		18: 1 E	18: 1 D						
0	CP$COMME		18: 7 E	18: 7 D						
2	CP$CONFI		18:19 E	18:19 D						
1	CP$CONSO		18: 7 E	18: 7 D						
0	CP$DATAS		18:21 E	18:21 D						
1	CP$DEBUG		18: 1 E	18: 1 D						
5	CP$DEC		18: 5 E	18: 5 D						
1	CP$DEFAU		18:27 E	18:27 D						
2	CP$DELAY		18: 7 E	18: 7 D						
1	CP$DISCO		18:13 E	18:13 D						
3	CP$DISPL		18:17 E	18:17 D						
31	CP$DKERR		18: 1 E	18: 1 D						
16	CP$DKSTA		18: 1 E	18: 1 D						
0	CP$DROP		18:11 E	18:11 D						
26	CP$DSPL		18: 1 E	18: 1 D						
0	CP$END		18: 3 E	18: 3 D						
1	CP$ENTER		18:15 E	18:15 D						
2	CP$ERROR		18: 1 E	18: 1 D						
0	CP$ESIGN		18: 9 E	18: 9 D						
2	CP$FLUSH		18:15 E	18:15 D						
2	CP$GREAT		18: 5 E	18: 5 D						
23	CP$HELP		18: 1 E	18: 1 D						
3	CP$IACON		18: 3 E	18: 3 D						

4	CP$INC	18: 5 E	18: 5 D
10	CP$INITI	18:17 E	18:17 D
0	CP$JOB	18:23 E	18:23 D
1	CP$KILL	18:11 E	18:11 D
3	CP$LESS	18: 5 E	18: 5 D
2	CP$LIMIT	18:13 E	18:13 D
3	CP$LINK	18: 1 E	18: 1 D
1	CP$LOGOF	18: 3 E	18: 3 D
2	CP$LOGON	18: 3 E	18: 3 D
27	CP$LPATH	18: 1 E	18: 1 D
0	CP$MESSA	18:19 E	18:19 D
1	CP$MINUS	18: 5 E	18: 5 D
1	CP$MODE	18:17 E	18:17 D
11	CP$MONIT	18: 1 E	18: 1 D
3	CP$MSG	18:19 E	18:19 D
0	CP$NSC	18:29 E	18:29 D
3	CP$OPERA	18:15 E	18:15 D
10	CP$PAUSE	18: 7 E	18: 7 D
10	CP$PERIO	18: 1 E	18: 1 D
0	CP$PLUS	18: 5 E	18: 5 D
3	CP$POLL	18: 7 E	18: 7 D
2	CP$RECOV	18:11 E	18:11 D
4	CP$RETRE	18: 7 E	18: 7 D
3	CP$RELEA	18:11 E	18:11 D
5	CP$REMOV	18:17 E	18:17 D
1	CP$REPLY	18:19 E	18:19 D
4	CP$RERUN	18:11 E	18:11 D
3	CP$RESTO	18:13 E	18:13 D
5	CP$RESUM	18:11 E	18:11 D
4	CP$ROUTE	18:13 E	18:13 D
24	CP$RSTAT	18: 1 E	18: 1 D
6	CP$RUN	18:17 E	18:17 D
0	CP$SAVE	18:25 E	18:25 D
4	CP$SCROL	18: 1 E	18: 1 D
5	CP$SET	18: 7 E	18: 7 D
6	CP$SHUTD	18:11 E	18:11 D
6	CP$SNAP	18: 7 E	18: 7 D
7	CP$STAGE	18: 7 E	18: 7 D
13	CP$STATC	18: 1 E	18: 1 D
5	CP$STATI	18: 1 E	18: 1 D
6	CP$STATU	18: 1 E	18: 1 D
7	CP$STOP	18:17 E	18:17 D

7	CP$STORA	18: 1 E	18: 1 D
20	CP$STP	18: 1 E	18: 1 D
5	CP$STREA	18:13 E	18:13 D
22	CP$STRS	18: 1 E	18: 1 D
1	CP$SUBMI	18:25 E	18:25 D
17	CP$SUMMA	18: 1 E	18: 1 D
7	CP$SUSPE	18:11 E	18:11 D
30	CP$SWAP	18: 1 E	18: 1 D
6	CP$SWEEP	18:13 E	18:13 D
7	CP$SWITC	18:13 E	18:13 D
14	CP$TAPE	18: 1 E	18: 1 D
15	CP$TJOB	18: 1 E	18: 1 D
0	CP$TLINK	18:31 E	18:31 D
1	CP$UNLOA	18:31 E	18:31 D
21	CP$XJST	18: 1 E	18: 1 D
423	CV$ALTER	18:17 E	18:17 D
423	CV$ASSIG	18:17 E	18:17 D
430	CV$ATSIG	18:27 E	18:27 D
413	CV$BABEL	18: 1 E	18: 1 D
423	CV$BREAK	18:17 E	18:17 D
421	CV$CHANN	18:13 E	18:13 D
422	CV$CLASS	18:15 E	18:15 D
413	CV$CLEAR	18: 1 E	18: 1 D
413	CV$CNC	18: 1 E	18: 1 D
416	CV$COMME	18: 7 E	18: 7 D
424	CV$CONFI	18:19 E	18:19 D
416	CV$CONSO	18: 7 E	18: 7 D
425	CV$DATAS	18:21 E	18:21 D
413	CV$DEBUG	18: 1 E	18: 1 D
415	CV$DEC	18: 5 E	18: 5 D
430	CV$DEFAU	18:27 E	18:27 D
416	CV$DELAY	18: 7 E	18: 7 D
421	CV$DISCO	18:13 E	18:13 D
423	CV$DISPL	18:17 E	18:17 D
413	CV$DKERR	18: 1 E	18: 1 D
413	CV$DKSTA	18: 1 E	18: 1 D
420	CV$DROP	18:11 E	18:11 D
413	CV$DSPL	18: 1 E	18: 1 D
414	CV$END	18: 3 E	18: 3 D
422	CV$ENTER	18:15 E	18:15 D
413	CV$ERROR	18: 1 E	18: 1 D
417	CV$ESIGN	18: 9 E	18: 9 D

422	CV$FLUSH	18:15 E	18:15 D
415	CV$GREAT	18: 5 E	18: 5 D
413	CV$HELP	18: 1 E	18: 1 D
414	CV$IACON	18: 3 E	18: 3 D
415	CV$INC	18: 5 E	18: 5 D
423	CV$INITI	18:17 E	18:17 D
426	CV$JOB	18:23 E	18:23 D
420	CV$KILL	18:11 E	18:11 D
415	CV$LESS	18: 5 E	18: 5 D
421	CV$LIMIT	18:13 E	18:13 D
413	CV$LINK	18: 1 E	18: 1 D
414	CV$LOGOF	18: 3 E	18: 3 D
414	CV$LOGON	18: 3 E	18: 3 D
413	CV$LPATH	18: 1 E	18: 1 D
424	CV$MESSA	18:19 E	18:19 D
415	CV$MINUS	18: 5 E	18: 5 D
423	CV$MODE	18:17 E	18:17 D
413	CV$MONIT	18: 1 E	18: 1 D
424	CV$MSG	18:19 E	18:19 D
431	CV$NSC	18:29 E	18:29 D
422	CV$OPERA	18:15 E	18:15 D
416	CV$PAUSE	18: 7 E	18: 7 D
413	CV$PERIO	18: 1 E	18: 1 D
415	CV$PLUS	18: 5 E	18: 5 D
416	CV$POLL	18: 7 E	18: 7 D
420	CV$RECOV	18:11 E	18:11 D
416	CV$RETRE	18: 7 E	18: 7 D
420	CV$RELEA	18:11 E	18:11 D
423	CV$REMOV	18:17 E	18:17 D
424	CV$REPLY	18:19 E	18:19 D
420	CV$RERUN	18:11 E	18:11 D
421	CV$RESTO	18:13 E	18:13 D
420	CV$RESUM	18:11 E	18:11 D
421	CV$ROUTE	18:13 E	18:13 D
413	CV$RSTAT	18: 1 E	18: 1 D
423	CV$RUN	18:17 E	18:17 D
427	CV$SAVE	18:25 E	18:25 D
413	CV$SCROL	18: 1 E	18: 1 D
416	CV$SET	18: 7 E	18: 7 D
420	CV$SHUTD	18:11 E	18:11 D
416	CV$SNAP	18: 7 E	18: 7 D
416	CV$STAGE	18: 7 E	18: 7 D

413	CV$STATC	18: 1 E	18: 1 D
413	CV$STATI	18: 1 E	18: 1 D
413	CV$STATU	18: 1 E	18: 1 D
423	CV$STOP	18:17 E	18:17 D
413	CV$STORA	18: 1 E	18: 1 D
413	CV$STP	18: 1 E	18: 1 D
421	CV$STREA	18:13 E	18:13 D
413	CV$STRS	18: 1 E	18: 1 D
427	CV$SUBMI	18:25 E	18:25 D
413	CV$SUMMA	18: 1 E	18: 1 D
420	CV$SUSPE	18:11 E	18:11 D
413	CV$SWAP	18: 1 E	18: 1 D
421	CV$SWEEP	18:13 E	18:13 D
421	CV$SWITC	18:13 E	18:13 D
413	CV$TAPE	18: 1 E	18: 1 D
413	CV$TJOB	18: 1 E	18: 1 D
432	CV$TLINK	18:31 E	18:31 D
432	CV$UNLOA	18:31 E	18:31 D
413	CV$XJST	18: 1 E	18: 1 D
7	DP$BABEL	20: 3 E	20: 3 D
22	DP$CNC	20: 3 E	20: 3 D
0	DP$DEBUG	20: 3 E	20: 3 D
27	DP$DKERR	20: 3 E	20: 3 D
13	DP$DKSTA	20: 3 E	20: 3 D
24	DP$DSPL	20: 3 E	20: 3 D
1	DP$ERROR	20: 3 E	20: 3 D
20	DP$HELP	20: 3 E	20: 3 D
2	DP$LINK	20: 3 E	20: 3 D
6	DP$MONIT	20: 3 E	20: 3 D
23	DP$NSC	20: 3 E	20: 3 D
25	DP$NSCLP	20: 3 E	20: 3 D
21	DP$RSTAT	20: 3 E	20: 3 D
0	DP$SPLIT	20: 5 E	20: 5 D
10	DP$STATC	20: 3 E	20: 3 D
3	DP$STATI	20: 3 E	20: 3 D
4	DP$STATU	20: 3 E	20: 3 D
5	DP$STORA	20: 3 E	20: 3 D
15	DP$STP	20: 3 E	20: 3 D
17	DP$STRS	20: 3 E	20: 3 D
14	DP$SUMMA	20: 3 E	20: 3 D
26	DP$SWAP	20: 3 E	20: 3 D
11	DP$TAPE	20: 3 E	20: 3 D

12	DP$TJOB	20: 3 E	20: 3 D
16	DP$XJST	20: 3 E	20: 3 D
444	DV$BABEL	20: 3 E	20: 3 D
444	DV$CNC	20: 3 E	20: 3 D
444	DV$DEBUG	20: 3 E	20: 3 D
444	DV$DKERR	20: 3 E	20: 3 D
444	DV$DKSTA	20: 3 E	20: 3 D
444	DV$DSPL	20: 3 E	20: 3 D
444	DV$ERROR	20: 3 E	20: 3 D
444	DV$HELP	20: 3 E	20: 3 D
444	DV$LINK	20: 3 E	20: 3 D
444	DV$MONIT	20: 3 E	20: 3 D
444	DV$NSC	20: 3 E	20: 3 D
444	DV$NSCLP	20: 3 E	20: 3 D
444	DV$RSTAT	20: 3 E	20: 3 D
445	DV$SPLIT	20: 5 E	20: 5 D
444	DV$STATC	20: 3 E	20: 3 D
444	DV$STATI	20: 3 E	20: 3 D
444	DV$STATU	20: 3 E	20: 3 D
444	DV$STORA	20: 3 E	20: 3 D
444	DV$STP	20: 3 E	20: 3 D
444	DV$STRS	20: 3 E	20: 3 D
444	DV$SUMMA	20: 3 E	20: 3 D
444	DV$SWAP	20: 3 E	20: 3 D
444	DV$TAPE	20: 3 E	20: 3 D
444	DV$TJOB	20: 3 E	20: 3 D
444	DV$XJST	20: 3 E	20: 3 D
75	O$ACOM	5:12 E	5:12 D
371	O$ACQTRM	11:18 E	11:18 D
372	O$ACQUIR	11:19 E	11:19 D
7	O$ADEM	3:34 E	3:34 D
6	O$AMAP	3:33 E	3:33 D
373	O$AMPEX	11:20 E	11:20 D
10	O$AMSG	3:35 E	3:35 D
637	O$ASD	16: 7 E	16: 7 D
640	O$ASDINP	16: 8 E	16: 8 D
641	O$ASDOUT	16: 9 E	16: 9 D
374	O$BABEL	11:21 E	11:21 D
76	O$BADDAT	5:13 E	5:13 D
375	O$BARDAT	11:22 E	11:22 D
144	O$BCOM	6:25 E	6:25 D
11	O$BEGIN	3:36 E	3:36 D

376	O$BLOCK	11:23 E	11:23 D	
377	O$BMAGET	11:24 E	11:24 D	
400	O$BMGET	11:25 E	11:25 D	
135	O$BMXAIO	6:10 E	6:10 D	
136	O$BMXCON	6:11 E	6:11 D	
137	O$BMXCPU	6:12 E	6:12 D	
140	O$BMXDEM	6:13 E	6:13 D	
141	O$BMXOPE	6:14 E	6:14 D	
142	O$BMXSIO	6:15 E	6:15 D	
143	O$BMXTPO	6:16 E	6:16 D	
12	O$BTD	3:37 E	3:37 D	
13	O$BTH	3:38 E	3:38 D	
14	O$BTO	3:39 E	3:39 D	
145	O$BUFMAN	6:26 E	6:26 D	
401	O$BXDIS	11:26 E	11:26 D	
402	O$BXSET	11:27 E	11:27 D	
146	O$BYPASS	6:27 E	6:27 D	
15	O$CALL	3:40 E	3:40 D	
16	O$CDEM	3:41 E	3:41 D	
403	O$CFINIT	11:28 E	11:28 D	
404	O$CFREAD	11:29 E	11:29 D	
222	O$CHNTES	8: 3 E	8: 3 D	
253	O$CLEAR	9: 2 E	9: 2 D	
254	O$CLEARO	9: 3 E	9: 3 D	
405	O$CLI	11:30 E	11:30 D	
406	O$CLINIT	11:31 E	11:31 D	
17	O$CLKSNC	3:42 E	3:42 D	
20	O$CLOCK	4: 1 E	4: 1 D	
407	O$CLUSTR	11:32 E	11:32 D	
410	O$CMGET	11:33 E	11:33 D	
411	O$CNCDIS	11:34 E	11:34 D	
412	O$COMBO	11:35 E	11:35 D	
413	O$COMM01	11:36 E	11:36 D	18: 1
414	O$COMM02	11:37 E	11:37 D	18: 3
415	O$COMM03	11:38 E	11:38 D	18: 5
416	O$COMM04	11:39 E	11:39 D	18: 7
417	O$COMM05	11:40 E	11:40 D	18: 9
420	O$COMM06	11:41 E	11:41 D	18:11
421	O$COMM07	11:42 E	11:42 D	18:13
422	O$COMM08	12: 1 E	12: 1 D	18:15
423	O$COMM09	12: 2 E	12: 2 D	18:17
424	O$COMM10	12: 3 E	12: 3 D	18:19

425	O$COMM11	12: 4 E	12: 4 D	
426	O$COMM12	12: 5 E	12: 5 D	18:23
427	O$COMM13	12: 6 E	12: 6 D	18:25
430	O$COMM14	12: 7 E	12: 7 D	18:27
431	O$COMM15	12: 8 E	12: 8 D	18:29
432	O$COMM16	12: 9 E	12: 9 D	18:31
357	O$CONC	10:36 E	10:36 D	
360	O$CONCER	10:37 E	10:37 D	
361	O$CONCID	10:38 E	10:38 D	
362	O$CONCIO	10:39 E	10:39 D	
21	O$CONFIG	4: 2 E	4: 2 D	
147	O$CONMAN	6:28 E	6:28 D	
433	O$CONSL	12:10 E	12:10 D	
255	O$COPY	9: 4 E	9: 4 D	
256	O$COPY0	9: 5 E	9: 5 D	
257	O$COPY1	9: 6 E	9: 6 D	
260	O$COPY2	9: 7 E	9: 7 D	
261	O$COPY3	9: 8 E	9: 8 D	
262	O$COPY4	9: 9 E	9: 9 D	
263	O$COPY5	9:10 E	9:10 D	
223	O$CPSPIN	8: 4 E	8: 4 D	
224	O$CPTEST	8: 5 E	8: 5 D	
434	O$CPUGET	12:11 E	12:11 D	
22	O$CRAY.	4: 3 E	4: 3 D	
435	O$CRAY10	12:12 E	12:12 D	
23	O$CRTDEM	4: 4 E	4: 4 D	
77	O$D3ECC	5:14 E	5:14 D	
100	O$D3ERR	5:15 E	5:15 D	
101	O$D3IOR	5:16 E	5:16 D	
102	O$D3LOG	5:17 E	5:17 D	
103	O$D3MSG	5:18 E	5:18 D	
104	O$D3RLR	5:19 E	5:19 D	
105	O$D3SKR	5:20 E	5:20 D	
106	O$D3SLR	5:21 E	5:21 D	
107	O$D40ECC	5:22 E	5:22 D	
110	O$D40ERR	5:23 E	5:23 D	
111	O$D40IOR	5:24 E	5:24 D	
112	O$D4DEM	5:25 E	5:25 D	
113	O$D4ECC	5:26 E	5:26 D	
114	O$D4ERR	5:27 E	5:27 D	
115	O$D4IOR	5:28 E	5:28 D	
116	O$D4LOG	5:29 E	5:29 D	

117	O$D4MSG	5:30	E	5:30	D	
120	O$D4RES	5:31	E	5:31	D	
121	O$D4RLR	5:32	E	5:32	D	
122	O$D4SKR	5:33	E	5:33	D	
123	O$D4SLR	5:34	E	5:34	D	
124	O$D4STAT	5:35	E	5:35	D	
436	O$DBGET	12:13	E	12:13	D	
125	O$DD49	5:36	E	5:36	D	
126	O$DDIAG	5:37	E	5:37	D	
264	O$DDUMP	9:11	E	9:11	D	
265	O$DDUMP0	9:12	E	9:12	D	
437	O$DECOD2	12:14	E	12:14	D	
440	O$DECODE	12:15	E	12:15	D	
266	O$DEF	9:13	E	9:13	D	
267	O$DELETO	9:14	E	9:14	D	
270	O$DELETE	9:15	E	9:15	D	
441	O$DELMSG	12:16	E	12:16	D	
442	O$DESCRI	12:17	E	12:17	D	
443	O$DEVDAT	12:18	E	12:18	D	
127	O$DISK	5:38	E	5:38	D	
24	O$DISKIO	4: 5	E	4: 5	D	
444	O$DISP01	12:19	E	12:19	D	20: 3
445	O$DISP02	12:20	E	12:20	D	20: 5
446	O$DISPLA	12:21	E	12:21	D	
25	O$DIVIDE	4: 6	E	4: 6	D	
447	O$DKDIS	12:22	E	12:22	D	
26	O$DKDMP	4: 7	E	4: 7	D	
450	O$DKERR	12:23	E	12:23	D	
451	O$DKERR1	12:24	E	12:24	D	
27	O$DKIOEX	4: 8	E	4: 8	D	
30	O$DKLOOK	4: 9	E	4: 9	D	
452	O$DKSET	12:25	E	12:25	D	
453	O$DKSETO	12:26	E	12:26	D	
454	O$DKSTAT	12:27	E	12:27	D	
271	O$DLOAD	9:16	E	9:16	D	
272	O$DLOAD0	9:17	E	9:17	D	
150	O$DSCGET	6:29	E	6:29	D	
455	O$DSPL	12:28	E	12:28	D	
273	O$DSTAT	9:18	E	9:18	D	
274	O$DSTAT0	9:19	E	9:19	D	
31	O$DTB	4:10	E	4:10	D	
225	O$ECHOCP	8: 6	E	8: 6	D	

275	O$EDDELE	9:20 E	9:20 D
276	O$EDINSI	9:21 E	9:21 D
277	O$EDIT	9:22 E	9:22 D
300	O$EDIT0	9:23 E	9:23 D
301	O$EDIT1	9:24 E	9:24 D
302	O$EDPRNT	9:25 E	9:25 D
303	O$EDREPL	9:26 E	9:26 D
304	O$EDTYPL	9:27 E	9:27 D
363	O$ENDCON	10:40 E	10:40 D
364	O$ENTRID	10:41 E	10:41 D
32	O$ERR	4:11 E	4:11 D
456	O$ERRDIS	12:29 E	12:29 D
33	O$ERRDMP	4:12 E	4:12 D
130	O$ERRECK	5:39 E	5:39 D
457	O$ERROR	12:30 E	12:30 D
461	O$F132	12:32 E	12:32 D
460	O$F80	12:31 E	12:31 D
305	O$FDUMP	9:28 E	9:28 D
306	O$FDUMPO	9:29 E	9:29 D
366	O$FEIMSG	11: 7 E	11: 7 D
367	O$FEIR	11: 8 E	11: 8 D
370	O$FEIW	11: 9 E	11: 9 D
34	O$FIND	4:13 E	4:13 D
131	O$FIRECO	5:40 E	5:40 D
307	O$FLAW	9:30 E	9:30 D
310	O$FLOAD	9:31 E	9:31 D
311	O$FLOADO	9:32 E	9:32 D
462	O$FMGET	12:33 E	12:33 D
603	O$FNSC	15: 4 E	15: 4 D
312	O$FORMAT	9:33 E	9:33 D
313	O$FORMTO	9:34 E	9:34 D
642	O$FSCOPY	16:10 E	16:10 D
314	O$FSTAT	9:35 E	9:35 D
315	O$FSTATO	9:36 E	9:36 D
35	O$GETEXT	4:14 E	4:14 D
463	O$GRAPH	12:34 E	12:34 D
464	O$GRCDIS	12:35 E	12:35 D
465	O$GRPDIS	12:36 E	12:36 D
466	O$GRUDIS	12:37 E	12:37 D
467	O$HDRPAG	12:38 E	12:38 D
470	O$HELP	12:39 E	12:39 D
471	O$HELP2	12:40 E	12:40 D

472	O$HELPD	12:41 E	12:41 D	
473	O$HPDATA	12:42 E	12:42 D	
474	O$HSPGET	13: 1 E	13: 1 D	
226	O$HSPTES	8: 7 E	8: 7 D	
573	O$IACMD	14:30 E	14:30 D	
574	O$IACON	14:31 E	14:31 D	
575	O$IACON1	14:32 E	14:32 D	
576	O$IAFUNC	14:33 E	14:33 D	
577	O$IAIOP	14:34 E	14:34 D	
600	O$IAIOP1	14:35 E	14:35 D	
601	O$IAMSG	14:36 E	14:36 D	
602	O$IAOUT	14:37 E	14:37 D	
132	O$ICOM	5:41 E	5:41 D	
475	O$ICONSL	13: 2 E	13: 2 D	
476	O$IDEBUG	13: 3 E	13: 3 D	
477	O$IDLGET	13: 4 E	13: 4 D	
500	O$IDRCT	13: 5 E	13: 5 D	
501	O$IFRMT	13: 6 E	13: 6 D	
37	O$INDD29	4:16 E	4:16 D	
40	O$INDD39	4:17 E	4:17 D	
41	O$INDD40	4:18 E	4:18 D	
42	O$INDD49	4:19 E	4:19 D	
43	O$INDD50	4:20 E	4:20 D	
36	O$INFORM	4:15 E	4:15 D	
316	O$INIT	9:37 E	9:37 D	
317	O$INITO	9:38 E	9:38 D	
624	O$ISPDRV	15:31 E	15:31 D	
502	O$KEYBD	13: 7 E	13: 7 D	
503	O$LCP	13: 8 E	13: 8 D	
320	O$LINGET	9:39 E	9:39 D	
504	O$LINK	13: 9 E	13: 9 D	
321	O$LINPUT	9:40 E	9:40 D	
44	O$LISTO	4:21 E	4:21 D	
45	O$LISTP	4:22 E	4:22 D	
505	O$LOGON	13:10 E	13:10 D	
661	[illegible]MAX	16:26 E	16:26 D	16:29 F
506	O$MESSAG	13:11 E	13:11 D	
5	O$MFCHK	3:24 E	3:24 D	3:24 F
46	O$MFCLR	4:23 E	4:23 D	
47	O$MFINIT	4:24 E	4:24 D	
227	O$MOSTES	8: 8 E	8: 8 D	
50	O$MSGHND	4:25 E	4:25 D	

507	O$MSTAT	13:12 E	13:12 D	
510	O$MSTDIS	13:13 E	13:13 D	
51	O$MULTIP	4:26 E	4:26 D	
511	O$NCNDIS	13:14 E	13:14 D	
512	O$NEWDIS	13:15 E	13:15 D	
604	O$NIDEND	15: 5 E	15: 5 D	
52	O$NOBEAT	4:27 E	4:27 D	
605	O$NSC	15: 6 E	15: 6 D	
606	O$NSCEND	15: 7 E	15: 7 D	
607	O$NSCID	15: 8 E	15: 8 D	
610	O$NSCIO	15: 9 E	15: 9 D	
513	O$NSCLP	13:16 E	13:16 D	
611	O$NSCMSG	15:10 E	15:10 D	
514	O$NSCNET	13:17 E	13:17 D	
612	O$NSCRW	15:11 E	15:11 D	
515	O$NSCSTP	13:18 E	13:18 D	
516	O$OBIT	13:19 E	13:19 D	
517	O$OFRMT	13:20 E	13:20 D	
520	O$ONLINE	13:21 E	13:21 D	
53	O$OPSYS	4:28 E	4:28 D	
54	O$OTB	4:29 E	4:29 D	
55	O$OUTCAL	4:30 E	4:30 D	
56	O$OVLNUM	1: 4	4:31 E	4:31 D
643	O$PART	16:11 E	16:11 D	
57	O$PATCH	4:32 E	4:32 D	
322	O$PLOT	9:41 E	9:41 D	
323	O$PLOTO	9:42 E	9:42 D	
324	O$PLOTT	10: 1 E	10: 1 D	
521	O$POST	13:22 E	13:22 D	
325	O$PROC	10: 2 E	10: 2 D	
326	O$PROCO	10: 3 E	10: 3 D	
522	O$PROTIN	13:23 E	13:23 D	
523	O$PROTOC	13:24 E	13:24 D	
327	O$PRTAPE	10: 4 E	10: 4 D	
524	O$QUEUE	13:25 E	13:25 D	
525	O$READ	13:26 E	13:26 D	
365	O$REMVID	10:42 E	10:42 D	
331	O$RENAMO	10: 6 E	10: 6 D	
330	O$RENAME	10: 5 E	10: 5 D	
133	O$REPORT	5:42 E	5:42 D	
205	O$RSTRT	7:24 E	7:24 D	
613	O$SCPIO	15:12 E	15:12 D	

60	O$SCRUB	4:33 E	4:33 D	
206	O$SDMP0	7:25 E	7:25 D	
207	O$SDMP1	7:26 E	7:26 D	
221	O$SDMP10	7:36 E	7:36 D	
210	O$SDMP2	7:27 E	7:27 D	
211	O$SDMP3	7:28 E	7:28 D	
212	O$SDMP3A	7:29 E	7:29 D	
213	O$SDMP4	7:30 E	7:30 D	
214	O$SDMP5	7:31 E	7:31 D	
215	O$SDMP6	7:32 E	7:32 D	
216	O$SDMP7	7:33 E	7:33 D	
217	O$SDMP8	7:34 E	7:34 D	
220	O$SDMP9	7:35 E	7:35 D	
0	O$SDMPA	3:19 E	3:19 D	3:19 F
1	O$SDMPB	3:20 E	3:20 D	3:20 F
2	O$SDMPC	3:21 E	3:21 D	3:21 F
3	O$SDMPD	3:22 E	3:22 D	3:22 F
4	O$SDMPE	3:23 E	3:23 D	3:23 F
61	O$SETIME	4:34 E	4:34 D	
625	O$SL2DRV	15:32 E	15:32 D	
526	O$SNAP	13:27 E	13:27 D	
527	O$SOROC	13:28 E	13:28 D	
530	O$SSDGRA	13:29 E	13:29 D	
230	O$SSDTES	8: 9 E	8: 9 D	
531	O$STADIS	13:30 E	13:30 D	
532	O$STAGEI	13:31 E	13:31 D	
533	O$STAGEO	13:32 E	13:32 D	
332	O$START	10: 7 E	10: 7 D	
333	O$START0	10: 8 E	10: 8 D	
334	O$START1	10: 9 E	10: 9 D	
335	O$START2	10:10 E	10:10 D	
336	O$START3	10:11 E	10:11 D	
534	O$STATCL	13:33 E	13:33 D	
535	O$STATIN	13:34 E	13:34 D	
536	O$STATIO	13:35 E	13:35 D	
62	O$STATS	4:35 E	4:35 D	
537	O$STIO	13:36 E	13:36 D	
540	O$STMSG	13:37 E	13:37 D	
231	O$STOP	8:10 E	8:10 D	
541	O$STPLOT	13:38 E	13:38 D	
542	O$STRLAM	13:39 E	13:39 D	
543	O$STRSTA	13:40 E	13:40 D	

544	O$SISGET	13:41 E	13:41 D
545	O$SITAPI	13:42 E	13:42 D
546	O$SITAPO	14: 1 E	14: 1 D
547	O$SIUBPR	14: 2 E	14: 2 D
550	O$SIXDKI	14: 3 E	14: 3 D
551	O$SIXDKO	14: 4 E	14: 4 D
552	O$SUMDAY	14: 5 E	14: 5 D
553	O$SUMHOW	14: 6 E	14: 6 D
554	O$SUMTIM	14: 7 E	14: 7 D
555	O$SWAP	14: 8 E	14: 8 D
556	O$SYNTAX	14: 9 E	14: 9 D
63	O$SYSS	4:36 E	4:36 D
557	O$SYSTAT	14:10 E	14:10 D
64	O$SYSTEX	4:37 E	4:37 D
151	O$TAPDIS	6:30 E	6:30 D
560	O$TAPEC	14:11 E	14:11 D
152	O$TAPETO	6:31 E	6:31 D
153	O$TAPEND	6:32 E	6:32 D
154	O$TAPERR	6:33 E	6:33 D
155	O$TAPFUN	6:34 E	6:34 D
156	O$TAPMOV	6:35 E	6:35 D
157	O$TCART	6:36 E	6:36 D
65	O$TCOM	4:38 E	4:38 D
160	O$TCTELL	6:37 E	6:37 D
161	O$TDEM	6:38 E	6:38 D
162	O$TDEM0	6:39 E	6:39 D
163	O$TDEM1	6:40 E	6:40 D
66	O$TDUMP	4:39 E	4:39 D
561	O$TEC455	14:12 E	14:12 D
614	O$TERMNS	15:13 E	15:13 D
164	O$TERROR	6:41 E	6:41 D
165	O$TEX	6:42 E	6:42 D
67	O$TIME	4:40 E	4:40 D
562	O$TJOB	14:13 E	14:13 D
563	O$TKSTAT	14:14 E	14:14 D
70	O$TRACE	4:41 E	4:41 D
71	O$TRACK	4:42 E	4:42 D
134	O$TRANSI	6: 1 E	6: 1 D
166	O$TRBOC	7: 1 E	7: 1 D
167	O$TRCER	7: 2 E	7: 2 D
170	O$TRCLN	7: 3 E	7: 3 D
171	O$TRDCK	7: 4 E	7: 4 D

172	O$IRDCKO	7: 5 E	7: 5 D
173	O$IREQC	7: 6 E	7: 6 D
174	O$IRIDB	7: 7 E	7: 7 D
175	O$IRINR	7: 8 E	7: 8 D
176	O$IRORN	7: 9 E	7: 9 D
177	O$IRRDB	7:10 E	7:10 D
200	O$IRRDF	7:11 E	7:11 D
201	O$IRREDO	7:12 E	7:12 D
202	O$IRSET	7:13 E	7:13 D
203	O$IRTELL	7:14 E	7:14 D
204	O$IRWRT	7:15 E	7:15 D
72	O$ISTASH	5: 1 E	5: 1 D
644	O$ITTKEY	16:12 E	16:12 D
645	O$ITTY	16:13 E	16:13 D
646	O$ITTYIN	16:14 E	16:14 D
647	O$ITTYNIT	16:15 E	16:15 D
650	O$ITTYNSC	16:16 E	16:16 D
651	O$ITTYOUT	16:17 E	16:17 D
564	O$IURNPG	14:15 E	14:15 D
337	O$UBTAPE	10:12 E	10:12 D
617	O$UCCLS	15:23 E	15:23 D
622	O$UCDRV	15:26 E	15:26 D
626	O$UCDRV1	15:33 E	15:33 D
627	O$UCDRV2	15:34 E	15:34 D
630	O$UCDRV3	15:35 E	15:35 D
631	O$UCDRV4	15:36 E	15:36 D
632	O$UCDRV5	15:37 E	15:37 D
633	O$UCDRV6	15:38 E	15:38 D
634	O$UCDRV7	15:39 E	15:39 D
635	O$UCDRV8	15:40 E	15:40 D
636	O$UCDRV9	15:41 E	15:41 D
616	O$UCOPN	15:22 E	15:22 D
620	O$UCRD	15:24 E	15:24 D
615	O$UCSHL	15:21 E	15:21 D
621	O$UCWRT	15:25 E	15:25 D
623	O$UCXFR	15:27 E	15:27 D
652	O$UDISK	16:18 E	16:18 D
653	O$UFILE	16:19 E	16:19 D
565	O$UNBLK	14:16 E	14:16 D
566	O$UPDATE	14:17 E	14:17 D
654	O$UPRINT	16:20 E	16:20 D
655	O$UPRT1	16:21 E	16:21 D

656	O$UPRT2		16:22 E	16:22 D						
657	O$UPRT3		16:23 E	16:23 D						
73	O$USURP		5: 2 E	5: 2 D						
660	O$UTAPE		16:24 E	16:24 D						
74	O$WATCH		5: 3 E	5: 3 D						
235	O$XCLOCK		8:22 E	8:22 D						
236	O$XCLS		8:23 E	8:23 D						
237	O$XDISK		8:24 E	8:24 D						
240	O$XDISKA		8:25 E	8:25 D						
241	O$XDISKB		8:26 E	8:26 D						
232	O$XDK		8:11 E	8:11 D						
340	O$XFMACC		10:13 E	10:13 D						
341	O$XFMCLS		10:14 E	10:14 D						
342	O$XFMCRE		10:15 E	10:15 D						
343	O$XFMDEL		10:16 E	10:16 D						
344	O$XFMDIR		10:17 E	10:17 D						
345	O$XFMDST		10:18 E	10:18 D						
346	O$XFMFIL		10:19 E	10:19 D						
350	O$XFMFLW		10:21 E	10:21 D						
351	O$XFMFND		10:22 E	10:22 D						
347	O$XFMFST		10:20 E	10:20 D						
352	O$XFMGET		10:23 E	10:23 D						
353	O$XFMIO		10:24 E	10:24 D						
354	O$XFMNIT		10:25 E	10:25 D						
355	O$XFMPUT		10:26 E	10:26 D						
356	O$XFMSTR		10:27 E	10:27 D						
567	O$XFRMT		14:18 E	14:18 D						
570	O$XJSTAT		14:19 E	14:19 D						
572	O$XMP4XP		14:21 E	14:21 D						
571	O$XMPXP		14:20 E	14:20 D						
233	O$XMT		8:12 E	8:12 D						
242	O$XOPN		8:27 E	8:27 D						
234	O$XPR		8:13 E	8:13 D						
243	O$XPRINT		8:28 E	8:28 D						
244	O$XPRNTA		8:29 E	8:29 D						
245	O$XPRNTB		8:30 E	8:30 D						
246	O$XTAPE		8:31 E	8:31 D						
247	O$XTAPEA		8:32 E	8:32 D						
250	O$XTAPEB		8:33 E	8:33 D						
251	O$XTAPEC		8:34 E	8:34 D						
252	O$XTAPED		8:35 E	8:35 D						
2	OV@ENT@V	$APIEXT	3:19	3:20	3:21	3:22	3:23	3:24	3:33	3:34

3:35	3:36	3:37	3:38	3:39	3:40	3:41	3:42
4: 1	4: 2	4: 3	4: 4	4: 5	4: 6	4: 7	4: 8
4: 9	4:10	4:11	4:12	4:13	4:14	4:15	4:16
4:17	4:18	4:19	4:20	4:21	4:22	4:23	4:24
4:25	4:26	4:27	4:28	4:29	4:30	4:31	4:32
4:33	4:34	4:35	4:36	4:37	4:38	4:39	4:40
4:41	4:42	5: 1	5: 2	5: 3	5:12	5:13	5:14
5:15	5:16	5:17	5:18	5:19	5:20	5:21	5:22
5:23	5:24	5:25	5:26	5:27	5:28	5:29	5:30
5:31	5:32	5:33	5:34	5:35	5:36	5:37	5:38
5:39	5:40	5:41	5:42	6: 1	6:10	6:11	6:12
6:13	6:14	6:15	6:16	6:25	6:26	6:27	6:28
6:29	6:30	6:31	6:32	6:33	6:34	6:35	6:36
6:37	6:38	6:39	6:40	6:41	6:42	7: 1	7: 2
7: 3	7: 4	7: 5	7: 6	7: 7	7: 8	7: 9	7:10
7:11	7:12	7:13	7:14	7:15	7:24	7:25	7:26
7:27	7:28	7:29	7:30	7:31	7:32	7:33	7:34
7:35	7:36	8: 3	8: 4	8: 5	8: 6	8: 7	8: 8
8: 9	8:10	8:11	8:12	8:13	8:22	8:23	8:24
8:25	8:26	8:27	8:28	8:29	8:30	8:31	8:32
8:33	8:34	8:35	9: 2	9: 3	9: 4	9: 5	9: 6
9: 7	9: 8	9: 9	9:10	9:11	9:12	9:13	9:14
9:15	9:16	9:17	9:18	9:19	9:20	9:21	9:22
9:23	9:24	9:25	9:26	9:27	9:28	9:29	9:30
9:31	9:32	9:33	9:34	9:35	9:36	9:37	9:38
9:39	9:40	9:41	9:42	10: 1	10: 2	10: 3	10: 4
10: 5	10: 6	10: 7	10: 8	10: 9	10:10	10:11	10:12
10:13	10:14	10:15	10:16	10:17	10:18	10:19	10:20
10:21	10:22	10:23	10:24	10:25	10:26	10:27	10:36
10:37	10:38	10:39	10:40	10:41	10:42	11: 7	11: 8
11: 9	11:18	11:19	11:20	11:21	11:22	11:23	11:24
11:25	11:26	11:27	11:28	11:29	11:30	11:31	11:32
11:33	11:34	11:35	11:36	11:37	11:38	11:39	11:40
11:41	11:42	12: 1	12: 2	12: 3	12: 4	12: 5	12: 6
12: 7	12: 8	12: 9	12:10	12:11	12:12	12:13	12:14
12:15	12:16	12:17	12:18	12:19	12:20	12:21	12:22
12:23	12:24	12:25	12:26	12:27	12:28	12:29	12:30
12:31	12:32	12:33	12:34	12:35	12:36	12:37	12:38
12:39	12:40	12:41	12:42	13: 1	13: 2	13: 3	13: 4
13: 5	13: 6	13: 7	13: 8	13: 9	13:10	13:11	13:12
13:13	13:14	13:15	13:16	13:17	13:18	13:19	13:20
13:21	13:22	13:23	13:24	13:25	13:26	13:27	13:28

			13:29	13:30	[illegible]	13:32	13:33	13:34	13:35	13:36
			13:37	13:38	13:39	13:40	13:41	13:42	14: 1	14: 2
			14: 3	14: 4	14: 5	14: 6	14: 7	14: 8	14: 9	14:10
			14:11	14:12	14:13	14:14	14:15	14:16	14:17	14:18
			14:19	14:20	14:21	14:30	14:31	14:32	14:33	14:34
			14:35	14:36	14:37	15: 4	15: 5	15: 6	15: 7	15: 8
			15: 9	15:10	15:11	15:12	15:13	15:21	15:22	15:23
			15:24	15:25	15:26	15:27	15:31	15:32	15:33	15:34
			15:35	15:36	15:37	15:38	15:39	15:40	15:41	16: 7
			16: 8	16: 9	16:10	16:11	16:12	16:13	16:14	16:15
			16:16	16:17	16:18	16:19	16:20	16:21	16:22	16:23
			16:24	16:26						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
700	OVLTBL	$APTEXT	16:29 F							
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							

SYMBOL QUALIFIER IS OVLNUM

```
0     <macro>                    OVERLAY    PATCH                                          PATCH.4

                                 COMMENT    'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                                  ****************************************      PATCH.5
                                  *                                      *      PATCH.6
                                  *   PATCH:  An overlay to allow patching overlay in MOS     *      PATCH.7
                                  *           from the CRT.  Overlay name and address are     *      PATCH.8
                                  *           specified for display, if change wanted the     *      PATCH.9
                                  *           contents of cell must be specified.             *      PATCH.10
                                  *           Ex:  PATCH ACOM 122      .Display address 122   *      PATCH.11
                                  *                 Display:  ACOM 000122 000027              *      PATCH.12
                                  *                PATCH ACOM 122 0    .Set 122 to 0          *      PATCH.13
                                  *                 Display:  ACOM 000122 000000              *      PATCH.14
                                  *                PATCH KERN 12345 7 .Set Kernel addr. to 7  *      PATCH.15
                                  *                                                         *      PATCH.16
                                  ***************************************************************  PATCH.17

6     <macro>                     REGDEFS   ,(PA,PB,PC,PD,PE,PF,PG,PH,PI,PJ,PK,PL,PM,PN,PO,PP),      PATCH.19
                                            (PQ,PR,PS,PT,PU,PV,PW,PX,ONUM)                            PATCH.20
                410     %S1       EQUALS    $REGORG                                                   $REG    3
                411     %S2       EQUALS    $REGORG                                                   $REG    3
                412     %S3       EQUALS    $REGORG                                                   $REG    3
                413     %S4       EQUALS    $REGORG                                                   $REG    3
                414     %S5       EQUALS    $REGORG                                                   $REG    3
                415     %T1       EQUALS    $REGORG                                                   $REG    3
                416     %T2       EQUALS    $REGORG                                                   $REG    3
                417     %T3       EQUALS    $REGORG                                                   $REG    3
                420     %T4       EQUALS    $REGORG                                                   $REG    3
                421     %T5       EQUALS    $REGORG                                                   $REG    3
                422     %T6       EQUALS    $REGORG                                                   $REG    3
                423     %W1       EQUALS    $REGORG                                                   $REG    3
                424     %W2       EQUALS    $REGORG                                                   $REG    3
                425     %W3       EQUALS    $REGORG                                                   $REG    3
                426     %W4       EQUALS    $REGORG                                                   $REG    3
                427     %W5       EQUALS    $REGORG                                                   $REG    3
                430     PA        EQUALS    $REGORG                                                   $REG    3
                431     PB        EQUALS    $REGORG                                                   $REG    3
                432     PC        EQUALS    $REGORG                                                   $REG    3
                433     PD        EQUALS    $REGORG                                                   $REG    3
                434     PE        EQUALS    $REGORG                                                   $REG    3
                435     PF        EQUALS    $REGORG                                                   $REG    3
                436     PG        EQUALS    $REGORG                                                   $REG    3
                437     PH        EQUALS    $REGORG                                                   $REG    3
                440     PI        EQUALS    $REGORG                                                   $REG    3
```

```
441     PJ      EQUALS  $REGORG                                                    $REG    3
442     PK      EQUALS  $REGORG                                                    $REG    3
443     PL      EQUALS  $REGORG                                                    $REG    3
444     PM      EQUALS  $REGORG                                                    $REG    3
445     PN      EQUALS  $REGORG                                                    $REG    3
446     PO      EQUALS  $REGORG                                                    $REG    3
447     PP      EQUALS  $REGORG                                                    $REG    3
450     PQ      EQUALS  $REGORG                                                    $REG    3
451     PR      EQUALS  $REGORG                                                    $REG    3
452     PS      EQUALS  $REGORG                                                    $REG    3
453     PT      EQUALS  $REGORG                                                    $REG    3
454     PU      EQUALS  $REGORG                                                    $REG    3
455     PV      EQUALS  $REGORG                                                    $REG    3
456     PW      EQUALS  $REGORG                                                    $REG    3
457     PX      EQUALS  $REGORG                                                    $REG    3
460     ONUM    EQUALS  $REGORG                                                    $REG    3
```

```
                                        * Get memory and      it the request parameters there          PATCH.22
  6   <macro>                           GETMEM    40,PO          .Get free memory                     PATCH.23
 26   103340                            P = GRETMAN, A # 0       .Give up if not available            PATCH.24
 27   010430  054000                    B = [PA]                 .Register number                     PATCH.25
 31   020446  024447                    PP = PO                  .Deposit                             PATCH.26
 33   010017  024450                    PQ = 17                  .Number of parcels to save           PATCH.27
 35                             GLLOD   *                                                             PATCH.28
 35   060000  034447                    (PP) = (B)               .Save params in memory               PATCH.29
 37   056000                            B = B + 1                .                                    PATCH.30
 40   026447                            PP = PP + 1              .Next                                PATCH.31
 41   027450                            PQ = PQ - 1              .Done?                               PATCH.32
 42   107005                            P = GLLOD, A # 0         .                                    PATCH.33
                                                                                                      PATCH.34
                                                                                                      PATCH.35
                                        * Translate input to overlay name                             PATCH.36
 43   010000  024442                    PK = 0                   .Kernel flag                         PATCH.37
 45   010000  024451                    PR = 0                   .Character number                    PATCH.38
 47   072362                            R = GALPHA               .Get the name                        PATCH.39
 50   072430                            R = GETNUM               .Address                             PATCH.40
 51   020456  024443                    PL = PW                  .                                    PATCH.41
 53   072425                            R = GETNUM               .New contents                        PATCH.42
 54   020456  024444                    PM = PW                  .If PX = 0, don't modify             PATCH.43
                                                                                                      PATCH.44
 56   <macro>                           $IF       ((R!PO) = 'KE'R),AND,((R!PO+1) = 'RN'R)            PATCH.45
                                                                                  .>>>>> 0              $IF      1
 73   010444  024442                      R!PK = O'444                 .Set Kernel flag               PATCH.46
 75   070277                              P = GKERN                                                   PATCH.47
 76   <macro>                           $ENDIF                                                        PATCH.48
                                                                                  .<<<<< 0              $ENDIF   1


                             *          Locate the overlay in buffer memory.                          PATCH.50
                                                                                                      PATCH.51
 76   010000  024460                    R!ONUM = 0                                                    PATCH.52
                                                                                                      PATCH.53
100                          FIND       *                                                             PATCH.54
100   <macro>                           FIND      R!ONUM,R!PC,R!PD,R!PJ,TYPE=NUMBER                   PATCH.55
116   103274                            P = GERRO, A # 0             .If overlay not found           PATCH.56
                                                                                                      PATCH.57
117   026460                            R!ONUM = R!ONUM + 1          .Next overlay number            PATCH.58
120   020446  012020  024434            R!PE = R!PO + O'20                                           PATCH.59
123   <macro>                           MOSR      R!PC,R!PD,R!PE,1   .Read overlay name             PATCH.60
```

```
                                                                                                PATCH.61
                                        * Compare MOS overlay name with that keyed in           PATCH.62
143   020440  004001  024435            PF = PI > 1              .Number of characters - full parcels  PATCH.63
146   020440  011001  024452            PS = PI & 1              .                                   PATCH.64
151   020452  102002  026435            PF = PF + 1, PS # 0      .Partial parcel?                    PATCH.65
154   020446  024436                    PG = PO                  .Keyed in name                      PATCH.66
156                           GOAGAN    *                                                            PATCH.67
156   030436  033434  107060            P = FIND, (R!PG) # (R!PE)     .If no match                   PATCH.68
161   026434                            PE = PE + 1              .Next                               PATCH.69
162   026436                            PG = PG + 1              .                                   PATCH.70
163   027435                            PF = PF - 1              .Parcels                            PATCH.71
164   107006                            P = GOAGAN, A # 0        .Find if now zero                   PATCH.72
                                                                                                    PATCH.73
165   020441  005002  024441            R!PJ = R!PJ < 2               .Overlay size in parcels       PATCH.74
170   020443  023441  101220            P = GERRO, PL >= PJ      .Within overlay?                    PATCH.75
                                                                                                    PATCH.76
                                        * Read in address to display and/or modify                  PATCH.77
173   020443  004002  022433            PD = PL > 2 + PD         .MOS lower plus address             PATCH.78
      024433
177   100002  026432                    PC = PC + 1, C # 0       .If overflow                        PATCH.79
201   020446  012004  024434            PE = PO + 4              .Local address                      PATCH.80
204    <macro>                          MOSR   PC,PD,PE,1        .Read one word                      PATCH.81
224   020443  011003  022434            PN = PL & 3 + PE         .Address of thing to display        PATCH.82
      024445
                                                                                                    PATCH.83
                                        * Modify if desired, and display old and new                PATCH.84
                                                                                                    PATCH.85
230                           GMODFY    *                                                            PATCH.86
230   020440  004001  022446            PG = PI > 1 + PO + 20    .PR has number of characters        PATCH.87
      012020  024436
235   020440  011001  024452            PS = PI & 1              .Odd?                               PATCH.88
240   102007                            P = GBYT, A = 0          .?                                  PATCH.89
241   030436  015000 /177400            (PG) = (PG) & 177400 + 40 .Insert space                      PATCH.90
      012040  034436
246   070004                            P = GBYX                 .                                   PATCH.91
247   014000 /020040  034436    GBYT    (PG) = '  'H             .Two space characters               PATCH.92
252   026436                    GBYX    PG = PG + 1              .Next spot                          PATCH.93
253   020443  024452                    PS = PL                  .Number to convert                  PATCH.94
255   020436  024453                    PT = PG                  .Destiny                            PATCH.95
257   072310                            R = GIOASCI              .Convert to ascii                   PATCH.96
260   010003  025436                    PG = PG + 3              .Next spot                          PATCH.97
262   014000 /020040  034436            (PG) = '  'H             .Blank fill                         PATCH.98
```

```
262   026436                          PG = PG + 1                 .                                  PATCH.99
266   020457  103003                  P = GMODD, PX # 0           .Modify?                           PATCH.100
270   030445  024444                  PM = (PN)                   .No, get current contents          PATCH.101
272                         GMODD     *                                                              PATCH.102
272   020444  034445                  (PN) = PM                   .Modify                            PATCH.103
274   020444  024452                  PS = PM                     .Display modification              PATCH.104
276   020436  024453                  PT = PG                     .Destination                       PATCH.105
300   072267                          R = GTOASCI                 .Convert                           PATCH.106
301   020442  103034                  P = GSEND, PK # 0           .Don't write if Kernel             PATCH.107
303   020457  102032                  P = GSEND, PX = 0           .Modify?                           PATCH.108
305   020443  013004  100103          P = GERRO, PL < 4           .Can't change overlay name         PATCH.109
310    <macro>                        MOSW    PC,PD,PE,1          .Write out modification            PATCH.110
330    <macro>                        FLUSH                            .Flush overlays from memory   PATCH.111
336                         GSEND     *                                                              PATCH.112
336   010003  025436                  PG = PG + 3                 .Next                              PATCH.113
340   010000  034436                  (PG) = 0                    .End of message                    PATCH.114
342                         GMSG      *                                                              PATCH.115
342   020446  012020  024447          PP = PO + 20                .                                  PATCH.116
345    <macro>                        MSG   PP                    .                                  PATCH.117
356    <macro>                        RELMEM    PO                .Release memory                    PATCH.118
366                         GRETMAN   *                                                              PATCH.119
366    <macro>                        RETURN                      .                                  PATCH.120
                                                                                                     PATCH.121
374                         GKERN     *                                                              PATCH.122
374   020446  012020  024436          PG = PO + 20                .Output buffer                     PATCH.123
377   014000 /045505  034436          (PG) = 'KE'H                .                                  PATCH.124
402   026436                          PG = PG + 1                 .Next                              PATCH.125
403   014000 /051116  034436          (PG) = 'RN'H                .                                  PATCH.126
406   026436                          PG = PG + 1                 .                                  PATCH.127
407   020443  024445                  PN = PL                     .Address to modify                 PATCH.128
411   071161                          P = GMODFY                  .Go, modify                        PATCH.129
                                                                                                     PATCH.130
                                                                                                     PATCH.131
                                                                                                     PATCH.132
                                      *  Error in key-in                                             PATCH.133
412                         GERRO     *                                                              PATCH.134
412   014000 /000620  022000          PH = GPERRM + R!%B          .Message to send                   PATCH.135
      024437
416   020446  012020  024436          PG = PO + 20                .                                  PATCH.136
421                         GMOVE     *                                                              PATCH.137
421   030437  034436                  (PG) = (PH)                 .Move message to free mem          PATCH.138
423   030436  102004                  P = GONER, (PG) = 0         .Done?                             PATCH.139
```

```
425   026436                          PG = PG + 1                .                                    PATCH.140
426   026437                          PH = PH + 1                .                                    PATCH.141
427   071006                          P = GMOVE                  .                                    PATCH.142
430                         GONER     *                                                               PATCH.143
430   071066                          P = GMSG                   .                                    PATCH.144
                                      ************************************************************    PATCH.145
                                      *                                                          *    PATCH.146
                                      *   SUBROUTINES:                                           *    PATCH.147
                                      *                                                          *    PATCH.148
                                      ************************************************************    PATCH.149
                                                                                                      PATCH.150
                                      *  Get number of numeric characters, set SP(40) to 00          PATCH.151
                                      *   Returns number of characters, sets first space to 00       PATCH.152
                                      *                                                               PATCH.153
431                         GALPHA    *                                                               PATCH.154
431   020451  004001  024455          PV = PR > 1                .Parcel                              PATCH.155
434   020451  011001  024454          PU = PR & 1                .Odd or even char.                   PATCH.156
437   020455  022446  024452          PS = PV + PO               .Parcel to examine                   PATCH.157
442   030452  024453                  PT = (PS)                  .Contents                            PATCH.158
444   020454  103004  020453          PT = PT > 10, PU = 0       .Top char.?                          PATCH.159
      004010  024453
451   020453  011377  024453          PT = PT & 377              .Bottom                              PATCH.160
454   020453  013012  106044          P = GERRO, PT = 12         .Carriage return                     PATCH.161
457   020453  013040  102006          P = GDONE, PT = 40         .Space?                              PATCH.162
462   026451                          PR = PR + 1                .next                                PATCH.163
463   020451  013010  107034          P = GALPHA, PR # 10        .Another                             PATCH.164
466   071054                          P = GERRO                  .Name too big                        PATCH.165
467                         GDONE     *                                                               PATCH.166
467   020454  102005  030452          (PS) = (PS) & 177400, PU # 0 .Odd char.                         PATCH.167
      015000 /177400  034452
475   020451  024440                  PI = PR                    .Save size of overlay name           PATCH.168
477   001000                          EXIT                       .                                    PATCH.169
                                                                                                      PATCH.170
                                      *  GETNUM:  Gets address or value from buffer                   PATCH.171
                                      *           returns value in PW                                 PATCH.172
500                         GETNUM    *                                                               PATCH.173
500   010000  024457                  PX = 0                     .Number of characters                PATCH.174
502   010000  024456                  PW = 0                     .Value                               PATCH.175
504   010000  024455                  PV = 0                     .Space count                         PATCH.176
506                         GIIT      *                                                               PATCH.177
506   020451  004001  022446          PS = PR > 1 + PO           .Parcel                              PATCH.178
      024452
```

```
514  020451  011001  024454          PU = PR & 1                 .Odd or even character          PATCH.17
515  026451                          PR = PR + 1                 .For next parcel                PATCH.180
516  030452  024453                  PT = (PS)                   .                               PATCH.181
520  020454  103004  020453          PT = PT > 10, PU = 0        .Upper?                         PATCH.182
     004010  024453
525  020453  011377  024453          PT = PT & 377               .Lower                          PATCH.183
530  020453  013040  102030          P = GFINI, PT = 40          .Space?                         PATCH.184
533  020453  102026                  P = GFINI, PT = 0           .Delimiter?                     PATCH.185
535  020457  013006  106125          P = GERRO, PX = 6           .Number too large?              PATCH.186
540  020453  013067  100003          P = GERRO, PT > 67          .Check numeric                  PATCH.187
     102002  071132
545  020453  013060  104135          P = GERRO, PT < 60          .Check numeric                  PATCH.188
550  026457                          PX = PX + 1                 .                               PATCH.189
551  020453  011007  024453          PT = PT & 7                 .Convert to octal               PATCH.190
554  020456  005003  022453          PW = PW < 3 + PT            .                               PATCH.191
     024456
560  026455                          PV = PV + 1                 .Count character processed      PATCH.192
561  071053                          P = GTTT                    .                               PATCH.193
562                          GFINI   *                                                           PATCH.194
562  020455  102002  001000          EXIT, PV # 0                .Need two characters to exit    PATCH.195
565  026455                          PV = PV + 1                 .                               PATCH.196
566  071060                          P = GTTT                    .                               PATCH.197
                                                                                                 PATCH.198
                                                                                                 PATCH.199
                                     * Convert octal to ascii, store in designated spot          PATCH.200
                                     *   Entry: PS = number to convert, PT = destination         PATCH.201
567                          GTOASCI *                                                           PATCH.202
567  020453  012002  024456          PW = PT + 2                 .Low order bits                 PATCH.203
572                          GTONXT  *                                                           PATCH.204
572  020452  011007  012060          PU = PS & 7 + 60            .First byte                     PATCH.205
     024454
576  020452  004003  024452          PS = PS > 3                 .Next                           PATCH.206
601  020452  011007  012060          (PW) = PS & 7 + 60 < 10 + PU .Store first 2 bytes           PATCH.207
     005010  022454  034456
607  020452  004003  024452          PS = PS > 3                 .                               PATCH.208
612  020456  023453  103002          EXIT, PW = PT               .Filled?                        PATCH.209
     001000
616  027456                          PW = PW - 1                 .No, set next destination       PATCH.210
617  071025                          P = GTONXT                  .                               PATCH.211
                                                                                                 PATCH.212
620                          GPERRM  'PATCH ERROR'H                                              PATCH.213
626  000000                          00                                                          PATCH.214
```

```
                                                                    PATCH.215
627   077770                77770          .Patch area              PATCH.216
630   077771                77771          .                        PATCH.217
631   077772                77772          .                        PATCH.218
632   077773                77773          .                        PATCH.219
633   077774                77774          .                        PATCH.220
634   077775                77775          .                        PATCH.221
635   077776                77776          .                        PATCH.222
636   077777                77777          .                        PATCH.223
                            END                                    PATCH.224
```

0X	%B		1: 4 D	6:37						
0X	%EX		4: 2 D	4: 2	4:37 D	4:37	4:42 D	4:42	5:22 D	5:22
			6:12 D	6:12	6:13 D	6:13	6:19 D	6:19	6:20 D	6:20
			6:22 D	6:22						
10	%GBLNUM	$APTEXT	2:17 F	2:17						
400	%GBLREG	$APTEXT	2:17							
53	F$FIND	$APTEXT	4:37							
54	F$FLUSH	$APTEXT	6:13							
30	F$GMEM	$APTEXT	4: 2							
20	F$MESG	$APTEXT	6:19							
46	F$MOSR	$APTEXT	4:42	5:22						
47	F$MOSW	$APTEXT	6:12							
52	F$RETURN	$APTEXT	6:22							
31	F$RMEM	$APTEXT	6:20							
0	F$$WAIT	$APTEXT	4:42	5:22	6:12					
0X	O$PATCH		1: 4 D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							

SYMBOL QUALIFIER IS PATCH

461	$REGORG		2:17 D	2:17 F	2:18	2:18 D	2:19	2:19 D	2:20	2:20 D
			2:21	2:21 D	2:22	2:22 D	2:22 F	2:23	2:23 D	2:24
			2:24 D	2:25	2:25 D	2:26	2:26 D	2:27	2:27 D	2:28
			2:28 D	2:28 F	2:29	2:29 D	2:30	2:30 D	2:31	2:31 D
			2:32	2:32 D	2:33	2:33 D	2:33 F	2:34	2:34 D	2:35
			2:35 D	2:36	2:36 D	2:37	2:37 D	2:38	2:38 D	2:39
			2:39 D	2:40	2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1
			3: 1 D	3: 2	3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5
			3: 5 D	3: 6	3: 6 D	3: 7	3: 7 D	3: 7 F	3: 8	3: 8 D
			3: 9	3: 9 D	3:10	3:10 D	3:11	3:11 D	3:12	3:12 D
			3:13	3:13 D	3:14	3:14 D	3:15	3:15 D	3:16	3:16 D
			3:16 F	4: 2	4:37	4:42	5:22	6:12	6:13	6:19
			6:19 F	6:20	6:22					
20	%NP		1: 4	3: 7 D						
430	%P		1: 4	2:33 D	3: 7	6:19 F	6:19			
410	%S1		2:17	2:18 D	4:25					
411	%S2		2:17	2:19 D						
412	%S3		2:17	2:20 D						
413	%S4		2:17	2:21 D						

414	%S5	2:17	2:22 D						
415	%T1	2:23 D							
416	%T2	2:24 D							
417	%T3	2:25 D							
420	%T4	2:26 D							
421	%T5	2:27 D							
422	%T6	2:28 D							
423	%W1	2:29 D							
424	%W2	2:30 D							
425	%W3	2:31 D							
426	%W4	2:32 D							
427	%W5	2:33 D							
100	FIND,	4:36 D	5: 8						
431	GALPHA	4:18	7:15 D	7:26					
247	GBYI	5:32	5:36 D						
252	GBYX	5:35	5:37 D						
467	GDONE	7:24	7:28 D						
412	GERRO	4:38	5:15	6:11	6:36 D	7:23	7:27	8: 9	8:11
		8:12							
500	GETNUM	4:19	4:21	7:36 D					
562	GFINI	8: 7	8: 8	8:19 D					
374	GKERN	4:27	6:24 D						
35	GLLOD	4: 7 D	4:12						
272	GMODD	6: 2	6: 4 D						
230	GMODFY	5:28 D	6:31						
421	GMOVE	6:40 D	7: 3						
342	GMSG	6:17 D	7: 5						
156	GOAGAN	5: 7 D	5:12						
430	GONER	6:42	7: 4 D						
620	GPERRM	6:37	8:41 D						
366	GRETMAN	4: 3	6:21 D						
336	GSEND	6: 9	6:10	6:14 D					
567	GTOASCI	5:40	6: 8	8:27 D					
572	GTONXT	8:29 D	8:39						
506	GTTT	7:40 D	8:18	8:22					
460	ONUM	3:16 D	4:34	4:37	4:40				
430	PA	2:34 D	4: 4						
431	PB	2:35 D							
432	PC	2:36 D	4:37	4:42	5:20	5:22	6:12		
433	PD	2:37 D	4:37	4:42	5:18	5:22	6:12		
434	PE	2:38 D	4:41	4:42	5: 8	5: 9	5:21	5:22	5:23
		6:12							

435	PF	2:39 D	5: 3	[illegible]	5:11				
436	PG	2:40 D	5: 6	5: 8	5:10	5:30	5:33	5:34	5:36
		5:37	5:39	5:41	5:42	6: 1	6: 7	6:15	6:16
		6:25	6:26	6:27	6:28	6:29	6:39	6:41	6:42
		7: 1							
437	PH	2:41 D	6:37	6:41	7: 2				
440	PI	2:42 D	5: 3	5: 4	5:29	5:31	7:31		
441	PJ	3: 1 D	4:37	5:14	5:15				
442	PK	3: 2 D	4:16	4:26	6: 9				
443	PL	3: 3 D	4:20	5:15	5:18	5:23	5:38	6:11	6:30
444	PM	3: 4 D	4:22	6: 3	6: 5	6: 6			
445	PN	3: 5 D	5:23	6: 3	6: 5	6:30			
446	PO	3: 6 D	4: 2	4: 5	4:25	4:41	5: 6	5:21	5:29
		6:18	6:20	6:25	6:39	7:18	7:41		
447	PP	3: 7 D	4: 5	4: 8	4:10	6:18	6:19		
450	PQ	3: 8 D	4: 6	4:11					
451	PR	3: 9 D	4:17	7:16	7:17	7:25	7:26	7:31	7:41
		8: 1	8: 2						
452	PS	3:10 D	5: 4	5: 5	5:31	5:38	6: 6	7:18	7:19
		7:29	7:30	7:41	8: 3	8:30	8:32	8:33	8:35
453	PT	3:11 D	5:39	6: 7	7:19	7:20	7:21	7:22	7:23
		7:24	8: 3	8: 4	8: 5	8: 6	8: 7	8: 8	8:10
		8:12	8:14	8:15	8:28	8:36			
454	PU	3:12 D	7:17	7:20	7:29	8: 1	8: 4	8:30	8:34
455	PV	3:13 D	7:16	7:18	7:39	8:17	8:20	8:21	
456	PW	3:14 D	4:20	4:22	7:38	8:15	8:28	8:34	8:36
		8:38							
457	PX	3:15 D	6: 2	6:10	7:37	8: 9	8:13		

```
0     <macro>                    OVERLAY SCRUB                                                    SCRUB.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ***************************************************************************   SCRUB.5
                    *  THE SCRUB OVERLAY IS ACTIVATED PERIODICALLY TO REFRESH AREAS OF        *   SCRUB.6
                    *  IOP LOCAL MEMORY WHICH ARE, IN GENERAL, READ-ONLY IN NATURE.            *   SCRUB.7
                    *  THE PURPOSE OF THIS REFRESH IS TO ENSURE THAT ALL 3 COPIES OF 2AW       *   SCRUB.8
                    *  VOTING MEMORY RETAIN VALID DATA. THE AREA REFRESHED IS FROM ADDRESS     *   SCRUB.9
                    *  0 UP TO THE END OF THE RESIDENT AREA. THE DATA IS WRITTEN 1 SECTOR      *   SCRUB.10
                    *  AT A TIME OUT TO MOS AND THEN READ BACK IN. A FLUSH IS THEN ISSUED      *   SCRUB.11
                    *  TO CAUSE ALL OVERLAYS TO BE RE-LOADED.                                  *   SCRUB.12
                    ***************************************************************************   SCRUB.13

6     <macro>                       REGDEFS ,,(MSU,MSL,LENGTH,PO)                                 SCRUB.15
            410     %S1       EQUALS    $REGORG                                                        $REG     3
            411     %S2       EQUALS    $REGORG                                                        $REG     3
            412     %S3       EQUALS    $REGORG                                                        $REG     3
            413     %S4       EQUALS    $REGORG                                                        $REG     3
            414     %S5       EQUALS    $REGORG                                                        $REG     3
            415     %T1       EQUALS    $REGORG                                                        $REG     3
            416     %T2       EQUALS    $REGORG                                                        $REG     3
            417     %T3       EQUALS    $REGORG                                                        $REG     3
            420     %T4       EQUALS    $REGORG                                                        $REG     3
            421     %T5       EQUALS    $REGORG                                                        $REG     3
            422     %T6       EQUALS    $REGORG                                                        $REG     3
            423     %W1       EQUALS    $REGORG                                                        $REG     3
            424     %W2       EQUALS    $REGORG                                                        $REG     3
            425     %W3       EQUALS    $REGORG                                                        $REG     3
            426     %W4       EQUALS    $REGORG                                                        $REG     3
            427     %W5       EQUALS    $REGORG                                                        $REG     3
            430     MSU       EQUALS    $REGORG                                                        $REG     3
            431     MSL       EQUALS    $REGORG                                                        $REG     3
            432     LENGTH    EQUALS    $REGORG                                                        $REG     3
            433     PO        EQUALS    $REGORG                                                        $REG     3

                              EXT       %INIT                                                         SCRUB.17
                              EXT       CA                                                            SCRUB.18
                              EXT       CB                                                            SCRUB.19
                              EXT       CC                                                            SCRUB.20
                              EXT       CD                                                            SCRUB.21
                              EXT       CE                                                            SCRUB.22
                              EXT       %MEMORY                                                       SCRUB.23
                              EXT       %MYID                                                         SCRUB.24
```

```
                    EXT      MTBUFF                                          SCRUB.25

21450     REFRESH   EQUALS   D'9000          .15 MINUTE REFRESH RATE         SCRUB.27
```

```
                                 *                                                                  SCRUB.29
                                 * FIND END OF RESIDENT CODE FOR THIS IOP                          SCRUB.30
                                 *                                                                  SCRUB.31
                                                                                                    SCRUB.32
  6    <macro>                   $IF (R!%MYID = MIOP)          .IF MIOP                           SCRUB.33
                                                                                .>>>>> 0                $IF      1
 11    014000 /000000  024432      R!LENGTH = MTBUFF           .DON'T SCRUB MTBUFF                SCRUB.34
 14    <macro>                   $ELSE                         .OTHERS                            SCRUB.35
 15    020000  024432              R!LENGTH = R!%MEMORY        .END OF RESIDENT CODE              SCRUB.36
 17    <macro>                   $ENDIF                                                           SCRUB.37
                                                                                .<<<<< 0                $ENDIF   1
 17    020432  012003  004002    R!LENGTH = R!LENGTH + 3 > 2   .CONVERT TO WORD LENGTH            SCRUB.38
       024432
                                                                                                    SCRUB.39
                                 *                                                                  SCRUB.40
                                 * WAIT FOR TIMED ACTIVATION                                       SCRUB.41
                                 *                                                                  SCRUB.42
                                                                                                    SCRUB.43
 23                        BEGIN  *                                                                SCRUB.44
 23    <macro>                   PAUSE REFRESH                 .WAIT FOR NEXT REFRESH             SCRUB.45
                                                                                                    SCRUB.46
                                 *                                                                  SCRUB.47
                                 * ALLOCATE AN MOS BUFFER                                          SCRUB.48
                                 *                                                                  SCRUB.49
                                                                                                    SCRUB.50
 35    010001  024433            R!PO = 1                      .FLAG                              SCRUB.51
 37    <macro>                   $UNTIL (R!PO = 0)             .UNTIL MEMORY FOUND                SCRUB.52
 41    <macro>                     MGET R!MSU,R!MSL            .REQUEST                           SCRUB.53
 67    024433                      R!PO = A                    .STATUS                            SCRUB.54
 70    <macro>                     $IF (A # 0)                 .IF NONE                           SCRUB.55
                                                                                .>>>>> 0                $IF      1
 72    <macro>                       GIVEUP                    .WAIT FOR IT TO BECOME AVAILABLE   SCRUB.56
101    <macro>                     $ENDIF                                                         SCRUB.57
                                                                                .<<<<< 0                $ENDIF   1
101    <macro>                   $ENDTIL                                                          SCRUB.58
                                                                                                    SCRUB.59
                                 *                                                                  SCRUB.60
                                 * WRITE/READ LOCAL MEMORY 1 SECTOR AT A TIME                      SCRUB.61
                                 *                                                                  SCRUB.62
                                                                                                    SCRUB.63
102    002000  000000            I = 0                         .DISABLE INTERRUPTS                SCRUB.64
104    010000  024000            R!%INIT = 0                   .DISABLE TRACING                   SCRUB.65
```

```
106   020430  024000                  R!CB = R!MSU                  .MOS BUFFER ADDRESS                SCRUB.66
110   020431  024000                  R!CC = R!MSL                  .                                  SCRUB.67
112   010000  024000                  R!CD = 0                      .LOCAL ADDRESS                     SCRUB.68
114   014000 /001000  024000          R!CE = D'512                  .1 SECTOR                          SCRUB.69
117   020432  024423                  R!%W1 = R!LENGTH              .SAVE LENGTH                       SCRUB.70
121   072050                          R = SCRUB                     .SCRUB MEMORY                      SCRUB.71
                                                                                                       SCRUB.72
                                      *                                                                SCRUB.73
                                      * SCRUB MIOP-ONLY CODE IN MIOP                                   SCRUB.74
                                      *                                                                SCRUB.75
                                                                                                       SCRUB.76
122    <macro>                        $IF (R!%MYID = MIOP)          .IF MIOP                           SCRUB.77
                                                                                        .>>>>> 0          $IF      1
125   014000 /000000  016000            R!CD = MTBUFF + MX$BFSZ     .END OF MTBUFF                     SCRUB.78
     /004000  024000
132   014000 /001000  024000            R!CE = D'512                .1 SECTOR                          SCRUB.79
135   020000  023000  024423            R!%W1 = R!%MEMORY - R!CD    .SAVE LENGTH                       SCRUB.80
140   020423  004002  024423            R!%W1 = R!%W1 > 2           .WORD SIZE                         SCRUB.81
                                                                                                       SCRUB.82
143   072026                            R = SCRUB                   .SCRUB MEMORY                      SCRUB.83
144    <macro>                        $ENDIF                                                          SCRUB.84
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                       SCRUB.85
144   010001  024000                  R!%INIT = 1                   .RESTORE TRACE                     SCRUB.86
146    <macro>                        MPUT R!MSU,R!MSL              .RETURN MOS BUFFER                 SCRUB.87
162    <macro>                        FLUSH                         .REFRESH OVERLAY AREA              SCRUB.88
170   071145                          P = BEGIN                     .LOOP                              SCRUB.89
                                                                                                       SCRUB.90
                                ***********************************************************************SCRUB.91
                                                                                                       SCRUB.92
                                      *                                                                SCRUB.93
                                      * SCRUB MEMORY                                                   SCRUB.94
                                      *                                                                SCRUB.95
                                                                                                       SCRUB.96
171                             SCRUB *                                                                SCRUB.97
                                                                                                       SCRUB.98
171    <macro>                        $UNTIL (R!%W1 = 0)            .UNTIL REFRESH DONE                SCRUB.99
173    <macro>                          $IF (R!%W1 < D'512)         .IF LESS THAN 1 SECTOR TO GO       SCRUB.100
                                                                                        .>>>>> 0          $IF      1
200   020423  024000                     R!CE = R!%W1               .SET SIZE                          SCRUB.101
202    <macro>                          $ENDIF                                                        SCRUB.102
                                                                                        .<<<<< 0          $ENDIF   1
```

```
20.   010003  024000                R!CA = MIO$BMW            .SET CODE TO WRITE          SCRUB.10
204    <macro>                      RKERN COMIO               .ISSUE                      SCRUB.104
                                                                                           SCRUB.105
                                    * READ BACK TO LOCAL                                   SCRUB.106
                                                                                           SCRUB.107
206   010001  024000                R!CA = MIO$BMR            .SET CODE TO READ           SCRUB.108
210    <macro>                      RKERN COMIO               .ISSUE                      SCRUB.109
212   020000  005002  022000        R!CD = R!CE < 2 + R!CD    .NEXT ADDRESS               SCRUB.110
      024000
216   020423  023000  024423        R!%W1 = R!%W1 - R!CE      .LEFT TO DO                 SCRUB.111
221    <macro>                    $ENDTIL                                                  SCRUB.112
222   001000                      EXIT                                                     SCRUB.113
                                                                                           SCRUB.114
                                  END                                                      SCRUB.115
```

```
   0X  %ACTIVE                 4:28 D    4:28
   0X  %B                      1: 4 D
   0X  %EX                     4:20 D    4:20      4:28 D    4:28      4:32 D    4:32      5:25 D    5:25
                               5:26 D    5:26
  10   %GBLNUM    $APTEXT      2:12 F    2:12
 400   %GBLREG    $APTEXT      2:12
   0X  %INIT                   2:35 D    4:42      5:24
   0   %KERNEL    $APTEXT      6: 2 D    6: 2 B    6: 7 D    6: 7 B
   0X  %MEMORY                 2:41 D    4: 9      5:17
   0X  %MYID                   2:42 D    4: 6      5:13
  20   AD@P1@N    $APTEXT      4:28      4:28 F
  14   AD@P1@P    $APTEXT      4:28 F    4:28
   0   AD@P1@S    $APTEXT      4:28      4:28 F
  20   AD@P2@N    $APTEXT      4:28      4:28 F
  15   AD@P2@P    $APTEXT      4:28 F    4:28
   0   AD@P2@S    $APTEXT      4:28      4:28 F
  20   AD@RC@N    $APTEXT      4:28      4:28 F
  13   AD@RC@P    $APTEXT      4:28 F    4:28
   0   AD@RC@S    $APTEXT      4:28      4:28 F
   0X  CA                      2:36 D    6: 1      6: 6
   0X  CB                      2:37 D    5: 1
   0X  CC                      2:38 D    5: 2
   0X  CD                      2:39 D    5: 3      5:15      5:17      6: 8
   0X  CE                      2:40 D    5: 4      5:16      5:40      6: 8      6:10
   0X  COMIO                   6: 2 D    6: 2      6: 7 D    6: 7
  54   F$FLUSH    $APTEXT      5:26
   4   F$GIVEUP   $APTEXT      4:32
  35   F$MGET     $APTEXT      4:28
  36   F$MPUT     $APTEXT      5:25
   7   F$PAUSE    $APTEXT      4:20
   1   MIO$BMR    $APTEXT      6: 6
   3   MIO$BMW    $APTEXT      6: 1
   0   MIOP       $APTEXT      4: 6      5:13
   0X  MIBUFF                  3: 1 D    4: 7      5:14
4000   MX$BFSZ    $APTEXT      5:14
   0X  O$SCRUB                 1: 4 D    1: 4
  17   OV@NUM@N   $APTEXT      1: 4
   1   OV@TYP@N   $APTEXT      1: 4
  11   SM@LST@N   $APTEXT      1: 4
   7   SM@NUM@N   $APTEXT      1: 4

       SYMBOL QUALIFIER IS SCRUB
```

434	$REGORG	2:12 D	2:12 F	2:13	2:13 D	2:14	2:14 D	2:15	2:15 D
		2:16	2:16 D	2:17	2:17 D	2:17 F	2:18	2:18 D	2:19
		2:19 D	2:20	2:20 D	2:21	2:21 D	2:22	2:22 D	2:23
		2:23 D	2:23 F	2:24	2:24 D	2:25	2:25 D	2:26	2:26 D
		2:27	2:27 D	2:28	2:28 D	2:28 F	2:29	2:29 D	2:30
		2:30 D	2:31	2:31 D	2:32	2:32 D	2:32 F	4:20	4:20 F
		4:28	4:32	4:32 F	5:25	5:26			
0	%NP	1: 4	2:28 D						
430	%P	1: 4	2:28 D	2:28	4:20 F	4:20	4:32 F	4:32	
410	%S1	2:12	2:13 D	4:28					
411	%S2	2:12	2:14 D						
412	%S3	2:12	2:15 D						
413	%S4	2:12	2:16 D						
414	%S5	2:12	2:17 D						
415	%T1	2:18 D							
416	%T2	2:19 D							
417	%T3	2:20 D							
420	%T4	2:21 D							
421	%T5	2:22 D							
422	%T6	2:23 D							
423	%W1	2:24 D	5: 5	5:17	5:18	5:37	5:39	5:40	6:10
424	%W2	2:25 D							
425	%W3	2:26 D							
426	%W4	2:27 D							
427	%W5	2:28 D							
23	BEGIN	4:19 D	5:27						
432	LENGTH	2:31 D	4: 7	4: 9	4:12	5: 5			
431	MSL	2:30 D	4:28	5: 2	5:25				
430	MSU	2:29 D	4:28	5: 1	5:25				
433	PO	2:32 D	4:26	4:27	4:29				
21450	REFRESH	3: 4 D	4:20						
171	SCRUB	5: 6	5:20	5:35 D					

```
0     <macro>                    OVERLAY   SYSS,TYPE=DATA                                    SYSS.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'              OVERLAY  1
```

```
                                  ***************************  ****************************************   SYSS.5
                                  *                                                                *   SYSS.6
                                  *        SYSS      IOP initialization                            *   SYSS.7
                                  *                                                                *   SYSS.8
                                  *        SYSS is loaded into memory to perform initialization for *   SYSS.9
                                  *        IOP'S 1, 2, and 3.  It configures the IOP as described by *   SYSS.10
                                  *        AMAP and the system directory.                          *   SYSS.11
                                  *                                                                *   SYSS.12
                                  ******************************************************************   SYSS.13


                                           EXT       %B                                                SYSS.15
                                           BASEREG   %B                                                SYSS.16
                                                                                                       SYSS.17
6     <macro>                              REGDEFS   ,,(BD,BE,BF,CF,EA,EB,EO,IA,IB,IC,ID,MA,MB,MC,MD,ME,  SYSS.18
                                                       MF,MG,MH,MI,MJ,MK,ML,MM,MN,MP)                  SYSS.19
                410                 %S1       EQUALS    $REGORG                                          $REG    3
                411                 %S2       EQUALS    $REGORG                                          $REG    3
                412                 %S3       EQUALS    $REGORG                                          $REG    3
                413                 %S4       EQUALS    $REGORG                                          $REG    3
                414                 %S5       EQUALS    $REGORG                                          $REG    3
                415                 %T1       EQUALS    $REGORG                                          $REG    3
                416                 %T2       EQUALS    $REGORG                                          $REG    3
                417                 %T3       EQUALS    $REGORG                                          $REG    3
                420                 %T4       EQUALS    $REGORG                                          $REG    3
                421                 %T5       EQUALS    $REGORG                                          $REG    3
                422                 %T6       EQUALS    $REGORG                                          $REG    3
                423                 %W1       EQUALS    $REGORG                                          $REG    3
                424                 %W2       EQUALS    $REGORG                                          $REG    3
                425                 %W3       EQUALS    $REGORG                                          $REG    3
                426                 %W4       EQUALS    $REGORG                                          $REG    3
                427                 %W5       EQUALS    $REGORG                                          $REG    3
                430                 BD        EQUALS    $REGORG                                          $REG    3
                431                 BE        EQUALS    $REGORG                                          $REG    3
                432                 BF        EQUALS    $REGORG                                          $REG    3
                433                 CF        EQUALS    $REGORG                                          $REG    3
                434                 EA        EQUALS    $REGORG                                          $REG    3
                435                 EB        EQUALS    $REGORG                                          $REG    3
                436                 EO        EQUALS    $REGORG                                          $REG    3
                437                 IA        EQUALS    $REGORG                                          $REG    3
                440                 IB        EQUALS    $REGORG                                          $REG    3
                441                 IC        EQUALS    $REGORG                                          $REG    3
```

```
442     ID      EQUALS  $REGORG                                         $REG    3
443     MA      EQUALS  $REGORG                                         $REG    3
444     MB      EQUALS  $REGORG                                         $REG    3
445     MC      EQUALS  $REGORG                                         $REG    3
446     MD      EQUALS  $REGORG                                         $REG    3
447     ME      EQUALS  $REGORG                                         $REG    3
450     MF      EQUALS  $REGORG                                         $REG    3
451     MG      EQUALS  $REGORG                                         $REG    3
452     MH      EQUALS  $REGORG                                         $REG    3
453     MI      EQUALS  $REGORG                                         $REG    3
454     MJ      EQUALS  $REGORG                                         $REG    3
455     MK      EQUALS  $REGORG                                         $REG    3
456     ML      EQUALS  $REGORG                                         $REG    3
457     MM      EQUALS  $REGORG                                         $REG    3
460     MN      EQUALS  $REGORG                                         $REG    3
461     MP      EQUALS  $REGORG                                         $REG    3

                EXT     %ACTIVE                                 SYSS.21
                EXT     %DEBBA                                  SYSS.22
                EXT     %DEBUG                                  SYSS.23
                EXT     %HISP                                   SYSS.24
                EXT     %HISPBUF                                SYSS.25
                EXT     %INIT                                   SYSS.26
                EXT     %MEMORY                                 SYSS.27
                EXT     %MYID                                   SYSS.28
                EXT     %OVPT                                   SYSS.29
                EXT     %SSD                                    SYSS.30
                EXT     %SSDBUF                                 SYSS.31
                EXT     %UCT                                    SYSS.32
                EXT     BBDAL                                   SYSS.33
                EXT     BBDISK                                  SYSS.34
                EXT     BBSOFT                                  SYSS.35
                EXT     BCRT                                    SYSS.36
                EXT     BXCR                                    SYSS.37
                EXT     CA                                      SYSS.38
                EXT     CB                                      SYSS.39
                EXT     CC                                      SYSS.40
                EXT     CD                                      SYSS.41
                EXT     CE                                      SYSS.42
                EXT     DEBLEN                                  SYSS.43
                EXT     DEBSTART                                SYSS.44
```

```
EXT      DY                                                    SYSS.45
EXT      EACT                                                  SYSS.46
EXT      EBUF                                                  SYSS.47
EXT      EBUFSIZ                                               SYSS.48
EXT      EITB                                                  SYSS.49
EXT      EJ                                                    SYSS.50
EXT      EK                                                    SYSS.51
EXT      EL                                                    SYSS.52
EXT      EMEM                                                  SYSS.53
EXT      EMEMSIZ                                               SYSS.54
EXT      ENDBOOT                                               SYSS.55
EXT      ESMD                                                  SYSS.56
EXT      HB                                                    SYSS.57
EXT      HC                                                    SYSS.58
EXT      IUCIO                                                 SYSS.59
EXT      MAAQ                                                  SYSS.60
EXT      MAOQ                                                  SYSS.61
EXT      MATT                                                  SYSS.62
EXT      MATX                                                  SYSS.63
EXT      MBAQ                                                  SYSS.64
EXT      MIAQ                                                  SYSS.65
EXT      MBBQ                                                  SYSS.66
EXT      MBCQ                                                  SYSS.67
EXT      MBDQ                                                  SYSS.68
EXT      O$AMAP                                                SYSS.69
EXT      O$BEGIN                                               SYSS.70
EXT     STOPCRAY                                               SYSS.71
EXT      SUMMORE                                               SYSS.72
EXT      TLOC                                                  SYSS.73
EXT      TMOSL                                                 SYSS.74
EXT      TMOSU                                                 SYSS.75
EXT      XHSPCH                                                SYSS.76
```

```
                                  *        Clear local memory.                                                    SYSS.78
                                                                                                                  SYSS.79
     6   020000  016000 /002130           R!%W1 = R!%B + AZZZ                                                    SYSS.80
         024423                                                                                                   SYSS.81
    12    <macro>                         $UNTIL    (R!%W1 = R!%DEBUG)  .Until upper memory cleared              SYSS.82
    15   010000  034423                     (R!%W1) = 0                                                           SYSS.83
    17   026423                             R!%W1 = R!%W1 + 1                                                     SYSS.84
    20    <macro>                         $ENDTIL                                                                SYSS.85

    21   020000  016000 /002130           R!MA = R!%B + AZZZ            .System directory local address          SYSS.87
         024443
    25   010000  024000                   R!CB = 0                      .System directory MOS address            SYSS.88
    27   014000 /000000  024000           R!CC = ENDBOOT                                                         SYSS.89
    32   020443  024000                   CD = MA                       .LOCAL ADDRESS                           SYSS.90
    34   010012  024000                   CE = 12                       .SIZE OF SYSDIR                          SYSS.91
    36   010001  024000                   CA = MIO$BMR                  .READ MOS                                SYSS.92
    40    <macro>                         RKERN COMIO                                                            SYSS.93
                                                                                                                  SYSS.94
                                          * SET UP OVERLAY AREAS                                                 SYSS.95
    42   020000  005002  022443           MD = CE < 2 + MA              .AMAP, in 3rd overlay after SYSDI        SYSS.96
         024446
    46   014000 /000000  005002           R!%W1 = O$AMAP < OT$POWER + R!%OVPT                                    SYSS.97
         022000  024423
    53    <macro>                         GET       R!CB,OT@MUP,R!%W1   .MOS address of AMAP                     SYSS.98
    56    <macro>                         GET       R!CC,OT@MLO,R!%W1                                            SYSS.99
    63    <macro>                         GET       R!CE,OT@WRD,R!%W1   .AMAP word length                        SYSS.100
    66   020446  024000                   R!CD = R!MD                                                            SYSS.101
    70   010001  024000                   CA = MIO$BMR                  .READ MOS                                SYSS.102
    72    <macro>                         RKERN COMIO                                                            SYSS.103
                                                                                                                  SYSS.104
    74   020446  012013  022000           R!MH = R!MD + O'13 + R!%MYID                                           SYSS.105
         024452
   100   030452  024452                   MH = (MH)                                                              SYSS.106
   102    <macro>                         $PUNTIF   (MH = 0)            .If undefined.                           SYSS.107
   105   020452  022446  024452           MH = MH + MD                  .ADDRESS OF 'A?' DESCR.                  SYSS.108

                                  *        Initialize the overlay area.  Three nodes are needed: a header,        SYSS.110
```

```
                                *       a trailer, and a r     which contains all the overlay space        SYSS.11
                                *       defined.  The register %MEMORY points to the header node which     SYSS.112
                                *       serves as a queue header for the available space list.             SYSS.113

110   020000  024000                    R!%MEMORY = R!%B                .Overlay space overwrites SYSS     SYSS.115
112    <macro>                          STORE     MD$ID,MD@ID,R!%MEMORY  .Set up header node              SYSS.116
115    <macro>                          STORE     MD$HEAD,MD@TYP,R!%MEMORY                                SYSS.117
122    <macro>                          PUT       R!%MEMORY,MD@PRE,R!%MEMORY                              SYSS.118
127    <macro>                          PUT       R!%MEMORY,MD@SUC,R!%MEMORY                              SYSS.119
134    <macro>                          PUT       R!%MEMORY,MD@FOR,R!%MEMORY                              SYSS.120
141    <macro>                          PUT       R!%MEMORY,MD@BAK,R!%MEMORY                              SYSS.121
                                                                                                          SYSS.122
146    <macro>                          ADDRESS   R!%W1,MD@LEN,R!%MEMORY  .Node with available space      SYSS.123
151    <macro>                          STORE     MD$ID,MD@ID,R!%W1                                       SYSS.124
154    <macro>                          STORE     MD$FREE,MD@TYP,R!%W1                                    SYSS.125
161    <macro>                          LINK      R!%W1,MD@SUC,MD@PRE,R!%MEMORY  .Link with header       SYSS.126
212    <macro>                          LINK      R!%W1,MD@FOR,MD@BAK,R!%MEMORY  .Add to available list  SYSS.127
                                                                                                          SYSS.128
243   020452  012003  024410             R!%W2 = (MH + 3)                .Overlay space to be allocated     SYSS.129
      030410  024424
250   020424  017000 /011000             R!%W2 = OVL$MIN, R!%W2 < OVL$MIN  .Insure minimum allocation     SYSS.130
      101004  014000 /011000
      024424
257    <macro>                          ADDRESS   R!%W3,MD@LEN,R!%W1                                      SYSS.131
262   020425  022424  024425             R!%W3 = R!%W3 + R!%W2           .Set up trailer node               SYSS.132
265    <macro>                          STORE     MD$ID,MD@ID,R!%W3                                       SYSS.133
270    <macro>                          STORE     MD$HEAD,MD@TYP,R!%W3                                    SYSS.134
275    <macro>                          LINK      R!%W3,MD@SUC,MD@PRE,R!%W1                               SYSS.135
                                                                                                          SYSS.136
326    <macro>                          ADDRESS   R!MG,MD@LEN,R!%W3   .Point to free memory              SYSS.137


                                        *         MA -> SYSDIR                                            SYSS.139
                                        *         MD -> AMAP                                              SYSS.140
                                        *         MG -> NEXT FREE MEMORY PIECE                            SYSS.141
                                        *         MH -> 'A' PARAMETERS IN AMAP                            SYSS.142
```

```
                                        *                                                                   SYSS.144
                                                * INITIALIZE LOCAL BUFFERS (512 WORDS)                      SYSS.145
                                                *  IA HAS ADDRESS OF START                                  SYSS.146
331   020452  012001  024410                    ME = (MH + 1)                  .NUMBER OF DISK BUFFERS     SYSS.147
      030410  024447
336   020447  102050                            P = APAS, ME = 0               .ZERO SPECIFIED?            SYSS.148
340   010040  023447  005013                    IA = 40 - ME < 13              .START OF 512W DISK BUFFERS SYSS.149
      024437
344   014000 /000000  024461                    MP = EBUF                      .DISK BUFFER CHAIN          SYSS.150
347   020461  012002  024410                    (MP + 2) = ME                  .EBNM .                     SYSS.151
      020447  034410
354   014000 /000000  024410                    (EBUFSIZ) = ME                 .SAVE INITIAL COUNT         SYSS.152
      020447  034410
361   020437  034461                            (MP) = IA                      .EBUF .FIRST                SYSS.153
363   027447                            ARPB    ME = ME - 1                    .COUNT DOWN                 SYSS.154
364   020447  102011                            P = ARPF, ME = 0               .LAST??                     SYSS.155
366   020437  016000 /004000                    (IA) = IA + 4000               .MORE                       SYSS.156
      034437
372   014000 /004000  025437                    IA = IA + 4000                 .4000 PARCELS, 1000 WORDS   SYSS.157
375   071012                                    P = ARPB                       .END?                       SYSS.158
376   010000  034437                    ARPF    (IA) = 0                       .YES                        SYSS.159
400   020461  012001  024410                    (MP + 1) = IA                  .EBUH .LAST                 SYSS.160
      020437  034410
405   030461  024437                            IA = (MP)                      .EBUF .DISK BUFFER START    SYSS.161
                                                *                                                           SYSS.162
                                                *  INITIALIZE LOCAL FREE MEMORY CHAIN                       SYSS.163
407   020451  024444                    APAS    MB = MG                        .LOCAL MEMORY               SYSS.164
411   014000 /000000  024447                    ME = EMEM                      .FREE MEMORY DESCRIPTER     SYSS.165
414   020444  034447                            (ME) = MB                      .EMEM .SET FIRST LINK       SYSS.166
416   026447                                    R!ME = R!ME + 1                                            SYSS.167
417   010000  034444                            (MB) = 0                       .FIRST LINK (ONLY)          SYSS.168
421   020444  012001  024410                    (MB + 1) = IA - MB & 177774    .UPDATE SIZE                SYSS.169
      020437  023444  015000
     /177774  034410
431   034447                                    (R!ME) = A                     .Free memory size           SYSS.170
432    <macro>                                  $PUNTIF   (IA < MB)            .If chains overlap.         SYSS.171


                                        *       Set up local memory message packets.                        SYSS.173
                                                                                                            SYSS.174
```

```
43    020452  012004  024410        R!MC = (R!MH + 4)          .Number of packets to allocate     SYSS.17
      030410  024445
                                                                                                 SYSS.176
443    <macro>                      $UNTIL    (R!MC = 0)       .Until all packets allocated       SYSS.177
445   010040  024000                  R!HB = 0'40               .Message packet size               SYSS.178
447   077000 /001665                  R = GTS                   .Allocate local memory             SYSS.179
451   020000  024000                  R!DY = R!HC                                                  SYSS.180
453    <macro>                        RKERN   HONN              .Place on available queue          SYSS.181
455   027445                          R!MC = R!MC - 1                                              SYSS.182
456    <macro>                      $ENDTIL                                                       SYSS.183


                              *     Allocate local memory software stack area.                    SYSS.185
                                                                                                  SYSS.186
457   010100  005002  024000        R!HB = SM$SIZE < 2        .Size for SMOD                     SYSS.187
462   077000 /001665                R = GTS                   .                                  SYSS.188
464   014000 /000000  024410        (ESMD) = R!HC             .SAVE ADDRESS                      SYSS.189
      020000  034410
471    <macro>                      STORE EACT,SM@ACT,R!HC    .PONTER TO ACTIVITY CHAIN          SYSS.190


                              *     Set up the allocation table for buffer-memory DALs.           SYSS.192
                                                                                                  SYSS.193
474   020446  012007  022000        R!%W1 = R!MD + 7 + R!%MYID                                    SYSS.194
      024423
500   030423  004003  024424        R!%W2 = (R!%W1) > 3       .Number of allocatable packets     SYSS.195
503   010003  024425                R!%W3 = 3                 .Packet size as power of 2         SYSS.196
505   020000  005001  022443        R!%W1 = R!%MYID < 1 + R!MA                                    SYSS.197
      024423
511   030423  024426                R!%W4 = (R!%W1)           .Base address in buffer memory     SYSS.198
513   020423  012001  024410        R!%W5 = (R!%W1 + 1)                                           SYSS.199
      030410  024427
520   014000 /000000  024423        R!%W1 = BBDAL             .Allocation table address          SYSS.200
523   077000 /002035                R = INITMAP               .Set up allocation table           SYSS.201
```

```
                                *       Initialize buffer memory allocation table.                      SYSS.203
                                                                                                        SYSS.204
525   020000  005002  022443           R!%W1 = R!%MYID < 2 + R!MA + O'20                               SYSS.205
      012020  024423
532   020423  012003  024410           R!%W2 = (R!%W1 + 3)            .Number of sectors for this IOP  SYSS.206
      030410  024424
                                                                                                        SYSS.207
                                        * UPDATE MOS BUFFER COUNT IN AMAP                               SYSS.208
                                        * AND WRITE BACK OUT                                            SYSS.209
                                                                                                        SYSS.210
537   034452                            (R!MH) = A                          .SET COUNT IN AMAP          SYSS.211
540   010004  024000                    R!CA = MIO$BMWI                                                 SYSS.212
542    <macro>                          RKERN COMIO                                                     SYSS.213
544   010011  024425                    R!%W3 = D'9                    .Size expressed as a power of 2  SYSS.214
546   030423  024426                    R!%W4 = (R!%W1)                .Buffer memory base address      SYSS.215
550   020423  012001  024410            R!%W5 = (R!%W1 + 1)                                             SYSS.216
      030410  024427
555   014000 /000000  024423            R!%W1 = BBDISK                 .Allocation table address        SYSS.217
560   077000 /002035                    R = INITMAP                    .Set up allocation table         SYSS.218


                                *       Allocate software stack space and initialize the allocation     SYSS.220
                                *       table.                                                          SYSS.221
                                                                                                        SYSS.222
                                        IFE       SM$SIZE,NE,O'100,1                                    SYSS.223
       <skipped>                        ERROR                          .Software stack size inconsistenc SYSS.224
                                                                                                        SYSS.225
562   020452  012002  024410            R!CA = (R!MH + 2) + 7 > 3      .Number of buffers required      SYSS.226
      030410  012007  004003
      024000
571   077000 /001706                    R = STEALIT                                                     SYSS.227
573   014000 /000000  024423            R!%W1 = BBSOFT                 .Allocation table address        SYSS.228
576   020452  012002  024410            R!%W2 = (R!MH + 2) + 7 & O'177770 .Number of stacks             SYSS.229
      030410  012007  015000
     /177770  024424
606   010006  024425                    R!%W3 = 6                      .Stack size shift count          SYSS.230
610   020000  024426                    R!%W4 = R!CB                   .Buffer memory base address      SYSS.231
612   020000  024427                    R!%W5 = R!CC                                                    SYSS.232
614   077000 /002035                    R = INITMAP                    .Set up allocation table         SYSS.233
```

```
                                        *                                                   SYSS.23
                                        * GET MEMORY FOR USER CHANNEL TABLE HEADER          SYSS.236
                                        *                                                   SYSS.237
                                                                                            SYSS.238
616   010004  024000                    R!HB = UC@HDR@P              .SIZE                  SYSS.239
620   077000 /001665                    R = GTS                      .ALLOCATE LOCAL MEMORY  SYSS.240
                                                                                            SYSS.241
622   020000  024000                    R!%UCT = R!HC                .SAVE ADDRESS          SYSS.242
                                                                                            SYSS.243
624    <macro>                          STORE UC$NAM,UC@NAM,R!%UCT   .SET TABLE NAME 'UC'   SYSS.244
627    <macro>                          STORE      0,UC@POP,R!%UCT   .POPULATION            SYSS.245
634    <macro>                          STORE      0,UC@FST,R!%UCT   .FIRST LINK            SYSS.246
641    <macro>                          STORE      0,UC@LST,R!%UCT   .LAST LINK             SYSS.247
                                                                                            SYSS.248
                                        * INITIALIZE 'A' -> 'A' MSG QUEUES                  SYSS.249
                                                                                            SYSS.250
646   020446  012013  022000            R!MB = R!MD + O'13 + R!%MYID                        SYSS.251
      024444
652   020446  012013  024445            MC = MD + 13                 .AMAP AO DESCR.        SYSS.252
655   014000 /000000  024461            MP = MAAQ                    .GET FROM SYS INDEX    SYSS.253
660   020461  024447                    ME = MP                      .MAAQ .I/P QUEUE AO    SYSS.254
662   014000 /000000  024450            MF = MAOQ                    .O/P QUEUE             SYSS.255
665   030445  102020          ABMM      P = ABMS, (MC) = 0           .EXIST                 SYSS.256
667   020444  023445  102015            P = ABMS, MB = MC            .SENSE SELF            SYSS.257
672   010226  024000                    HB = MSGQSZI                 .Size of table to build SYSS.258
674   077000 /001640                    R = ABLD                     .Build I/P message table SYSS.259
676   020000  034447                    (ME) = HC                    .SAVE ADDRESS FOR I/P  SYSS.260
700   010100  024000                    HB = MSGQSZO                 .Build O/P message table SYSS.261
702   077000 /001640                    R = ABLD                     .BUILD O/P TABLE       SYSS.262
704   020000  034450                    (MF) = HC                    .SAVE ADDRESS          SYSS.263
706   010004  025447          ABMS      ME = ME + 4                  .TRY ANOTHER           SYSS.264
710   026450                            MF = MF + 1                                         SYSS.265
711   026445                            MC = MC + 1                  .NEXT A                SYSS.266
712   020461  012020  024410            P = ABMM, ME # MP + 20       .DONE?? (MAAQ + 20)    SYSS.267
      020447  023410  107032
                                                                                            SYSS.268
                                        *  INITIALIZE 'MATT' AND 'MATX' - TABLES TO TRANSLATE SYSS.269
                                        *    CHANNEL TO LOGICAL 'A' PROCESSOR               SYSS.270
                                        * MH POINTS TO AMAP PARAMETERS FOR THIS 'A'         SYSS.271
720   020452  012006  024410            MI = (MH + 6)                .CHAN CONFIGURATION INFO SYSS.272
      030410  024453
725   020453  022446  024453            MI = MI + MD                 .CHANNEL CONFIG THIS 'A' SYSS.273
```

```
 730   014000 /000000  024460          MN = MATX                    .'A' TO 'A' CHN'L TABLE          SYSS.274
 733   020460  024455                  MK = MN                      .MATX .TABLE TO CLEAR            SYSS.275
 735   010000  034455          AMCL    (MK) = 0                     .CLEAR 'MATX'                    SYSS.276
 737   026455                          MK = MK + 1                  .NEXT                            SYSS.277
 740   020460  012004  024410          P = AMCL, MK # MN + 4        .DONE??                          SYSS.278
       020455  023410  107010
 746   020453  012001  024454          MJ = MI + 1                  .CHANNEL 6                       SYSS.279
 751   014000 /000000  024455          MK = MATT                    .CHNL TABLE                      SYSS.280
 754   030454  024456          AMOR    ML = (MJ)                    .P-TYPE                          SYSS.281
 756   020456  126000 /001032          P = AMAM, ML = 0             .DEFINED?                        SYSS.282
 761   020456  011077  013030          ML = ML & 77 - 30            .P-TYPE 30 IS AO                 SYSS.283
       024456
 765    <macro>                        $PUNTIF   (ML >= 4)          .Bad AMAP.                       SYSS.284
 771   020456  034455                  (MK) = ML                    .INIT 'MATT'                     SYSS.285
 773   020460  022456  024457          MM = MN + ML                 .MATX .INIT 'MATX'               SYSS.286
 776   020454  023453  012005          MF = MJ - MI + 5             .CALCULATE CHANNEL NO.           SYSS.287
       024450
1002   020450  034457                  (MM) = MF                    .INSERT CHANNEL                  SYSS.288
1004   020456  005002  022461          MM = ML < 2 + MP + 3         .FIND 'MAAQ' TABLE CELL          SYSS.289
       012003  024457
1011   020456  005002  016000          ME = ML < 2 + MBAQ + 3       .BMX TABLE CELL FOR I/P          SYSS.290
      /000000  012003  024447
1017   020456  005002  016000          R!MC = R!ML < 2 + MIAQ + 3   .DD49 table cell for I/P chan.  SYSS.291
      /000000  012003  024445
1025   020450  034457                  (MM) = MF                    .STORE CHANNEL NO.               SYSS.292
1027   034447                          (ME) = A                     .SAVE CHANNEL FOR BMX TABLE      SYSS.293
1030   034445                          (R!MC) = A                   .Save channel for DD49 table     SYSS.294
1031   070003                          P = AMAO                     .NEXT                            SYSS.295
1032   010077  034455          AMAM    (MK) = 77                    .NO 'A', UNDEFINED FLAG          SYSS.296
1034   026455                  AMAO    MK = MK + 1                                                   SYSS.297
1035   010002  025454                  MJ = MJ + 2                  .NEXT CHANNEL                    SYSS.298
1037   020453  012007  024410          P = AMOR, MJ # MI + 7        .SENSE DONE                      SYSS.299
       020454  023410  107070
                                                                                                     SYSS.300
                                       * SET UP INTERRUPT JUMP TABLE                                 SYSS.301
                                                                                                     SYSS.302
1045   020453  012001  024454          MJ = MI + 1                  .'A' CHANNEL INFO                SYSS.303
1050   030453  022453  013005          MK = (MI) + MI - 5           .HIGHEST CHNL TO PROCESS         SYSS.304
       024455
1054   014000 /000000  012006          ML = EITB + 6                .INTERRUPT ENTRANCE TABLR        SYSS.305
       024456
1060                           ANEXT    *                                                            SYSS.306
```

```
106     020454  023453  012005          IB = MJ - MI + 5            .Channel number                 SYSS.30
        024440
1064    030454  102217                  P = AMNENT, (MJ) = 0        .If empty channel entry         SYSS.308
1066    030454  024000                  HB = (MJ)                   .Channel type                   SYSS.309
1070    020000  004017  024410          P = AMHIGHS, 0 = HB > 17    .If not possible CRT channel    SYSS.310
        010000  023410  102102
1076    020000  015000 /037777          HB = HB & 37777                                             SYSS.311
        024000
1102    010000  024431                  BE = 0                      .Set 1440 CRT type              SYSS.312
1104    020000  013021  102020          P = AMCRT, HB = CH$C1       .If 1440 CRT                    SYSS.313
1107    010001  024431                  BE = 1                                                      SYSS.314
1111    020000  013020  102013          P = AMCRT, HB = CH$C0       .If 455 CRT                     SYSS.315
1114    010002  024431                  BE = 2                                                      SYSS.316
1116    020000  013022  102006          P = AMCRT, HB = CH$CS       .If SOROC                       SYSS.317
1121    010003  024431                  BE = 3                                                      SYSS.318
1123    020000  013023  103052          P = AMHIGHS, HB # CH$AMPEX  .If not AMPEX                   SYSS.319
                                                                                                    SYSS.320
1126                            AMCRT   *                                                           SYSS.321
1126    010070  024000                  HB = CRTABSIZ               .Allocate table space for this CR SYSS.322
1130    072535                          R = GIS                                                     SYSS.323
1131    020440  004001  011003          BD = IB > 1 & 3 + BCRT      .CRT table entry                SYSS.324
        016000 /000000  024430
1137    020000  034430                  (BD) = HC                                                   SYSS.325
1141    020000  012010  024410          (HC + 10) = HC + 20         .CRT buffer                     SYSS.326
        020000  012020  034410
1147    020000  012011  024410          (HC + 11) = BE              .CRT type                       SYSS.327
        020431  034410
1154    020000  012012  024410          (HC + 12) = IB + 1          .O/P channel number             SYSS.328
        020440  012001  034410
1162    020440  054000                  B = IB                      .Channel                        SYSS.329
1164    160000                          IOB: 0                      .Master clear                   SYSS.330
1165    167000                          IOB: 7                      .Allow interrupts               SYSS.331
1166    056000                          B = B + 1                   .O/P channel                    SYSS.332
1167     <macro>                        $IF       (IOB = BZ)                                        SYSS.333
                                                                                .>>>>> 0                $IF      1
1172    042000  104001                    WAIT,   IOB # DN                                          SYSS.334
1174     <macro>                        $ENDIF                                                      SYSS.335
                                                                                .<<<<< 0                $ENDIF   1
1174    160000                          IOB: 0                                                      SYSS.336
1175    167000                          IOB: 7                      .Allow interrupts               SYSS.337
1176    070104                          P = AMDESC                                                  SYSS.338
1177                            AMHIGHS *                                                           SYSS.339
```

```
1177   030454  011777  024410          P = AMUCHN, CH$CH # (R!MJ) & O'777 .If not high speed channel      SYSS.340
       010035  023410  103066
1205   030454  004011  011037          R!%W1 = (R!MJ) > D'9 & O'37   .Target memory type                 SYSS.341
       024423
1211    <macro>                        $IF (R!%W1 = TM$CMEM)         .If Central memory                 SYSS.342
                                                                                        .>>>>> 0          $IF      1
1214   020440  024000                   R!%HISP = R!IB               .HISP input channel number         SYSS.343
1216   014000 /000000  024410           (XHSPCH) = R!IB              .Save for debugger                 SYSS.344
       020440  034410
1223    <macro>                         $IF (I@IOPMOD # 'C'R),OR,(R!IB # O'14) .Bypass not available   SYSS.345
                                                                                        .>>>>> 1          $IF      1
1232    <macro>                          RKERN HGETBUF               .Dedicated disk buffer             SYSS.346
1234    <macro>                          $PUNTIF (R!DY = 0),CODE=PT$LOCAL                               SYSS.347
1237   020000  024000                    R!%HISPBUF = R!DY            .Save                              SYSS.348
1241    <macro>                         $ENDIF                                                          SYSS.349
                                                                                        .<<<<< 1          $ENDIF   1
1241    <macro>                        $ELSEIF (R!%W1 = TM$SSD)      .If SSD                            SYSS.350
1246   020440  024000                   R!%SSD = R!IB                .SSD input channel number          SYSS.351
1250    <macro>                         $IF (I@IOPMOD # 'C'R),OR,(R!IB # O'14) .Bypass not available   SYSS.352
                                                                                        .>>>>> 1          $IF      1
1257    <macro>                          RKERN HGETBUF               .Dedicated disk buffer             SYSS.353
1261    <macro>                          $PUNTIF (R!DY = 0),CODE=PT$LOCAL                               SYSS.354
1264   020000  024000                    R!%SSDBUF = R!DY             .Save                              SYSS.355
1266    <macro>                         $ENDIF                                                          SYSS.356
                                                                                        .<<<<< 1          $ENDIF   1
1266    <macro>                        $ELSE                                                            SYSS.357
1267    <macro>                          $PUNTIF CODE=PT$BADTM       .Illegal target memory             SYSS.358
1271    <macro>                        $ENDIF                                                           SYSS.359
                                                                                        .<<<<< 0          $ENDIF   1
1271   070011                          P = AMDESC                                                       SYSS.360
1272                          AMUCHN   *                                                                SYSS.361
1272   030454  017000 /100043          P = AMNENT, (R!MJ) # 100043   .If not User channel               SYSS.362
       103007
1276   014000 /000000  034456           (ML) = IUCIO                 .Interrupt handler addr            SYSS.363
1301   070003                          P = AMNENT                                                       SYSS.364
1302                          AMDESC   *                                                                SYSS.365
1302   026456                          ML = ML + 1                                                      SYSS.366
1303   026454                          MJ = MJ + 1                  .NEXT CHAN                         SYSS.367
1304   026454                 AMNENT   MJ = MJ + 1                  .NEXT CHANNEL                      SYSS.368
1305   026456                          ML = ML + 1                  .NEXT EITB CELL                    SYSS.369
1306   020455  023454  100003          P = ANEXT, MK > MJ           .Done with all?                    SYSS.370
       102002  071232
```

```
131.   014000 /000000  024410          P = PBMX, (BXCR)              .If no CRT channel for the KERNEL    SYSS.37
       030410  102031
1320   014000 /000000  024410          HB = (BXCR) > 1 & 3 + BCRT    .CRT table entry                     SYSS.372
       030410  004001  011003
       016000 /000000  024000
1331    <macro>                        $PUNTIF   ((HB) = 0)          .If KERNEL CRT not in AMAP           SYSS.373
1334   030000  024000                  HB = (HB)                                                          SYSS.374
1336    <macro>                        $PUNTIF   ((BXCR) # (HB + 12))  .If CRT channels do not match      SYSS.375
1350                          PBMX     *                                                                  SYSS.376
                                                                                                          SYSS.377
                                       *                                                                  SYSS.378
                                       * BUILD BMX MESSAGE QUEUES                                         SYSS.379
                                       *                                                                  SYSS.380
                                                                                                          SYSS.381
1350    <macro>                        $IF (I@BMXSYS # 0)            .IF DEFINED                          SYSS.382
                                                                                     .>>>>> 0                $IF      1
1353    <macro>                          $IF (R!%MYID = BIOP) OR (R!%MYID = DIOP)                         SYSS.383
                                                                                     .>>>>> 1                $IF      1
1362    <macro>                            $IF (R!%HISP # 0)           .IF HIGH SPEED CHANNEL             SYSS.384
                                                                                     .>>>>> 2                $IF      1
1365   010040  024000                        HB = BMXQSZI              .LENGTH                            SYSS.385
1367   072251                                R = ABLD                  .BUILD XIOP MESSAGE QUEUE          SYSS.386
1370   014000 /000000  024410                (MBDQ) = HC               .                                  SYSS.387
       020000  034410
1375    <macro>                            $ENDIF                                                         SYSS.388
                                                                                     .<<<<< 2                $ENDIF   1
1375    <macro>                          $ELSEIF (R!%MYID = XIOP)                                         SYSS.389
1402   010040  024000                      HB = BMXQSZI              .SIZE OF QUEUE                     SYSS.390
1404   072234                              R = ABLD                  .BUILD MIOP MESSAGE QUEUE          SYSS.391
1405   014000 /000000  024410              (MBAQ) = HC               .                                  SYSS.392
       020000  034410
1412   010040  024000                      HB = BMXQSZI              .SIZE OF QUEUE                     SYSS.393
1414   072224                              R = ABLD                  .BUILD BIOP MESSAGE QUEUE          SYSS.394
1415   014000 /000000  024410              (MBBQ) = HC               .                                  SYSS.395
       020000  034410
1422   010040  024000                      HB = BMXQSZI              .SIZE OF QUEUE                     SYSS.396
1424   072214                              R = ABLD                  .BUILD DIOP QUEUE                  SYSS.397
1425   014000 /000000  024410              (MBCQ) = HC               .                                  SYSS.398
       020000  034410
1432    <macro>                          $ENDIF                                                         SYSS.399
                                                                                     .<<<<< 1                $ENDIF   1
1432    <macro>                        $ENDIF                                                             SYSS.400
```

```
                                                                                .<<<<< 0          $ENDIF    1
                                                                                                  SYSS.401
                                        *                                                         SYSS.402
                                        * TRACE INITIALIZATION                                   SYSS.403
                                        *                                                         SYSS.404
                                                                                                  SYSS.405
                                        * LOCAL MEMORY BUFFER                                    SYSS.406
                                                                                                  SYSS.407
1432   014000 /001000  024000           HB = LBUFL                  .MEMORY SIZE                 SYSS.408
1435   072230                           R = GTS                     .Get table space             SYSS.409
1436   014000 /000000  024410           (TLOC) = HC                 .Save base address           SYSS.410
       020000  034410
                                                                                                  SYSS.411
                                        * MOS BUFFER                                             SYSS.412
                                                                                                  SYSS.413
1443   014000 /020000  012777           R!CA = MBUFL + O'777 > D'9  .Trace buffer sector size    SYSS.414
       004011  024000
1450   072236                           R = STEALIT                 .Allocate buffer memory      SYSS.415
1451   020000  024437                   R!IA = R!CB                                              SYSS.416
1453   020000  024440                   R!IB = R!CC                                              SYSS.417
1455   014000 /000000  024410           (TMOSU) = IA                .ENTER UPPER ADDRESS         SYSS.418
       020437  034410
1462   014000 /000000  024410           (TMOSL) = IB                .ENTER LOWER ADDRESS         SYSS.419
       020440  034410
                                                                                                  SYSS.420
                                        * CLEAR MOS BUFFER                                       SYSS.421
                                                                                                  SYSS.422
1467   014000 /001000  004002           ME = LBUFL > 2              .LOCAL BUFFER WORDS          SYSS.423
       024447
1473                           BCLR     *                                                        SYSS.424
1473   020437  024000                   CB = IA                     .UPPER ADDRESS               SYSS.425
1475   020440  024000                   CC = IB                     .LOWER ADDRESS               SYSS.426
1477   014000 /000000  024410           CD = (TLOC)                 .LOCAL ADDRESS               SYSS.427
       030410  024000
1504   014000 /001000  004002           CE = LBUFL > 2              .WORDS TO WRITE              SYSS.428
       024000
1510   010004  024000                   CA = MIO$BMWI               .WRITE - NO WAIT             SYSS.429
1512    <macro>                         RKERN COMIO                                              SYSS.430
1514    <macro>                         DADD IA,IB,CE               .INCREMENT ADDRESS           SYSS.431
1521   020447  022000  024447           ME = ME + CE                .COUNT WRITTEN CHUNKS        SYSS.432
1524   041000  017000 /020000           P = BCLR, A < MBUFL         .LOOP TILL BUFFER CLEARED    SYSS.433
       104034
```

```
                                                                                                SYSS.43
                                 *                                                              SYSS.435
                                 * ALLOCATE BUFFER FOR SUMMARY                                  SYSS.436
                                 *                                                              SYSS.437
                                                                                                SYSS.438
1530  010003  024000             R!CA = SUM$IBL                .Number of summary buffers       SYSS.439
1532  072154                     R = STEALIT                                                    SYSS.440
1533  014000 /000000  024423     R!%W1 = SUMMORE               .ADDRESS SAVE CELL               SYSS.441
1536  020000  034423             (R!%W1) = R!CB                .                                SYSS.442
1540  020423  012001  024410     (R!%W1 + 1) = R!CC            .                                SYSS.443
      020000  034410
                                                                                                SYSS.444
                                                                                                SYSS.445
                                 * FINISH INITIALIZATION                                        SYSS.446
                                                                                                SYSS.447
                                 *       INITIALIZE PERIPHERALS, CLOCK                          SYSS.448
1545  147002                     PXS: 7                        .ENABLE EXIT STACK INTERRUPTS    SYSS.449
1546  147003                     LME: 7                        .ENABLE LOCAL MEMORY ERRORS      SYSS.450
1547  147004                     RTC: 7                        .ENABLE REAL TIME CLOCK          SYSS.451
                                                                                                SYSS.452
                               *  Create first overlay to execute.                              SYSS.453
                                                                                                SYSS.454
1550  014000 /000000  024000     EJ = O$BEGIN                  .Finish dead start process       SYSS.455
1553  010000  024060             EK = 0                                                         SYSS.456
1555  014000 /177777  024000     R!EL = AD$NODEM               .NOT DEMON ACTIVITY              SYSS.457
1560   <macro>                   RKERN EREB                    .INITIALIZE NEW ACTIVITY         SYSS.458


                               *  Copy the debugger into a disk buffer if it is active.         SYSS.460
                                                                                                SYSS.461
1562   <macro>                   $IF       (R!%DEBUG # 0)      .If Debugger is active           SYSS.462
                                                                          .>>>>> 0                $IF      1
1565   <macro>                     RKERN   HGETBUF             .Allocate a local disk buffer    SYSS.463
1567   <macro>                     $PUNTIF (R!DY = 0)          .If no buffer available          SYSS.464
1572   <macro>                     COPY    R!%DEBUG,R!DY,DEBLEN<2                               SYSS.465
1612  020000  024000               R!%DEBUG = R!DY                                              SYSS.466
1614  014000 /000000  005002       R!%W1 = DEBSTART < 2        . TEMP ************************  SYSS.467
      024423
1620  020000  023423  024000       R!%DEBBA = R!%DEBUG - R!%W1 . TEMP ************************  SYSS.468
1623   <macro>                   $ENDIF                                                         SYSS.469
```

```
                                                                                .<<<<< 0          $ENDIF   1
1623    <macro>                 $IF (R!%MYID = BIOP)        .MAKE SURE CRAY QUIET                SYSS.470
                                                                                .>>>>> 0          $IF      1
1627    <macro>                   RKERN STOPCRAY                                                 SYSS.471
1631    <macro>                 $ENDIF                                                           SYSS.472
                                                                                .<<<<< 0          $ENDIF   1


                        *       Initialization is complete.  Enter the Kernel idle loop.         SYSS.474
                                                                                                 SYSS.475
1631    010001  024000          R!%INIT = 1                 .Indicate initialization complete    SYSS.476
1633    014000 /177777  024000  R!%ACTIVE = AD$IDLE         .SHOW NO ACTIVITY RUNNING            SYSS.477
1636    <macro>                 PKERN     ESWP              .Enter Idle Loop                     SYSS.478
```

```
                                        * GET MEMORY AND INITIALIZE MESSAGE QUEUE                    SYSS.480
                                        *  ENTER WITH HB = SIZE                                      SYSS.481
1640                            ABLD    *                                                            SYSS.482
1640  072025                            R = GTS                     .Get and clear table space       SYSS.483
1641  020000  012012  034000            (HC) = HC + 12              .SET UP MSG HEAD                 SYSS.484
1644  020000  012001  024410            (HC + 1) = HC + HB          .TAIL                            SYSS.485
      020000  022000  034410
1652  020000  012002  024410            (HC + 2) = (HC)             .CURRENT POINTER                 SYSS.486
      030000  034410
1657  020000  012003  024410            (HC + 3) = (HC)             .NEXT TO FILL CELL PTR.          SYSS.487
      030000  034410
1664  001000                            EXIT                        .YES, EXIT                       SYSS.488


                                ***     GTS - Get table space.                                       SYSS.490
                                *                                                                    SYSS.491
                                *       Entry     HB = Space needed.                                 SYSS.492
                                *       Exit      HC = First parcial address of allocated space.     SYSS.493
                                *                 To location PUNT if space is not available.        SYSS.494
                                *                 Note - the space is cleared out.                   SYSS.495
                                                                                                     SYSS.496
                                                                                                     SYSS.497
1665                            GTS     *                                                            SYSS.498
1665   <macro>                          RKERN     HMA                 .A.locate local memory         SYSS.499
1667   <macro>                          $PUNTIF   (R!HC = 0),CODE=PT$LOCAL  .If memory not available SYSS.500
                                                                                                     SYSS.501
1672   <macro>                          CLEAR     START=HC,COUNT=HB   .Clear out entry.              SYSS.502
1705  001000                            EXIT                          .Return                        SYSS.503


                                *       STEALIT   Allocate permanent buffer-memory buffer            SYSS.505
                                *                                                                    SYSS.506
                                *       This routine allocates a permanent buffer in buffer memory.  SYSS.507
                                *       The bits corresponding to the buffer are, in effect, removed SYSS.508
                                *       from the buffer memory allocation table and thus the buffers SYSS.509
                                *       cannot be released.  Note that the buffer memory allocation  SYSS.510
                                *       table must be initialized before this routine is called.     SYSS.511
                                *                                                                    SYSS.512
```

```
                                *       Entry:    R!CA - Number of 512-word blocks requested          SYSS.513
                                *                                                                   SYSS.514
                                *       Exit:     R!CB,R!CC - Buffer address                          SYSS.515
                                                                                                    SYSS.516
1706                            STEALIT *                                                           SYSS.517
1706    <macro>                         $PUNTIF   (R!CA = 0)          .If zero-length buffer        SYSS.518
1711    014000 /000000  024423          R!%W1 = BBDISK                .Buffer memory allocation table SYSS.519
1714    <macro>                         GET       R!CC,BB@TOT,R!%W1   .Number of entries in map     SYSS.520
1721    <macro>                         GET       R!%W2,BB@MAP,R!%W1  .Bit map address              SYSS.521
                                                                                                    SYSS.522
1726    <macro>                         $UNTIL    (R!CA = 0)          .Until all sectors are allocated SYSS.523
1730    027000                            R!CA = R!CA - 1             .Decrement the allocation count SYSS.524
1731    027000                            R!CC = R!CC - 1             .Decrement available-entry count SYSS.525
1732    <macro>                           $PUNTIF (A = 0)             .If all space exhausted       SYSS.526
1734    <macro>                           PUTBIT  0,R!CC,R!%W2        .Clear bit in map             SYSS.527
1756    <macro>                         $ENDTIL                                                     SYSS.528
                                                                                                    SYSS.529
1757    <macro>                         PUT       R!CC,BB@TOT,R!%W1   .Update map size              SYSS.530
1764    010000  024000                  R!CB = 0                                                    SYSS.531
1766    <macro>                         GET       R!%W2,BB@SHF,R!%W1                                SYSS.532
1773    <macro>                         LSHIFT    R!CB,R!CC,R!%W2     .Calculate relative address   SYSS.533
2012    <macro>                         GET       R!%W2,BB@BAU,R!%W1                                SYSS.534
2017    <macro>                         GET       R!%W3,BB@BAL,R!%W1                                SYSS.535
2024    <macro>                         DADD      R!CB,R!CC,R!%W2,R!%W3  .Calculate real address  SYSS.536
2034    001000                          EXIT                                                        SYSS.537


                                *       INITMAP   Initialize allocation table                       SYSS.539
                                *                                                                   SYSS.540
                                *       INITMAP sets up the allocation table header, allocates local SYSS.541
                                *       memory for the bit map, and sets a bit for each entry.      SYSS.542
                                *                                                                   SYSS.543
                                *       Entry:    R!%W1 - Allocation table address                    SYSS.544
                                *                 R!%W2 - Number of entries in map                    SYSS.545
                                *                 R!%W3 - Entry size expressed as a power of 2        SYSS.546
                                *                 R!%W4,R!%W5 - Entry base address                    SYSS.547
                                                                                                    SYSS.548
2035                            INITMAP *                                                           SYSS.549
2035    <macro>                         $PUNTIF   (R!%W2 = 0)         .If no entries in table       SYSS.550
2040    <macro>                         PUT       R!%W2,BB@TOT,R!%W1  .Set up header fields         SYSS.551
```

```
2045    <macro>                         PUT       R!%W3,BL     ,R!%W1                                    SYSS.552
2052    <macro>                         PUT       R!%W4,BB@BAU,R!%W1                                     SYSS.553
2057    <macro>                         PUT       R!%W5,BB@BAL,R!%W1                                     SYSS.554
2064    020424  012017  004004          R!HB = R!%W2 + D'15 > 4        .Calculate parcel size of map      SYSS.555
        024000
2070    073203                          R = GIS                        .Allocate local memory for map     SYSS.556
2071    <macro>                         PUT       R!HC,BB@MAP,R!%W1                                      SYSS.557
                                                                                                         SYSS.558
2076    <macro>                         $UNTIL    (R!%W2 = 0)          .Until all entries are processed  SYSS.559
2100    027424                            R!%W2 = R!%W2 - 1                                              SYSS.560
2101    <macro>                           PUTBIT  1,R!%W2,R!HC         .Set appropriate bit in map       SYSS.561
2123    <macro>                         $ENDTIL                                                          SYSS.562
                                                                                                         SYSS.563
2124    001000                          EXIT                                                             SYSS.564


 426W                        0          BSS       0                                                      SYSS.566
2130                            AZZZ    *                              .END OF INITIALIZATION             SYSS.567
                                                                                                         SYSS.568
                                        END                                                              SYSS.569
```

0X	%ACTIVE		3:19 D	17:14						
0X	%B		2:12 D	2:13 D	5: 3	5:12	6: 6	8: 6 B	8:18 B	8:38 B
			9:19 B	9:33 B	9:41 B	10: 6 B	10:26 B	10:29 B	11:10 B	
0X	%DEBBA		3:20 D	16:41						
0X	%DEBUG		3:21 D	5: 6	16:34	16:37	16:38	16:41		
10	%GBLNUM	$APTEXT	2:16 F	2:16						
400	%GBLREG	$APTEXT	2:16							
0X	%HISP		3:22 D	13: 7	14:20					
0X	%HISPBUF		3:23 D	13:14						
0X	%INIT		3:24 D	17:13						
0	%KERNEL	$APTEXT	5:19 D	5:19 B	5:31 D	5:31 B	8: 8 D	8: 8 B	9:13 D	9:13 B
			13:12 D	13:12 B	13:21 D	13:21 B	15:38 D	15:38 B	16:26 D	16:26 B
			16:35 D	16:35 B	17: 4 D	17: 4 B	17:15 D	17:15 B	18:26 D	18:26 B
0X	%MEMORY		3:25 D	6: 6	6: 7	6: 8	6: 9	6:10	6:11	6:12
			6:14	6:17	6:18					
0X	%MYID		3:26 D	5:33	8:28	8:32	9: 3	10:17	14:18	14:27
			17: 3							
0X	%OVPT		3:27 D	5:24						
0X	%PUNT		5:36 D	5:36	7:36 D	7:36	11:13 D	11:13	13:13 D	13:13
			13:22 D	13:22	13:27 D	13:27	14: 6 D	14: 6	14: 8 D	14: 8
			16:36 D	16:36	18:27 D	18:27	19: 6 D	19: 6	19:14 D	19:14
			19:41 D	19:41						
0X	%SSD		3:28 D	13:18						
0X	%SSDBUF		3:29 D	13:23						
0X	%UCI		3:30 D	10: 8	10:10	10:11	10:12	10:13		
177777	AD$IDLE	$APTEXT	17:14							
177777	AD$NODEM	$APTEXT	16:25							
20	BB@BAL@N	$APTEXT	19:23	19:23 F	20: 3 F					
6	BB@BAL@P	$APTEXT	19:23 F	19:23	20: 3 F	20: 3				
0	BB@BAL@S	$APTEXT	19:23	19:23 F	20: 3 F					
20	BB@BAU@N	$APTEXT	19:22	19:22 F	20: 2 F					
5	BB@BAU@P	$APTEXT	19:22 F	19:22	20: 2 F	20: 2				
0	BB@BAU@S	$APTEXT	19:22	19:22 F	20: 2 F					
20	BB@MAP@N	$APTEXT	19: 9	19: 9 F	20: 7 F					
3	BB@MAP@P	$APTEXT	19: 9 F	19: 9	20: 7 F	20: 7				
0	BB@MAP@S	$APTEXT	19: 9	19: 9 F	20: 7 F					
20	BB@SHF@N	$APTEXT	19:20	19:20 F	20: 1 F					
4	BB@SHF@P	$APTEXT	19:20 F	19:20	20: 1 F	20: 1				
0	BB@SHF@S	$APTEXT	19:20	19:20 F	20: 1 F					
20	BB@IOI@N	$APTEXT	19: 8	19: 8 F	19:18 F	19:42 F				
2	BB@IOI@P	$APTEXT	19: 8 F	19: 8	19:18 F	19:18	19:42 F	19:42		
0	BB@IOI@S	$APTEXT	19: 8	19: 8 F	19:18 F	19:42 F				

0X	BBDAL		3:31 D	8:37						
0X	BBDISK		3:32 D	9:18	19: 7					
0X	BBSOFT		3:33 D	9:34						
0X	BCRT		3:34 D	12:22	14: 5					
1	BIOP	$APTEXT	14:18	17: 3						
40	BMXQSZI	$APTEXT	14:21	14:28	14:32	14:36				
0X	BXCR		3:35 D	14: 1	14: 3	14: 8				
0X	CA		3:36 D	5:18	5:30	9:12	9:31	15:17	15:37	16: 6
			19: 6	19:11	19:12					
0X	CB		3:37 D	5:14	5:26	9:39	15:19	15:31	16: 9	19:19
			19:21	19:24						
0X	CC		3:38 D	5:15	5:27	9:40	15:20	15:32	16:10	19: 8
			19:13	19:15	19:18	19:21	19:24			
0X	CD		3:39 D	5:16	5:29	15:34				
0X	CE		3:40 D	5:17	5:22	5:28	15:35	15:39	15:40	
23	CH$AMPEX	$APTEXT	12:16							
20	CH$CO	$APTEXT	12:12							
21	CH$C1	$APTEXT	12:10							
35	CH$CH	$APTEXT	13: 1							
22	CH$CS	$APTEXT	12:14							
0X	COMIO		5:19 D	5:19	5:31 D	5:31	9:13 D	9:13	15:38 D	15:38
70	CRTABSIZ	$APTEXT	12:19							
0X	DEBLEN		3:41 D	16:37						
0X	DEBSTART		3:42 D	16:39						
2	DIOP	$APTEXT	14:18							
0X	DY		4: 1 D	8: 7	13:13	13:14	13:22	13:23	16:36	16:37
			16:38							
0X	EACT		4: 2 D	8:21						
0X	EBUF		4: 3 D	7: 9						
0X	EBUFSIZ		4: 4 D	7:12						
0X	ETTB		4: 5 D	11:40						
0X	EJ		4: 6 D	16:23						
0X	EK		4: 7 D	16:24						
0X	EL		4: 8 D	16:25						
0X	EMEM		4: 9 D	7:28						
0X	EMEMSIZ		4:10 D							
0X	ENDBOOT		4:11 D	5:15						
0X	ERTB		16:26 D	16:26						
0X	ESMD		4:12 D	8:19						
0X	ESWP		17:15 D	17:15						
0X	HB		4:13 D	8: 5	8:17	10: 5	10:25	10:28	12: 4	12: 5
			12: 7	12:10	12:12	12:14	12:16	12:19	14: 5	14: 6

			14: 7	14: 8	14:21	14:28	14:32	14:36	15: 9	18: 7
			18:29	20: 4						
0X	HC		4:14 D	8: 7	8:19	8:21	10: 8	10:27	10:30	12:23
			12:24	12:26	12:28	14:23	14:30	14:34	14:38	15:11
			18: 5	18: 6	18: 8	18:10	18:27	18:29	20: 7	20:11
0X	HGETBUF		13:12 D	13:12	13:21 D	13:21	16:35 D	16:35		
0X	HMA		18:26 D	18:26						
0X	HONN		8: 8 D	8: 8						
1	I@BMXSYS	$APTEXT	14:16							
103	I@IOPMOD	$APTEXT	13:11	13:20						
0X	IUCIO		4:15 D	13:34						
1000	LBUFL	$APTEXT	15: 9	15:28	15:35					
3	LME ,	$APTEXT	16:18							
0X	MAAQ		4:16 D	10:20						
0X	MAOQ		4:17 D	10:22						
0X	MATT		4:18 D	11: 8						
0X	MATX		4:19 D	11: 1						
0X	MBAQ		4:20 D	11:21	14:30					
0X	MBBQ		4:22 D	14:34						
0X	MBCQ		4:23 D	14:38						
0X	MBDQ		4:24 D	14:23						
20000	MBUFL	$APTEXT	15:16	15:41						
1	MD$FREE	$APTEXT	6:16							
0	MD$HEAD	$APTEXT	6: 8	6:28						
46504	MD$ID	$APTEXT	6: 7	6:15	6:27					
20	MD@BAK@N	$APTEXT	6:12 F	6:18 F						
5	MD@BAK@P	$APTEXT	6:12 F	6:12	6:18 F	6:18				
0	MD@BAK@S	$APTEXT	6:12 F	6:18 F						
20	MD@FOR@N	$APTEXT	6:11 F	6:18	6:18 F					
4	MD@FOR@P	$APTEXT	6:11 F	6:11	6:18 F	6:18				
0	MD@FOR@S	$APTEXT	6:11 F	6:18	6:18 F					
20	MD@ID@N	$APTEXT	6: 7 F	6:15 F	6:27 F					
0	MD@ID@P	$APTEXT	6: 7 F	6:15 F	6:27 F					
0	MD@ID@S	$APTEXT	6: 7 F	6:15 F	6:27 F					
10	MD@LEN@P	$APTEXT	6:14	6:25	6:31					
20	MD@PRE@N	$APTEXT	6: 9 F	6:17 F	6:29 F					
2	MD@PRE@P	$APTEXT	6: 9 F	6: 9	6:17 F	6:17	6:29 F	6:29		
0	MD@PRE@S	$APTEXT	6: 9 F	6:17 F	6:29 F					
20	MD@SUC@N	$APTEXT	6:10 F	6:17	6:17 F	6:29	6:29 F			
1	MD@SUC@P	$APTEXT	6:10 F	6:10	6:17 F	6:17	6:29 F	6:29		
0	MD@SUC@S	$APTEXT	6:10 F	6:17	6:17 F	6:29	6:29 F			
20	MD@TYP@N	$APTEXT	6: 8 F	6:16 F	6:28 F					

3	MD@TYP@P	$APTEXT	6: 8	F	6: 8			F	6:16		6:28	F	6:28				
0	MD@TYP@S	$APTEXT	6: 8	F	6:16	F	6:28	F									
0X	MIAQ		4:21	D	11:23												
1	MIO$BMR	$APTEXT	5:18		5:30												
4	MIO$BMWI	$APTEXT	9:12		15:37												
226	MSGQSZI	$APTEXT	10:25														
100	MSGQSZO	$APTEXT	10:28														
0X	O$AMAP		4:25	D	5:24												
0X	O$BEGIN		4:26	D	16:23												
0X	O$SYSS		1: 4	D	1: 4												
2	OT$POWER	$APTEXT	5:24														
20	OT@MLO@N	$APTEXT	5:27		5:27	F											
1	OT@MLO@P	$APTEXT	5:27	F	5:27												
0	OT@MLO@S	$APTEXT	5:27		5:27	F											
17	OT@MUP@M	$APTEXT	5:26														
4	OT@MUP@N	$APTEXT	5:26														
0	OT@MUP@P	$APTEXT	5:26	F													
14	OT@MUP@S	$APTEXT	5:26		5:26	F											
14	OT@WRD@N	$APTEXT	5:28		5:28	F											
0	OT@WRD@P	$APTEXT	5:28	F													
0	OT@WRD@S	$APTEXT	5:28		5:28	F											
17	OV@NUM@N	$APTEXT	1: 4														
1	OV@TYP@N	$APTEXT	1: 4														
11000	OVL$MIN	$APTEXT	6:22		6:23												
113	PT$BADTM	$APTEXT	13:27														
0	PT$HUH	$APTEXT	5:36		7:36		11:13		14: 6		14: 8		16:36		19: 6		19:14
			19:41														
14	PT$LOCAL	$APTEXT	13:13		13:22		18:27										
2	PXS	$APTEXT	16:17														
4	RTC	$APTEXT	16:19														
100	SM$SIZE	$APTEXT	8:17		9:27	F											
20	SM@ACT@N	$APTEXT	8:21	F													
0	SM@ACT@P	$APTEXT	8:21	F													
0	SM@ACT@S	$APTEXT	8:21	F													
11	SM@FST@N	$APTEXT	1: 4														
7	SM@NUM@N	$APTEXT	1: 4														
0X	STORCRAY		4:27	D	17: 4	D	17: 4										
3	SUM$TBL	$APTEXT	16: 6														
0X	SUMMORE		4:28	D	16: 8												
0X	TLOC		4:29	D	15:11		15:33										
0	TM$CMEM	$APTEXT	13: 6														
2	TM$SSD	$APTEXT	13:17														

```
   0X  TMOSL                  4:30 D   15:23
   0X  TMOSU                  4:31 D   15:21
52503  UC$NAM     $APTEXT    10:10
   20  UC@FSI@N   $APTEXT    10:12 F
    2  UC@FSI@P   $APTEXT    10:12 F   10:12
    0  UC@FSI@S   $APTEXT    10:12 F
    4  UC@HDR@P   $APTEXT    10: 5
   20  UC@LSI@N   $APTEXT    10:13 F
    3  UC@LSI@P   $APTEXT    10:13 F   10:13
    0  UC@LSI@S   $APTEXT    10:13 F
   20  UC@NAM@N   $APTEXT    10:10 F
    0  UC@NAM@P   $APTEXT    10:10 F
    0  UC@NAM@S   $APTEXT    10:10 F
   20  UC@POP@N   $APTEXT    10:11 F
    1  UC@POP@P   $APTEXT    10:11 F   10:11
    0  UC@POP@S   $APTEXT    10:11 F
   0X  XHSPCH                 4:32 D   13: 8
    3  XIOP       $APTEXT    14:27

       SYMBOL QUALIFIER IS SYSS

    0  $CONST                 6: 7 D    6: 7 F    6: 7      6: 8 D    6: 8 F    6:15 D    6:15 F    6:15
                              6:16 D   6:16 F    6:16      6:27 D   6:27 F    6:27      6:28 D   6:28 F
                              8:21 D   8:21 F    8:21     10:10 D  10:10 F   10:10     10:11 D  10:11 F
                             10:12 D  10:12 F   10:13 D   10:13 F
  462  $REGORG                2:16 D   2:16 F    2:17      2:17 D    2:18      2:18 D    2:19      2:19 D
                              2:20     2:20 D    2:21      2:21 D   2:21 F    2:22      2:22 D    2:23
                              2:23 D   2:24      2:24 D    2:25      2:25 D    2:26      2:26 D    2:27
                              2:27 D   2:27 F    2:28      2:28 D    2:29      2:29 D    2:30      2:30 D
                              2:31     2:31 D    2:32      2:32 D   2:32 F    2:33      2:33 D    2:34
                              2:34 D   2:35      2:35 D    2:36      2:36 D    2:37      2:37 D    2:38
                              2:38 D   2:39      2:39 D    2:40      2:40 D    2:41      2:41 D    2:42
                              2:42 D   3: 1      3: 1 D    3: 2      3: 2 D    3: 3      3: 3 D    3: 4
                              3: 4 D   3: 5      3: 5 D    3: 6      3: 6 D    3: 7      3: 7 D    3: 8
                              3: 8 D   3: 9      3: 9 D   3:10      3:10 D    3:11      3:11 D   3:12
                              3:12 D  3:13      3:13 D   3:14      3:14 D    3:15      3:15 D   3:16
                              3:16 D  3:16 F
    0  %NP                    2:32 D
  430  %P                     2:32 D   2:32
  410  %S1                    2:16     2:17 D    5:27      6: 8      6: 9      6:10      6:11      6:12
                              6:16     6:17      6:18      6:20      6:28      6:29      7: 4      7:10
                              7:11     7:12      7:13      7:22      7:23      7:32      7:34      8: 1
```

		8:19	8:20		9: 5	9:16	9:30	9:35	10:11
		10:12	10:13	10:34	10:35	10:40	11: 5	11: 6	11:32
		11:33	12: 5	12: 6	12:24	12:25	12:26	12:27	12:28
		12:29	13: 1	13: 2	13: 8	13: 9	14: 1	14: 3	14: 8
		14:23	14:24	14:30	14:31	14:34	14:35	14:38	14:39
		15:11	15:12	15:21	15:22	15:23	15:24	15:33	16:10
		16:11	18: 6	18: 7	18: 8	18: 9	18:10	18:11	19: 8
		19: 9	19:18	19:20	19:22	19:23	19:42	20: 1	20: 2
		20: 3	20: 7						
411	%S2	2:16	2:18 D	14: 8					
412	%S3	2:16	2:19 D						
413	%S4	2:16	2:20 D						
414	%S5	2:16	2:21 D						
415	%T1	2:22 D	6:17	6:18	6:29	16:37	18:29	19:15	19:21
		20:11							
416	%T2	2:23 D	16:37	18:29	19:15	20:11			
417	%T3	2:24 D	16:37						
420	%T4	2:25 D							
421	%T5	2:26 D							
422	%T6	2:27 D							
423	%W1	2:28 D	5: 3	5: 6	5: 7	5: 8	5:25	5:26	5:27
		5:28	6:14	6:15	6:16	6:17	6:18	6:25	6:29
		8:28	8:30	8:32	8:34	8:35	8:37	9: 4	9: 5
		9:15	9:16	9:18	9:34	13: 3	13: 6	13:17	16: 8
		16: 9	16:10	16:39	16:41	19: 7	19: 8	19: 9	19:18
		19:20	19:22	19:23	19:42	20: 1	20: 2	20: 3	20: 7
424	%W2	2:29 D	6:21	6:22	6:23	6:26	8:30	9: 6	9:37
		19: 9	19:15	19:20	19:21	19:22	19:24	19:41	19:42
		20: 4	20: 9	20:10	20:11				
425	%W3	2:30 D	6:25	6:26	6:27	6:28	6:29	6:31	8:31
		9:14	9:38	19:23	19:24	20: 1			
426	%W4	2:31 D	8:34	9:15	9:39	20: 2			
427	%W5	2:32 D	8:36	9:17	9:40	20: 3			
1640	ABLD	10:26	10:29	14:22	14:29	14:33	14:37	18: 3 D	
665	ABMM	10:23 D	10:35						
706	ABMS	10:23	10:24	10:31 D					
1032	AMAM	11:10	11:29 D						
1034	AMAO	11:28	11:30 D						
735	AMCL	11: 3 D	11: 6						
1126	AMCRT	12:10	12:12	12:14	12:18 D				
1302	AMDESC	12:41	13:30	13:36 D					
1177	AMHEGHS	12: 6	12:16	12:42 D					

1304	AMNENT	12: 3	13:32	13:35	13:39 D				
754	AMOR	11: 9 D	11:33						
1272	AMUCHN	13: 2	13:31 D						
1060	ANEXT	11:42 D	13:42						
407	APAS	7: 6	7:27 D						
363	ARPB	7:15 D	7:20						
376	ARPF	7:16	7:21 D						
2130	AZZZ	5: 3	5:12	20:20 D					
1473	BCLR	15:30 D	15:41						
430	BD	2:33 D	12:22	12:23					
431	BE	2:34 D	12: 9	12:11	12:13	12:15	12:26		
432	BF	2:35 D							
433	CF	2:36 D							
434	EA	2:37 D							
435	EB	2:38 D							
436	EO	2:39 D							
1665	GTS	8: 6	8:18	10: 6	12:20	15:10	18: 4	18:25 D	20: 6
437	IA	2:40 D	7: 7	7:14	7:17	7:19	7:21	7:22	7:24
		7:32	7:36	15:19	15:21	15:31	15:39		
440	IB	2:41 D	12: 1	12:21	12:28	12:30	13: 7	13: 8	13:11
		13:18	13:20	15:20	15:23	15:32	15:39		
441	IC	2:42 D							
442	ID	3: 1 D							
2035	INITMAP	8:38	9:19	9:41	19:40 D				
443	MA	3: 2 D	5:12	5:16	5:22	8:32	9: 3		
444	MB	3: 3 D	7:27	7:29	7:31	7:32	7:33	7:36	10:17
		10:24							
445	MC	3: 4 D	8: 2	8: 4	8: 9	10:19	10:23	10:24	10:33
		11:24	11:27						
446	MD	3: 5 D	5:22	5:29	5:33	5:37	8:28	10:17	10:19
		10:42							
447	ME	3: 6 D	7: 5	7: 6	7: 7	7:10	7:12	7:15	7:16
		7:28	7:29	7:30	7:35	10:21	10:27	10:31	10:34
		11:22	11:26	15:28	15:40				
450	MF	3: 7 D	10:22	10:30	10:32	11:16	11:18	11:25	
451	MG	3: 8 D	6:31	7:27					
452	MH	3: 9 D	5:33	5:35	5:36	5:37	6:20	7: 4	8: 1
		9:11	9:30	9:35	10:40				
453	MI	3:10 D	10:41	10:42	11: 7	11:16	11:32	11:37	11:38
		12: 1							
454	MJ	3:11 D	11: 7	11: 9	11:16	11:31	11:32	11:37	12: 1
		12: 3	12: 4	13: 1	13: 3	13:32	13:38	13:39	13:41

455	MK	3:12 D	11: 2	3	11: 4	11: 5	11: 8	11:14	11:29
		11:30	11:38	13:41					
456	ML	3:13 D	11: 9	11:10	11:11	11:13	11:14	11:15	11:19
		11:21	11:23	11:40	13:34	13:37	13:40		
457	MM	3:14 D	11:15	11:18	11:20	11:25			
460	MN	3:15 D	11: 1	11: 2	11: 5	11:15			
461	MP	3:16 D	7: 9	7:10	7:14	7:22	7:24	10:20	10:21
		10:34	11:19						
1350	PBMX	14: 2	14: 9 D						
1706	STEALIT	9:33	15:18	16: 7	19: 5 D				

```
0     <macro>                    OVERLAY   ACOM                                                    ACOM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ***************************************************************  ACOM.5
                    *                                                              *  ACOM.6
                    *        ACOM is a demon overlay that handles inter-IOP communication *  ACOM.7
                    *        for DCU4 disks.  The function codes supported are:    *  ACOM.8
                    *                                                              *  ACOM.9
                    *                  MC$MSTIO - Start disk I/O                   *  ACOM.10
                    *                  MC$RLDAL - Release MOS DAL                  *  ACOM.11
                    *                  MC$CPURD - Move data from target memory to MOS *  ACOM.12
                    *                  MC$CPUWR - Move data from MOS to target memory *  ACOM.13
                    *                  MC$CSTAT - Send packet to mainframe         *  ACOM.14
                    *                  MC$CPRDN - Target memory to MOS move done   *  ACOM.15
                    *                  MC$CPWDN - MOS to target memory move done   *  ACOM.16
                    *                                                              *  ACOM.17
                    ***************************************************************  ACOM.18
                                                                                       ACOM.19
6   <macro>                 REGDEFS   ,,(LIM,ACT,QPTR,QUE,MSG,IFC,DAL,CHN,DCB,CYL,HED,SEC,  ACOM.20
                    ,                 FNC,IOP,SC0,SC1,SC2,SC3,CPU,CPL,LN0,LN1,OFF,LLN,TOT,  ACOM.21
                    ,                 TM,TMP)                                          ACOM.22
            410     %S1       EQUALS  $REGORG                                          $REG    3
            411     %S2       EQUALS  $REGORG                                          $REG    3
            412     %S3       EQUALS  $REGORG                                          $REG    3
            413     %S4       EQUALS  $REGORG                                          $REG    3
            414     %S5       EQUALS  $REGORG                                          $REG    3
            415     %T1       EQUALS  $REGORG                                          $REG    3
            416     %T2       EQUALS  $REGORG                                          $REG    3
            417     %T3       EQUALS  $REGORG                                          $REG    3
            420     %T4       EQUALS  $REGORG                                          $REG    3
            421     %T5       EQUALS  $REGORG                                          $REG    3
            422     %T6       EQUALS  $REGORG                                          $REG    3
            423     %W1       EQUALS  $REGORG                                          $REG    3
            424     %W2       EQUALS  $REGORG                                          $REG    3
            425     %W3       EQUALS  $REGORG                                          $REG    3
            426     %W4       EQUALS  $REGORG                                          $REG    3
            427     %W5       EQUALS  $REGORG                                          $REG    3
            430     LIM       EQUALS  $REGORG                                          $REG    3
            431     ACT       EQUALS  $REGORG                                          $REG    3
            432     QPTR      EQUALS  $REGORG                                          $REG    3
            433     QUE       EQUALS  $REGORG                                          $REG    3
            434     MSG       EQUALS  $REGORG                                          $REG    3
            435     IFC       EQUALS  $REGORG                                          $REG    3
            436     DAL       EQUALS  $REGORG                                          $REG    3
            437     CHN       EQUALS  $REGORG                                          $REG    3
```

```
440     DCB     EQUALS    $REGORG                                   $REG    3
441     CYL     EQUALS    $REGORG                                   $REG    3
442     HED     EQUALS    $REGORG                                   $REG    3
443     SEC     EQUALS    $REGORG                                   $REG    3
444     FNC     EQUALS    $REGORG                                   $REG    3
445     IOP     EQUALS    $REGORG                                   $REG    3
446     SC0     EQUALS    $REGORG                                   $REG    3
447     SC1     EQUALS    $REGORG                                   $REG    3
450     SC2     EQUALS    $REGORG                                   $REG    3
451     SC3     EQUALS    $REGORG                                   $REG    3
452     CPU     EQUALS    $REGORG                                   $REG    3
453     CPL     EQUALS    $REGORG                                   $REG    3
454     LN0     EQUALS    $REGORG                                   $REG    3
455     LN1     EQUALS    $REGORG                                   $REG    3
456     OFF     EQUALS    $REGORG                                   $REG    3
457     LLN     EQUALS    $REGORG                                   $REG    3
460     TOT     EQUALS    $REGORG                                   $REG    3
461     TM      EQUALS    $REGORG                                   $REG    3
462     TMP     EQUALS    $REGORG                                   $REG    3
                                                                ACOM.23
                EXT       %MYID                                 ACOM.24
                EXT       %LSPI                                 ACOM.25
                EXT       CA                                    ACOM.26
                EXT       CB                                    ACOM.27
                EXT       CC                                    ACOM.28
                EXT       CD                                    ACOM.29
                EXT       CE                                    ACOM.30
                EXT       D$ACOM                                ACOM.31
                EXT       D$AMSG                                ACOM.32
                EXT       DA                                    ACOM.33
                EXT       DB                                    ACOM.34
                EXT       DC                                    ACOM.35
                EXT       DCCB                                  ACOM.36
                EXT       DH                                    ACOM.37
                EXT       DS                                    ACOM.38
                EXT       DT                                    ACOM.39
                EXT       DY                                    ACOM.40
                EXT       DZ                                    ACOM.41
                EXT       ED                                    ACOM.42
                EXT       EDES                                  ACOM.43
                EXT       EH                                    ACOM.44
                EXT       EI                                    ACOM.45
```

```
EXT     IMDAL                                                   ACOM.46
EXT     MAAQ                                                    ACOM.47
EXT     MIMDALQ                                                 ACOM.48
EXT     MQEN                                                    ACOM.49
EXT     TMCB                                                    ACOM.50
```

```
                                        *  Initialize for scanning all A <=> A input channel queues.      ACOM.52
                                                                                                          ACOM.53
 6   010404  024431                     R!ACT = 404                  .Force scan of all queues           ACOM.54
10   014000 /000000  024432             R!QPTR = MAAQ                .MIOP queue                         ACOM.55
13    <macro>                           $IF (R!%MYID = MIOP),OR,(R!%MYID = DIOP)                         ACOM.56
                                                                                        .>>>>> 0            $IF      1
21   010004  025432                       R!QPTR = R!QPTR + 4        .Look at BIOP first                 ACOM.57
23    <macro>                           $ELSE                                                            ACOM.58
24   010010  025432                       R!QPTR = R!QPTR + D'8      .Look at DIOP first                 ACOM.59
26    <macro>                           $ENDIF                                                           ACOM.60
                                                                                        .<<<<< 0            $ENDIF   1
26   010010  012001  024430             R!LIM = ACOM$LIM + 1         .Processing limit                   ACOM.61
31   070006                             P = MEMQ                     .First pass                         ACOM.62
                                                                                                          ACOM.63
                                        *  Check for messages.                                           ACOM.64
                                                                                                          ACOM.65
32                              MEMN    *                                                                ACOM.66
32   010000  024431                     R!ACT = 0                    .Snow one pass already made         ACOM.67
34   014000 /000000  024432             R!QPTR = MAAQ                .MIOP Queue                         ACOM.68
37                              MEMQ    *                                                                ACOM.69
37   030432  024433                     R!QUE = (R!QPTR)             .Queue address                      ACOM.70
41   102010                             P = MEMS, A = 0              .Sense no queue                     ACOM.71
42   002000  000000                     I = 0                                                            ACOM.72
44   020433  012004  024410             P = MEMX, (R!QUE + QU$POP) # 0 .If any entries to process       ACOM.73
     030410  103015
51                              MEMS    *                                                                ACOM.74
51   010004  025432                     R!QPTR = R!QPTR + 4          .Next queue                         ACOM.75
53   020432  017000 /000000             P = MEMQ, R!QPTR # MQEN      .If more queues to check            ACOM.76
     107017
57    <macro>                           $IF (R!ACT = 0)              .If full pass has been made         ACOM.77
                                                                                        .>>>>> 0            $IF      1
62    <macro>                             PKERN ESWP                 .Idle down                          ACOM.78
64    <macro>                           $ENDIF                                                           ACOM.79
                                                                                        .<<<<< 0            $ENDIF   1
64   071032                             P = MEMN                     .Check again                        ACOM.80
65                              MEMX    *                                                                ACOM.81
65   003000                             I = 1                                                            ACOM.82
66   027430                             R!LIM = R!LIM - 1            .Messages processed                 ACOM.83
67   002000  000000                     I = 0                                                            ACOM.84
71    <macro>                           $IF (A = 0)                  .If giveup limit reached            ACOM.85
                                                                                        .>>>>> 0            $IF      1
73   014000 /000000                       A = D$ACOM                 .ACOM demon pointer                 ACOM.86
```

```
        <macro>                          RKERN DGIVEUP          .Relinquish control if necessary     ACOM.8
  77    010010  024430                   R!LIM = ACOM$LIM       .Reload limit                       ACOM.88
 101    <macro>                        $ENDIF                                                       ACOM.89
                                                                                .<<<<< 0                $ENDIF   1
 101    010404  024431                 R!ACT = 404              .Set processing flag                ACOM.90
 103    014000 /000000  012002         R!%W1 = EDES + 2         .DAL population                     ACOM.91
        024423
 107    <macro>                        $IF ((R!%W1) < 4)        .If running low                     ACOM.92
                                                                                .>>>>> 0                $IF      1
 113    <macro>                          PAUSE 1,A1=0,A2=-1     .Give up control                    ACOM.93
 123    010010  024430                   R!LIM = ACOM$LIM       .Reset limit                        ACOM.94
 125    071073                           P = MEMN               .Start over                         ACOM.95
 126    <macro>                        $ENDIF                                                       ACOM.96
                                                                                .<<<<< 0                $ENDIF   1
 126    <macro>                        RKERN HOFF               .Get a 'DAL'                        ACOM.97
 130    020000  024436                 R!DAL = R!DY             .DAL address                        ACOM.98
 132    <macro>                        $PUNTIF (A = 0)          .DAL count shows one available      ACOM.99
 134    020433  024000                 R!ED = R!QUE             .Queue                              ACOM.100
 136    <macro>                        RKERN ETOQ               .Take next entry                    ACOM.101
 140    020000  024434                 R!MSG = R!EH             .Accumulator msg                    ACOM.102
                                                                                                     ACOM.103
                                       * Check for queue overflow entry.                             ACOM.104
                                       * If there, queue it and re-enable input channel.             ACOM.105
                                                                                                     ACOM.106
 142    020433  012006  024423         R!%W1 = R!QUE + 6        .Overflow entry cell                ACOM.107
 145    030423  024000                 R!EH = (R!%W1)           .Overflow entry                     ACOM.108
 147    <macro>                        $IF (A # 0)              .If overflow                        ACOM.109
                                                                                .>>>>> 0                $IF      1
 151    020433  024000                   R!ED = R!QUE           .Put it on queue                    ACOM.110
 153    <macro>                          RKERN EPOQ                                                 ACOM.111
 155    <macro>                          $PUNTIF (R!EH = 0)     .Make sure it was queued            ACOM.112
 160    010000  034423                   (R!%W1) = 0            .Clear overflow cell                ACOM.113
 162    020432  012003  024410           B = (R!QPTR + 3)       .Re-enable channel                  ACOM.114
        030410  054000
 167    167000                           IOB: 7                                                     ACOM.115
 170    026430                           R!LIM = R!LIM + 1      .Let one more entry through         ACOM.116
 171    <macro>                        $ENDIF                                                       ACOM.117
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                     ACOM.118
                                       *  Calculate MOS address of DAL and read into local DAL.     ACOM.119
                                                                                                     ACOM.120
 171    <macro>                        RGET R!IOP,DA@NUM,R!MSG  .IOP that owns MOS DAL space        ACOM.121
```

```
175   020445  005002  016000        R!%W1 = R!IOP < 2 + MAAQ        .Find correct queue entry        ACOM.122
     /000000  024423
202   026423                        R!%W1 = R!%W1 + 1               .Cell for MOS base address       ACOM.123
203    <macro>                      RGET R!%W2,DA@IND,R!MSG         .MOS index of msg                ACOM.124
207   020424  005003  024424        R!%W2 = R!%W2 < 3               .Times 10 Octal                  ACOM.125
212   030423  024000                R!CB = (R!%W1)                  .Address upper                   ACOM.126
214   026423                        R!%W1 = R!%W1 + 1                                                ACOM.127
215   030423  022424  024000        R!CC = (R!%W1) + R!%W2          .Calculate address lower         ACOM.128
220   100002  026000                R!CB = R!CB + 1, C # 0          .If overflow                     ACOM.129
222   020436  024000                R!CD = R!DAL                    .Local Address                   ACOM.130
224   010010  024000                R!CE = DA@LEN@W                 .Word size                       ACOM.131
226   010002  024000                R!CA = MIO$BMRI                 .Read - no wait                  ACOM.132
230    <macro>                      RKERN COMIO                     .                                ACOM.133
232   020000  024423                R!%W1 = R!CB                    .Save address                    ACOM.134
234   020000  024424                R!%W2 = R!CC                    .                                ACOM.135
236   077000 /002305                R = CHNCHK                      .Wait For Completion             ACOM.136
240    <macro>                      GET A,DA@DIO,R!DAL              .MOS address upper               ACOM.137
244    <macro>                      $PUNTIF (A # R!%W1)             .Mismatch                        ACOM.138
247    <macro>                      GET A,DA@DL1,R!DAL              .MOS address lower               ACOM.139
253    <macro>                      $PUNTIF (A # R!%W2)             .Mismatch                        ACOM.140
256    <macro>                      GET R!IFC,DA@IFC,R!DAL          .Function Code                   ACOM.141
263    <macro>                      $PUNTIF (A = 0)                 .Illegal function code           ACOM.142
265    <macro>                      ADDRESS R!%W1,DA@CHN,R!DAL      .Channel cell                    ACOM.143
270   030423  024423                R!%W1 = (R!%W1)                 .Parameters for trace            ACOM.144
272    <macro>                      GET R!%W2,DA@DVO,R!DAL          .Device address upper            ACOM.145
                                                                                                      ACOM.146
277    <macro>                      KRECORD TF$ACOM,(R!IFC,R!DAL,R!MSG,R!%W1,R!%W2)                  ACOM.147
                                              *  Internal function code                             ACOM.148
                                              *  DAL address                                        ACOM.149
                                              *  Accumulator message                                ACOM.150
                                              *  Device type (5)/IOP (2)/Channel (9)                ACOM.151
                                              *  Cylinder (11)/Head group (5)                       ACOM.152
                                                                                                      ACOM.153
                                       *  Process the DAL according to internal function code,      ACOM.154
                                                                                                      ACOM.155
316   020435  013007  100004        P = IOAMSG, R!IFC > MC$CPWDN    .If not DCU4 disk request        ACOM.156
      102003  075000 /002427
324   020435  013001  102276        P = MDSK, R!IFC = MC$MSTIO      .Start disk I/O                  ACOM.157
327   020435  013002  126000        P = MSPECREL, R!IFC = MC$RLDAL  .Release MOS DAL                 ACOM.158
     /001642
333   020435  013005  102145        P = MANA, R!IFC = MC$CSTAT      .Status to mainframe             ACOM.159
336   020435  013006  126000        P = MOTH, R!IFC = MC$CPRDN      .Target memory to MOS done       ACOM.160
```

```
        001646
342     020435   013007   126000          P = MOIZ, R!IFC = MOSCPWDN    .MOS to Target memory done          ACOM.16
        /002200
```

```
                                                                                ACOM.163
                    ************************************************************
                    *                                                          *  ACOM.164
                    *  MC$CPURD, MC$CPUWR:  Move data across high speed channel *  ACOM.165
                    *       from Target  =>  MOS (MC$CPURD), or                *  ACOM.166
                    *       from MOS  =>  Target (MC$CPUWR)                    *  ACOM.167
                    *                                                          *  ACOM.168
                    ************************************************************  ACOM.169
                                                                                ACOM.170
                    * Extract parameters from DAL.                              ACOM.171
                                                                                ACOM.172
346   003000        I = 1                                                       ACOM.173
347    <macro>      GET R!LN1,DA@LN1,R!DAL          .Word length                ACOM.174
354    <macro>      GET R!SC0,DA@BM0,R!DAL          .MOS address                ACOM.175
361    <macro>      GET R!SC1,DA@BM1,R!DAL                                      ACOM.176
366    <macro>      GET R!TM,DA@TM,R!DAL            .Target memory              ACOM.177
374    <macro>      GET R!CPU,DA@TM0,R!DAL          .Target memory address (upper)  ACOM.178
403    <macro>      GET R!CPL,DA@TM1,R!DAL          .Target memory address (lower)  ACOM.179
                                                                                ACOM.180
                    * Issue transfer request.                                  ACOM.181
                                                                                ACOM.182
410    <macro>      $IF (R!IFC = MC$CPURD)          .Target memory to MOS       ACOM.183
                                                                     .>>>>> 0       $IF      1
414   010000  024450   R!SC2 = FS$IN                .Function code              ACOM.184
416   010006  024435   R!IFC = MC$CPRDN             .Code for response          ACOM.185
420    <macro>      $ELSE                                                       ACOM.186
421   010001  024450   R!SC2 = FS$OUT               .MOS to target memory       ACOM.187
423    <macro>        GET R!%W1,DA@OFF,R!DAL       .Adjust MOS for partial I/O ACOM.188
431    <macro>        DADD R!SC0,R!SC1,R!%W1                                   ACOM.189
436   010007  024435   R!IFC = MC$CPWDN            .Code for response          ACOM.190
440    <macro>      $ENDIF                                                      ACOM.191
                                                                     .<<<<< 0       $ENDIF   1
440    <macro>      TRANSFER R!SC2,R!TM,R!CPU,R!CPL,R!SC0,R!SC1,R!LN1 .Issue   ACOM.192
                                                                                ACOM.193
                    *  Send status back to requesting IOP.                     ACOM.194
                                                                                ACOM.195
467    <macro>      PUT R!IFC,DA@IFC,R!DAL          .Respond data moved         ACOM.196
474   002000  000000 I = 0                                                      ACOM.197
476   077000 /002414 R = IOPMSG                     .Send msg to originating IOP ACOM.198
500   075000 /002277 P = MARX                       .Release DAL and look for more  ACOM.199
                                                                                ACOM.200
                                                                                ACOM.201
                    ************************************************************  ACOM.202
```

```
                                 *                                                          *    ACOM.20
                                 *  MC$CSTAT: Send response to mainframe                    *    ACOM.204
                                 *                                                          *    ACOM.205
                                 ************************************************************    ACOM.206
                                                                                                 ACOM.207
502                        MANA  *                                                               ACOM.208
                                                                                                 ACOM.209
                                 *  Release the MOS DAL                                         ACOM.210
                                                                                                 ACOM.211
502    <macro>                   $IF (R!IOP = R!%MYID)          .If this IOP owns the MOS DAL   ACOM.212
                                                                                .>>>>> 0         $IF      1
506   077000 /002326               R = MRELDAL                  .Release MOS DAL                ACOM.213
510    <macro>                   $ELSE                                                         ACOM.214
511    <macro>                     STORE MC$RLDAL,DA@IFC,R!DAL .Function code                  ACOM.215
516   077000 /002414               R = IOPMSG                   .Ask owner to release MOS DAL   ACOM.216
520    <macro>                   $ENDIF                                                        ACOM.217
                                                                                .<<<<< 0         $ENDIF   1
                                                                                                 ACOM.218
                                 * If response for striped group, route to AMSG for handling.  ACOM.219
                                                                                                 ACOM.220
520    <macro>                   GET R!%W1,DA@DIL,R!DAL         .Destination ID                 ACOM.221
526    <macro>                   GET R!%W2,DA@UNS,R!DAL         .Units in group                 ACOM.222
534    <macro>                   $IF (R!%W1 = RQ$DISK) AND (R!%W2 # 1) .If striped disk response ACOM.223
                                                                                .>>>>> 0         $IF      1
544    <macro>                     STORE AM$BDRSP,DA@IFC,R!DAL .Code for AMSG                  ACOM.224
551   075000 /002427               P = TOAMSG                   .Let AMSG process               ACOM.225
553    <macro>                   $ENDIF                                                        ACOM.226
                                                                                .<<<<< 0         $ENDIF   1
                                                                                                 ACOM.227
                                 *  Send reply to the mainframe.                                ACOM.228
                                                                                                 ACOM.229
553    <macro>                   $IF (R!IOP = R!%MYID)          .If DAL originated in this IOP  ACOM.230
                                                                                .>>>>> 0         $IF      1
557    <macro>                     FLDSUB 1,CPI@DP,R!%LSPI      .Adjust disk request population ACOM.231
563    <macro>                   $ENDIF                                                        ACOM.232
                                                                                .<<<<< 0         $ENDIF   1
563    <macro>                   GET R!%W1,DA@DID,R!DAL         .Swap DID and SID               ACOM.233
570    <macro>                   GET R!%W2,DA@SID,R!DAL         .Source                         ACOM.234
575    <macro>                   PUT R!%W1,DA@SID,R!DAL         .New source                     ACOM.235
602    <macro>                   PUT R!%W2,DA@DID,R!DAL         .New destination                ACOM.236
607    <macro>                   SEND R!DAL                     .Send DAL to mainframe          ACOM.237
623   071552                     P = MEMS                       .Next msg                       ACOM.238
```

```
                                                                                ACOM.239
                                                                                ACOM.240
                                 ************************************************ACOM.241
                                 *                                              *ACOM.242
                                 *  MC$MSTIO:  Start disk I/O.                  *ACOM.243
                                 *     If read request, check read ahead info areas to see *ACOM.244
                                 *     if data already in - or on its way.      *ACOM.245
                                 *                                              *ACOM.246
                                 ************************************************ACOM.247
                                                                                ACOM.248
624                        MDSK  *                                              ACOM.249
624   003000                     I = 1                                          ACOM.250
625    <macro>                   GET R!CHN,DA@CHN,R!DAL        .Requested channel ACOM.251
633   020437  013040  125000     P = MBAD, R!CHN >= O'40       .If too big      ACOM.252
     /002357
637   020437  013020  124000     P = MBAD, R!CHN < O'20        .If too small    ACOM.253
     /002357
643   020437  016000 /000000     R!DCB = R!CHN + DCCB - O'20   .Channel Table   ACOM.254
      013020  024440
650   030440  024440             R!DCB = (R!DCB)               .Disk Control Block (DCB) ACOM.255
652   126000 /002357             P = MBAD, A = 0               .If channel not assigned ACOM.256
                                                                                ACOM.257
                                 *  Validate target memory.                     ACOM.258
                                                                                ACOM.259
654    <macro>                   GET R!TM,DA@TM,R!DAL          .Target memory   ACOM.260
662   041000  013003  125000     P = MBAD, A >= TM$MAX         .Too big         ACOM.261
     /002357
666   020461  016000 /000000     R!TMP = (R!TM + TMCB)         .Pointer to TM control table ACOM.262
      024410  030410  024462
674    <macro>                   $IF (A = 0)                   .Path not defined ACOM.263
                                                                        .>>>>> 0     $IF      1
676    <macro>                     $IF (R!TM = TM$SSD)         .If SSD          ACOM.264
                                                                        .>>>>> 1     $IF      1
702    <macro>                       STORE DAR$BKD,DA@RC,R!DAL .Set return code ACOM.265
712   075000 /002367               P = MBAD1                   .Send to mainframe ACOM.266
714    <macro>                     $ELSE                                        ACOM.267
715   075000 /002357               P = MBAD                    .Return bad status ACOM.268
717    <macro>                     $ENDIF                                       ACOM.269
                                                                        .<<<<< 1     $ENDIF   1
717    <macro>                   $ENDIF                                         ACOM.270
                                                                        .<<<<< 0     $ENDIF   1
                                                                                ACOM.271
```

```
                                 *  Validate disk     ess.                                          ACOM.2
                                                                                                    ACOM.27
  717    <macro>                 GET A,DA@TYP,R!DAL            .Device type                        ACOM.274
  724    <macro>                 $IF (A = DSK$DD19)            .If DD19                            ACOM.275
                                                                                .>>>>> 0                $IF      1
  727   010633  024423             R!%W1 = D1$CPD              .Cylinders/disk                     ACOM.276
  731    <macro>                 $ELSE                         .DD29                               ACOM.277
  732   014000 /001467  024423     R!%W1 = D2$CPD                                                  ACOM.278
  735    <macro>                 $ENDIF                                                            ACOM.279
                                                                                .<<<<< 0                $ENDIF   1
  735    <macro>                 GET R!CYL,DA@CYL,R!DAL        .Cylinder from MDAL                 ACOM.280
  743   041000  023423  125000   P = MBAD, A >= R!%W1          .Out of range                       ACOM.281
       /002357
  747    <macro>                 GET R!HED,DA@HED,R!DAL        .Head group                         ACOM.282
  755   041000  013012  125000   P = MBAD, A >= D2$TPC         .Illegal                            ACOM.283
       /002357
  761    <macro>                 GET R!SEC,DA@SEC,R!DAL        .Sector                             ACOM.284
  767   041000  013022  125000   P = MBAD, A >= D2$SPT         .Illegal                            ACOM.285
       /002357
  773    <macro>                 GET R!FNC,DA@FC,R!DAL         .Function code                      ACOM.286
                                                                                                    ACOM.287
                                 *  Validate transfer length.                                      ACOM.288
                                                                                                    ACOM.289
 1001    <macro>                 GET R!LN0,DA@LN0,R!DAL        .Get the request length             ACOM.290
 1006    <macro>                 GET R!LN1,DA@LN1,R!DAL                                            ACOM.291
 1013    <macro>                 $IF (R!LN1 = 0)                                                   ACOM.292
                                                                                .>>>>> 0                $IF      1
 1016   020454  126000 /002357     P = MBAD, R!LN0 = 0         .If bad request length              ACOM.293
 1021    <macro>                 $ENDIF                                                            ACOM.294
                                                                                .<<<<< 0                $ENDIF   1
 1021   020454  004011  024410   P = MBAD, 0 # R!LN0 > D'9     .Limit transfers to 65K             ACOM.295
        010000  023410  127000
       /002357
 1030    <macro>                 $IF (R!TM = TM$SSD)           .If target is SSD                 R164101A.1
                                                                                .>>>>> 0                $IF      1
 1034   020455  011077  024410     P = MBAD, 0 # R!LN1 & MSK$SSD .Must be mod 64                 R164101A.2
        010000  023410  127000
       /002357
 1043    <macro>                 $ENDIF                                                            R164101A.3
                                                                                .<<<<< 0                $ENDIF   1
 1043    <macro>                 GET R!%W1,DA@IM0,R!DAL        .Starting address from MDAL         ACOM.296
 1052    <macro>                 GET R!%W2,DA@IM1,R!DAL        .                                   ACOM.297
```

```
                                                                                                ACOM.298
                                 *  Include the target memory base address.                     ACOM.299
                                                                                                ACOM.300
1057    <macro>                  GET R!%W3,TM@FW0,R!TMP          .Target memory base address   ACOM.301
1064    <macro>                  GET R!%W4,TM@FW1,R!TMP                                        ACOM.302
1071    <macro>                  DADD R!%W1,R!%W2,R!%W3,R!%W4    .Add base                     ACOM.303
1101    <macro>                  $IF (R!TM = TM$SSD)             .If target is SSD             R164101A.4
                                                                                .>>>>> 0                $IF      1
1105   020424  011077  024410      P = MBAD, 0 # R!%W2 & MSK$SSD .Must be mod 64               R164101A.5
       010000  023410  127000
      /002357
1114    <macro>                  $ENDIF                                                        R164101A.6
                                                                                .<<<<< 0                $ENDIF   1
1114    <macro>                  PUT R!%W1,DA@TM0,R!DAL          .Save in MDAL                 ACOM.304
1130    <macro>                  PUT R!%W2,DA@TM1,R!DAL          .                             ACOM.305
                                                                                                ACOM.306
                                 *  Check ending address.                                        ACOM.307
                                                                                                ACOM.308
1135    <macro>                  DADD R!%W1,R!%W2,R!LN0,R!LN1    .Ending address               ACOM.309
1145    <macro>                  GET R!%W3,TM@LM0,R!TMP          .Target memory limit address  ACOM.310
1152    <macro>                  GET R!%W4,TM@LM1,R!TMP                                        ACOM.311
1157   020423  023425  100004    P = MBAD, R!%W1 > R!%W3         .Too big                      ACOM.312
       102003  075000 /002357
1165    <macro>                  $IF (R!%W1 = R!%W3)                                           ACOM.313
                                                                                .>>>>> 0                $IF      1
1171   020424  023426  100004      P = MBAD, R!%W2 > R!%W4       .Too big                      ACOM.314
       102003  075000 /002357
1177    <macro>                  $ENDIF                                                        ACOM.315
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                ACOM.316
                                 *  Calculate number of sectors for request.                     ACOM.317
                                                                                                ACOM.318
1177    <macro>                  GET R!OFF,DA@OFF,R!DAL          .First sector offset          ACOM.319
1205   020443  005011  022456    R!SC3 = R!SEC < D'9 + R!OFF     .Starting word address        ACOM.320
       024451
1211    <macro>                  DADD R!LN0,R!LN1,R!SC3          .Add transfer length          ACOM.321
1216   020455  011777  024457    R!LLN = R!LN1 & 777             .Last length                  ACOM.322
1221   027455                    R!LN1 = R!LN1 - 1               .End sector number            ACOM.323
1222   101002  027454            R!LN0 = R!LN0 - 1, C = 0        .If underflow, adjust         ACOM.324
1224    <macro>                  RSHIFT R!LN0,R!LN1,D'9          .Divide by D'512              ACOM.325
1243   020455  023443  012001    R!TOT = R!LN1 - R!SEC + 1       .Total sectors for request    ACOM.326
       024460
```

```
1247   020457  103004  014000          R!LLN = D2$WPS, R.     = 0      .If full sector                          ACOM.32
       /001000  024457
1254   020460  013001  103004          R!LLN = R!LLN - R!OFF, R!TOT = 1 .If less than one sector                ACOM.328
       020457  023456  024457
1262    <macro>                        PUT R!LLN,DA@LLN,R!DAL          .Put last length in DAL                  ACOM.329
1276    <macro>                        PUT R!TOT,DA@TOT,R!DAL          .Number of sectors to do                 ACOM.330
                                                                                                                ACOM.331
                                       * Place the master DAL on the DCB queue.  Don't process this            ACOM.332
                                       * request any further if another request is already on queue.           ACOM.333
                                                                                                                ACOM.334
1303   002000  000000                  I = 0                                                                    ACOM.335
1305   020436  024000                  R!EH = R!DAL                    .MDAL address                            ACOM.336
1307    <macro>                        ADDRESS R!ED,DB@MDL,R!DCB       .MDAL queue                              ACOM.337
1312    <macro>                        RKERN  ENQU                     .Put on DCB queue                        ACOM.338
1314    <macro>                        GET R!%W1,DB@FLG,R!DCB          .Disk activity flags                     ACOM.339
1316   020444  011001                  A = R!FNC & 1                   .Decode function                         ACOM.340
1320    <macro>                        $IF (A = 0)                     .If a write request                      ACOM.341
                                                                                        .>>>>> 0                $IF      1
                                                                                                                ACOM.342
                                         *            Write request                                             ACOM.343
                                         *                                                                      ACOM.344
                                         *  Abort read aheads in progress and build                             ACOM.345
                                         *  initial EDALs.                                                      ACOM.346
                                                                                                                ACOM.347
1322    <macro>                          STORE 0,DA@WBH,R!DAL          .Initial write-behind sector           ACOM.348
1327    <macro>                          $IF ((R!ED) = R!DAL)          .If current is first on queue          ACOM.349
                                                                                        .>>>>> 1                $IF      1
1333    <macro>                            $IF (R!%W1 = 0) OR (0 # R!%W1 & 4) .If quiet or read-ahead          ACOM.350
                                                                                        .>>>>> 2                $IF      1
1344   077000 /002444                      R = ABORT                     .Abort read aheads                     ACOM.351
1346    <macro>                            $ENDIF                                                               ACOM.352
                                                                                        .<<<<< 2                $ENDIF   1
1346    <macro>                          $ENDIF                                                                 ACOM.353
                                                                                        .<<<<< 1                $ENDIF   1
1346    <macro>                          GET A,DA@DIA,R!DAL            .Diagnostic flag                       ACOM.354
1353    <macro>                          $IF (A # 0)                   .If diagnostic write                   ACOM.355
                                                                                        .>>>>> 1                $IF      1
1355    <macro>                            GET A,DA@RCC,R!DAL          .Write behind count                    ACOM.356
1361    <macro>                            $IF (A = 0)                 .If not suppressed                     ACOM.357
                                                                                        .>>>>> 2                $IF      1
1363    <macro>                              STORE DAF$WRBH,DA@FC,R!DAL .Adjust function code                   ACOM.358
1373    <macro>                            $ENDIF                                                               ACOM.359
```

```
                                                                .<<<<< 2          $ENDIF   1
1373    <macro>          $ENDIF                                                  ACOM.360
                                                                .<<<<< 1          $ENDIF   1
1373    020440  024000   R!DT = R!DCB              .Let DISK start write-behind  ACOM.361
1375    <macro>          RKERN DISCO                                             ACOM.362
1377    <macro>        $ELSE                                                     ACOM.363
                                                                                  ACOM.364
                         *            Read request                               ACOM.365
                         *                                                       ACOM.366
                         *  If channel is quiet or reading ahead, try to match   ACOM.367
                         *  the first sector of the request to a read-ahead sector. ACOM.368
                                                                                  ACOM.369
1400    <macro>          $IF (R!%W1 # 0) AND (R!%W1 # 4) .If not read-ahead and busy ACOM.370
                                                                .>>>>> 1          $IF      1
1407    070215             P = GENDALS              .Skip                        ACOM.371
1410    <macro>          $ELSEIF ((R!ED) # R!DAL)   .If not first request        ACOM.372
1415    070207             P = GENDALS              .Skip read ahead check       ACOM.373
1416    <macro>          $ENDIF                                                  ACOM.374
                                                                .<<<<< 1          $ENDIF   1
                                                                                  ACOM.375
                         *  Don't use read ahead data for transfers to BMR.      ACOM.376
                                                                                  ACOM.377
1416    <macro>          $IF (R!TM = TM$BMR)        .If target is BMR            ACOM.378
                                                                .>>>>> 1          $IF      1
1422    077000 /002444     R = ABORT                .Abort read aheads           ACOM.379
1424    070200             P = GENDALS                                           ACOM.380
1425    <macro>          $ENDIF                                                  ACOM.381
                                                                .<<<<< 1          $ENDIF   1
                                                                                  ACOM.382
                         * Check each read-ahead sector for a match if the track ACOM.383
                         * of the request matches the read-ahead track.          ACOM.384
                                                                                  ACOM.385
1425    <macro>          GET R!%W2,DA@SUB,R!DAL     .DAL subfunction code        ACOM.386
1434    <macro>          GET A,DB@SUB,R!DCB         .Read ahead subfnc. code     ACOM.387
1440    <macro>          $IF (A # R!%W2)            .If mismatch                 ACOM.388
                                                                .>>>>> 1          $IF      1
1443    077000 /002444     R = ABORT                .Abort read aheads           ACOM.389
1445    070157             P = GENDALS                                           ACOM.390
1446    <macro>          $ENDIF                                                  ACOM.391
                                                                .<<<<< 1          $ENDIF   1
1446    <macro>          GET R!%W5,DB@RAF,R!DCB     .Read-aheads done            ACOM.392
1453    <macro>          $IF (A # 0)                .If any                      ACOM.393
```

```
                                                                .>>>>> 1          $IF      1
1455   <macro>          GET R!SC1,DB@OUT,R!DCB     .'OUT' pointer                R16410BA.1
1462   <macro>          GET R!SCO,DB@UNF,R!DCB     .'LAST' pointer               R16410BA.2
1467   <macro>          ADDRESS R!SC2,DA@CYL,R!DAL .Cylinder/head of request        ACOM.396
1472   030450  024450   R!SC2 = (R!SC2)                                            ACOM.397
1474   024451           R!SC3 = A                  .Copy                          ACOM.398
1475   020442  012001  024425  R!%W3 = R!HED + 1   .Calculate next track address   ACOM.399
1500   <macro>          $IF (A >= D2$TPC)          .If cylinder boundary          ACOM.400
                                                                .>>>>> 2          $IF      1
1504   010000  024425     R!%W3 = 0                .Head group zero               ACOM.401
1506   020441  012001  024426  R!%W4 = R!CYL + 1   .Increment request cylinder     ACOM.402
1511   <macro>            RPUT R!%W4,DA@CYL,R!SC3 .Restore                        ACOM.403
1522   <macro>          $ENDIF                                                    ACOM.404
                                                                .<<<<< 2          $ENDIF   1
1522   <macro>          RPUT R!%W3,DA@HED,R!SC3    .Restore                       ACOM.405
1532   <macro>          $UNTIL (R!%W5 = 0)         .For each read-ahead entry     ACOM.406
1534   <macro>            ADDRESS R!%W3,RA@CYL,R!SC1 .Cyl/head of current entry   ACOM.407
1537   030425  024425     R!%W3 = (R!%W3)                                         ACOM.408
1541   <macro>            $IF (R!%W3 # R!SC2),AND,(R!%W3 # R!SC3) .No match poss  ACOM.409
                                                                .>>>>> 2          $IF      1
1551   077000 /002444       R = ABORT              .Abort read aheads             ACOM.410
1553   070051               P = GENDALS            .Skip rest of check            ACOM.411
1554   <macro>            $ENDIF                                                  ACOM.412
                                                                .<<<<< 2          $ENDIF   1
1554   <macro>            GET R!%W2,RA@SEC,R!SC1  .Read-ahead sector             ACOM.413
1562   <macro>            $IF (R!%W3 = R!SC2) AND (R!%W2 = R!SEC) .If match found ACOM.414
                                                                .>>>>> 2          $IF      1
1572   010000  024427       R!%W5 = 0              .Set scan done                 ACOM.415
1574   <macro>            $ELSE                                                   ACOM.416
1575   <macro>              $IF (R!SC1 = R!SCO)    .Advance 'OUT'                 ACOM.417
                                                                .>>>>> 3          $IF      1
1601   <macro>                GET R!SC1,DB@INF,R!DCB .Wrap if at 'LAST'           ACOM.418
1606   <macro>              $ELSE                                                 ACOM.419
1607   010006  025447         R!SC1 = R!SC1 + RA@LEN@P .Advance                   ACOM.420
1611   <macro>              $ENDIF                                                ACOM.421
                                                                .<<<<< 3          $ENDIF   1
1611   027427               R!%W5 = R!%W5 - 1      .Decrement counter             ACOM.422
1612   <macro>              FLDSUB 1,DB@RAF,R!DCB .Decrement read-ahead count     ACOM.423
1616   <macro>            $ENDIF                                                  ACOM.424
                                                                .<<<<< 2          $ENDIF   1
1616   <macro>          $ENDTIL                                                   ACOM.425
1617   <macro>          PUT R!SC1,DB@OUT,R!DCB     .Update 'OUT'                  ACOM.426
```

```
1624    <macro>                         $ENDIF                                                          ACOM.427
                                                                                .<<<<< 1                $ENDIF    1
1624    <macro>                         $ENDIF                                                          ACOM.428
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                        ACOM.429
1624                            GENDALS *                                                               ACOM.430
1624    <macro>                         $IF (0 = R!%W1 & O'1120)        .If not in error recovery       ACOM.431
                                                                                .>>>>> 0                $IF       1
1634    020440  024000                    R!DH = R!DCB                  .DCB address                    ACOM.432
1636    <macro>                           RKERN DBUD                    .Make up initial E-DALs         ACOM.433
1640    <macro>                         $ENDIF                                                          ACOM.434
                                                                                .<<<<< 0                $ENDIF    1
1640    075000 /000051                  P = MEMS                        .Check for more requests        ACOM.435
                                                                                                        ACOM.436
                                                                                                        ACOM.437
                                        ************************************************************    ACOM.438
                                        *                                                          *    ACOM.439
                                        *  MC$RLDAL:     Release MOS DAL space                     *    ACOM.440
                                        *                                                          *    ACOM.441
                                        ************************************************************    ACOM.442
                                                                                                        ACOM.443
1642                            MSPECREL *                                                              ACOM.444
1642    077000 /002326                  R = MRELDAL                     .Release MOS DAL                ACOM.445
1644    075000 /002277                  P = MARX                        .Release local DAL, next msg    ACOM.446
                                                                                                        ACOM.447
                                                                                                        ACOM.448
                                        ************************************************************    ACOM.449
                                        *                                                          *    ACOM.450
                                        *  MC$CPRDN:  Target memory to MOS move done.                *    ACOM.451
                                        *                                                          *    ACOM.452
                                        ************************************************************    ACOM.453
                                                                                                        ACOM.454
1646                            MOTH    *                                                               ACOM.455
1646    003000                          I = 1                                                           ACOM.456
1647    <macro>                         GET R!SC0,DA@MDL,R!DAL          .M-DAL address                  ACOM.457
1654    <macro>                         FLDADD 1,DA@DAT,R!SC0           .Count DAL Returned             ACOM.458
1660    <macro>                         GET R!SC1,DA@TOT,R!DAL          .E-DAL Address                  ACOM.459
1665    <macro>                         STORE 2,DA@DAT,R!SC1            .Data In MOS                    ACOM.460
1672    <macro>                         GET R!FNC,DA@FC,R!SC1           .Function code                  ACOM.461
1700    <macro>                         GET R!CHN,DA@CHN,R!SC1          .Disk channel of request        ACOM.462
1706    014000 /000000  022437          R!DCB = DCCB + R!CHN - O'20     .DCB table pointer              ACOM.463
        013020  024440
```

```
1711   030440  024440                R!DCB = (R!DCB)                 .DCB Address                          ACOM.46
                                                                                                            ACOM.465
                                      *  Send early write behind reply to mainframe when                   ACOM.466
                                      *  all data is in MOS.                                               ACOM.467
                                                                                                            ACOM.468
1715    <macro>                       GET A,DA@SID,R!SC1              .Last sector flag                     ACOM.469
1721    <macro>                       $IF (A # 0)                     .If last EDAL                         ACOM.470
                                                                                        .>>>>> 0              $IF      1
1724    <macro>                         $IF (R!FNC = DAF$WRBH),OR,(R!FNC = DAF$WBFR) .If write behind       ACOM.471
                                                                                        .>>>>> 1              $IF      1
1734    <macro>                           STORE MC$CSTAT,DA@IFC,R!SC0 .Internal function code in MDAL       ACOM.472
1741   014000 /000000  024424             R!%W2 = IMDAL                .Use immediate DAL                   ACOM.473
1744    <macro>                           COPY R!SC0,R!%W2,DA@@LH+DA@@LE .Copy MDAL                         ACOM.474
                                                                                                            ACOM.475
                                          *  Include error info if any errors ocurred.                      ACOM.476
                                                                                                            ACOM.477
1753    <macro>                           GET A,DA@RC,R!SC0            .Last error status                   ACOM.478
1770    <macro>                           $IF (A # DAR$OK)             .If error                            ACOM.479
                                                                                        .>>>>> 2              $IF      1
1773   020424  012030  024410               (R!%W2 + DA@CYL@P) = (R!DCB + DB@ECY@P) .Error cyl/head        ACOM.480
       020440  012027  024411
       030411  034410
2003   020424  012031  024410               (R!%W2 + DA@SEC@P) = (R!DCB + DB@ESC@P) .Error sec/offset      ACOM.481
       020440  012030  024411
       030411  034410
2013    <macro>                             STORE 0,DA@ERC,R!SC0       .Clear errcount for final reply      ACOM.482
2020    <macro>                           $ENDIF                                                          ACOM.483
                                                                                        .<<<<< 2              $ENDIF   1
2020   002000  000000                     I = 0                                                            ACOM.484
2022   020424  024000                     R!DS = R!%W2                 .DAL to send                         ACOM.485
2024    <macro>                           RKERN DMGA                   .Send to MIOP                        ACOM.486
2026    <macro>                         $ENDIF                                                            ACOM.487
                                                                                        .<<<<< 1              $ENDIF   1
2026    <macro>                       $ENDIF                                                              ACOM.488
                                                                                        .<<<<< 0              $ENDIF   1
2026    <macro>                       STORE 0,DA@MES,R!DAL            .Clear accumulator msg                ACOM.489
2033   002000  000000                 I = 0                                                                 ACOM.490
2035   020436  024000                 R!DY = R!DAL                    .DAL With Response                    ACOM.491
2037    <macro>                       RKERN HONN                      .Put It Back                          ACOM.492
2041   020447  024436                 R!DAL = R!SC1                   .E-DAL                                ACOM.493
2043   020444  013005  102125         P = MDVH, R!FNC = DAF$RBW       .If read before partial write         ACOM.494
2046    <macro>                       GET R!%W1,DB@EDL,R!DCB          .E-DAL chain                          ACOM.495
```

```
2053   023436  102006                P = MFIRST, A = R!DAL          .If first on chain                ACOM.496
2055   020436  033423  102113        P = MDVH, R!DAL = (R!%W1)      .If second on chain               ACOM.497
2060   075000 /000051                P = MEMS                       .Check for more                   ACOM.498
                                                                                                       ACOM.499
                                     *  If EDAL first on queue, read data into local buffer         ACOM.500
                                     *  from MOS and attempt to start I/O.                          ACOM.501
                                                                                                       ACOM.502
2062                          MFIRST *                                                               ACOM.503
2062    <macro>                      GET A,DB@FLG,R!DCB             .Check channel activity           ACOM.504
2063   011777  024423                R!%W1 = A & O'777              .Strip off irrelevant flags       ACOM.505
2065   013004  102104                P = MDVH, A = 4                .Doing read ahead                 ACOM.506
2067    <macro>                      RGET A,DBF@ER,R!%W1            .Error recovery flag              ACOM.507
2072   127000 /000051                P = MEMS, A # 0                .If active                        ACOM.508
2074    <macro>                      STORE 0,DBF@DS,R!DCB           .Ensure disabled bit clear        ACOM.509
2100    <macro>                      GET A,DA@LOC,R!DAL             .Local buffer pointer cell        ACOM.510
2104    <macro>                      $IF (A = 0)                    .If not allocated                 ACOM.511
                                                                                   .>>>>> 0            $IF      1
2106    <macro>                        RKERN HGETBUF                .Get Local Buffer                 ACOM.512
2110    <macro>                        $IF (R!DY = 0)               .If no buffer available           ACOM.513
                                                                                   .>>>>> 1            $IF      1
2113    <macro>                          STORE O'42,DB@FLG,R!DCB    .Indicate disabled status         ACOM.514
2115   070055                            P = MDVH                   .Activate disk demon to recover   ACOM.515
2116    <macro>                        $ENDIF                                                        ACOM.516
                                                                                   .<<<<< 1            $ENDIF   1
2116    <macro>                        PUT R!DY,DA@LOC,R!DAL        .Put in E-DAL                     ACOM.517
2123    <macro>                      $ENDIF                                                          ACOM.518
                                                                                   .<<<<< 0            $ENDIF   1
2123   020436  024000                R!DY = R!DAL                   .DAL address                      ACOM.519
2125    <macro>                      RKERN DDIO                     .Set up MOS I/O params            ACOM.520
2127    <macro>                      GET R!OFF,DA@OFF,R!DAL         .Word offset                      ACOM.521
2135   020456  005002  022000        R!CD = R!OFF < 2 + R!CD        .Adjust local buffer              ACOM.522
       024000
2141   010002  024000                R!CA = MIO$BMRI                .Read - no wait                   ACOM.523
2143    <macro>                      RKERN COMIO                                                      ACOM.524
2145   072140                        R = CHNCHK                     .Wait For Completion              ACOM.525
2146    <macro>                      STORE 1,DA@DAT,R!DAL           .Mark data in local               ACOM.526
2153   020440  024000                R!DA = R!DCB                   .DCB                              ACOM.527
2155    <macro>                      GET R!DB,DA@DVO,R!DAL          .Cyl,  Head                       ACOM.528
2162    <macro>                      RKERN DIOH                     .Position Head                    ACOM.529
2164   020000  103005                P = MDVH, R!DS # 0             .If seek required                 ACOM.530
2166   020436  024000                R!DC = R!DAL                   .EDAL                             ACOM.531
2170    <macro>                      RKERN DIOA                     .Start I/O                        ACOM.532
```

```
2171                    MDVH    *                                                                   ACOM.53
2172  020440  024000            R!DT = R!DCB                .DCB address                           ACOM.534
2174   <macro>                  RKERN DISCO                 .Activate the disk demon               ACOM.535
2176  075000 /000051            P = MEMS                    .Check for more                        ACOM.536
                                                                                                   ACOM.537
                                                                                                   ACOM.538
                                *************************************************************  ACOM.539
                                *                                                           *      ACOM.540
                                *  MC$CPWDN:  MOS to Target memory move done.               *      ACOM.541
                                *                                                           *      ACOM.542
                                *************************************************************  ACOM.543
                                                                                                   ACOM.544
2200                    MOTZ    *                                                                   ACOM.545
2200   <macro>                  GET R!SCO,DA@MDL,R!DAL      .M-DAL address                         ACOM.546
2205   <macro>                  GET R!%W1,DA@RC,R!DAL       .Status of Slave                       ACOM.547
2213   <macro>                  $IF (A # DAR$OK)            .If error                              ACOM.548
                                                                                .>>>>> 0                $IF      1
2215   <macro>                    PUT R!%W1,DA@RC,R!SCO     .Set status into master DAL            ACOM.549
2230   <macro>                  $ENDIF                                                             ACOM.550
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                   ACOM.551
                                *  If last sector of read, send status to mainframe.               ACOM.552
                                                                                                   ACOM.553
2230   <macro>                  GET A,DA@SID,R!DAL          .Last sector flag                      ACOM.554
2234   <macro>                  $IF (A # 0)                 .If last EDAL                          ACOM.555
                                                                                .>>>>> 0                $IF      1
2236   <macro>                    STORE MC$CSTAT,DA@IFC,R!SCO .Send Status to COS                  ACOM.556
2243  020446  024000              R!DS = R!SCO              .M-DAL                                 ACOM.557
2245   <macro>                    RKERN DMGA                .Send                                  ACOM.558
2247  020446  024000              R!DY = R!SCO              .Master DAL to Return                  ACOM.559
2251   <macro>                    RKERN HONN                .To Free Pool                          ACOM.560
2253   <macro>                  $ENDIF                                                             ACOM.561
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                   ACOM.562
                                *  Release resources.                                              ACOM.563
                                                                                                   ACOM.564
2253   <macro>                  GET R!CB,DA@BM0,R!DAL       .MOS buffer address                    ACOM.565
2260   <macro>                  GET R!CC,DA@BM1,R!DAL                                              ACOM.566
2265   <macro>                  $IF (R!CB # 0),OR,(R!CC # 0) .If a buffer allocated                ACOM.567
                                                                                .>>>>> 0                $IF      1
2272  010001  024000              R!CA = 1                  .Release 1 item                        ACOM.568
2274   <macro>                    RKERN PUTDISK             .Release the buffer                    ACOM.569
```

```
2276   <macro>                          $ENDIF                                                    ACOM.570
                                                                          .<<<<< 0                  $ENDIF    1
2276   072030                           R = MRELDAL                  .Release DAL MOS space        ACOM.571
2277                          MARX      *                                                          ACOM.572
2277   020436  024000                   R!DY = R!DAL                 .Local Address of DAL         ACOM.573
2301   <macro>                          RKERN HONN                   .Put It Back                  ACOM.574
2303   075000 /000051                   P = MEMS                     .Check for more               ACOM.575
```

```
                                 ***************************************************************   ACOM.577
                                 *          SUBROUTINES                                        *   ACOM.578
                                 ***************************************************************   ACOM.579
                                                                                                   ACOM.580
                                         ************************************************************   ACOM.581
                                         *  CHNCHK:  MOS or HSP error check and recovery.         *   ACOM.582
                                         ************************************************************   ACOM.583
                                                                                                   ACOM.584
2305                             CHNCHK  *                                                          ACOM.585
2305    <macro>                          $UNTIL (IOB # BZ)              .Until channel complete     ACOM.586
2307    003000                             I = 1                        .Unlock                     ACOM.587
2310    <macro>                            RKERN CHNWTDN                .Wait                       ACOM.588
2312    002000  000000                     I = 0                        .Lock                       ACOM.589
2314    <macro>                            $IF (IOB = DN) AND (IOB = BZ) .If error                  ACOM.590
                                                                                .>>>>> 0                $IF      1
2322    <macro>                              RKERN CEREC                .All done                   ACOM.591
2324    <macro>                            $ENDIF                                                   ACOM.592
                                                                                .<<<<< 0                $ENDIF   1
2324    <macro>                          $ENDTIL                                                    ACOM.593
2325    001000                           EXIT                                                       ACOM.594
                                                                                                   ACOM.595
                                         ************************************************************   ACOM.596
                                         *  MRELDAL:  Release MOS DAL space.                      *   ACOM.597
                                         ************************************************************   ACOM.598
                                                                                                   ACOM.599
2326                             MRELDAL *                                                          ACOM.600
2326    <macro>                          STORE 0,DA@MES,R!DAL           .Clear accumulator msg      ACOM.601
2333    <macro>                          GET R!CB,DA@DL0,R!DAL          .MOS DAL address            ACOM.602
2340    <macro>                          GET R!CC,DA@DL1,R!DAL                                      ACOM.603
2345    <macro>                          $IF (R!CB # 0),OR,(R!CC # 0)   .If a buffer allocated      ACOM.604
                                                                                .>>>>> 0                $IF      1
2352    010001  024000                     R!CA = 1                     .Release 1 item             ACOM.605
2354    <macro>                            RKERN PUTDAL                 .Release the DAL            ACOM.606
2356    <macro>                          $ENDIF                                                     ACOM.607
                                                                                .<<<<< 0                $ENDIF   1
2356    001000                           EXIT                                                       ACOM.608
                                                                                                   ACOM.609
                                         ************************************************************   ACOM.610
                                         *  MBAD:  Process illegal parameter in request.          *   ACOM.611
                                         ************************************************************   ACOM.612
                                                                                                   ACOM.613
2357                             MBAD    *                                                          ACOM.614
```

```
2357    <macro>                          STORE DAR$PAR,DA@RC,R!DAL      .Illegal parameter                ACOM.615
2367                             MBAD1   *                                                                ACOM.616
2367    <macro>                          STORE MC$CSTAT,DA@IFC,R!DAL    .Status to mainframe              ACOM.617
2374    002000  000000                   I = 0                                                            ACOM.618
2376    020436  024000                   R!DS = R!DAL                   .DAL address                      ACOM.619
2400    <macro>                          RKERN DMGA                     .Send to MIOP                     ACOM.620
2402    <macro>                          RELDAL R!DAL                   .Release local DAL                ACOM.621
2412    075000 /000051                   P = MEMS                       .Check for more requests          ACOM.622
                                                                                                          ACOM.623
                                         ***************************************************************  ACOM.624
                                         *  IOPMSG: Send message back to originating IOP.              *  ACOM.625
                                         ***************************************************************  ACOM.626
                                                                                                          ACOM.627
2414                             IOPMSG  *                                                                ACOM.628
2414    020436  024000                   R!DS = R!DAL                   .DAL address                      ACOM.629
2416    010001  024000                   R!CE = 1                       .Write one word                   ACOM.630
2420    <macro>                          RKERN DALWRT                   .Write to MOS                     ACOM.631
2422    020445  024000                   R!EI = R!IOP                   .Target IOP                       ACOM.632
2424    <macro>                          RKERN EMSGIOP                  .Send accumulator msg             ACOM.633
2426    001000                           EXIT                                                             ACOM.634
                                                                                                          ACOM.635
                                         ***************************************************************  ACOM.636
                                         *  TOAMSG: Queue DAL for AMSG demon.                          *  ACOM.637
                                         ***************************************************************  ACOM.638
                                                                                                          ACOM.639
2427                             TOAMSG  *                                                                ACOM.640
2427    020436  024000                   R!DY = R!DAL                   .Put on AMSG queue                ACOM.641
2431    014000 /000000  024000           R!DZ = MIMDALQ                                                   ACOM.642
2434    <macro>                          RKERN HPUT                                                       ACOM.643
2436    014000 /000000                   A = D$AMSG                     .AMSG demon pointer               ACOM.644
2440    <macro>                          RKERN DACT                     .Activate the demon               ACOM.645
2442    075000 /000051                   P = MEMS                       .Check for more                   ACOM.646
                                                                                                          ACOM.647
                                         ***************************************************************  ACOM.648
                                         *  ABORT:  Abort read aheads.                                 *  ACOM.649
                                         ***************************************************************  ACOM.650
                                                                                                          ACOM.651
2444                             ABORT   *                                                                ACOM.652
2444    <macro>                          STORE 0,DB@RAF,R!DCB           .Clear read-aheads done           ACOM.653
2451    <macro>                          $IF (0 # R!%W1 & 4)            .If read-ahead in progress        ACOM.654
                                                                                          .>>>>> 0          $IF      1
2460    <macro>                            STORE 1,DBF@AB,R!DCB         .Set abort flag                   ACOM.655
```

```
246     <macro>     $ELSE                                                          ACOM.65
2466    <macro>       GET R!%W2,DB@INF,R!DCB      .Reset pointers to 'FIRST'       ACOM.657
2473    <macro>       PUT R!%W2,DB@IN,R!DCB       .'IN'                            ACOM.658
2500    <macro>       PUT R!%W2,DB@OUT,R!DCB      .'OUT'                           ACOM.659
2505    <macro>     $ENDIF                                                         ACOM.660
                                                                  .<<<<< 0             $ENDIF   1
2505    001000      EXIT                                                           ACOM.661
                                                                                   ACOM.662
                    END                                                            ACOM.663
```

0X	%ACTIVE		10:41 D	10:41						
0X	%B		1: 4 D	7:16 B	7:37 B	7:39 B	7:42 B	8: 2 B	9:38 B	9:39 B
			10:12 B	10:15 B	10:26 B	11:14 B	11:16 B	11:21 B	11:26 B	11:35 B
			11:37 B	12:12 B	12:15 B	12:18 B	12:28 B	12:32 B	12:37 B	13:10 B
			13:23 B	13:27 B	14:30 B	15:25 B	15:37 B	16:21 B	17:13 B	17:23 B
			17:24 B	18: 8 B	18:10 B	18:19 B	19: 3 B	19:13 B	20: 4 B	21: 7 B
			23: 8 B	23:32 B						
0X	%EX		6:10 D	6:10	9:32 D	9:32	10:41 D	10:41	23: 7 D	23: 7
10	%GBLNUM	$APTEXT	2:18 F	2:18						
400	%GBLREG	$APTEXT	2:18							
0	%KERNEL	$APTEXT	5:32 D	5:32 B	6: 1 D	6: 1 B	6:15 D	6:15 B	6:19 D	6:19 B
			6:30 D	6:30 B	7:13 D	7:13 B	14:14 D	14:14 B	15: 5 D	15: 5 B
			17:10 D	17:10 B	18:31 D	18:31 B	18:39 D	18:39 B	19:18 D	19:18 B
			19:29 D	19:29 B	19:34 D	19:34 B	19:39 D	19:39 B	19:42 D	19:42 B
			20: 3 D	20: 3 B	20:29 D	20:29 B	20:31 D	20:31 B	20:42 D	20:42 B
			21: 6 D	21: 6 B	22:12 D	22:12 B	22:16 D	22:16 B	22:33 D	22:33 B
			23: 6 D	23: 6 B	23:17 D	23:17 B	23:19 D	23:19 B	23:29 D	23:29 B
			23:31 D	23:31 B						
0X	%LSPI		3:22 D	10:34						
0X	%MYID		3:21 D	5: 6	10:11	10:33				
0X	%PUNT		6:17 D	6:17	6:31 D	6:31	7:18 D	7:18	7:20 D	7:20
			7:22 D	7:22						
0X	%TR1		7:27 D	7:27						
0X	%TR2		7:27 D	7:27						
0X	%TR3		7:27 D	7:27						
0X	%TR4		7:27 D	7:27						
0X	%TR5		7:27 D	7:27						
0X	%TRAC		7:27 D	7:27						
0X	%TVNT		7:27 D	7:27						
10	ACOM$LIM	$APTEXT	5:12	6: 2	6:11					
20	AD@RC@N	$APTEXT	10:41	10:41 F						
13	AD@RC@P	$APTEXT	10:41 F	10:41						
0	AD@RC@S	$APTEXT	10:41	10:41 F						
42	AMSBDRSP	$APTEXT	10:25							
0X	CA		3:23 D	7:12	19:33	20:41	22:32			
0X	CB		3:24 D	7: 6	7: 9	7:14	20:37	20:40	22:28	22:31
0X	CC.		3:25 D	7: 8	7:15	20:38	20:40	22:29	22:31	
0X	CD		3:26 D	7:10	19:31					
0X	CE		3:27 D	7:11	23:16					
0X	CEREC		22:16 D	22:16						
0X	CHNWIDN		22:12 D	22:12						
0X	COMIO		7:13 D	7:13	19:34 D	19:34				

20	CPI@DP@N	$APIEXT	10:34	F							
12	CPI@DP@P	$APIEXT	10:34	F	10:34						
0X	D$ACOM		3:28	D	5:42						
0X	D$AMSG		3:29	D	23:30						
633	D1$CPD	$APIEXT	12: 6								
1467	D2$CPD	$APIEXT	12: 8								
22	D2$SPI	$APIEXT	12:18								
12	D2$IPC	$APIEXT	12:15		16: 9						
1000	D2$WPS	$APIEXT	14: 1								
0X	DA		3:30	D	19:37						
30	DA@@LE	$APIEXT	18:13								
10	DA@@LH	$APIEXT	18:13								
20	DA@BM0@N	$APIEXT	9:13		9:13	F	20:37		20:37	F	
34	DA@BM0@P	$APIEXT	9:13	F	9:13		20:37	F	20:37		
0	DA@BM0@S	$APIEXT	9:13		9:13	F	20:37		20:37	F	
20	DA@BM1@N	$APIEXT	9:14		9:14	F	20:38		20:38	F	
35	DA@BM1@P	$APIEXT	9:14	F	9:14		20:38	F	20:38		
0	DA@BM1@S	$APIEXT	9:14		9:14	F	20:38		20:38	F	
177	DA@CHN@M	$APIEXT	11:13		17:40						
7	DA@CHN@N	$APIEXT	11:13		17:40						
27	DA@CHN@P	$APIEXT	7:23		11:13	F	11:13		17:40	F	17:40
11	DA@CHN@S	$APIEXT	11:13		11:13	F	17:40		17:40	F	
3777	DA@CYI@M	$APIEXT	16:12								
13	DA@CYI@N	$APIEXT	12:11		12:11	F	16:12	F	16:12		
30	DA@CYI@P	$APIEXT	12:11	F	12:11		16: 4		18:20		
0	DA@CYI@S	$APIEXT	12:11		12:11	F	16:12	F	16:12		
20	DA@DAI@N	$APIEXT	17:36	F	17:38	F	19:36	F			
13	DA@DAI@P	$APIEXT	17:36	F	17:36		17:38	F	17:38		19:36 F 19:36
0	DA@DAI@S	$APIEXT	17:38	F	19:36	F					
1	DA@DIA@N	$APIEXT	14:35		14:35	F					
15	DA@DIA@P	$APIEXT	14:35	F	14:35						
0	DA@DIA@S	$APIEXT	14:35		14:35	F					
20	DA@DID@N	$APIEXT	10:37		10:37	F	10:40	F			
10	DA@DID@P	$APIEXT	10:37	F	10:37		10:40	F	10:40		
0	DA@DID@S	$APIEXT	10:37		10:37	F	10:40	F			
377	DA@DIL@M	$APIEXT	10:21								
10	DA@DIL@N	$APIEXT	10:21								
10	DA@DIL@P	$APIEXT	10:21	F	10:21						
10	DA@DIL@S	$APIEXT	10:21		10:21	F					
20	DA@DLO@N	$APIEXT	7:17		7:17	F	22:28		22:28	F	
2	DA@DLO@P	$APIEXT	7:17	F	7:17		22:28	F	22:28		
0	DA@DLO@S	$APIEXT	7:17		7:17	F	22:28		22:28	F	

20	DA@DL1@N	$APTEXT	7:19	7:19 F	22:29	22:29 F				
3	DA@DL1@P	$APTEXT	7:19 F	7:19	22:29 F	22:29				
0	DA@DL1@S	$APTEXT	7:19	7:19 F	22:29	22:29 F				
20	DA@DVO@N	$APTEXT	7:25	7:25 F	19:38	19:38 F				
30	DA@DVO@P	$APTEXT	7:25 F	7:25	19:38 F	19:38				
0	DA@DVO@S	$APTEXT	7:25	7:25 F	19:38	19:38 F				
20	DA@ERC@N	$APTEXT	18:26 F							
17	DA@ERC@P	$APTEXT	18:26 F	18:26						
0	DA@ERC@S	$APTEXT	18:26 F							
377	DA@FC@M	$APTEXT	14:41 F							
10	DA@FC@N	$APTEXT	12:20	12:20 F	14:41 F	14:41	17:39	17:39 F		
26	DA@FC@P	$APTEXT	12:20 F	12:20	14:41 F	14:41	17:39 F	17:39		
0	DA@FC@S	$APTEXT	12:20	12:20 F	14:41 F	14:41	17:39	17:39 F		
37	DA@HED@M	$APTEXT	12:14	16:15						
5	DA@HED@N	$APTEXT	12:14	16:15 F						
30	DA@HED@P	$APTEXT	12:14 F	12:14						
13	DA@HED@S	$APTEXT	12:14	12:14 F	16:15 F					
177740	DA@HED@X	$APTEXT	16:15							
20	DA@IFC@N	$APTEXT	7:21	7:21 F	9:36 F	10:14 F	10:25 F	18:11 F	20:27 F	23: 3 F
1	DA@IFC@P	$APTEXT	7:21 F	7:21	9:36 F	9:36	10:14 F	10:14	10:25 F	10:25
			18:11 F	18:11	20:27 F	20:27	23: 3 F	23: 3		
0	DA@IFC@S	$APTEXT	7:21	7:21 F	9:36 F	10:14 F	10:25 F	18:11 F	20:27 F	23: 3 F
7777	DA@IND@M	$APTEXT	7: 4							
14	DA@IND@N	$APTEXT	7: 4							
4	DA@IND@S	$APTEXT	7: 4	7: 4 F						
10	DA@LEN@W	$APTEXT	7:11							
1777	DA@LLN@M	$APTEXT	14: 5							
12	DA@LLN@N	$APTEXT	14: 5 F							
15	DA@LLN@P	$APTEXT	14: 5 F	14: 5						
6	DA@LLN@S	$APTEXT	14: 5 F							
176000	DA@LLN@X	$APTEXT	14: 5							
20	DA@LNO@N	$APTEXT	12:24	12:24 F						
32	DA@LNO@P	$APTEXT	12:24 F	12:24						
0	DA@LNO@S	$APTEXT	12:24	12:24 F						
20	DA@LN1@N	$APTEXT	9:12	9:12 F	12:25	12:25 F				
33	DA@LN1@P	$APTEXT	9:12 F	9:12	12:25 F	12:25				
0	DA@LN1@S	$APTEXT	9:12	9:12 F	12:25	12:25 F				
20	DA@LOC@N	$APTEXT	19:15	19:15 F	19:25 F					
37	DA@LOC@P	$APTEXT	19:15 F	19:15	19:25 F	19:25				
0	DA@LOC@S	$APTEXT	19:15	19:15 F	19:25 F					
20	DA@MDL@N	$APTEXT	17:35	17:35 F	20:14	20:14 F				
17	DA@MDL@P	$APTEXT	17:35 F	17:35	20:14 F	20:14				

0	DA@MDL@S	$APTEXT	17:35	17:35 F	2 4	20:14 F				
20	DA@MES@N	$APTEXT	18:36 F	22:27 F						
5	DA@MES@P	$APTEXT	18:36 F	18:36	22:27 F	22:27				
0	DA@MES@S	$APTEXT	18:36 F	22:27 F						
3	DA@NUM@M	$APTEXT	6:42							
2	DA@NUM@N	$APTEXT	6:42	6:42 F						
2	DA@NUM@S	$APTEXT	6:42	6:42 F						
777	DA@OFF@M	$APTEXT	9:27	13:33	19:30					
11	DA@OFF@N	$APTEXT	9:27	13:33	19:30					
31	DA@OFF@P	$APTEXT	9:27 F	9:27	13:33 F	13:33	19:30 F	19:30		
7	DA@OFF@S	$APTEXT	9:27	9:27 F	13:33	13:33 F	19:30	19:30 F		
377	DA@RC@M	$APTEXT	11:34 F	18:17	20:15	20:18	23: 1 F			
10	DA@RC@N	$APTEXT	11:34 F	18:17	20:15	20:18 F	23: 1 F			
26	DA@RC@P	$APTEXT	11:34 F	11:34	18:17 F	18:17	20:15 F	20:15	20:18 F	20:18
			23: 1 F	23: 1						
10	DA@RC@S	$APTEXT	11:34 F	18:17	18:17 F	20:15	20:15 F	20:18 F	23: 1 F	
177400	DA@RC@X	$APTEXT	11:34	20:18	23: 1					
20	DA@RCC@N	$APTEXT	14:38	14:38 F						
36	DA@RCC@P	$APTEXT	14:38 F	14:38						
0	DA@RCC@S	$APTEXT	14:38	14:38 F						
7	DA@SEC@N	$APTEXT	12:17	12:17 F						
31	DA@SEC@P	$APTEXT	12:17 F	12:17	18:23					
0	DA@SEC@S	$APTEXT	12:17	12:17 F						
20	DA@SID@N	$APTEXT	10:38	10:38 F	10:39 F	18: 6	18: 6 F	20:24	20:24 F	
11	DA@SID@P	$APTEXT	10:38 F	10:38	10:39 F	10:39	18: 6 F	18: 6	20:24 F	20:24
0	DA@SID@S	$APTEXT	10:38	10:38 F	10:39 F	18: 6	18: 6 F	20:24	20:24 F	
37	DA@SUB@M	$APTEXT	15:33							
5	DA@SUB@N	$APTEXT	15:33	15:33 F						
15	DA@SUB@P	$APTEXT	15:33 F	15:33						
1	DA@SUB@S	$APTEXT	15:33	15:33 F						
37777	DA@TM0@M	$APTEXT	9:16	12:41	13:14					
16	DA@TM0@N	$APTEXT	9:16	12:41	13:14 F					
24	DA@TM0@P	$APTEXT	9:16 F	9:16	12:41 F	12:41	13:14 F	13:14		
2	DA@TM0@S	$APTEXT	9:16	9:16 F	12:41	12:41 F	13:14 F			
140000	DA@TM0@X	$APTEXT	13:14							
20	DA@TM1@N	$APTEXT	9:17	9:17 F	12:42	12:42 F	13:15 F			
25	DA@TM1@P	$APTEXT	9:17 F	9:17	12:42 F	12:42	13:15 F	13:15		
0	DA@TM1@S	$APTEXT	9:17	9:17 F	12:42	12:42 F	13:15 F			
2	DA@TM@N	$APTEXT	9:15	9:15 F	11:25	11:25 F				
24	DA@TM@P	$APTEXT	9:15 F	9:15	11:25 F	11:25				
0	DA@TM@S	$APTEXT	9:15	9:15 F	11:25	11:25 F				
20	DA@TOT@N	$APTEXT	14: 6 F	17:37	17:37 F					

14	DA@TOT@P	$APTEXT	14: 6 F	14: 6	17:37 F	17:37			
0	DA@TOT@S	$APTEXT	14: 6 F	17:37	17:37 F				
5	DA@TYP@N	$APTEXT	12: 3	12: 3 F					
27	DA@TYP@P	$APTEXT	12: 3 F	12: 3					
0	DA@TYP@S	$APTEXT	12: 3	12: 3 F					
4	DA@UNS@N	$APTEXT	10:22	10:22 F					
20	DA@UNS@P	$APTEXT	10:22 F	10:22					
0	DA@UNS@S	$APTEXT	10:22	10:22 F					
20	DA@WBH@N	$APTEXT	14:25 F						
37	DA@WBH@P	$APTEXT	14:25 F	14:25					
0	DA@WBH@S	$APTEXT	14:25 F						
0X	DACT		23:31 D	23:31					
5	DAF$RBW	$APTEXT	18:41						
6	DAF$WBFR	$APTEXT	18:10						
14	DAF$WRBH	$APTEXT	14:41	18:10					
0X	DALWRT		23:17 D	23:17					
21	DAR$BKD	$APTEXT	11:34						
0	DAR$OK	$APTEXT	18:19	20:17					
20	DAR$PAR	$APTEXT	23: 1						
0X	DB		3:31 D	19:38					
27	DB@ECY@P	$APTEXT	18:21						
20	DB@EDL@N	$APTEXT	18:42	18:42 F					
2	DB@EDL@P	$APTEXT	18:42 F	18:42					
0	DB@EDL@S	$APTEXT	18:42	18:42 F					
30	DB@ESC@P	$APTEXT	18:24						
20	DB@FLG@N	$APTEXT	14:15	14:15 F	19: 9	19: 9 F	19:21 F		
0	DB@FLG@P	$APTEXT	14:15 F	19: 9 F	19:21 F				
0	DB@FLG@S	$APTEXT	14:15	14:15 F	19: 9	19: 9 F	19:21 F		
20	DB@IN@N	$APTEXT	24: 3 F						
44	DB@IN@P	$APTEXT	24: 3 F	24: 3					
0	DB@IN@S	$APTEXT	24: 3 F						
20	DB@INF@N	$APTEXT	16:32	16:32 F	24: 2	24: 2 F			
12	DB@INF@P	$APTEXT	16:32 F	16:32	24: 2 F	24: 2			
0	DB@INF@S	$APTEXT	16:32	16:32 F	24: 2	24: 2 F			
5	DB@MDL@P	$APTEXT	14:13						
20	DB@OUT@N	$APTEXT	16: 2	16: 2 F	16:42 F	24: 4 F			
45	DB@OUT@P	$APTEXT	16: 2 F	16: 2	16:42 F	16:42	24: 4 F	24: 4	
0	DB@OUT@S	$APTEXT	16: 2	16: 2 F	16:42 F	24: 4 F			
20	DB@RAT@N	$APTEXT	15:41	15:41 F	16:38 F	23:39 F			
11	DB@RAT@P	$APTEXT	15:41 F	15:41	16:38 F	16:38	23:39 F	23:39	
0	DB@RAT@S	$APTEXT	15:41	15:41 F	23:39 F				
20	DB@SUB@N	$APTEXT	15:34	15:34 F					

24	DB@SUB@P	$APTEXT	15:34 F	15:34							
0	DB@SUB@S	$APTEXT	15:34	15:34 F							
20	DB@UNI@N	$APTEXT	16: 3	16: 3 F							
46	DB@UNI@P	$APTEXT	16: 3 F	16: 3							
0	DB@UNI@S	$APTEXT	16: 3	16: 3 F							
1	DBI@AB@M	$APTEXT	23:42 F								
1	DBI@AB@N	$APTEXT	23:42 F	23:42							
0	DBI@AB@P	$APTEXT	23:42 F								
0	DBI@AB@S	$APTEXT	23:42 F	23:42							
1	DBI@DS@M	$APTEXT	19:14 F								
1	DBI@DS@N	$APTEXT	19:14 F								
0	DBI@DS@P	$APTEXT	19:14 F								
12	DBI@DS@S	$APTEXT	19:14 F								
177737	DBI@DS@X	$APTEXT	19:14								
1	DBI@ER@M	$APTEXT	19:12								
1	DBI@ER@N	$APTEXT	19:12	19:12 F							
13	DBI@ER@S	$APTEXT	19:12	19:12 F							
0X	DBUD		17:10 D	17:10							
0X	DC		3:32 D	19:41							
0X	DCCB		3:33 D	11:18	17:41						
0X	DDIO		19:29 D	19:29							
0X	DGIVEUP		6: 1 D	6: 1							
0X	DH		3:34 D	17: 9							
0X	DIOA		19:42 D	19:42							
0X	DIOH		19:39 D	19:39							
2	DIOP	$APTEXT	5: 6								
0X	DISCO		15: 5 D	15: 5	20: 3 D	20: 3					
0X	DMGA		18:31 D	18:31	20:29 D	20:29	23: 6 D	23: 6			
0X	DS		3:35 D	18:30	19:40	20:28	23: 5	23:15			
1	DSK$DD19	$APTEXT	12: 5								
0X	DT		3:36 D	15: 4	20: 2						
0X	DY		3:37 D	6:16	18:38	19:20	19:25	19:28	20:30	21: 5	
			23:27								
0X	DZ		3:38 D	23:28							
0X	ED		3:39 D	6:18	6:29	14:13	14:27	15:16			
0X	EDIS		3:40 D	6: 6							
0X	EH		3:41 D	6:20	6:26	6:31	14:12				
0X	EI		3:42 D	23:18							
0X	EMSGTOP		23:19 D	23:19							
0X	ENQU		14:14 D	14:14							
0X	EPOQ		6:30 D	6:30							
0X	ESWP		5:32 D	5:32							

0X	ETOQ		6:19	D	6:19						
7	F$PAUSE	$APTEXT	6:10								
27	F$RDAL	$APTEXT	23: 7								
34	F$SEND	$APTEXT	10:41								
45	F$TRANSF	$APTEXT	9:32								
0	FS$IN	$APTEXT	9:23								
1	FS$OUT	$APTEXT	9:26								
0	FS$WAIT	$APTEXT	9:32								
0X	HGETBUF		19:18	D	19:18						
0X	HOFF		6:15	D	6:15						
0X	HONN		18:39	D	18:39	20:31	D	20:31	21: 6	D	21: 6
0X	HPUT		23:29	D	23:29						
0X	IMDAL		4: 1	D	18:12						
0X	MAAQ		4: 2	D	5: 4	5:19		7: 1			
6	MC$CPRDN	$APTEXT	7:42		9:24						
3	MC$CPURD	$APTEXT	9:22								
7	MC$CPWDN	$APTEXT	7:36		8: 2	9:29					
5	MC$CSTAT	$APTEXT	7:41		18:11	20:27		23: 3			
1	MC$MSTIO	$APTEXT	7:38								
2	MC$RLDAL	$APTEXT	7:39		10:14						
0X	MIMDALQ		4: 3	D	23:28						
2	MIO$BMRI	$APTEXT	7:12		19:33						
0	MIOP	$APTEXT	5: 6								
0X	MQEN		4: 4	D	5:28						
77	MSK$SSD	$APTEXT	12:36		13: 9						
0X	O$ACOM		1: 4	D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4								
1	OV@TYP@N	$APTEXT	1: 4								
0	PI$HUH	$APTEXT	6:17		6:31	7:18		7:20	7:22		
0X	PUTDAL		22:33	D	22:33						
0X	PUTDISK		20:42	D	20:42						
4	QU$POP	$APTEXT	5:24								
1	RA@CYL@P	$APTEXT	16:17								
6	RA@LEN@P	$APTEXT	16:34								
7	RA@SEC@N	$APTEXT	16:25		16:25	F					
2	RA@SEC@P	$APTEXT	16:25	F	16:25						
0	RA@SEC@S	$APTEXT	16:25		16:25	F					
101	RQ$DISK	$APTEXT	10:24								
11	SM@FSI@N	$APTEXT	1: 4								
7	SM@NUM@N	$APTEXT	1: 4								
14	TF$ACOM	$APTEXT	7:27								
1	TM$BMR	$APTEXT	15:24								

3	IM$MAX	$APTEXT	11:26		
2	IM$SSD	$APTEXT	11:33	12:35	13: 8
20	IM@IW0@N	$APTEXT	13: 4	13: 4 F	
1	IM@IW0@P	$APTEXT	13: 4 F	13: 4	
0	IM@IW0@S	$APTEXT	13: 4	13: 4 F	
20	IM@IW1@N	$APTEXT	13: 5	13: 5 F	
2	IM@IW1@P	$APTEXT	13: 5 F	13: 5	
0	IM@IW1@S	$APTEXT	13: 5	13: 5 F	
20	IM@LM0@N	$APTEXT	13:20	13:20 F	
3	IM@LM0@P	$APTEXT	13:20 F	13:20	
0	IM@LM0@S	$APTEXT	13:20	13:20 F	
20	IM@LM1@N	$APTEXT	13:21	13:21 F	
4	IM@LM1@P	$APTEXT	13:21 F	13:21	
0	IM@LM1@S	$APTEXT	13:21	13:21 F	
0X	IMCB		4: 5 D	11:28	

SYMBOL QUALIFIER IS ACOM

1	$CONST	10:14 D	10:14 F	10:14	10:25 D	10:25 F	10:25	11:34 D	11:34 F
		11:34	14:25 D	14:25 F	14:41 D	14:41 F	14:41	17:38 D	17:38 F
		17:38	18:11 D	18:11 F	18:11	18:26 D	18:26 F	18:36 D	18:36 F
		19:14 D	19:14 F	19:21 D	19:21 F	19:21	19:36 D	19:36 F	19:36
		20:27 D	20:27 F	20:27	22:27 D	22:27 F	23: 1 D	23: 1 F	23: 1
		23: 3 D	23: 3 F	23: 3	23:39 D	23:39 F	23:42 D	23:42 F	23:42
463	$REGORG	2:18 D	2:18 F	2:19	2:19 D	2:20	2:20 D	2:21	2:21 D
		2:22	2:22 D	2:23	2:23 D	2:23 F	2:24	2:24 D	2:25
		2:25 D	2:26	2:26 D	2:27	2:27 D	2:28	2:28 D	2:29
		2:29 D	2:29 F	2:30	2:30 D	2:31	2:31 D	2:32	2:32 D
		2:33	2:33 D	2:34	2:34 D	2:34 F	2:35	2:35 D	2:36
		2:36 D	2:37	2:37 D	2:38	2:38 D	2:39	2:39 D	2:40
		2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2
		3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6
		3: 6 D	3: 7	3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3:10
		3:10 D	3:11	3:11 D	3:12	3:12 D	3:13	3:13 D	3:14
		3:14 D	3:15	3:15 D	3:16	3:16 D	3:17	3:17 D	3:18
		3:18 D	3:19	3:19 D	3:19 F	6:10	9:32	9:32 F	10:41
		23: 7							
0	%NP	1: 4	2:34 D						
430	%P	1: 4	2:34 D	2:34	9:32 F	9:32			
410	%S1	2:18	2:19 D	5:24	6:33	7:17	7:19	7:21	7:25
		9:12	9:13	9:14	9:15	9:16	9:17	9:27	9:36
		10:14	10:21	10:22	10:25	10:34	10:37	10:38	10:39

		10:40	10:41	11:13	11:25	11:28	11:29	11:34	12: 3
		12:11	12:14	12:17	12:20	12:24	12:25	12:31	12:32
		12:36	12:37	12:41	12:42	13: 4	13: 5	13: 9	13:10
		13:14	13:15	13:20	13:21	13:33	14: 5	14: 6	14:25
		14:29	14:35	14:38	14:41	15:33	15:34	15:41	16: 2
		16: 3	16:25	16:32	16:38	16:42	17: 8	17:35	17:36
		17:37	17:38	17:39	17:40	18: 6	18:11	18:17	18:20
		18:22	18:23	18:25	18:26	18:36	18:42	19:15	19:25
		19:30	19:36	19:38	20:14	20:15	20:18	20:24	20:27
		20:37	20:38	22:27	22:28	22:29	23: 1	23: 3	23:39
		23:41	24: 2	24: 3	24: 4				
411	%S2	2:18	2:20 D	18:21	18:24				
412	%S3	2:18	2:21 D						
413	%S4	2:18	2:22 D						
414	%S5	2:18	2:23 D						
415	%T1	2:24 D	13:40	18:13					
416	%T2	2:25 D	18:13						
417	%T3	2:26 D	18:13						
420	%T4	2:27 D							
421	%T5	2:28 D							
422	%T6	2:29 D							
423	%W1	2:30 D	6: 6	6: 9	6:25	6:26	6:32	7: 2	7: 3
		7: 6	7: 7	7: 8	7:14	7:18	7:23	7:24	7:27
		9:27	9:28	10:21	10:24	10:37	10:39	12: 6	12: 8
		12:12	12:41	13: 6	13:14	13:19	13:22	13:25	14:15
		14:29	15:14	17: 8	18:42	19: 2	19:10	19:12	20:15
		20:18	23:41						
424	%W2	2:31 D	7: 4	7: 5	7: 8	7:15	7:20	7:25	7:27
		10:22	10:24	10:38	10:40	12:42	13: 6	13: 9	13:15
		13:19	13:26	15:33	15:36	16:25	16:27	18:12	18:13
		18:20	18:23	18:30	24: 2	24: 3	24: 4		
425	%W3	2:32 D	13: 4	13: 6	13:20	13:22	13:25	16: 7	16:10
		16:15	16:17	16:18	16:20	16:27			
426	%W4	2:33 D	13: 5	13: 6	13:21	13:26	16:11	16:12	
427	%W5	2:34 D	15:41	16:16	16:28	16:37			
2444	ABORT	14:30	15:25	15:37	16:21	23:38 D			
431	ACT	2:36 D	5: 3	5:18	5:31	6: 5			
437	CHN	2:42 D	11:13	11:14	11:16	11:18	17:40	17:41	
2305	CHNCHK	7:16	19:35	22: 9 D					
453	CPI	3:12 D	9:17	9:32					
452	CPU	3:11 D	9:16	9:32					
441	CYL	3: 2 D	12:11	16:11					

436	DAI	2:41 D	6:16		7:17	7:19	7:21	7:23	7:25
		7:27	9:12	9:13	9:14	9:15	9:16	9:17	9:27
		9:36	10:14	10:21	10:22	10:25	10:37	10:38	10:39
		10:40	10:41	11:13	11:25	11:34	12: 3	12:11	12:14
		12:17	12:20	12:24	12:25	12:41	12:42	13:14	13:15
		13:33	14: 5	14: 6	14:12	14:25	14:27	14:35	14:38
		14:41	15:16	15:33	16: 4	17:35	17:37	18:36	18:38
		18:40	19: 1	19: 2	19:15	19:25	19:28	19:30	19:36
		19:38	19:41	20:14	20:15	20:24	20:37	20:38	21: 5
		22:27	22:28	22:29	23: 1	23: 3	23: 5	23: 7	23:15
		23:27							
440	DCB	3: 1 D	11:19	11:20	14:13	14:15	15: 4	15:34	15:41
		16: 2	16: 3	16:32	16:38	16:42	17: 9	17:42	18: 1
		18:20	18:23	18:42	19: 9	19:14	19:21	19:37	20: 2
		23:39	23:42	24: 2	24: 3	24: 4			
444	FNC	3: 5 D	12:20	14:16	17:39	18:10	18:41		
1624	GENDALS	15:15	15:17	15:26	15:38	16:22	17: 6 D		
442	HED	3: 3 D	12:14	16: 7					
435	IFC	2:40 D	7:21	7:27	7:36	7:38	7:39	7:41	7:42
		8: 2	9:22	9:24	9:29	9:36			
445	IOP	3: 6 D	6:42	7: 1	10:11	10:33	23:18		
2414	IOPMSG	9:38	10:15	23:14 D					
430	LIM	2:35 D	5:12	5:38	6: 2	6:11	6:36		
457	LLN	3:16 D	13:37	14: 1	14: 2	14: 3	14: 4	14: 5	
454	LNO	3:13 D	12:24	12:28	12:31	13:19	13:36	13:39	13:40
455	LN1	3:14 D	9:12	9:32	12:25	12:27	12:36	13:19	13:36
		13:37	13:38	13:40	13:41				
502	MANA	7:41	10: 6 D						
2277	MARX	9:39	17:24	21: 4 D					
2357	MBAD	11:14	11:16	11:21	11:26	11:37	12:12	12:15	12:18
		12:28	12:32	12:37	13:10	13:23	13:27	22:42 D	
2367	MBAD1	11:35	23: 2 D						
624	MDSK	7:38	11:11 D						
2172	MDVH	18:41	19: 2	19:11	19:22	19:40	20: 1 D		
32	MEMN	5:17 D	5:35	6:12					
37	MEMQ	5:13	5:20 D	5:28					
51	MEMS	5:22	5:26 D	10:42	17:13	19: 3	19:13	20: 4	21: 7
		23: 8	23:32						
65	MEMX	5:25	5:36 D						
2062	MFIRST	19: 1	19: 8 D						
1646	MOTH	7:42	17:33 D						
2200	MOTZ	8: 2	20:13 D						

2326	MRELDAL	10:12	17:23	21: 3	22:26 D				
434	MSG	2:39 D	6:20	6:42	7: 4	7:27			
1642	MSPECREL	7:39	17:22 D						
456	OFF	3:15 D	13:33	13:34	14: 4	19:30	19:31		
432	QPTR	2:37 D	5: 4	5: 7	5: 9	5:19	5:21	5:27	5:28
		6:33							
433	QUE	2:38 D	5:21	5:24	6:18	6:25	6:29		
446	SC0	3: 7 D	9:13	9:28	9:32	16: 3	16:31	17:35	17:36
		18:11	18:13	18:17	18:26	20:14	20:18	20:27	20:28
		20:30							
447	SC1	3: 8 D	9:14	9:28	9:32	16: 2	16:17	16:25	16:31
		16:32	16:34	16:42	17:37	17:38	17:39	17:40	18: 6
		18:40							
450	SC2	3: 9 D	9:23	9:26	9:32	16: 4	16: 5	16:20	16:27
451	SC3	3:10 D	13:34	13:36	16: 6	16:12	16:15	16:20	
443	SEC	3: 4 D	12:17	13:34	13:41	16:27			
461	TM	3:18 D	9:15	9:32	11:25	11:28	11:33	12:35	13: 8
		15:24							
462	TMP	3:19 D	11:29	13: 4	13: 5	13:20	13:21		
2427	TOAMSG	7:37	10:26	23:26 D					
460	TOT	3:17 D	13:41	14: 3	14: 6				

```
<macro>                    OVERLAY   D4DEM                                          D4DEM.4

                           COMMENT   'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                    ***********************************************************************   D4DEM.5
                    *                                                                     *   D4DEM.6
                    *        D4DEM     Disk demon for DD39/DD49 disks.                      *   D4DEM.7
                    *                                                                     *   D4DEM.8
                    ***********************************************************************   D4DEM.9
                                                                                              D4DEM.10
6     <macro>                 REGDEFS   ,,(NUMDONE,DCB,CHAN,DAL,STATUS,MCB,MOSU,MOSL,DTR,FLG,   D4DEM.11
                    ,ENTRY,CYL,HED,SEC,SC0,SC1,TOTAL,FNC,DONE,SEQ,BUFU,BUFL,CPU,CPL,WBC,QUE,   D4DEM.12
                    ,BINFO,BPTR,DINFO,DPTR,UNIT,TYPE,CODE,CPD,TPC,SPT,DIA,SUB,DPT,GIVE,TDIR,   D4DEM.13
                    ,HIFLG,HOFLG,SIFLG,SOFLG,MSFLG,LOCAL,LEN,NUM,TM,TMP,REL)                R13739AP.1
410                 %S1       EQUALS    $REGORG                                                  $REG     3
411                 %S2       EQUALS    $REGORG                                                  $REG     3
412                 %S3       EQUALS    $REGORG                                                  $REG     3
413                 %S4       EQUALS    $REGORG                                                  $REG     3
414                 %S5       EQUALS    $REGORG                                                  $REG     3
415                 %T1       EQUALS    $REGORG                                                  $REG     3
416                 %T2       EQUALS    $REGORG                                                  $REG     3
417                 %T3       EQUALS    $REGORG                                                  $REG     3
420                 %T4       EQUALS    $REGORG                                                  $REG     3
421                 %T5       EQUALS    $REGORG                                                  $REG     3
422                 %T6       EQUALS    $REGORG                                                  $REG     3
423                 %W1       EQUALS    $REGORG                                                  $REG     3
424                 %W2       EQUALS    $REGORG                                                  $REG     3
425                 %W3       EQUALS    $REGORG                                                  $REG     3
426                 %W4       EQUALS    $REGORG                                                  $REG     3
427                 %W5       EQUALS    $REGORG                                                  $REG     3
430                 NUMDONE   EQUALS    $REGORG                                                  $REG     3
431                 DCB       EQUALS    $REGORG                                                  $REG     3
432                 CHAN      EQUALS    $REGORG                                                  $REG     3
433                 DAL       EQUALS    $REGORG                                                  $REG     3
434                 STATUS    EQUALS    $REGORG                                                  $REG     3
435                 MCB       EQUALS    $REGORG                                                  $REG     3
436                 MOSU      EQUALS    $REGORG                                                  $REG     3
437                 MOSL      EQUALS    $REGORG                                                  $REG     3
440                 DTR       EQUALS    $REGORG                                                  $REG     3
441                 FLG       EQUALS    $REGORG                                                  $REG     3
442                 ENTRY     EQUALS    $REGORG                                                  $REG     3
443                 CYL       EQUALS    $REGORG                                                  $REG     3
444                 HED       EQUALS    $REGORG                                                  $REG     3
445                 SEC       EQUALS    $REGORG                                                  $REG     3
446                 SC0       EQUALS    $REGORG                                                  $REG     3
447                 SC1       EQUALS    $REGORG                                                  $REG     3
```

```
450     TOTAL   EQUALS  $REGORG                                         $REC    3
451     FNC     EQUALS  $REGORG                                         $REG    3
452     DONE    EQUALS  $REGORG                                         $REG    3
453     SEQ     EQUALS  $REGORG                                         $REG    3
454     BUFU    EQUALS  $REGORG                                         $REG    3
455     BUFL    EQUALS  $REGORG                                         $REG    3
456     CPU     EQUALS  $REGORG                                         $REG    3
457     CPL     EQUALS  $REGORG                                         $REG    3
460     WBC     EQUALS  $REGORG                                         $REG    3
461     QUE     EQUALS  $REGORG                                         $REG    3
462     BINFO   EQUALS  $REGORG                                         $REG    3
463     BPTR    EQUALS  $REGORG                                         $REG    3
464     DINFO   EQUALS  $REGORG                                         $REG    3
465     DPTR    EQUALS  $REGORG                                         $REG    3
466     UNIT    EQUALS  $REGORG                                         $REG    3
467     TYPE    EQUALS  $REGORG                                         $REG    3
470     CODE    EQUALS  $REGORG                                         $REG    3
471     CPD     EQUALS  $REGORG                                         $REG    3
472     TPC     EQUALS  $REGORG                                         $REG    3
473     SPT     EQUALS  $REGORG                                         $REG    3
474     DIA     EQUALS  $REGORG                                         $REG    3
475     SUB     EQUALS  $REGORG                                         $REG    3
476     DPT     EQUALS  $REGORG                                         $REG    3
477     GIVE    EQUALS  $REGORG                                         $REG    3
500     TDIR    EQUALS  $REGORG                                         $REG    3
501     HIFLG   EQUALS  $REGORG                                         $REG    3
502     HOFLG   EQUALS  $REGORG                                         $REG    3
503     SIFLG   EQUALS  $REGORG                                         $REG    3
504     SOFLG   EQUALS  $REGORG                                         $REG    3
505     MSFLG   EQUALS  $REGORG                                         $REG    3
506     LOCAL   EQUALS  $REGORG                                         $REG    3
507     LEN     EQUALS  $REGORG                                         $REG    3
510     NUM     EQUALS  $REGORG                                         $REG    3
511     TM      EQUALS  $REGORG                                         $REG    3
512     TMP     EQUALS  $REGORG                                         $REG    3
513     REL     EQUALS  $REGORG                                         $REG    3
                                                                 D4DEM.15
                EXT     %DCBQ                                    D4DEM.16
                EXT     %HISP                                    D4DEM.17
                EXT     %HISPBUF                                 D4DEM.18
                EXT     %MEMIO                                   D4DEM.19
                EXT     %MYID                                    D4DEM.20
```

```
EXT     %SSD                                                    D4DEM.21
EXT     %SSDBUF                                                 D4DEM.22
EXT     BMLOCL                                                  D4DEM.23
EXT     BNFLG                                                   D4DEM.24
EXT     CA                                                      D4DEM.25
EXT     CB                                                      D4DEM.26
EXT     CC                                                      D4DEM.27
EXT     CD                                                      D4DEM.28
EXT     CE                                                      D4DEM.29
EXT     CEREC                                                   D4DEM.30
EXT     CHNWTDN                                                 D4DEM.31
EXT     COMIO                                                   D4DEM.32
EXT     D$D4DEM                                                 D4DEM.33
EXT     DCCB                                                    D4DEM.34
EXT     DDALGETA                                                D4DEM.35
EXT     DGIVEUP                                                 D4DEM.36
EXT     DMGA                                                    D4DEM.37
EXT     DS                                                      D4DEM.38
EXT     DT                                                      D4DEM.39
EXT     D4ACT                                                   D4DEM.40
EXT     ED                                                      D4DEM.41
EXT     EH                                                      D4DEM.42
EXT     EI                                                      D4DEM.43
EXT     EG                                                      D4DEM.44
EXT     EMSGIOP                                                 D4DEM.45
EXT     ENQU                                                    D4DEM.46
EXT     ESWP                                                    D4DEM.47
EXT     HB                                                      D4DEM.48
EXT     HC                                                      D4DEM.49
EXT     HILOCL                                                  D4DEM.50
EXT     HMA                                                     D4DEM.51
EXT     HMD                                                     D4DEM.52
EXT     HOLOCL                                                  D4DEM.53
EXT     IMDAL                                                   D4DEM.54
EXT     O$D3ERR                                                 D4DEM.55
EXT     O$D40ERR                                            R13739AP.2
EXT     O$D4ERR                                                 D4DEM.56
EXT     QTIME                                                   D4DEM.57
EXT     SILOCL                                                  D4DEM.58
EXT     SOLOCL                                                  D4DEM.59
EXT     TMCB                                                    D4DEM.60
```

```
                                 ***************************  ****************************************   D4DEM.62
                                 *                                                                    *   D4DEM.63
                                 *        START     This section is the main loop for the overlay.    *   D4DEM.64
                                 *                                                                    *   D4DEM.65
                                 **********************************************************************   D4DEM.66
                                                                                                        D4DEM.67
  6                              START    *                                                             D4DEM.68
  6   003000                              I = 1                                                         D4DEM.69
  7   010000  024435                      R!MCB = 0                    .Initialize                      D4DEM.70
 11   024477                              R!GIVE = A                   .Giveup flag                     D4DEM.71
 12   072155                              R = DALQ                     .Process DAL start queue         D4DEM.72
 13   077000 /002543                      R = MEMIO                    .Move data                       D4DEM.73
 15   077000 /001345                      R = DCBQ                     .Process DCB queue               D4DEM.74
 17   072042                              R = DONEQ                    .Process DAL done queue          D4DEM.75
 20    <macro>                            $IF (R!GIVE # 0)             .If time to giveup               D4DEM.76
                                                                                        .>>>>> 0            $IF      1
 23   020435  137000 /005104               R = RELMEM, R!MCB # 0       .Release memory                  D4DEM.77
 26   002000  000000                       I = 0                                                        D4DEM.78
 30   014000 /000000                       A = D$D4DEM                 .Demon flag                      D4DEM.79
 32    <macro>                             RKERN DGIVEUP               .Give up                         D4DEM.80
 34   071026                               P = START                   .Take it from the top            D4DEM.81
 35    <macro>                            $ELSE                                                         D4DEM.82
 36   002000  000000                       I = 0                                                        D4DEM.83
 40    <macro>                             GET A,DCB@DA,R!%DCBQ        .DAL start queue                 D4DEM.84
 44   107036                               P = START, A # 0            .If something there              D4DEM.85
 45    <macro>                             GET A,DCB@DN,R!%DCBQ        .DAL done queue                  D4DEM.86
 51   107043                               P = START, A # 0            .If something there              D4DEM.87
 52    <macro>                             GET A,DCB@QU,R!%DCBQ        .DCB queue                       D4DEM.88
 53   107045                               P = START, A # 0            .If something there              D4DEM.89
 54   077000 /003563                       R = QCHECK                  .Check MEMIO queues              D4DEM.90
 56   107050                               P = START, A # 0            .If something there              D4DEM.91
 57    <macro>                             PKERN ESWP                  .All done                        D4DEM.92
 61    <macro>                            $ENDIF                                                        D4DEM.93
                                                                                        .<<<<< 0            $ENDIF   1
```

```
                                 ********************************************************************   D4DEM.95
                                 *                                                                  *   D4DEM.96
                                 *        DONEQ     This section processes the done queue for       *   D4DEM.97
                                 *                  for all channels.                               *   D4DEM.98
                                 *                                                                  *   D4DEM.99
                                 ********************************************************************   D4DEM.100
                                                                                                        D4DEM.101
 61                              DONEQ    *                                                             D4DEM.102
 61    <macro>                            ADDRESS R!QUE,DCB@DN,R!%DCBQ  .DAL done queue address        D4DEM.103
 64    <macro>                            $UNTIL ((R!QUE) = 0)          .Until queue empty             D4DEM.104
 66   030461  024433                        R!DAL = (R!QUE)             .First on queue                D4DEM.105
 70    <macro>                              GET R!FNC,DL@FC,R!DAL       .Function code                 D4DEM.106
 76    <macro>                              $IF (R!FNC = DAF$WRBH),OR,(R!FNC = DAF$WBFR)               D4DEM.107
                                                                                   .>>>>> 0              $IF      1
105    <macro>                                GET A,DA@IFC,R!DAL        .Internal fnc code             D4DEM.108
111    <macro>                                $IF (A # IM$STAT)         .If first reply not sent     R16410KA.1
                                                                                   .>>>>> 1              $IF      1
114   010001  024477                           R!GIVE = 1              .Giveup, let ICOM send it    R16410KA.2
116   001000                                   EXIT                                                 R16410KA.3
117    <macro>                                $ENDIF                                               R16410KA.4
                                                                                   .<<<<< 1            $ENDIF    1
117    <macro>                              $ENDIF                                                     D4DEM.110
                                                                                   .<<<<< 0            $ENDIF    1
117   002000  000000                        I = 0                                                      D4DEM.111
121   020461  024000                        R!ED = R!QUE                .Queue address                 D4DEM.112
123    <macro>                              RKERN EDQU                  .Take first entry              D4DEM.113
125   003000                                I = 1                                                      D4DEM.114
126    <macro>                              $IF (0 # R!FNC & O'20)      .Internal request              D4DEM.115
                                                                                   .>>>>> 0              $IF      1
135    <macro>                                ADDRESS R!%W2,DA@DID,R!DAL .Push/pop cell                D4DEM.116
140    <macro>                                POP R!%W2                 .Pop the activity              D4DEM.117
150    <macro>                              $ELSE                                                      D4DEM.118
151   020451  013014  137000                  R = STAT, R!FNC # DAF$WRBH .Status to mainframe          D4DEM.119
     /004501
155    <macro>                                RELDAL R!DAL              .Release local DAL             D4DEM.120
165    <macro>                              $ENDIF                                                     D4DEM.121
                                                                                   .<<<<< 0            $ENDIF    1
165    <macro>                            $ENDTIL                                                      D4DEM.122
166   001000                              EXIT                                                         D4DEM.123
```

```
                                 *************************.  ****************************************   D4DEM.12
                                 *                                                                *   D4DEM.126
                                 *        DALQ:     This section processes the DAL start queue for *   D4DEM.127
                                 *                  all channels.                                 *   D4DEM.128
                                 *                                                                *   D4DEM.129
                                 ******************************************************************   D4DEM.130
                                                                                                     D4DEM.131
167                              DALQ     *                                                          D4DEM.132
167    <macro>                           ADDRESS R!QUE,DCB@DA,R!%DCBQ  .DAL start queue address    D4DEM.133
172                              NEXTDAL  *                                                          D4DEM.134
172    002000  000000                     I = 0                                                      D4DEM.135
174    020461  024000                     R!ED = R!QUE                  .Queue address              D4DEM.136
176    <macro>                            RKERN EDQU                    .Take first entry           D4DEM.137
200    020000  024433                     R!DAL = R!EH                  .DAL address                D4DEM.138
202    003000                             I = 1                                                      D4DEM.139
203    103002  001000                     EXIT, A = 0                   .Queue empty                D4DEM.140
205    <macro>                            GET R!CHAN,DL@CHN,R!DAL       .Channel                    D4DEM.141
213    020432  013020  016000             R!DCB = R!CHAN - O'20 + DCCB  .DCB cell                   D4DEM.142
      /000000  024431
220    030431  024431                     R!DCB = (R!DCB)               .DCB address                D4DEM.143
222    077000 /007247                     R = STRIP                     .Get DAL parameters         D4DEM.144
                                                                                                     D4DEM.145
                                 *        Check for a contiguous request.                            D4DEM.146
                                                                                                     D4DEM.147
224    020451  011020  024410             P = QDAL, 0 # R!FNC & O'20    .If internal request        D4DEM.148
       010000  023410  103450
232    <macro>                            GET R!SCO,DK@DL1,R!DCB        .Last DAL on queue          D4DEM.149
237    <macro>                            $IF (A # 0)                   .If previous request        D4DEM.150
                                                                                      .>>>>> 0          $IF      1
241    <macro>                              GET A,DL@FC,R!SCO           .Function code              D4DEM.151
246    023451  103432                       P = QDAL, A # R!FNC         .If mismatch                D4DEM.152
250    <macro>                              GET A,DL@DIA,R!SCO          .Diagnostic flag            D4DEM.153
255    023474  103423                       P = QDAL, A # R!DIA         .If mismatch                D4DEM.154
257    <macro>                              GET A,DL@SUB,R!SCO          .Subfunction code           D4DEM.155
265    023475  103413                       P = QDAL, A # R!SUB         .If mismatch                D4DEM.156
267    <macro>                              GET A,DL@TM,R!SCO           .Target memory              D4DEM.157
274    023511  103404                       P = QDAL, A # R!TM          .If mismatch                D4DEM.158
276    <macro>                              GET R!%W1,DL@UNT,R!SCO      .Unit                       D4DEM.159
305    <macro>                              GET R!%W2,DL@CYL,R!SCO      .Cylinder                   D4DEM.160
313    <macro>                              GET R!%W3,DL@HED,R!SCO      .Head group                 D4DEM.161
321    <macro>                              GET R!%W4,DL@SEC,R!SCO      .Sector                     D4DEM.162
327    <macro>                              GET R!%W5,DL@TOT,R!SCO      .Total sectors              D4DEM.163
```

```
334   020426  022427  024426          R!%W4 = R!%W4 + R!%W5        .Calculate ending address          D4DEM.164
337    <macro>                        $UNTIL (R!%W4 < R!SPT)                                          D4DEM.165
342   020426  023473  024426            R!%W4 = R!%W4 - R!SPT      .Decrement                         D4DEM.166
345   026425                            R!%W3 = R!%W3 + 1          .Next track                        D4DEM.167
346    <macro>                          $IF (A = R!TPC)            .If end of cylinder                D4DEM.168
                                                                                        .>>>>> 1          $IF      1
351   010000  024425                      R!%W3 = 0                .Reset                             D4DEM.169
353   026424                              R!%W2 = R!%W2 + 1        .Next cylinder                     D4DEM.170
354    <macro>                          $ENDIF                                                          D4DEM.171
                                                                                        .<<<<< 1          $ENDIF   1
354    <macro>                        $ENDTIL                                                         D4DEM.172
355    <macro>                      $ELSE                        .If only request                   D4DEM.173
356   020511  013001  102321            P = QDAL, R!TM = TM$BMR    .If BMR, don't use read ahead      D4DEM.174
361    <macro>                          GET A,DK@DIA,R!DCB         .Diagnostic mode flag              D4DEM.175
367   023474  103311                    P = QDAL, A # R!DIA        .If mismatch                       D4DEM.176
371    <macro>                          GET A,DK@SUB,R!DCB         .Read/write subfunction            D4DEM.177
376   023475  103302                    P = QDAL, A # R!SUB        .If not the same                   D4DEM.178
400    <macro>                          GET A,DK@TM,R!DCB          .Target memory                     D4DEM.179
405   023511  103273                    P = QDAL, A # R!TM         .If mismatch                       D4DEM.180
407    <macro>                          GET R!%W1,DK@RAU,R!DCB     .Unit                              D4DEM.181
415    <macro>                          GET R!%W2,DK@RAC,R!DCB     .Cylinder                          D4DEM.182
422    <macro>                          GET R!%W3,DK@RAH,R!DCB     .Head group                        D4DEM.183
431    <macro>                          GET R!%W4,DK@RAS,R!DCB     .Sector                            D4DEM.184
                                                                                                        D4DEM.185
                                *     Check for match on other than the first read ahead sector.        D4DEM.186
                                                                                                        D4DEM.187
437   020512  023000  103213            P = NODICE, R!TMP # R!%MYID .If remote path                    D4DEM.188
442   020451  011001  024410            P = NODICE, 0 = R!FNC & 1  .If write                           D4DEM.189
      010000  023410  102205
450    <macro>                          GET R!%W5,DK@RAD,R!DCB     .Read aheads done                  D4DEM.190
456   041000  013002  100174            P = NODICE, A < 2          .Not enough to check               D4DEM.191
461   020466  023423  103171            P = NODICE, R!UNIT # R!%W1 .Wrong unit                        D4DEM.192
464   020443  023424  103166            P = NODICE, R!CYL # R!%W2  .Wrong cylinder                    D4DEM.193
467   020444  023425  100163            P = NODICE, R!HED < R!%W3  .Wrong head                        D4DEM.194
472    <macro>                          $IF (R!HED = R!%W3),AND,(R!SEC <= R!%W4) .Check sector          D4DEM.195
                                                                                        .>>>>> 1          $IF      1
504   070150                              P = NODICE                                                    D4DEM.196
505    <macro>                          $ENDIF                                                          D4DEM.197
                                                                                        .<<<<< 1          $ENDIF   1
565   010000  024441                    R!FLG = 0                  .Initialize for search             D4DEM.198
507   024507                            R!LEN = A                                                     D4DEM.199
510    <macro>                          $UNTIL (R!LEN = R!%W5),OR,(R!FLG # 0) .Find match             D4DEM.200
```

```
516   026426                          R!%W4 = R!%W4              .Next sector                          D4DEM.20
516   <macro>                         $IF (A = R!SPT)            .Check head switch                    D4DEM.202
                                                                                        .>>>>> 1          $IF      1
521   010000  024426                    R!%W4 = 0                .First sector                         D4DEM.203
523   026425                            R!%W3 = R!%W3 + 1        .Next head                            D4DEM.204
524   <macro>                         $ENDIF                                                           D4DEM.205
                                                                                        .<<<<< 1          $ENDIF   1
524   <macro>                         $IF (R!%W3 = R!HED),AND,(R!%W4 = R!SEC)                          D4DEM.206
                                                                                        .>>>>> 1          $IF      1
534   010001  024441                    R!FLG = 1                .Match found                          D4DEM.207
536   <macro>                         $ENDIF                                                           D4DEM.208
                                                                                        .<<<<< 1          $ENDIF   1
536   026507                            R!LEN = R!LEN + 1        .Count sector difference              D4DEM.209
537   <macro>                       $ENDTIL                                                            D4DEM.210
540   020441  102140                P = QDAL, R!FLG = 0          .If no match                          D4DEM.211
542   002000  000060                I = 0                                                              D4DEM.212
544   <macro>                       PUT R!HED,DK@RAH,R!DCB       .Adjust read ahead address            D4DEM.213
560   <macro>                       PUT R!SEC,DK@RAS,R!DCB                                             D4DEM.214
573   <macro>                       FLDSUB R!LEN,DK@RAD,R!DCB    .Adjust read ahead count              D4DEM.215
616   <macro>                       GET R!NUM,DK@CNT,R!DCB       .Number of MOS buffers                D4DEM.216
624   <macro>                       GET A,DK@BIO,R!DCB           .Adjust pointer to next R.A.          D4DEM.217
631   022507  024463                R!BPTR = A + R!LEN                                                 D4DEM.218
633   041000  023510  100004        R!BPTR = R!BPTR - R!NUM, A >= R!NUM .Check wrap                    D4DEM.219
      020463  023510  024463
641   <macro>                       PUT R!BPTR,DK@BIO,R!DCB                                            D4DEM.220
654   <macro>                     $ENDIF                                                               D4DEM.221
                                                                                        .<<<<< 0          $ENDIF   1
654                       NODICE    *                                                                  D4DEM.222
654   020423  023466  103023        P = QDAL, R!%W1 # R!UNIT     .If mismatch                          D4DEM.223
657   020424  023443  103020        P = QDAL, R!%W2 # R!CYL                                            D4DEM.224
662   020425  023444  103015        P = QDAL, R!%W3 # R!HED                                            D4DEM.225
665   020426  023445  103012        P = QDAL, R!%W4 # R!SEC                                            D4DEM.226
670   <macro>                       STORE 1,DL@CON,R!DAL         .Set continuation flag                D4DEM.227
701                       QDAL      *                                                                  D4DEM.228
701   002000  000000                I = 0                                                              D4DEM.229
703   <macro>                       ADDRESS R!ED,DK@DLO,R!DCB    .DCB DAL queue                        D4DEM.230
706   020433  024000                R!EH = R!DAL                 .Entry                                D4DEM.231
710   <macro>                       RKERN ENQU                   .Queue it                             D4DEM.232
712   <macro>                       GET R!FLG,DK@ACT,R!DCB       .Channel activity                     D4DEM.233
714   020451  011020  024410        P = ABORTRA, 0 # R!FNC & O'20 .Internal request                    D4DEM.234
      010000  023410  127000
     /001321
```

```
                                                                                D4DEM.235
                        *       Write request processing.                       D4DEM.236
                                                                                D4DEM.237
 723    <macro>                 $IF (0 = R!FNC & 1)          .If write request  D4DEM.238
                                                                  .>>>>> 0          $IF      1
 733    020441  013002  127000    P = ABORTRA, R!FLG # DK$WRIT .Abort read aheads D4DEM.239
       /001321
 737    <macro>                   GET A,DKS@CR,R!DCB         .Create error recovery flag D4DEM.240
 744    107552                    P = NEXTDAL, A # 0         .If error detected  D4DEM.241
 745    <macro>                   GET R!DAL,DK@DL0,R!DCB     .First on queue     D4DEM.242
 752    024447                    R!SC1 = A                  .Save               D4DEM.243
 753    077000 /007274            R = DALPAR                 .Get parameters     D4DEM.244
 755    077000 /006732            R = FILL                   .Ensure local buffers are full D4DEM.245
 757    <macro>                   GET R!DAL,DK@DL1,R!DCB     .New DAL            D4DEM.246
 764    <macro>                   $IF (R!DAL = (R!SC1))      .If next in line    D4DEM.247
                                                                  .>>>>> 1          $IF      1
 771    <macro>                     GET R!%W2,DL@TOT,R!SC1   .Total              D4DEM.248
 776    <macro>                     GET R!%W3,DL@DON,R!SC1   .Done               D4DEM.249
1003    <macro>                     $IF (1 = R!%W2 - R!%W3)  .If writing last sector D4DEM.250
                                                                  .>>>>> 2          $IF      1
1013    <macro>                       GET A,DL@CON,R!DAL     .Connect flag       D4DEM.251
1021    106627                        P = NEXTDAL, A = 0     .Not contiguous     D4DEM.252
1022    <macro>                       GET R!ENTRY,DK@CLB,R!DCB .Current buffer descriptor D4DEM.253
1027    077000 /007565                R = NEXTBUF            .Next               D4DEM.254
1031    <macro>                       STORE 1,LB@CP,R!ENTRY  .Set continue pending D4DEM.255
1042    <macro>                     $ENDIF                   D4DEM.256
                                                                  .<<<<< 2          $ENDIF   1
1042    <macro>                   $ENDIF                     D4DEM.257
                                                                  .<<<<< 1          $ENDIF   1
                                                                                D4DEM.258
                        *       Read request processing.                        D4DEM.259
                                                                                D4DEM.260
1042    <macro>                 $ELSEIF (R!FLG = 0),OR,(0 # R!FLG & O'104) .Idle, rlse, or RA. D4DEM.261
1054    <macro>                   GET R!SCO,DK@RAD,R!DCB     .Read aheads done   D4DEM.262
1062    <macro>                   $IF (R!FLG = DK$RAHD),OR,(R!SCO # 0) .If some to check D4DEM.263
                                                                  .>>>>> 1          $IF      1
1070    <macro>                     GET A,DK@INV,R!DCB       .Invalid data flag  D4DEM.264
1076    107704                      P = NEXTDAL, A # 0       .If read ahead data invalid D4DEM.265
1077    <macro>                     GET A,DL@CON,R!DAL       .Continue flag      D4DEM.266
1105    <macro>                     $IF (A = 0)              .If no match        D4DEM.267
                                                                  .>>>>> 2          $IF      1
1107    <macro>                       STORE 1,DK@INV,R!DCB   .Invalidate the read ahead data D4DEM.268
```

```
1120   070201                          P = ABORTRA              .Abort read aheads                D4DEM.26
1121    <macro>                      $ENDIF                                                        D4DEM.270
                                                                                .<<<<< 2              $ENDIF   1
1121    <macro>                      MINIMUM R!NUM,(R!SCO,R!TOTAL) .Number to steal                D4DEM.271
1132    <macro>                      FLDADD R!NUM,DK@SRA,R!DCB .Successful R.A.'s                  D4DEM.272
1140    <macro>                      FLDSUB R!NUM,DK@RAD,R!DCB .Decrement number done              D4DEM.273
1163    <macro>                      PUT R!NUM,DL@RAC,R!DAL    .Read ahead count                   D4DEM.274
1176    <macro>                      PUT R!NUM,DL@RAT,R!DAL    .Read ahead total                   D4DEM.275
1211    <macro>                      PUT R!NUM,DL@DON,R!DAL    .Sectors done                       D4DEM.276
1216    <macro>                      PUT R!NUM,DL@SEQ,R!DAL    .MEMIO sequence number              D4DEM.277
1223    <macro>                      $IF (R!NUM = R!TOTAL)     .IO request complete                D4DEM.278
                                                                                .>>>>> 2              $IF      1
1227    <macro>                        $IF (R!FLG = DK$RAHD),OR,(R!SCO > R!TOTAL) .Some left       D4DEM.279
                                                                                .>>>>> 3              $IF      1
1240   020445  022450  024445            R!SEC = R!SEC + R!TOTAL                                   D4DEM.280
1243   077000 /007255                    R = CALC                 .Calculate new start             D4DEM.281
1245    <macro>                          PUT R!SEC,DK@RAS,R!DCB .Save address                      D4DEM.282
1260    <macro>                          PUT R!HED,DK@RAH,R!DCB                                    D4DEM.283
1274    <macro>                          PUT R!CYL,DK@RAC,R!DCB                                    D4DEM.284
1301    <macro>                        $ENDIF                                                      D4DEM.285
                                                                                .<<<<< 3              $ENDIF   1
1301    <macro>                      $ELSEIF (R!FLG = DK$RAHD) .If reading ahead now               D4DEM.286
1306    <macro>                        STORE 0,DKF@RA,R!DCB    .Clear read ahead flag              D4DEM.287
1312    <macro>                        STORE 1,DKF@RD,R!DCB    .Set read flag                      D4DEM.288
1317    <macro>                      $ENDIF                                                        D4DEM.289
                                                                                .<<<<< 2              $ENDIF   1
1317    <macro>                    $ENDIF                                                          D4DEM.290
                                                                                .<<<<< 1              $ENDIF   1
1317    <macro>                  $ENDIF                                                            D4DEM.291
                                                                                .<<<<< 0              $ENDIF   1
1317   075000 /000172             P = NEXTDAL                  .Next                               D4DEM.292
                                                                                                    D4DEM.293
                          *        Abort the read ahead process.                                    D4DEM.294
                                                                                                    D4DEM.295
1321                      ABORTRA  *                                                                D4DEM.296
1321    <macro>                  $IF (R!FLG = 0),OR,(0 # R!FLG & O'104) .Idle, rise, or R.A.       D4DEM.297
                                                                                .>>>>> 0              $IF      1
1332    <macro>                    STORE 1,DK@ABT,R!DCB         .Set abort flag                   D4DEM.298
1343    <macro>                  $ENDIF                                                            D4DEM.299
                                                                                .<<<<< 0              $ENDIF   1
1343   075000 /000172             P = NEXTDAL                  .Next                               D4DEM.300
```

```
                                 ******************************************************************   D4DEM.302
                                 *                                                                *   D4DEM.303
                                 *        DCBQ      This section processes the DCB's on the DCB queue.  *   D4DEM.304
                                 *                  It starts disk I/O for new requests,                *   D4DEM.305
                                 *                  re-enables channels waiting for resources,          *   D4DEM.306
                                 *                  creates the error recovery activity,                *   D4DEM.307
                                 *                  and transfers read ahead and write behind data.     *   D4DEM.308
                                 *                                                                *   D4DEM.309
                                 ******************************************************************   D4DEM.310
                                                                                                        D4DEM.311
1345                             DCBQ     *                                                             D4DEM.312
1345   010000  024430                     R!NUMDONE = 0                  .Initialize                    D4DEM.313
                                                                                                        D4DEM.314
                                 *        Take the next DCB off queue.                                  D4DEM.315
                                                                                                        D4DEM.316
1347                             NEXT     *                                                             D4DEM.317
1347    <macro>                            $IF (R!NUMDONE >= D4D$LIM)    .If time to giveup control   D4DEM.318
                                                                                          .>>>>> 0          $IF      1
1353   010001  024477                       R!GIVE = 1                   .Set giveup flag              D4DEM.319
1355   001000                               EXIT                                                       D4DEM.320
1356    <macro>                            $ENDIF                                                      D4DEM.321
                                                                                          .<<<<< 0          $ENDIF   1
1356    <macro>                            GET R!DCB,DCB@QU,R!%DCBQ      .Next DCB on queue            D4DEM.322
1360   103002  001000                      EXIT, A = 0                   .If none                      D4DEM.323
1362   002000  000000                      I = 0                                                       D4DEM.324
1364    <macro>                            GET R!%W1,DK@LNK,R!DCB        .Pointer to next              D4DEM.325
1371    <macro>                            PUT R!%W1,DCB@QU,R!%DCBQ      .Set new head                 D4DEM.326
1373    <macro>                            $IF (R!%W1 = 0)               .If queue empty               D4DEM.327
                                                                                          .>>>>> 0          $IF      1
1376    <macro>                              STORE 0,DCB@QL,R!%DCBQ      .Clear tail                   D4DEM.328
1403    <macro>                            $ENDIF                                                      D4DEM.329
                                                                                          .<<<<< 0          $ENDIF   1
1403    <macro>                            STORE 0,DK@LNK,R!DCB          .Clear link                   D4DEM.330
1410    <macro>                            GET R!REL,DKS@UR,R!DCB        .Unit release flag          R13739AP.3
1417    <macro>                            STORE 0,DKS@UR,R!DCB          .Clear it                   R13739AP.4
1426   003000                              I = 1                                                       D4DEM.331
1427   026430                              R!NUMDONE = R!NUMDONE + 1     .Increment number done        D4DEM.332
1430    <macro>                            GET R!CHAN,DK@CHN,R!DCB       .Channel number               D4DEM.333
1436    <macro>                            GET R!TYPE,DK@TYP,R!DCB       .Device type                  D4DEM.334
                                                                                                        D4DEM.335
                                 *        Create the error recovery activity.                           D4DEM.336
                                                                                                        D4DEM.337
```

```
144     <macro>                         GET A,DKS@CR,R!DC.              .Create flag                          D4DEM.33
1452    <macro>                         $IF (A # 0)                     .If set                               D4DEM.339
                                                                                                .>>>>> 0          $IF      1
1454    <macro>                           $IF (R!TYPE = DSK$DD39)       .If DD39                              D4DEM.340
                                                                                                .>>>>> 1          $IF      1
1460    014000 /000000  024424            R!%W2 = O$D3ERR                                                     D4DEM.341
1463    <macro>                           $ELSEIF (R!TYPE = DSK$DD40)   .If DD40                              R13739AP.5
1470    014000 /000900  024424            R!%W2 = O$D40ERR                                                    R13739AP.6
1473    <macro>                           $ELSEIF (R!TYPE = DSK$DD49),OR,(R!TYPE = DSK$DD50)                  D4DEM.342
1503    014000 /000060  024424            R!%W2 = O$D4ERR                                                     D4DEM.343
1506    <macro>                           $ELSE                                                               D4DEM.344
1507    <macro>                             $PUNTIF CODE=PT$DEV         .Illegal device type                  D4DEM.345
1511    <macro>                           $ENDIF                                                              D4DEM.346
                                                                                                .<<<<< 1          $ENDIF   1
1511    <macro>                           CREATE R!%W2,(R!CHAN,R!DCB),PR1=0,TYPE=NUMBER                       D4DEM.347
1532    <macro>                           $PUNTIF (A # 0)               .If can't create                      D4DEM.348
1534    <macro>                           STORE 1,DKF@ER,R!DCB          .Set error recovery flag              D4DEM.349
1542    <macro>                           STORE 0,DKS@CR,R!DCB          .Clear create flag                    D4DEM.350
1551    071202                            P = NEXT                                                            D4DEM.351
1552    <macro>                         $ENDIF                                                                D4DEM.352
                                                                                                .<<<<< 0          $ENDIF   1
1552    <macro>                         GET A,DKF@ER,R!DCB              .Error recovery flag                  D4DEM.353
1555    107206                          P = NEXT, A # 0                 .If recovering now                    D4DEM.354
                                                                                                              D4DEM.355
                                *       Re-enable a channel waiting for resources.                            D4DEM.356
                                                                                                              D4DEM.357
1556    <macro>                         GET A,DKS@RS,R!DCB              .Waiting for resource flag            D4DEM.358
1564    <macro>                         $IF (A # 0)                     .If set                               D4DEM.359
                                                                                                .>>>>> 0          $IF      1
1566    <macro>                           STORE 0,DKS@RS,R!DCB          .Clear flag                           D4DEM.360
1575    <macro>                           GET R!DAL,DK@DLO,R!DCB        .MDAL address                         D4DEM.361
1602    102005                            P = IOGO, A = 0               .If already taken care of             D4DEM.362
1603    077000 /007247                    R = STRIP                     .Get MDAL parameters                  D4DEM.363
1605    077000 /004350                    R = INITB                     .Start I/O to disk                    D4DEM.364
1607    <macro>                         $ENDIF                                                                D4DEM.365
                                                                                                .<<<<< 0          $ENDIF   1
                                                                                                              D4DEM.366
                                *       Keep the I/O going.                                                   D4DEM.367
                                                                                                              D4DEM.368
1607                            IOGO    *                                                                     D4DEM.369
1607    002000  000000                  I = 0                                                                 D4DEM.370
1611    <macro>                         GET A,DK@SRV,R!DCB              .All service flags                    D4DEM.371
```

```
1616   107247                    P = NEXT, A # 0              .If service requested          D4DEM.372
1617    <macro>                  GET R!DAL,DK@DLO,R!DCB       .Current MDAL                  D4DEM.373
1624    <macro>                  GET R!FLG,DK@ACT,R!DCB       .Channel activity              D4DEM.374
1626    <macro>                  $IF (R!DAL # 0)              .If something there            D4DEM.375
                                                                                .>>>>> 0          $IF      1
1632   077000 /007247              R = STRIP                  .Get parameters from MDAL      D4DEM.376
                                                                                               D4DEM.377
                                  *  Check for useful read ahead data.                       D4DEM.378
                                                                                               D4DEM.379
1634    <macro>                     $IF (R!FNC = DAF$READ),AND,(R!TMP # R!%MYID)              D4DEM.380
                                                                                .>>>>> 1          $IF      1
1644    <macro>                       $IF (R!SEQ # 0)          .If part of request complete   D4DEM.381
                                                                                .>>>>> 2          $IF      1
1647   077000 /007605                 R = SENDTR               .Send DTR to BIOP              D4DEM.382
1651    <macro>                       $ENDIF                                                  D4DEM.383
                                                                                .<<<<< 2          $ENDIF   1
1651    <macro>                     $ENDIF                                                    D4DEM.384
                                                                                .<<<<< 1          $ENDIF   1
1651    <macro>                     $IF (R!FLG = 0),OR,(R!FLG = DK$RAHD) .If idle or R.A.     D4DEM.385
                                                                                .>>>>> 1          $IF      1
1660   020451  011001               A = R!FNC & 1                                             D4DEM.386
1662    <macro>                     $IF (A # 0)              .If read request               D4DEM.387
                                                                                .>>>>> 2          $IF      1
1664   020452  023450  126000         P = READAH, R!DONE = R!TOTAL .If request done           D4DEM.388
      /002062
1670    <macro>                     $ENDIF                                                    D4DEM.389
                                                                                .<<<<< 2          $ENDIF   1
                                                                                               D4DEM.390
                                  *  Abort read aheads, if necessary.                        D4DEM.391
                                                                                               D4DEM.392
1670    <macro>                     $IF (R!FLG # 0)          .If read aheads active         D4DEM.393
                                                                                .>>>>> 2          $IF      1
1673    <macro>                       STORE 1,DK@ABT,R!DCB   .Set abort flag                 D4DEM.394
1704   071335                         P = NEXT                                                D4DEM.395
1705    <macro>                     $ENDIF                                                    D4DEM.396
                                                                                .<<<<< 2          $ENDIF   1
1705    <macro>                     GET R!%W1,DK@RAD,R!DCB   .Read aheads done              D4DEM.397
1713    <macro>                     $IF (R!%W1 # 0),AND,(R!WBC = 0) .If R.A. abort needed     D4DEM.398
                                                                                .>>>>> 2          $IF      1
1723    <macro>                       GET R!LOCAL,DK@CLB,R!DCB .Current local buffer          D4DEM.399
1730    <macro>                       GET A,DK@FLB,R!DCB     .First local buffer              D4DEM.400
1734    <macro>                       $IF (A = R!LOCAL)      .If first                        D4DEM.401
```

```
                                                                  .>>>>> 3          $IF      1
1737    <macro>                          GET R!LOCAL,DK@LLB,R!DCB .Wrap to last          D4DEM.402
1744    <macro>                        $ELSE                                             D4DEM.403
1745   020506  013006  024506            R!LOCAL = R!LOCAL - LB@LE .Back up one          D4DEM.404
1750    <macro>                        $ENDIF                                            D4DEM.405
                                                                  .<<<<< 3          $ENDIF   1
1750    <macro>                          GET A,LB@FUL,R!LOCAL     .Buffer full flag      D4DEM.406
1755   127000 /002502                    P = REQUEUE, A # 0       .If data still there   D4DEM.407
1757   020512  024440                    R!DTR = R!TMP            .Save register         D4DEM.408
1761    <macro>                          GET R!TMP,DK@TMP,R!DCB   .IOP with path to TM   D4DEM.409
1770   077000 /004562                    R = READMCB              .Read in MCB           D4DEM.410
1772   126000 /002517                    P = WHOA, A = 0          .If no local memory    D4DEM.411
1774    <macro>                          STORE 0,DK@RAD,R!DCB     .Indicate R.A.'s gone  D4DEM.412
2003   020465  024463                    R!BPTR = R!DPTR          .Set BIOP pointer      D4DEM.413
2005   077000 /004755                    R = WRITB                .Update BIOP part of MCB D4DEM.414
2007   020440  024512                    R!TMP = R!DTR            .Restore register      D4DEM.415
2011    <macro>                        $ENDIF                                            D4DEM.416
                                                                  .<<<<< 2          $ENDIF   1
2011    <macro>                        STORE 0,DK@ABT,R!DCB       .Clear R.A. abort flag D4DEM.417
2020    <macro>                        STORE 0,DK@INV,R!DCB       .Clear invalid data flag D4DEM.418
2027   077000 /003615                  R = INITA                  .Initiate I/O          D4DEM.419
2031   002000  000000                  I = 0                                           R13739GA.1
2033    <macro>                        GET R!DAL,DK@DLO,R!DCB     .Restore register      D4DEM.420
2040   106471                          P = NEXT, A = 0            .If DAL has gone away  R13739GA.2
2041   020451  011001                  A = R!FNC & 1                                   R13739GA.3
2043   103017                          P = READAH, A # 0          .Process read aheads   R13739GA.4
2044   077000 /007274                  R = DALPAR                 .Restore parameters    D4DEM.423
2046   070205                          P = WRITEBH                .Check for write-behind D4DEM.424
2047    <macro>                      $ELSEIF (R!FLG = DK$RLSE)    .If release in progress D4DEM.425
2054   071505                          P = NEXT                   .Wait for release completion D4DEM.426
2055    <macro>                      $ELSEIF (R!FLG = DK$READ)    .If reading            D4DEM.427
                                                                                          D4DEM.428
                                    *  Transfer next read ahead sector.                   D4DEM.429
                                                                                          D4DEM.430
2062                        READAH  *                                                     D4DEM.431
2062   020460  106514                  P = NEXT, R!WBC = 0        .If none               D4DEM.432
2064   020512  023000  107517          P = NEXT, R!TMP # R!%MYID .If path not in this IOP D4DEM.433
2067   077000 /004562                  R = READMCB                .Read in MCB           D4DEM.434
2071   020463  023465  102407          P = REQUEUE, R!BPTR = R!DPTR .If data not in MOS yet D4DEM.435
2074   020463  024423                  R!%W1 = R!BPTR             .Offset to buffer      D4DEM.436
2076   077000 /004716                  R = GETMOS                 .Get MOS buffer address D4DEM.437
2100    <macro>                        GET R!SCO,DL@RAT,R!DAL     .Total R.A.'s for request D4DEM.438
```

```
2106   020446  023460  024447          R!SC1 = R!SC0 - R!WBC      .Next sector to do                D4DEM.439
2111   010001  024470                  R!CODE = FS$OUT            .Direction                        D4DEM.440
2113   077000 /005120                  R = MOVEIT                 .Transfer read ahead              D4DEM.441
2115    <macro>                        $IF (R!TM = TM$CMEM)       .If target is Central memory      D4DEM.442
                                                                                     .>>>>> 2          $IF      1
2120   020000  024424                    R!%W2 = R!%HISPBUF       .Transfer buffer                  D4DEM.443
2122    <macro>                        $ELSE                                                        D4DEM.444
2123   020000  024424                    R!%W2 = R!%SSDBUF        .SSD transfer buffer              D4DEM.445
2125    <macro>                        $ENDIF                                                       D4DEM.446
                                                                                     .<<<<< 2          $ENDIF   1
2125    <macro>                        $IF ((BNFLG) # 0),AND,(R!%W2 # 0) .If enabled                D4DEM.447
                                                                                     .>>>>> 2          $IF      1
2136    <macro>                          GET R!%W1,DL@BNC,R!DAL   .Block check flag and number      D4DEM.448
2143    <macro>                          $IF (0 # A > D'15)       .If requested                     D4DEM.449
                                                                                     .>>>>> 3          $IF      1
2151   020446  023460  024425              R!%W3 = R!SC0 - R!WBC .Sector offset                    D4DEM.450
2154   077000 /010453                      R = BLCK1              .Check block                      D4DEM.451
2156    <macro>                          $ENDIF                                                     D4DEM.452
                                                                                     .<<<<< 3          $ENDIF   1
2156    <macro>                        $ENDIF                                                       D4DEM.453
                                                                                     .<<<<< 2          $ENDIF   1
2156   077000 /010061                  R = ACTDEM                 .Put DCB back on queue            D4DEM.454
2160    <macro>                        FLDSUB 1,DL@RAC,R!DAL      .Decrement R.A. count             D4DEM.455
2203   102002  001000                  EXIT, A # 0                .If more to do                    D4DEM.456
2205   002000  000000                  I = 0                                                        D4DEM.457
2207    <macro>                        GET R!%W1,DK@EWB,R!DCB     .Request done signal              D4DEM.458
2216    <macro>                        $IF (R!SC0 = R!TOTAL),OR,(R!%W1 # 0) .Request done           D4DEM.459
                                                                                     .>>>>> 2          $IF      1
2224    <macro>                          STORE 0,DK@EWB,R!DCB     .Clear flag                       D4DEM.460
2233    <macro>                          ADDRESS R!ED,DK@DLO,R!DCB .DAL queue                       D4DEM.461
2236    <macro>                          RKERN EDQU               .Take off                         D4DEM.462
2240   077000 /004501                    R = STAT                 .Send status                      D4DEM.463
2242    <macro>                          RELDAL R!DAL             .Release DAL                      D4DEM.464
2252    <macro>                        $ENDIF                                                       D4DEM.465
                                                                                     .<<<<< 2          $ENDIF   1
2252    <macro>                      $ELSE                                                          D4DEM.466
                                                                                                    D4DEM.467
                              *      Facilitate chaining of write requests, if possible.            D4DEM.468
                                                                                                    D4DEM.469
2253                          WRITEBH *                                                             D4DEM.470
2253   010000  033433  102056         P = WRITEBH1, 0 = (R!DAL) .If only one DAL                   D4DEM.471
2256   077000 /006732                 R = FILL                   .Fill local buffers               D4DEM.472
```

```
2260   020433  106712                    P = NEXT, R!D     0        .If none to check                    D4DEM.47
2262   020451  011001  024410            P = NEXT, 0 # R!FNC & 1    .If read                          R16410FA.1
       010000  023410  107720
2270    <macro>                          $UNTIL (R!TOTAL > R!SEQ + R!WBC) .Locate W.B. candidate       D4DEM.474
2300   030433  024433                      R!DAL = (R!DAL)          .Next DAL                          D4DEM.475
2302   106733                              P = NEXT, A = 0          .If no more                        D4DEM.476
2303   077000 /007274                      R = DALPAR               .Get its parameters                D4DEM.477
2305    <macro>                          $ENDTIL                                                       D4DEM.478
2306    <macro>                          $IF (0 = R!FNC & 1)        .If write request                  D4DEM.479
                                                                                   .>>>>> 2                $IF      1
2315    <macro>                             GET R!%W1,DK@DLO,R!DCB   .First DAL on queue              R164100A.1
2322    <macro>                             GET A,DL@TM,R!%W1        .Target memory type              R164100A.2
2327   023511  107761                      P = NEXT, A # R!TM       .If current is different         R164100A.3
2331   077000 /007605                      R = SENDIR               .Send DTR if needed                D4DEM.480
                                                                                                         D4DEM.481
                                 *       Transfer next write behind sector.                              D4DEM.482
                                                                                                         D4DEM.483
2333                             WRITEBH1 *                                                              D4DEM.484
2333    <macro>                          $IF (R!FNC # DAF$WRBH),AND,(R!FNC # DAF$WBFR)                 D4DEM.485
                                                                                   .>>>>> 3                $IF      1
2343   075000 /001347                      P = NEXT                 .No write behind                   D4DEM.486
2345    <macro>                          $ENDIF                                                        D4DEM.487
                                                                                   .<<<<< 3                $ENDIF   1
2345   020460  022453  024410            P = NEXT, R!TOTAL <= R!WBC + R!SEQ .If completed             D4DEM.488
       020450  023410  100002
       103003  075000 /001347
2356    <macro>                          $IF (R!TMP # R!%MYID),OR,(R!TM = TM$BMR) .If no path      R16410DA.1
                                                                                   .>>>>> 3                $IF      1
2365   075000 /001347                      P = NEXT                 .No write behind                 R16410DA.2
2367    <macro>                          $ENDIF                                                      R16410DA.3
                                                                                   .<<<<< 3                $ENDIF   1
2367    <macro>                             GET R!%W1,DK@WBD,R!DCB   .# sectors in MOS                 D4DEM.490
2375    <macro>                             GET A,DK@WCC,R!DCB       .Write behind limit               D4DEM.491
2402   023423  126000 /001347              P = NEXT, A = R!%W1      .If at limit                       D4DEM.492
2405   002000  000000                      1 = 0                                                       D4DEM.493
2407   077000 /003563                      R = QCHECK               .Check for work on MEMIO queues    D4DEM.494
2411   103071                              P = REQUEUE, A # 0       .If MEMIO needed                   D4DEM.495
2412   077000 /004562                      R = READMCB              .Read in MCB                       D4DEM.496
2414   020463  024423                      R!%W1 = R!BPTR           .Offset to buffer                  D4DEM.497
2416   077000 /004716                      R = GETMOS               .Get MOS buffer address            D4DEM.498
2420   026460                              R!WBC = R!WBC + 1                                           D4DEM.499
2421   020450  023460  024447              R!SC1 = R!TOTAL - R!WBC .Next sector to do                  D4DEM.500
```

```
2424    <macro>                          PUT R!WBC,DL@RAC,R!DAL   .Increment                          D4DEM.501
2437    <macro>                          FLDADD 1,DK@WBD,R!DCB    .Increment                          D4DEM.502
2462   010000  024470                    R!CODE = FS$IN           .Direction                          D4DEM.503
2464   077000 /005120                    R = MOVEIT               .Transfer write behind              D4DEM.504
2466    <macro>                          $IF (R!TOTAL = R!WBC + R!SEQ) .If complete                   D4DEM.505
                                                                                  .>>>>> 3                $IF      1
2475   077000 /004415                      R = WBSTAT             .Send write behind status           D4DEM.506
2477   030433  103002                      P = REQUEUE, (R!DAL) # 0 .If more to do                    D4DEM.507
2501    <macro>                          $ELSE                                                        D4DEM.508
2502                          REQUEUE  *                                                              D4DEM.509
2502   077000 /010061                      R = ACTDEM             .Queue for next W.B.                D4DEM.510
2504    <macro>                          $ENDIF                                                       D4DEM.511
                                                                                  .<<<<< 3                $ENDIF   1
2504    <macro>                        $ENDIF                                                         D4DEM.512
                                                                                  .<<<<< 2                $ENDIF   1
2504    <macro>                      $ENDIF                                                           D4DEM.513
                                                                                  .<<<<< 1                $ENDIF   1
                                                                                                       D4DEM.514
                              *        Release buffers and channel.                                    D4DEM.515
                                                                                                       D4DEM.516
2504    <macro>                        $ELSEIF (R!FLG = 0)      .If channel idle                    D4DEM.517
2510   077000 /005454                    R = RELBUFS              .Release local buffers              D4DEM.518
2512   020513  137000 /005523            R = RELEASE, R!REL # 0   .If time to release the channel  R13739AP.7
2515    <macro>                        $ENDIF                                                       D4DEM.538
                                                                                  .<<<<< 0                $ENDIF   1
2515   075000 /001347                   P = NEXT                 .Next DCB                           D4DEM.539
                                                                                                       D4DEM.540
                              *        Slow down processing of requests if some                        D4DEM.541
                              *        resource is unavailable.                                        D4DEM.542
                                                                                                       D4DEM.543
2517                          WHOA     *                                                               D4DEM.544
2517   077000 /010061                   R = ACTDEM               .Put DCB on queue                   D4DEM.545
2521   002000  000000                   I = 0                                                        D4DEM.546
2523    <macro>                         GET A,DCB@QU,R!%DCBQ     .First on queue                     D4DEM.547
2524    <macro>                         $IF (A = R!DCB)          .If only one                        D4DEM.548
                                                                                  .>>>>> 0                $IF      1
2527    <macro>                          PAUSE 1,A1=0,A2=-1      .Pause 1/10 sec.                    D4DEM.549
2537   075000 /000006                    P = START                .Start over                         D4DEM.550
2541    <macro>                         $ENDIF                                                       D4DEM.551
                                                                                  .<<<<< 0                $ENDIF   1
2541   075000 /001347                   P = NEXT                 .Next request                       D4DEM.552
```

```
                                 *************************  ****************************************   D4DEM.55
                                 *                                                                *   D4DEM.555
                                 *        MEMIO     This section moves disk data between Local Memory   *   D4DEM.556
                                 *                  and either Central Memory or Buffer Memory.         *   D4DEM.557
                                 *                                                                *   D4DEM.558
                                 ******************************************************************   D4DEM.559
                                                                                                       D4DEM.560
2543                             MEMIO    *                                                            D4DEM.561
2543   010000  024430                     R!NUMDONE = 0                  .# of consecutive requests done   D4DEM.562
2545   024501                             R!HIFLG = A                    .High speed in queue flag         D4DEM.563
2546   024502                             R!HOFLG = A                    .High speed out queue flag        D4DEM.564
2547   024503                             R!SIFLG = A                    .SSD in queue flag                D4DEM.565
2550   024504                             R!SOFLG = A                    .SSD out queue flag               D4DEM.566
2551   024505                             R!MSFLG = A                    .MOS queue flag                   D4DEM.567
2552                             AGAIN    *                                                            D4DEM.568
2552   003000                             I = 1                                                        D4DEM.569
                                                                                                       D4DEM.570
                                 *        SSD In queue processing.                                     D4DEM.571
                                                                                                       D4DEM.572
2553                             SSDIN    *                                                            D4DEM.573
2553    <macro>                          $IF (R!SIFLG # 0)              .If I/O outstanding               D4DEM.574
                                                                                          .>>>>> 0          $IF      1
2556   010000  024503                       R!SIFLG = 0                  .Clear flag                       D4DEM.575
2560   020000  054000                       B = R!%SSD                   .SSD channel                      D4DEM.576
2562    <macro>                             ADDRESS R!QUE,MEM@SI,R!%MEMIO .Queue address                  D4DEM.577
2565   077000 /003511                       R = INHSCH                   .Complete input processing        D4DEM.578
2567    <macro>                          $ENDIF                                                          D4DEM.579
                                                                                          .<<<<< 0          $ENDIF   1
2567   020430  013777  101026           P = SSDOUT, R!NUMDONE >= MEM$LIM .If time to give up             D4DEM.580
2572    <macro>                         GET R!LOCAL,MEM@SI,R!%MEMIO    .Get next entry to process        D4DEM.581
2577    <macro>                         $IF (A # 0)                    .If something on queue            D4DEM.582
                                                                                          .>>>>> 0          $IF      1
2601   077000 /007405                       R = LBPAR                    .Local Buffer parameters          D4DEM.583
2603    <macro>                             $IF (R!FLG = 0)              .If I/O not initiated             D4DEM.584
                                                                                          .>>>>> 1          $IF      1
2606   077000 /010541                         R = GETCPU                 .Get CPU address                  D4DEM.585
2610   010012  024451                         R!FNC = MIO$SSII           .SSD in, immediate return         D4DEM.586
2612   077000 /010515                         R = DATAMOVE               .Start the I/O                    D4DEM.587
2614    <macro>                             $ENDIF                                                      D4DEM.588
                                                                                          .<<<<< 1          $ENDIF   1
2614   010001  024503                       R!SIFLG = 1                  .Queue active flag                D4DEM.589
2616   026430                               R!NUMDONE = R!NUMDONE + 1                                  D4DEM.590
```

```
2617    <macro>                         $ENDIF                                                          D4DEM.591
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                        D4DEM.592
                                *       SSD Out queue processing.                                       D4DEM.593
                                                                                                        D4DEM.594
2617                            SSDOUT  *                                                               D4DEM.595
2617    <macro>                         $IF (R!SOFLG # 0)               .If I/O outstanding             D4DEM.596
                                                                                .>>>>> 0                $IF       1
2622    010000  024504                    R!SOFLG = 0                   .Clear flag                     D4DEM.597
2624    020000  012001  054000            B = R!%SSD + 1                .SSD out channel                D4DEM.598
2627    <macro>                           ADDRESS R!QUE,MEM@SO,R!%MEMIO .Queue address                  D4DEM.599
2632    077000 /010071                    R = CHANCHK                   .Wait for completion            D4DEM.600
2634    077000 /010553                    R = RDDONE                    .Check for request completion   D4DEM.601
2636    <macro>                         $ENDIF                                                          D4DEM.602
                                                                                .<<<<< 0                $ENDIF    1
2636                            SSDONXT *                                                               D4DEM.603
2636    020430  013777  101042          P = HSPIN, R!NUMDONE >= MEM$LIM .If time to give up             D4DEM.604
2641    <macro>                         GET R!LOCAL,MEM@SO,R!%MEMIO   .Get next entry to process        D4DEM.605
2646    <macro>                         $IF (A # 0)                   .If something on queue            D4DEM.606
                                                                                .>>>>> 0                $IF       1
2650    077000 /007405                    R = LBPAR                     .Local Buffer parameters        D4DEM.607
2652    <macro>                           GET A,DK@DLO,R!DCB            .First DAL on queue             D4DEM.608
2656    <macro>                           $IF (A # R!DAL)               .If this not first              D4DEM.609
                                                                                .>>>>> 1                $IF       1
2661    010777  024430                      R!NUMDONE = MEM$LIM         .Set process limit              D4DEM.610
2663    070017                              P = HSPIN                   .Let R.A. transfers continue    D4DEM.611
2664    <macro>                           $ENDIF                                                        D4DEM.612
                                                                                .<<<<< 1                $ENDIF    1
2664    077000 /010422                    R = BLCK                      .Block number check             D4DEM.613
2666    <macro>                           $IF (R!FLG = 0)               .If I/O not initiated           D4DEM.614
                                                                                .>>>>> 1                $IF       1
2671    077000 /010541                      R = GETCPU                  .Get CPU address                D4DEM.615
2673    010014  024451                      R!FNC = MIO$SSOI            .SSD out, immediate return      D4DEM.616
2675    077000 /010515                      R = DATAMOVE                .Start the I/O                  D4DEM.617
2677    <macro>                           $ENDIF                                                        D4DEM.618
                                                                                .<<<<< 1                $ENDIF    1
2677    010001  024504                    R!SOFLG = 1                   .Queue active flag              D4DEM.619
2701    026430                            R!NUMDONE = R!NUMDONE + 1     .Increment number requests done D4DEM.620
2702    <macro>                         $ENDIF                                                          D4DEM.621
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                        D4DEM.622
                                *       High Speed In queue processing.                                 D4DEM.623
```

```
                                                                                                D4DEM.62
2702                          HSPIN    *                                                        D4DEM.625
2702   <macro>                         $IF (R!HIFLG # 0)              .If I/O outstanding       D4DEM.626
                                                                                .>>>>> 0          $IF      1
2705   010000  024501                    R!HIFLG = 0                   .Clear flag               D4DEM.627
2707   020000  054000                    B = R!%HISP                   .High speed in channel    D4DEM.628
2711   <macro>                           ADDRESS R!QUE,MEM@HI,R!%MEMIO .Queue address            D4DEM.629
2714   077000 /003511                    R = INHSCH                    .Complete input processing D4DEM.630
2716   <macro>                         $ENDIF                                                    D4DEM.631
                                                                                .<<<<< 0          $ENDIF   1
2716   020430  013777  101023          P = HSPOUT, R!NUMDONE >= MEM$LIM .If time to give up      D4DEM.632
2721   <macro>                         GET R!LOCAL,MEM@HI,R!%MEMIO   .Get next entry             D4DEM.633
2723   <macro>                         $IF (A # 0)                   .If something on queue      D4DEM.634
                                                                                .>>>>> 0          $IF      1
2725   077000 /007405                    R = LBPAR                                               D4DEM.635
2727   <macro>                           $IF (R!FLG = 0)               .If I/O not initiated     D4DEM.636
                                                                                .>>>>> 1          $IF      1
2732   077000 /010541                      R = GETCPU                  .Get CPU address          D4DEM.637
2734   010006  024451                      R!FNC = MIO$HSII            .HISP in, immediate return D4DEM.638
2736   077000 /010515                      R = DATAMOVE                .Start the I/O            D4DEM.639
2740   <macro>                           $ENDIF                                                  D4DEM.640
                                                                                .<<<<< 1          $ENDIF   1
2740   010001  024501                    R!HIFLG = 1                   .Queue active flag        D4DEM.641
2742   026430                            R!NUMDONE = R!NUMDONE + 1                               D4DEM.642
2743   <macro>                         $ENDIF                                                    D4DEM.643
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                  D4DEM.644
                              *         High Speed Out queue processing.                          D4DEM.645
                                                                                                  D4DEM.646
2743                          HSPOUT   *                                                        D4DEM.647
2743   <macro>                         $IF (R!HOFLG # 0)              .If I/O outstanding       D4DEM.648
                                                                                .>>>>> 0          $IF      1
2746   010000  024502                    R!HOFLG = 0                   .Clear flag               D4DEM.649
2750   020000  012001  054000            B = R!%HISP + 1               .High speed out channel   D4DEM.650
2753   <macro>                           ADDRESS R!QUE,MEM@HO,R!%MEMIO .Queue address            D4DEM.651
2756   077000 /010071                    R = CHANCHK                   .Wait for completion      D4DEM.652
2760   077000 /010553                    R = RDDONE                    .Check for request completion D4DEM.653
2762   <macro>                         $ENDIF                                                    D4DEM.654
                                                                                .<<<<< 0          $ENDIF   1
2762                          HSPONXT  *                                                        D4DEM.655
2762   020430  013777  125000          P = MOSIO, R!NUMDONE >= MEM$LIM .If time to give up       D4DEM.656
       /003041
```

```
2756    <macro>                 GET R!LOCAL,MEM@HO,R!%MEMIO   .Get next entry                    D4DEM.657
2773    <macro>                 $IF (A # 0)                   .If something on queue             D4DEM.658
                                                                                .>>>>> 0      $IF      1
2776   077000 /007405             R = LBPAR                                                       D4DEM.659
3000    <macro>                   GET A,DL@TOT,R!DAL          .Total sectors                     D4DEM.660
3004    <macro>                   $IF (A = R!SEQ)             .If last sector                    D4DEM.661
                                                                                .>>>>> 1      $IF      1
3011    <macro>                     GET A,DK@DLO,R!DCB        .First DAL on queue                D4DEM.662
3015    <macro>                     $IF (A # R!DAL)           .If this not first                 D4DEM.663
                                                                                .>>>>> 2      $IF      1
3020   010777  024430                 R!NUMDONE = MEM$LIM     .Set process limit                 D4DEM.664
3022   070017                         P = MOSIO               .Let R.A. transfers continue       D4DEM.665
3023    <macro>                     $ENDIF                                                        D4DEM.666
                                                                                .<<<<< 2      $ENDIF   1
3023    <macro>                   $ENDIF                                                          D4DEM.667
                                                                                .<<<<< 1      $ENDIF   1
3023   077000 /010422             R = BLCK                    .Block number check                D4DEM.668
3025    <macro>                   $IF (R!FLG = 0)             .If I/O not initiated              D4DEM.669
                                                                                .>>>>> 1      $IF      1
3030   077000 /010541               R = GETCPU                .Get CPU address                   D4DEM.670
3032   010010  024451               R!FNC = MIO$HSOI          .HISP out, immediate return        D4DEM.671
3034   077000 /010515               R = DATAMOVE              .Start the I/O                     D4DEM.672
3036    <macro>                   $ENDIF                                                          D4DEM.673
                                                                                .<<<<< 1      $ENDIF   1
3036   010001  024502             R!HOFLG = 1                 .Queue active flag                 D4DEM.674
3040   026430                     R!NUMDONE = R!NUMDONE + 1                                      D4DEM.675
3041    <macro>                 $ENDIF                                                            D4DEM.676
                                                                                .<<<<< 0      $ENDIF   1
                                                                                                  D4DEM.677
                        *       MOS queue processing.                                             D4DEM.678
                                                                                                  D4DEM.679
3041                    MOSIO   *                                                                 D4DEM.680
3041    <macro>                 $IF (R!MSFLG # 0)           .If I/O outstanding                D4DEM.681
                                                                                .>>>>> 0      $IF      1
3044   010000  024505             R!MSFLG = 0                 .Clear flag                        D4DEM.682
3046   010005  054000             B = [MOS]                   .Channel number                    D4DEM.683
3050    <macro>                   ADDRESS R!QUE,MEM@BM,R!%MEMIO .Queue address                   D4DEM.684
3053   077000 /010071             R = CHANCHK                 .Wait for completion               D4DEM.685
3055    <macro>                   $IF (0 = R!TDIR & 0'20)     .Not internal                      D4DEM.686
                                                                                .>>>>> 1      $IF      1
3064    <macro>                     $IF (0 = R!TDIR & 1)      .If disk write                     D4DEM.687
                                                                                .>>>>> 2      $IF      1
```

```
3075   077000 /010422                    R = BLCK                  .Check block number                D4DEM.68.
3075    <macro>                          $IF (R!TM = TM$BMR)       .If write from BMR               R16410DA.4
                                                                                        .>>>>> 3          $IF      1
3101   072414                              R = INHS                .Process HISP I/O done           R16410DA.5
3102   070102                              P = MOSDN                                                R16410DA.6
3103    <macro>                          $ENDIF                                                    R16410DA.7
                                                                                        .<<<<< 3        $ENDIF    1
3103    <macro>                        $ELSEIF (R!TM = TM$BMR)     .Disk read to BMR                D4DEM.689
3110   077000 /010553                    R = RDDONE                .Check for done                  D4DEM.690
3112    <macro>                        $ENDIF                                                        D4DEM.691
                                                                                        .<<<<< 2        $ENDIF    1
3112   020441  103102                  P = MOSNXT, R!FLG # 0       .If MCB alread updated           D4DEM.692
3114    <macro>                        $IF (0 # R!TDIR & 1),OR,(R!TMP # R!%MYID) .Read, no path       D4DEM.693
                                                                                        .>>>>> 2          $IF      1
3126   010001  024423                    R!%W1 = 1                 .Increment MCB pointer by 1      D4DEM.694
3130    <macro>                        $ELSE                                                         D4DEM.695
3131    <macro>                          GET A,DL@TOT,R!DAL        .Total for request               D4DEM.696
3135   041000  023453  100003            P = MOSDN, A > R!SEQ      .If not complete                 D4DEM.697
       102002  070043
3142    <macro>                          GET R!%W1,DL@RAC,R!DAL    .Write behind count              D4DEM.698
3150    <macro>                          FLDSUB R!%W1,DK@WBD,R!DCB .Decrement DCB count             D4DEM.699
3173   077000 /010061                    R = ACTDEM                .Put DCB on queue                D4DEM.700
3175    <macro>                        $ENDIF                                                        D4DEM.701
                                                                                        .<<<<< 2        $ENDIF    1
3175   077000 /005021                  R = WRITDIOP                .Update DIOP part of MCB         D4DEM.702
3177    <macro>                      $ELSE                         .Internal I/O                    D4DEM.703
3200   020500  013021  136000            R = RDDONE, R!TDIR = DAF$INTR .MDAL is done                D4DEM.704
      /010553
3204    <macro>                      $ENDIF                                                          D4DEM.705
                                                                                        .<<<<< 1        $ENDIF    1
3204                          MOSDN  *                                                              D4DEM.706
3204    <macro>                      STORE 0,DK@LCK,R!DCB        .Unlock the MCB                  D4DEM.707
3215    <macro>                    $ENDIF                                                            D4DEM.708
                                                                                        .<<<<< 0        $ENDIF    1
3215                          MOSNXT *                                                              D4DEM.709
3215    <macro>                    $IF (R!NUMDONE >= MEM$LIM)      .If time to give up              D4DEM.710
                                                                                        .>>>>> 0          $IF      1
3221   020501  022502  022505        A = R!HIFLG + R!HOFLG + R!MSFLG + R!SIFLG + R!SOFLG             D4DEM.711
       022503  022504
3226    <macro>                      $IF (A = 0)                 .If no outstanding I/O           D4DEM.712
                                                                                        .>>>>> 1          $IF      1
3230   010001  024477                  R!GIVE = 1                                                    D4DEM.713
```

```
3232   001000                          EXIT                                                          D4DEM.714
3233    <macro>                       $ELSE                        .Clean up                        D4DEM.715
3234   071462                          P = AGAIN                                                     D4DEM.716
3235    <macro>                       $ENDIF                                                         D4DEM.717
                                                                                   .<<<<< 1              $ENDIF    1
3235    <macro>                     $ENDIF                                                           D4DEM.718
                                                                                   .<<<<< 0              $ENDIF    1
3235    <macro>                     GET R!LOCAL,MEM@BM,R!%MEMIO   .Get next entry                    D4DEM.719
3242    <macro>                     $IF (A # 0)                   .If something there                D4DEM.720
                                                                                   .>>>>> 0              $IF       1
3244   026430                         R!NUMDONE = R!NUMDONE + 1   .Increment number requests done     D4DEM.721
3245   077000 /007405                 R = LBPAR                   .Get local buffer params           D4DEM.722
3247    <macro>                       $IF (R!DAL # 0)             .If not read ahead                 D4DEM.723
                                                                                   .>>>>> 1              $IF       1
3252    <macro>                         GET R!TDIR,DL@FC,R!DAL    .Transfer direction                D4DEM.724
3260    <macro>                       $ELSE                                                          D4DEM.725
3261   010001  024500                   R!TDIR = DAF$READ         .Read ahead                        D4DEM.726
3263    <macro>                       $ENDIF                                                         D4DEM.727
                                                                                   .<<<<< 1              $ENDIF    1
3263    <macro>                       $IF (R!TM = TM$BMR)         .If target is BMR                  D4DEM.728
                                                                                   .>>>>> 1              $IF       1
3267   020500  011001  024410           R = BLCK, 0 # R!TDIR & 1  .Check block on read               D4DEM.729
       010000  023410  137000
      /010422
3276    <macro>                         $IF (R!FLG = 0)           .If I/O not initiated              D4DEM.730
                                                                                   .>>>>> 2              $IF       1
3301   010002  024451                     R!FNC = MIO$BMRI         .Must be disk write                D4DEM.731
3303   077000 /010541                     R = GETCPU               .Get MOS addr for BMR              D4DEM.732
3305    <macro>                         $ENDIF                                                         D4DEM.733
                                                                                   .<<<<< 2              $ENDIF    1
3305    <macro>                         $ELSEIF (0 = R!TDIR & O'20) .If not internal I/O             D4DEM.734
3315   077000 /004562                   R = READMCB               .Read in MCB from MOS              D4DEM.735
3317   106545                           P = AGAIN, A = 0          .If no local memory                D4DEM.736
3320    <macro>                         $IF (0 = R!TDIR & 1)      .If write to disk                  D4DEM.737
                                                                                   .>>>>> 2              $IF       1
3327    <macro>                           $IF (R!BPTR = R!DPTR),AND,(R!FLG = 0) .Data not in MOS     D4DEM.738
                                                                                   .>>>>> 3              $IF       1
3336   071564                             P = AGAIN                                                  D4DEM.739
3337    <macro>                           $ENDIF                                                     D4DEM.740
                                                                                   .<<<<< 3              $ENDIF    1
3337   010002  024451                     R!FNC = MIO$BMRI         .MOS read, immediate return        D4DEM.741
3341    <macro>                           $IF (R!TMP = R!%MYID)    .If write behind                   D4DEM.742
```

```
                                                                  .>>>>> 3          $IF      1
3345   <macro>                    GET A,DL@TOT,R!DAL     .Total for request          D4DEM.743
3351   023453                     A = A - R!SEQ          .Offset into MCB            D4DEM.744
3352   025465                     R!DPTR = A + R!DPTR    .Add to current pointer     D4DEM.745
3353   041000  023510  100004     R!DPTR = R!DPTR - R!NUM, A >= R!NUM .Wrap          D4DEM.746
       020465  023510  024465
3361   <macro>                  $ENDIF                                               D4DEM.747
                                                                  .<<<<< 3          $ENDIF   1
3361   <macro>                  $ELSE                  .If read from disk            D4DEM.748
3362   <macro>                    $IF (R!FLG = 0)      .If I/O not started           D4DEM.749
                                                                  .>>>>> 3          $IF      1
3365   <macro>                      $IF (R!DPTR = R!BPTR - 1) .If table full         D4DEM.750
                                                                  .>>>>> 4          $IF      1
3374   070073                         P = MCBFULL        .Handle                     D4DEM.751
3375   <macro>                      $ELSEIF (R!DPTR = R!NUM - 1),AND,(R!BPTR = 0)    D4DEM.752
3410   070057                         P = MCBFULL        .Handle full table          D4DEM.753
3411   <macro>                      $ENDIF                                           D4DEM.754
                                                                  .<<<<< 4          $ENDIF   1
3411   <macro>                    $ENDIF                                             D4DEM.755
                                                                  .<<<<< 3          $ENDIF   1
3411   020433  137000 /010422     R = BLCK, R!DAL # 0    .Block number check         D4DEM.756
3414   010004  024451             R!FNC = MIO$BMWI       .MOS write, immediate return D4DEM.757
3416   <macro>                  $ENDIF                                               D4DEM.758
                                                                  .<<<<< 2          $ENDIF   1
3416   020465  024423           R!%W1 = R!DPTR           .Offset to buffer           D4DEM.759
3420   077000 /004716           R = GETMOS               .Get MOS buffer address     D4DEM.760
3422   <macro>                $ELSE                                                  D4DEM.761
3423   010004  024451           R!FNC = MIO$BMWI         .MOS write                  D4DEM.762
3425   020500  013022  103003   R!FNC = MIO$BMRI, R!TDIR = DAF$INTW .MOS read        D4DEM.763
       010002  024451
3432   <macro>                  GET R!BUFU,DL@BM0,R!DAL  .MOS address (upper)        D4DEM.764
3437   <macro>                  GET R!BUFL,DL@BM1,R!DAL  .(lower)                    D4DEM.765
3444   <macro>                $ENDIF                                                 D4DEM.766
                                                                  .<<<<< 1          $ENDIF   1
3444   <macro>                $IF (R!FLG = 0)            .If I/O not initiated       D4DEM.767
                                                                  .>>>>> 1          $IF      1
3447   077000 /010515           R = DATAMOVE             .Start the I/O              D4DEM.768
3451   <macro>                $ENDIF                                                 D4DEM.769
                                                                  .<<<<< 1          $ENDIF   1
3451   010001  024505         R!MSFLG = 1                .Queue active flag          D4DEM.770
3453   <macro>              $ELSEIF (R!MCB # 0)          .If memory allocated        D4DEM.771
3457   077000 /005104         R = RELMEM                 .Release it                 D4DEM.772
```

```
3461    <macro>                         $ENDIF                                                          D4DEM.773
                                                                                  .<<<<< 0                $ENDIF    1
                                                                                                          D4DEM.774
                                * Check for more work to do.                                              D4DEM.775
                                                                                                          D4DEM.776
3461    002000  000000                  I = 0                                                             D4DEM.777
3463    072100                          R = QCHECK                .Check queues for more work             D4DEM.778
3464    107712                          P = AGAIN, A # 0          .If more to do                          D4DEM.779
3465    003000                          I = 1                                                             D4DEM.780
3466    001000                          EXIT                                                              D4DEM.781
                                                                                                          D4DEM.782
                                * Handle MCB full situation.                                              D4DEM.783
                                                                                                          D4DEM.784
3467                            MCBFULL *                                                                 D4DEM.785
3467    002000  000000                  I = 0                                                             D4DEM.786
3471    <macro>                         GET A,DKF@RA,R!DCB        .Read ahead flag                        D4DEM.787
3474    <macro>                         $IF (A # 0)               .If active now                          D4DEM.788
                                                                                  .>>>>> 0                $IF       1
3476    <macro>                           STORE 1,DK@ABT,R!DCB    .Set abort flag                         D4DEM.789
3507    <macro>                         $ENDIF                                                            D4DEM.790
                                                                                  .<<<<< 0                $ENDIF    1
3507    075000 /002552                  P = AGAIN                                                         D4DEM.791
```

```
                                 *************************************************************   D4DEM.79
                                 *        SUBROUTINES                                        *   D4DEM.794
                                 *************************************************************   D4DEM.795
                                                                                                  D4DEM.796
                                 *************************************************************   D4DEM.797
                                 *                                                           *   D4DEM.798
                                 *        INHSCH :  Waits for a DN on an HSP or SSD channel.  Upon  *   D4DEM.799
                                 *                  completion of I/O, completes processing the queue  *   D4DEM.800
                                 *                  and activates appropriate demon.         *   D4DEM.801
                                 *                                                           *   D4DEM.802
                                 *        Entry:    B = Channel number                       *   D4DEM.803
                                 *                  R!QUE = Queue address for LMA entry      *   D4DEM.804
                                 *                                                           *   D4DEM.805
                                 *************************************************************   D4DEM.806
                                                                                                  D4DEM.807
3511                             INHSCH   *                                                       D4DEM.808
3511   077000 /010071                     R = CHANCHK                  .Wait for completion      D4DEM.809
3513   077000 /010422                     R = BLCK                     .Block number check       D4DEM.810
3515                             INHS     *                                                   R16410DA.8
3515    <macro>                           GET R!%W1,DL@FC,R!DAL        .Function code            D4DEM.811
3523    <macro>                           $IF (R!%W1 = DAF$WRBH),OR,(R!%W1 = DAF$WBFR) .W.B. request  D4DEM.812
                                                                                  .>>>>> 0          $IF      1
3532    <macro>                             GET R!WBC,DL@RAC,R!DAL     .Write behind count       D4DEM.813
3540    <macro>                             GET R!TOTAL,DL@TOT,R!DAL   .Total for request        D4DEM.814
3545    <macro>                             $IF (R!TOTAL = R!SEQ + R!WBC) .If time to send status  D4DEM.815
                                                                                  .>>>>> 1          $IF      1
3554   077000 /007471                       R = LIMITS                 .Get device limits        D4DEM.816
3556   077000 /004415                       R = WBSTAT                 .Send write behind status D4DEM.817
3560    <macro>                             $ENDIF                                               D4DEM.818
                                                                                  .<<<<< 1          $ENDIF   1
3560    <macro>                           $ENDIF                                                 D4DEM.819
                                                                                  .<<<<< 0          $ENDIF   1
3560   077000 /010061                     R = ACTDEM                   .Activate demon           D4DEM.820
3562   001000                             EXIT                                                   D4DEM.821
                                                                                                  D4DEM.822
                                 *************************************************************   D4DEM.823
                                 *                                                           *   D4DEM.824
                                 *        QCHECK:   Check for work to do on MEMIO queues.    *   D4DEM.825
                                 *                                                           *   D4DEM.826
                                 *        Exit:     A # 0 if something to do                 *   D4DEM.827
                                 *                                                           *   D4DEM.828
                                 *************************************************************   D4DEM.829
```

```
                                                                                D4DEM.830
3563                    QCHECK  *                                               D4DEM.831
3563     <macro>                GET A,MEM@HI,R!%MEMIO        .HISP in queue     D4DEM.832
3564    102002  001000          EXIT, A # 0                  .If something there D4DEM.833
3566     <macro>                GET A,MEM@HO,R!%MEMIO        .HISP out queue    D4DEM.834
3572    102002  001000          EXIT, A # 0                  .If something there D4DEM.835
3574     <macro>                GET A,MEM@SI,R!%MEMIO        .SSD in queue      D4DEM.836
3600    102002  001000          EXIT, A # 0                                     D4DEM.837
3602     <macro>                GET A,MEM@SO,R!%MEMIO        .SSD out queue     D4DEM.838
3606    102002  001000          EXIT, A # 0                                     D4DEM.839
3610     <macro>                GET A,MEM@BM,R!%MEMIO        .MOS queue         D4DEM.840
3614    001000                  EXIT                                            D4DEM.841
                                                                                D4DEM.842
                        ***********************************************************D4DEM.843
                        *                                                       *D4DEM.844
                        *       INITA:    Initiate I/O for a new MDAL request.  *D4DEM.845
                        *                                                       *D4DEM.846
                        *       Entry:    R!DAL   = MDAL address                *D4DEM.847
                        *                 R!DCB   = DCB address                 *D4DEM.848
                        *                 R!UNIT = Unit number                  *D4DEM.849
                        *                 R!FNC = Read/write code               *D4DEM.850
                        *                 R!CYL = Desired cylinder              *D4DEM.851
                        *                 R!HED = Desired head group            *D4DEM.852
                        *                 R!SEC = Desired sector                *D4DEM.853
                        *                 R!TM  = Target memory                 *D4DEM.854
                        *                 R!TMP = IOP with path to target memory *D4DEM.855
                        *                                                       *D4DEM.856
                        ***********************************************************D4DEM.857
                                                                                D4DEM.858
3615                    INITA   *                                               D4DEM.859
3615     <macro>                PUT R!DIA,DK@DIA,R!DCB       .Set in DCB        D4DEM.860
3631     <macro>                PUT R!SPT,DK@SPT,R!DCB       .Set sectors/track D4DEM.861
3644     <macro>                PUT R!CPD,DK@CPD,R!DCB       .Cylinders/disk    D4DEM.862
3651     <macro>                PUT R!TM,DK@TM,R!DCB         .Save target memory D4DEM.863
3664     <macro>                PUT R!TMP,DK@TMP,R!DCB       .Target memory IOP D4DEM.864
3700     <macro>                GET A,DL@RCC,R!DAL           .Read ahead count  D4DEM.865
3704     <macro>                $IF (A = O'177777),OR,(R!TM = TM$BMR) .If R.A.'s suppressed  R16410DA.9
                                                                    .>>>>> 0        $IF        1
3713    010000  024423            R!%W1 = 0                  .No read aheads    D4DEM.867
3715    010000  024424            R!%W2 = 0                  .No write behinds  D4DEM.868
3717     <macro>                $ELSE                                           D4DEM.869
3720     <macro>                  GET R!%W1,DK@DRC,R!DCB     .Default values    D4DEM.870
```

```
3726    <macro>                    GET R!%W2,DK@DV     DCB                                          D4DEM.8
3734    <macro>                  $ENDIF                                                            D4DEM.872
                                                                          .<<<<< 0                     $ENDIF    1
3734    <macro>                  PUT R!%W1,DK@RCC,R!DCB        .Set in DCB                         D4DEM.873
3747    <macro>                  PUT R!%W2,DK@WCC,R!DCB                                            D4DEM.874
3762    <macro>                  PUT R!SUB,DK@SUB,R!DCB        .Subfunction code                   D4DEM.875
3775    <macro>                  $IF (R!DIA = 1)               .If diagnostic request              D4DEM.876
                                                                          .>>>>> 0                     $IF       1
4002  010000  024425                R!%W3 = 0                   .No spiraling                       D4DEM.877
4004    <macro>                    STORE 0,DK@SPO,R!DCB        .Clear spiral offset                D4DEM.878
4011    <macro>                  $ELSE                                                             D4DEM.879
4012    <macro>                    GET R!%W3,DP@SPF,R!DPT      .Default spiral mode                D4DEM.880
4020    <macro>                    $IF (A # 0)                                                       D4DEM.881
                                                                          .>>>>> 1                     $IF       1
4022    <macro>                      GET R!%W1,DP@IDX,R!DPT    .Spiral index value                 D4DEM.882
4030  020443  021423  024423         R!%W1 = R!CYL & R!%W1     .Index                              D4DEM.883
4033    <macro>                      ADDRESS R!%W2,DP@SPI,R!DPT .Spiral offset table               D4DEM.884
4036  020424  022423  024424         R!%W2 = R!%W2 + R!%W1     .Add index to base                  D4DEM.885
4041  030424  024424                 R!%W2 = (R!%W2)           .Spiral index                       D4DEM.886
4043    <macro>                    $ELSE                                                             D4DEM.887
4044  010000  024424                 R!%W2 = 0                 .If spiraling not enabled           D4DEM.888
4046    <macro>                    $ENDIF                                                            D4DEM.889
                                                                          .<<<<< 1                     $ENDIF    1
4046    <macro>                    PUT R!%W2,DK@SPO,R!DCB      .Save in DCB                        D4DEM.890
4053    <macro>                  $ENDIF                                                            D4DEM.891
                                                                          .<<<<< 0                     $ENDIF    1
4053    <macro>                  PUT R!%W3,DK@SPF,R!DCB        .Set in DCB                         D4DEM.892
                                                                                                     D4DEM.893
                               *  Perform unit select or seek as needed.                            D4DEM.894
                                                                                                     D4DEM.895
4067    <macro>                  GET A,DK@SEL,R!DCB            .Select flag                        D4DEM.896
4074  127000 /005632             P = SELECT, A # 0             .If channel not selected          R13739AP.8
4076    <macro>                  GET A,DK@UNT,R!DCB            .Active unit                        D4DEM.898
4103  023466  127000 /005632     P = SELECT, A # R!UNIT        .If not the desired unit            D4DEM.899
4106  020451  011001             A = R!FNC & 1                                                     D4DEM.900
4110    <macro>                  $IF (A = 0)                   .If write                           D4DEM.901
                                                                          .>>>>> 0                     $IF       1
4112    <macro>                    $IF (R!TYPE = DSK$DD49),OR,(R!TYPE = DSK$DD50) .If DD49/DD50    D4DEM.902
                                                                          .>>>>> 1                     $IF       1
4121    <macro>                      GET R!%W1,DK@CE,R!DCB     .CE cylinder flag                   D4DEM.903
4130    <macro>                      GET R!%W3,DP@DCY,R!DPT    .Default cylinders/disk             D4DEM.904
4135    <macro>                      $IF (R!%W1 = 0),AND,(R!CYL >= R!%W3)                          D4DEM.905
```

```
                                                                  .>>>>> 2          $IF      1
4144  010001  024424          R!%W2 = OL$ECE          .Write-enable CE cylinders      D4DEM.906
4146  075000 /005767          P = CRDDIAG             .Create DDIAG                   D4DEM.907
4150   <macro>               $ELSEIF (R!%W1 # 0),AND,(R!CYL < R!%W3)                 D4DEM.908
4160  010002  024424          R!%W2 = OL$PCE          .Write-protect CE cylinders     D4DEM.909
4162  075000 /005767          P = CRDDIAG             .Create DDIAG                   D4DEM.910
4164   <macro>               $ENDIF                                                  D4DEM.911
                                                                  .<<<<< 2          $ENDIF   1
4164   <macro>               $ENDIF                                                  D4DEM.912
                                                                  .<<<<< 1          $ENDIF   1
4164   <macro>             $ENDIF                                                    D4DEM.913
                                                                  .<<<<< 0          $ENDIF   1
4164   <macro>             GET R!%W1,DK@CYL,R!DCB  .Current cylinder                 D4DEM.914
4171   <macro>             $IF (A # R!CYL)         .If seek needed                   D4DEM.915
                                                                  .>>>>> 0          $IF      1
4174   <macro>               GET A,DK@OWC,R!DCB                                    R13739AP.9
4200  127000 /005350         P = HEDSEL, A # 0       .Sync track buffer            R13739AP.10
4202  077000 /006022         R = CLEAR               .Clear channel                  D4DEM.916
4204  102002  001000         EXIT, A # 0             .If error                       D4DEM.917
4206  020443                 A = R!CYL               .Desired cylinder               D4DEM.918
4207  165000                 IOB: 5                  .Select cylinder                D4DEM.919
4210   <macro>               PUT R!CYL,DK@ACC,R!DCB  .Save accumulator parameter     D4DEM.920
4215   <macro>               STORE 1,DKF@SK,R!DCB    .Set seek flag                  D4DEM.921
4223   <macro>               PUT R!CYL,DK@CYL,R!DCB  .Set new cylinder               D4DEM.922
4230   <macro>               PUT R!HED,DK@NHD,R!DCB  .Head to select when seek done  D4DEM.923
4243  010024  024427         R!%W5 = D4SKLIM         .Timeout limit                  D4DEM.924
4245  077000 /006056         R = TIMER               .Put on timer queue             D4DEM.925
4247   <macro>               KRECORD TF$D4SK,(B,R!%W1,R!CYL,R!DCB)                   D4DEM.926
                                     *  Channel                                      D4DEM.927
                                     *  Old cylinder                                 D4DEM.928
                                     *  New cylinder                                 D4DEM.929
                                     *  DCB address                                  D4DEM.930
4266   <macro>             $ELSE                                                     D4DEM.931
4267   <macro>               GET R!%W1,DK@HED,R!DCB  .Current head group             D4DEM.932
4276  020467  013006  103044  P = INIT1, R!TYPE # DSK$DD40 .If not DD40            R13739AP.11
4301   <macro>               GET A,DK@SYN,R!DCB      .Check synchronization info   R13739AP.12
4307  102041                 P = INITB, A = 0        .If sync not needed           R13739AP.13
4310  011001  024424         R!%W2 = A & 1                                         R13739AP.14
4312  020451  011001         A = R!FNC & 1           .Compare with current function R13739AP.15
4314  023424  127000 /005350 P = HEDSEL, A # R!%W2   .If switch between rd & wrt    R13739AP.16
4317   <macro>               GET A,DK@OWC,R!DCB                                    R13739AP.17
4323  102025                 P = INITB, A = 0        .If no outstanding writes     R13739AP.18
```

```
4332   <macro>                          GET R!%W2,DK@SL     DCB                                  R13739AP.19
4332   026424                           R!%W2 = R!%W2 + 1            .Advance sector              R13739AP.20
4333   023473                           A = A - R!SPT                                             R13739AP.21
4334   <macro>                          $IF (A = 0)                  .If wrap                     R13739AP.22
                                                                                   .>>>>> 1          $IF      1
4336   024424                             R!%W2 = A                                               R13739AP.23
4337   026423                             R!%W1 = R!%W1 + 1          .Advance head (can't wrap)   R13739AP.24
4340   <macro>                          $ENDIF                                                    R13739AP.25
                                                                                   .<<<<< 1          $ENDIF   1
4340   020424  023445  127000             P = HEDSEL, R!%W2 # R!SEC  .If not contiguous           R13739AP.26
      /005350
4344                            INIT1   *                                                         R13739AP.27
4344   020423  023444  127000             P = HEDSEL, R!%W1 # R!HED  .Select head group           R13739AP.28
      /005350
4350   <macro>                          $ENDIF                                                      D4DEM.950
                                                                                   .<<<<< 0          $ENDIF   1
                                                                                                    D4DEM.951
                                        *        Start the I/O.                                     D4DEM.952
                                                                                                    D4DEM.953
4350                            INITB   *                                                           D4DEM.954
4350   <macro>                          $IF (R!FNC = DAF$READ),OR,(R!FNC = DAF$INTR) .If read       D4DEM.955
                                                                                   .>>>>> 0          $IF      1
4357   077000 /005244                     R = GETBUFS                .Allocate local buffers      D4DEM.956
4361   020434  137000 /006072             R = STARTIT, R!STATUS # 0  .Start I/O                   D4DEM.957
4364   <macro>                          $ELSE                        .If write                    D4DEM.958
4365   077000 /007605                     R = SENDTR                 .Send data request to BIOP   D4DEM.959
4367   020434  102002  001000             EXIT, R!STATUS # 0         .If data not available       D4DEM.960
4372   077000 /005244                     R = GETBUFS                .Allocate local buffers      D4DEM.961
4374   <macro>                          $IF (R!STATUS # 0)           .If enough available         D4DEM.962
                                                                                   .>>>>> 1          $IF      1
4377   002000  000000                       I = 0                                                  D4DEM.963
4401   077000 /006732                       R = FILL                 .Fill the rest with data      D4DEM.964
4403   <macro>                              GET R!DAL,DK@DLO,R!DCB   .Get first on queue again    D4DEM.965
4410   077000 /007274                       R = DALPAR               .Get its parameters           D4DEM.966
4412   077000 /006072                       R = STARTIT              .Start I/O                    D4DEM.967
4414   <macro>                          $ENDIF                                                    D4DEM.968
                                                                                   .<<<<< 1          $ENDIF   1
4414   <macro>                          $ENDIF                                                    D4DEM.969
                                                                                   .<<<<< 0          $ENDIF   1
4414   001000                           EXIT                                                      D4DEM.970
                                                                                                    D4DEM.971
                                        ************************************************************ D4DEM.972
```

```
                                *                                                                  *   D4DEM.973
                                *        WBSTAT:   Write Behind status processing.                 *   D4DEM.974
                                *                                                                  *   D4DEM.975
                                *        Entry:    R!DCB = DCB address                             *   D4DEM.976
                                *                  R!DAL = DAL address                             *   D4DEM.977
                                *                  R!TOTAL = Total sectors for request             *   D4DEM.978
                                *                  R!SPT = Sectors per track                       *   D4DEM.979
                                *                  R!TPC = Tracks per cylinder                     *   D4DEM.980
                                *                                                                  *   D4DEM.981
                                ********************************************************************   D4DEM.982
                                                                                                       D4DEM.983
4415                            WBSTAT   *                                                             D4DEM.984
4415    <macro>                          GET A,DK@DL1,R!DCB            .Last DAL                      D4DEM.985
4421   023433  103057                    P = STAT, A # R!DAL           .If more                       D4DEM.986
4423   077000 /007353                    R = DALADD                    .Get disk address from DAL     D4DEM.987
4425   020445  022450  024445            R!SEC = R!SEC + R!TOTAL       .Ending sector                 D4DEM.988
4430   077000 /007255                    R = CALC                      .Calculate ending address      D4DEM.989
4432    <macro>                          PUT R!SEC,DK@RAS,R!DCB        .Save in DCB                   D4DEM.990
4445    <macro>                          PUT R!HED,DK@RAH,R!DCB                                       D4DEM.991
4461    <macro>                          PUT R!CYL,DK@RAC,R!DCB                                       D4DEM.992
4466    <macro>                          PUT R!UNIT,DK@RAU,R!DCB                                      D4DEM.993
4501                            STAT     *                                                            D4DEM.994
4501   002000  000000                    I = 0                                                        D4DEM.995
4503    <macro>                          GET A,DA@IFC,R!DAL            .Internal function code        D4DEM.996
4507    <macro>                          $IF (A = IM$STAT)             .Early reply already sent      D4DEM.997
                                                                                        .>>>>> 0          $IF      1
4512   020433  024000                      R!DS = R!DAL                .DAL to send                   D4DEM.998
4514    <macro>                            RKERN DDALGETA              .Get an MOS DAL                D4DEM.999
4516    <macro>                            RKERN DMGA                  .Send to MIOP                  D4DEM.1000
4520    <macro>                          $ELSE                                                        D4DEM.1001
4521    <macro>                            STORE IM$STAT,DA@IFC,R!DAL  .Internal function code        D4DEM.1002
4526    <macro>                            GET A,DL@RC,R!DAL           .Response code                 D4DEM.1003
4533    <macro>                            $IF (A # DAR$OK)            .If error                      D4DEM.1004
                                                                                        .>>>>> 1          $IF      1
4535   020433  024000                        R!DS = R!DAL              .DAL to send                   D4DEM.1005
4537    <macro>                              RKERN DMGA                .Send to MIOP                  D4DEM.1006
4541    <macro>                              STORE 0,DL@ERC,R!DAL      .Clear errcount for next reply D4DEM.1007
4546    <macro>                            $ELSE                                                      D4DEM.1008
4547    <macro>                              GET R!EH,DA@MES,R!DAL     .A-to-A message                D4DEM.1009
4554   010000  024000                        R!EI = MIOP               .Target IOP                    D4DEM.1010
4556    <macro>                              RKERN EMSGIOP             .Send accumulator msg          D4DEM.1011
4560    <macro>                            $ENDIF                                                     D4DEM.1012
```

```
                                                                .<<<<< 1          $EN      1
4560    <macro>                 $ENDIF                                          D4DEM.1015
                                                                .<<<<< 0          $ENDIF   1
4560   003000                   I = 1                                           D4DEM.1014
4561   001000                   EXIT                                            D4DEM.1015
                                                                                D4DEM.1016
                        ************************************************************D4DEM.1017
                        *                                                       *D4DEM.1018
                        *       READMCB:  Gets the MOS Control Block info for this channel.  *D4DEM.1019
                        *                                                       *D4DEM.1020
                        *       Entry:    R!DCB = DCB address                   *D4DEM.1021
                        *                 R!IMP = IOP with path to memory       *D4DEM.1022
                        *                                                       *D4DEM.1023
                        *       Exit:     R!MCB = MCB local address             *D4DEM.1024
                        *                 R!MOSU = MCB MOS address (upper)      *D4DEM.1025
                        *                 R!MOSL = MCB MOS address (lower)      *D4DEM.1026
                        *                 R!BINFO = BIOP info address           *D4DEM.1027
                        *                 R!BPTR = Offset to next BIOP MOS buffer *D4DEM.1028
                        *                 R!DINFO = DIOP info address           *D4DEM.1029
                        *                 R!DPTR = Offset to next DIOP MOS buffer *D4DEM.1030
                        *                 R!NUM = Number of MOS buffers         *D4DEM.1031
                        *                 R!BUFU = Base address for buffers (upper) *D4DEM.1032
                        *                 R!BUFL = Base address for buffers (lower) *D4DEM.1033
                        *                 A = 0 if no local memory, 1 if all is OK *D4DEM.1034
                        *                                                       *D4DEM.1035
                        ************************************************************D4DEM.1036
                                                                                D4DEM.1037
4562                    READMCB *                                               D4DEM.1038
4562    <macro>                 GET R!BUFU,DK@BF0,R!DCB     .Base of MOS buffers D4DEM.1039
4567    <macro>                 GET R!BUFL,DK@BF1,R!DCB                         D4DEM.1040
4574    <macro>                 GET R!NUM,DK@CNT,R!DCB      .Number of buffers  D4DEM.1041
4602    <macro>                 $IF (R!TMP = R!%MYID)       .If path present    D4DEM.1042
                                                                .>>>>> 0          $IF      1
4606    <macro>                   GET R!BPTR,DK@BIO,R!DCB   .BIOP pointer       D4DEM.1043
4614    <macro>                   GET R!DPTR,DK@DIO,R!DCB   .DIOP pointer       D4DEM.1044
4622    <macro>                 $ELSE                                           D4DEM.1045
4623    <macro>                   GET R!MOSU,DK@MC0,R!DCB   .MOS address of MCB D4DEM.1046
4630    <macro>                   GET R!MOSL,DK@MC1,R!DCB                       D4DEM.1047
4635   002000  000000             I = 0                     .Ensure interrupts disabled D4DEM.1048
4637    <macro>                   $IF (R!MCB = 0)           .If no local memory D4DEM.1049
                                                                .>>>>> 1          $IF      1
4642   010014  024000               R!HB = CB@SZ@P          .MCB parcel size    D4DEM.1050
```

```
4644    <macro>                          RKERN HMA                  .Get it                          D4DEM.1051
4646   020000  024435                    R!MCB = R!HC               .Address                         D4DEM.1052
4650   103002  001000                    EXIT, A = 0                .If no memory available          D4DEM.1053
4652    <macro>                        $ENDIF                                                        D4DEM.1054
                                                                                   .<<<<< 1              $ENDIF    1
4652   010001  024000                  R!CA = MIO$BMR               .Buffer memory read              D4DEM.1055
4654   020436  024000                  R!CB = R!MOSU                .MOS address                     D4DEM.1056
4655   020437  024000                  R!CC = R!MOSL                                                 D4DEM.1057
4660   020435  024000                  R!CD = R!MCB                 .Local address                   D4DEM.1058
4662   010003  024000                  R!CE = CB@SZ                 .Word length                     D4DEM.1059
4664    <macro>                        RKERN COMIO                  .Read                            D4DEM.1060
4665   010000  005002  022435          R!DINFO = CB$DIOP < 2 + R!MCB .DIOP info                      D4DEM.1061
       024464
4672    <macro>                        GET R!DPTR,DD@PTR,R!DINFO    .Offset to next buffer           D4DEM.1062
4677   010001  005002  022435          R!BINFO = CB$BIOP < 2 + R!MCB .BIOP info                      D4DEM.1063
       024462
4703    <macro>                        GET R!BPTR,BB@PTR,R!BINFO    .Offset to next buffer           D4DEM.1064
4710   010002  005002  022435          R!%W1 = CB@LH < 2 + R!MCB    .Entry address                   D4DEM.1065
       024423
4714    <macro>                      $ENDIF                                                          D4DEM.1066
                                                                                   .<<<<< 0              $ENDIF    1
4714   010001                          A = 1                        .Good status                     D4DEM.1067
4715   001000                          EXIT                                                          D4DEM.1068
                                                                                                     D4DEM.1069
                                 ************************************************************        D4DEM.1070
                                 *                                                          *        D4DEM.1071
                                 *        GETMOS:   Get the MOS buffer address from the specified *  D4DEM.1072
                                 *                  offset.                                 *        D4DEM.1073
                                 *                                                          *        D4DEM.1074
                                 *        Entry:    R!%W1 = Offset                          *        D4DEM.1075
                                 *                  R!BUFU = MOS base address (upper)       *        D4DEM.1076
                                 *                  R!BUFL = MOS base address (lower)       *        D4DEM.1077
                                 *                                                          *        D4DEM.1078
                                 *        Exit:     R!BUFU = MOS address (upper)            *        D4DEM.1079
                                 *                  R!BUFL = MOS address (lower)            *        D4DEM.1080
                                 *                                                          *        D4DEM.1081
                                 ************************************************************        D4DEM.1082
                                                                                                     D4DEM.1083
4716                             GETMOS   *                                                          D4DEM.1084
4716   010000  024424                    R!%W2 = 0                    .Initialize                      D4DEM.1085
4720    <macro>                          LSHIFT R!%W2,R!%W1,D'9       .Multiply by D'512               D4DEM.1086
4737    <macro>                          DADD R!BUFU,R!BUFL,R!%W2,R!%W1 .Add to base                   D4DEM.1087
```

```
4747    001000                          EXIT                                                    D4DEM.1088
                                                                                                D4DEM.1089
                                 ******************************************************************D4DEM.1090
                                 *                                                                *D4DEM.1091
                                 *        WRITBIOP: Updates the BIOP portion of MCB.              *D4DEM.1092
                                 *                                                                *D4DEM.1093
                                 *        WRITDIOP: Updates the DIOP portion of MCB.              *D4DEM.1094
                                 *                                                                *D4DEM.1095
                                 *        Entry:    R!BINFO = BIOP info address                   *D4DEM.1096
                                 *                  R!BPTR = offset to next MOS buffer            *D4DEM.1097
                                 *                  R!DINFO = BIOP info address                   *D4DEM.1098
                                 *                  R!DPTR = offset to next MOS buffer            *D4DEM.1099
                                 *                  R!MOSU = MOS address of MCB (upper)           *D4DEM.1100
                                 *                  R!MOSL = MOS address of MCB (lower)           *D4DEM.1101
                                 *                  R!MCB = MCB local address                     *D4DEM.1102
                                 *                  R!NUM = Number of buffers                     *D4DEM.1103
                                 *                  R!%W1 = Increment for DIOP pointer            *D4DEM.1104
                                 *                  R!TMP = IOP with path to memory               *D4DEM.1105
                                 *                                                                *D4DEM.1106
                                 ******************************************************************D4DEM.1107
                                                                                                D4DEM.1108
4750                             WRITBIOP *                                                     D4DEM.1109
4750    026463                           R!BPTR = R!BPTR + 1          .Next entry               D4DEM.1110
4751    023510  103003  010000           R!BPTR = 0, A = R!NUM        .Wrap to top              D4DEM.1111
        024463
4755                             WRITB    *                                                     D4DEM.1112
4755    002000  000000                   I = 0                                                  D4DEM.1113
4757     <macro>                         $IF (R!TMP = R!%MYID)        .Path on this IOP         D4DEM.1114
                                                                                  .>>>>> 0          $IF      1
4763     <macro>                           PUT R!BPTR,DK@BIO,R!DCB    .Save BIOP pointer        D4DEM.1115
4775     <macro>                         $ELSE                                                  D4DEM.1116
5000     <macro>                           PUT R!BPTR,BB@PTR,R!BINFO  .Save BIOP pointer        D4DEM.1117
5005     <macro>                           DADD R!MOSU,R!MOSL,CB$BIOP,,R!CB,R!CC .MOS address   D4DEM.1118
5014    020462  024000                     R!CD = R!BINFO             .Local address            D4DEM.1119
5016    010001  024000                     R!CE = CB$BBSZ             .Length in words          D4DEM.1120
5020     <macro>                         $ENDIF                                                 D4DEM.1121
                                                                                  .<<<<< 0          $ENDIF   1
5020    070054                           P = WRITD                                              D4DEM.1122
5021                             WRITDIOP *                                                     D4DEM.1123
5021    020465  022423  024465           R!DPTR = R!DPTR + R!%W1      .Next entry               D4DEM.1124
5024    041000  023510  100004           R!DPTR = R!DPTR - R!NUM, A >= R!NUM .Wrap              D4DEM.1125
        020465  023510  024465
```

```
5032   002000  000000                I = 0                                                    D4DEM.1126
5034    <macro>                      $IF (R!TMP = R!%MYID)          .Path on this IOP         D4DEM.1127
                                                                                .>>>>> 0          $IF      1
5040    <macro>                        PUT R!DPTR,DK@DIO,R!DCB      .Save DIOP pointer        D4DEM.1128
5053    <macro>                      $ELSE                                                    D4DEM.1129
5054    <macro>                        PUT R!DPTR,DD@PTR,R!DINFO    .Save DIOP pointer        D4DEM.1130
5061    <macro>                        DADD R!MOSU,R!MOSL,CB$DIOP,,R!CB,R!CC                  D4DEM.1131
5070   020464  024000                  R!CD = R!DINFO               .Local memory address     D4DEM.1132
5072   010001  024000                  R!CE = CB$DDSZ               .Length to write          D4DEM.1133
5074    <macro>                      $ENDIF                                                   D4DEM.1134
                                                                                .<<<<< 0          $ENDIF   1
5074                          WRITD    *                                                       D4DEM.1135
5074    <macro>                      $IF (R!TMP # R!%MYID)          .Path not on this IOP     D4DEM.1136
                                                                                .>>>>> 0          $IF      1
5100   010003  024000                  R!CA = MIO$BMW               .Buffer memory write      D4DEM.1137
5102    <macro>                        RKERN COMIO                  .Write                    D4DEM.1138
5104                          RELMEM   *                                                       D4DEM.1139
5104   002000  000000                  I = 0                        .Ensure interrupts disabled D4DEM.1140
5106   010014  024000                  R!HB = CB@SZ@P               .Memory size              D4DEM.1141
5110   020435  024000                  R!HC = R!MCB                 .Address                  D4DEM.1142
5112    <macro>                        RKERN HMD                    .Release                  D4DEM.1143
5114   010000  024435                  R!MCB = 0                    .Clear                    D4DEM.1144
5116    <macro>                      $ENDIF                                                   D4DEM.1145
                                                                                .<<<<< 0          $ENDIF   1
5115   003000                       I = 1                                                    D4DEM.1146
5117   001000                       EXIT                                                     D4DEM.1147
                                                                                             D4DEM.1148
                              ***********************************************************    D4DEM.1149
                              *                                                         *    D4DEM.1150
                              *        MOVEIT:   Transfer next read ahead or write behind sector.  *    D4DEM.1151
                              *                                                         *    D4DEM.1152
                              *        Entry:    R!DAL = DAL address                    *    D4DEM.1153
                              *                  R!SC1 = Number sectors offset from start *    D4DEM.1154
                              *                  R!CODE = Transfer direction            *    D4DEM.1155
                              *                  R!BUFU = MOS address upper             *    D4DEM.1156
                              *                  R!BUFL = MOS address lower             *    D4DEM.1157
                              *                                                         *    D4DEM.1158
                              ***********************************************************    D4DEM.1159
                                                                                             D4DEM.1160
5120                          MOVEIT   *                                                       D4DEM.1161
5120   072033                       R = CPUCALC                  .Calculate CPU address    D4DEM.1162
5121    <macro>                      TRANSFER R!CODE,R!TM,R!CPU,R!CPL,R!BUFU,R!BUFL,D4$WPS   D4DEM.1163
```

```
5151   073201                    R = WRITBIOP                .Update BIOP part of MCB          D4DEM.11
5152   001000                    EXIT                                                          D4DEM.1162
                                                                                                D4DEM.1166
                          ***********************************************************************D4DEM.1167
                          *                                                                     *D4DEM.1168
                          *        CPUCALC:  Calculate Central Memory address.                  *D4DEM.1169
                          *                                                                     *D4DEM.1170
                          *        Entry:    R!SC1 = Number sectors,offset from start           *D4DEM.1171
                          *                  R!DAL = DAL address                                *D4DEM.1172
                          *                                                                     *D4DEM.1173
                          *        Exit:     R!CPU = CPU address (upper)                        *D4DEM.1174
                          *                  R!CPL = CPU address (lower)                        *D4DEM.1175
                          *                                                                     *D4DEM.1176
                          ***********************************************************************D4DEM.1177
                                                                                                D4DEM.1178
5153                      CPUCALC  *                                                            D4DEM.1179
5153    <macro>                    GET R!CPU,DL@IMO,R!DAL      .Starting target memory address   D4DEM.1180
5162    <macro>                    GET R!CPL,DL@TM1,R!DAL      .                                 D4DEM.1181
5167    <macro>                    $IF (R!SC1 # 0)             .If not first                     D4DEM.1182
                                                                                .>>>>> 0          $IF      1
5172    <macro>                      GET R!%W5,DL@UNS,R!DAL    .# physical units                 D4DEM.1183
5200   020447  024426                R!%W4 = R!SC1             .Save offset                      D4DEM.1184
5202    <macro>                      $UNTIL (R!%W5 = 1)        .Multiply by units                D4DEM.1185
5205   020447  022426  024447          R!SC1 = R!SC1 + R!%W4                                     D4DEM.1186
5210   027427                          R!%W5 = R!%W5 - 1                                         D4DEM.1187
5211    <macro>                      $ENDTIL                                                    D4DEM.1188
5212   010000  024424                R!%W2 = 0                 .Convert to words                 D4DEM.1189
5214    <macro>                      LSHIFT R!%W2,R!SC1,D'9                                      D4DEM.1190
5233    <macro>                      DADD R!CPU,R!CPL,R!%W2,R!SC1 .Add to CPU base address       D4DEM.1191
5243    <macro>                    $ENDIF                                                       D4DEM.1192
                                                                                .<<<<< 0          $ENDIF   1
5243   001000                      EXIT                                                          D4DEM.1193
                                                                                                D4DEM.1194
                          ***********************************************************************D4DEM.1195
                          *                                                                     *D4DEM.1196
                          *        GETBUFS:  Allocates local memory buffers for a disk request. *D4DEM.1197
                          *                                                                     *D4DEM.1198
                          *        Entry:    R!DCB = DCB address                                *D4DEM.1199
                          *                  R!FNC = DAL function code                          *D4DEM.1200
                          *                                                                     *D4DEM.1201
                          *        Exit:     R!STATUS = return status (0 = bad)                 *D4DEM.1202
                          *                  R!ENTRY  = Current LMA entry                       *D4DEM.1203
```

```
                                 *                                                                  *   D4DEM.1204
                                 ************************************************************************   D4DEM.1205
                                                                                                         D4DEM.1206
5244                             GETBUFS  *                                                             D4DEM.1207
5244    <macro>                          GET R!ENTRY,DK@CLB,R!DCB       .Current entry                 D4DEM.1208
5251   024506                            R!LOCAL = A                    .Save                          D4DEM.1209
5252                             GETBUFSA *                                                             D4DEM.1210
5252    <macro>                          GET R!STATUS,LB@LOC,R!ENTRY    .Local address                 D4DEM.1211
5257    <macro>                          $IF (A = 0)                    .If not allocated              D4DEM.1212
                                                                                        .>>>>> 0          $IF      1
5261    <macro>                            BGET  R!STATUS               .Get a buffer                  D4DEM.1213
5271    <macro>                            $IF (A # 0)                  .If no more buffers            D4DEM.1214
                                                                                        .>>>>> 1          $IF      1
5273    <macro>                              STORE 1,DKS@RS,R!DCB       .Set waiting for buffer flag   D4DEM.1215
5304   077000 /010061                        R = ACTDEM                 .Put DCB back on queue         D4DEM.1216
5306   070146                                P = RELBUFS                .Release any allocated buffers  D4DEM.1217
5307    <macro>                            $ENDIF                                                      D4DEM.1218
                                                                                        .<<<<< 1          $ENDIF   1
5307    <macro>                            PUT R!STATUS,LB@LOC,R!ENTRY .Save the address             D4DEM.1219
5314    <macro>                          $ENDIF                                                      D4DEM.1220
                                                                                        .<<<<< 0          $ENDIF   1
5314    <macro>                          $IF (R!FNC = DAF$PTRN)         .If patterning                 D4DEM.1221
                                                                                        .>>>>> 0          $IF      1
5320    <macro>                            GET R!%W1,DL@PAT,R!DAL       .Pick up 1 parcel pattern      D4DEM.1222
5325   014000 /001000  005002              R!%W2 = D4$WPS < 2           .Number of parcels in sector   D4DEM.1223
       024424
5331   020434  024425                      R!%W3 = R!STATUS             .Starting address              D4DEM.1224
5333    <macro>                            $UNTIL (R!%W2 = 0)           .Until all parcels patterned   D4DEM.1225
5335   020423  034425                        (R!%W3) = R!%W1                                           D4DEM.1226
5337   026425                                R!%W3 = R!%W3 + 1                                         D4DEM.1227
5340   027424                                R!%W2 = R!%W2 - 1                                         D4DEM.1228
5341    <macro>                            $ENDTIL                                                     D4DEM.1229
5342    <macro>                          $ENDIF                                                      D4DEM.1230
                                                                                        .<<<<< 0          $ENDIF   1
5342   077000 /007565                    R = NEXTBUF                    .Next entry                    D4DEM.1231
5344   020442  023506  107074            P = GETBUFSA, R!ENTRY # R!LOCAL .If more to do                D4DEM.1232
5347   001000                            EXIT                                                          D4DEM.1233
                                                                                                       R13739AP.29
                                 ************************************************************************R13739AP.30
                                 *                                                                    *R13739AP.31
                                 *        HEDSEL:   Select head group.                                *R13739AP.32
                                 *                                                                    *R13739AP.33
```

```
                        *        Entry:     B = channel number                                  *R13739AP.34
                        *                   R!DCB = DCB address                                 *R13739AP.35
                        *                   R!HED = new head group number                       *R13739AP.36
                        *                   R!TYPE = device type                                *R13739AP.37
                        *                                                                       *R13739AP.38
                        *        Exit:      A # 0 if error in clearing channel                  *R13739AP.39
                        *                                                                       *R13739AP.40
                        ************************************************************************R13739AP.41
                                                                                                 R13739AP.42
5350                    HEDSEL   *                                .Entry                         R13739AP.43
5350   077000 /006022           R = CLEAR                         .Clear channel                 R13739AP.44
5352   102002  001000           EXIT, A # 0                       .If unable to clear            R13739AP.45
5354    <macro>                 GET R!%W1,DK@HED,R!DCB            .Get old head group            R13739AP.46
5363    <macro>                 PUT R!HED,DK@HED,R!DCB            .Set new head group            R13739AP.47
5377    <macro>                 $IF (R!TYPE = DSK$DD40)                                          R13739AP.48
                                                                                  .>>>>> 0          $IF      1
5403   020444  005007             A = R!HED < 7                                                  R13739AP.49
5405    <macro>                 $ELSE                                                            R13739AP.50
5406   020444  005010             A = R!HED < D'8                                                R13739AP.51
5410    <macro>                 $ENDIF                                                           R13739AP.52
                                                                                  .<<<<< 0          $ENDIF   1
5410   024424                   R!%W2 = A                         .Save for function echo        R13739AP.53
5411   016000 /040000           A = A + O'40000                                                  R13739AP.54
5413   161000                   IOB: 1                            .Select head                   R13739AP.55
5414    <macro>                 PUT R!%W2,DK@ACC,R!DCB            .Save accumulator parameter    R13739AP.56
5421    <macro>                 STORE 1,DKF@HD,R!DCB              .Set head select flag          R13739AP.57
5427   010002  024427           R!%W5 = D4HDLIM                   .Timeout limit                 R13739AP.58
5431   077000 /006056           R = TIMER                         .Put on timer queue            R13739AP.59
5433    <macro>                 kRECORD TF$D4HD,(B,R!%W1,R!HED,R!DCB)                            R13739AP.60
                                      *  Channel                                                  R13739AP.61
                                      *  Old head group                                           R13739AP.62
                                      *  New head group                                           R13739AP.63
                                      *  DCB address                                              R13739AP.64
5452   010000                   A = 0                             .Clear status                  R13739AP.65
5453   001000                   EXIT                                                             R13739AP.66
                                                                                                    D4DEM.1234
                        ************************************************************************    D4DEM.1235
                        *                                                                       *   D4DEM.1236
                        *        RELBUFS:   Releases local memory buffers.                      *   D4DEM.1237
                        *                                                                       *   D4DEM.1238
                        *        Entry:     R!DCB = DCB address                                 *   D4DEM.1239
                        *                                                                       *   D4DEM.1240
```

```
                                 ************************************************************************   D4DEM.1241
                                                                                                            D4DEM.1242
5454                             RELBUFS  *                                                                 D4DEM.1243
5454    <macro>                           GET R!ENTRY,DK@CLB,R!DCB       .Current LMA entry               D4DEM.1244
5461   024506                             R!LOCAL = A                    .Save                            D4DEM.1245
5462                             RELBUFSA *                                                                 D4DEM.1246
5462    <macro>                           GET R!SCO,LB@LOC,R!ENTRY       .Local buffer address            D4DEM.1247
5467    <macro>                           $IF (A # 0)                    .If allocated                    D4DEM.1248
                                                                                          .>>>>> 0          $IF      1
5471    <macro>                             GET A,LB@FUL,R!ENTRY         .Buffer full flag                D4DEM.1249
5476   127000 /010061                       P = ACTDEM, A # 0            .If buffer is full               D4DEM.1250
5500    <macro>                             BRET R!SCO                   .Release                         D4DEM.1251
5510    <macro>                             STORE 0,LB@LOC,R!ENTRY       .Clear address                   D4DEM.1252
5515    <macro>                           $ENDIF                                                           D4DEM.1253
                                                                                          .<<<<< 0          $ENDIF   1
5515   077000 /007565                     R = NEXTBUF                    .Next entry                      D4DEM.1254
5517   020442  023506  107037             P = RELBUFSA, R!ENTRY # R!LOCAL .If more to do                  D4DEM.1255
5522   001000                             EXIT                                                             D4DEM.1256
                                                                                                           R13739AP.67
                                 ************************************************************************R13739AP.68
                                 *                                                                       *R13739AP.69
                                 *        RELEASE:  Release the active channel.                          *R13739AP.70
                                 *                                                                       *R13739AP.71
                                 *        Entry:    R!DCB = DCB address                                  *R13739AP.72
                                 *                  R!CHAN = channel number                              *R13739AP.73
                                 *                  R!TYPE = device type                                 *R13739AP.74
                                 *                                                                       *R13739AP.75
                                 *        Exit:     R!UNIT = active unit                                 *R13739AP.76
                                 *                                                                       *R13739AP.77
                                 ************************************************************************R13739AP.78
                                                                                                           R13739AP.79
5523                             RELEASE  *                                                                R13739AP.80
5523    <macro>                           GET A,DK@SEL,R!DCB             .Selected flag                   R13739AP.81
5530   103101                             P = RELX, A # 0                .If unit not selected            R13739AP.82
5531    <macro>                           GET A,DK@OWC,R!DCB                                              R13739AP.83
5535    <macro>                           $IF (A # 0)                    .If sync needed                  R13739AP.84
                                                                                          .>>>>> 0          $IF      1
5537   010000  024444                       R!HED = 0                                                     R13739AP.85
5541   071171                               P = HEDSEL                   .Sync DD40 write buffer          R13739AP.86
5542    <macro>                           $ENDIF                                                           R13739AP.87
                                                                                          .<<<<< 0          $ENDIF   1
5542   077000 /006022                     R = CLEAR                      .Clear the channel               R13739AP.88
```

```
5544   103065                          P = RELX, A # 0                 .If error                          R13739AP.89
5545    <macro>                        GET R!UNIT,DK@UNT,R!DCB         .Active unit number                R13739AP.90
5553    <macro>                        $IF (R!TYPE = DSK$DD39),OR,(R!TYPE = DSK$DD40) .If DD39/40         R13739AP.91
                                                                                        .>>>>> 0                $IF      1
5562   014000 /170000  022466            A = O'170000 + R!UNIT          .Parameter                         R13739AP.92
5565    <macro>                        $ELSEIF (R!TYPE = DSK$DD49)     .If DD49                           R13739AP.93
5572   014000 /172524  022466            A = O'172524 + R!UNIT          .Parameter                         R13739AP.94
5575    <macro>                        $ELSEIF (R!TYPE = DSK$DD50)     .If DD50                           R13739AP.95
5602   014000 /172534  022466            A = O'172534 + R!UNIT          .Parameter                         R13739AP.96
5605    <macro>                        $ELSE                                                              R13739AP.97
5606    <macro>                          $PUNTIF CODE=PT$DEV           .Illegal device type               R13739AP.98
5610    <macro>                        $ENDIF                                                             R13739AP.99
                                                                                        .<<<<< 0                $ENDIF   1
5610   024423                          R!%W1 = A                       .Save parameter                    R13739AP.100
5611   161000                          IOB: 1                          .Release                           R13739AP.101
5612    <macro>                        PUT R!%W1,DK@ACC,R!DCB          .Save accumulator parameter        R13739AP.102
5617    <macro>                        STORE 1,DKF@RL,R!DCB            .Set flag                          R13739AP.103
5625   010002  024427                  R!%W5 = D4HDLIM                 .Timeout limit                     R13739AP.104
5627   077000 /006056                  R = TIMER                       .Put on timer queue                R13739AP.105
5631                           RELX    *                                                                  R13739AP.106
5631   001000                          EXIT                                                               R13739AP.107
                                                                                                             D4DEM.1257
                               ***********************************************************************     D4DEM.1258
                               *                                                                     *     D4DEM.1259
                               *       SELECT:   Selects the specified channel.                      *     D4DEM.1260
                               *                                                                     *     D4DEM.1261
                               *       Entry:    R!DCB = DCB address                                 *     D4DEM.1262
                               *                 R!CHAN = Channel number                             *     D4DEM.1263
                               *                 R!UNIT = Unit number                                *     D4DEM.1264
                               *                 R!TYPE = Device type                                *     D4DEM.1265
                               *                                                                     *     D4DEM.1266
                               ***********************************************************************     D4DEM.1267
                                                                                                             D4DEM.1268
5632                           SELECT  *                                                                     D4DEM.1269
5632    <macro>                        GET A,DK@SEL,R!DCB                                                 R13739AP.108
5637   106114                          P = RELEASE, A = 0              .If unit selected                  R13739AP.109
5640   077000 /006022                  R = CLEAR                       .Clear channel                        D4DEM.1270
5642   102002  001000                  EXIT, A # 0                     .If error                             D4DEM.1271
5644    <macro>                        $IF (R!TYPE = DSK$DD39),OR,(R!TYPE = DSK$DD40) .If DD39/DD40       R13739AP.110
                                                                                        .>>>>> 0                $IF      1
5653   014000 /010000  022466            R!%W1 = O'10000 + R!UNIT       .Parameter                            D4DEM.1273
       024423
```

```
5657    <macro>                         $ELSEIF (R!TYPE = DSK$DD49)   .If DD49                          D4DEM.1274
5664   014000 /012524  022466             R!%W1 = O'12524 + R!UNIT    .Parameter                        D4DEM.1275
       024423
5670    <macro>                         $ELSEIF (R!TYPE = DSK$DD50)   .If DD50                          D4DEM.1276
5675   014000 /012534  022466             R!%W1 = O'12534 + R!UNIT    .Parameter                        D4DEM.1277
       024423
5701    <macro>                         $ELSE                                                           D4DEM.1278
5702    <macro>                           $PUNTIF CODE=PT$DEV         .Illegal device type              D4DEM.1279
5704    <macro>                         $ENDIF                                                          D4DEM.1280
                                                                                  .<<<<< 0                $ENDIF    1
5704   161000                           IOB: 1                      .Select                             D4DEM.1281
5705    <macro>                         PUT R!%W1,DK@ACC,R!DCB      .Save accumulator param             D4DEM.1282
5712    <macro>                         STORE 1,DKF@SL,R!DCB        .Set select flag                    D4DEM.1283
5720    <macro>                         GET A,DK@UNT,R!DCB          .Last unit selected                 D4DEM.1284
5725    <macro>                         $IF (A # R!UNIT)            .If different                       D4DEM.1285
                                                                                  .>>>>> 0                $IF       1
5730    <macro>                           STORE O'177777,DK@CYL,R!DCB .Force a seek                     D4DEM.1286
5736    <macro>                           STORE O'37,DK@HED,R!DCB                                       D4DEM.1287
5747    <macro>                           PUT R!UNIT,DK@UNT,R!DCB     .Set new unit                   D4DEM.1288
5762    <macro>                         $ENDIF                                                          D4DEM.1289
                                                                                  .<<<<< 0                $ENDIF    1
5762   010002  024427                   R!%W5 = D4HDLIM             .Timeout limit                      D4DEM.1290
5764   077000 /006056                   R = TIMER                   .Put on timer queue                 D4DEM.1291
5766   001000                           EXIT                                                            D4DEM.1292
                                                                                                         D4DEM.1293
                                 ********************************************************************    D4DEM.1294
                                 *                                                                  *    D4DEM.1295
                                 *        CRDDIAG:  Creates the online diagnostic function processor.  *    D4DEM.1296
                                 *                                                                  *    D4DEM.1297
                                 *        Entry:    R!DCB = DCB address                              *    D4DEM.1298
                                 *                  R!%W2 = Function code to perform                 *    D4DEM.1299
                                 *                                                                  *    D4DEM.1300
                                 ********************************************************************    D4DEM.1301
                                                                                                         D4DEM.1302
5767                             CRDDIAG  *                                                              D4DEM.1303
5767    <macro>                          CREATE DDIAG,(R!DCB,R!%W2),PRI=0 .Online diag. processor       D4DEM.1304
6011    <macro>                          $PUNTIF (A # 0)                                                D4DEM.1305
6013    <macro>                          STORE 1,DKF@ER,R!DCB        .Prevent other channel activity    D4DEM.1306
6021   001000                            EXIT                                                           D4DEM.1307
                                                                                                         D4DEM.1308
                                 ********************************************************************    D4DEM.1309
                                 *                                                                  *    D4DEM.1310
```

```
                                *        CLEAR:     Clear disk channel.                                  *   D4DEM.1311
                                *                                                                       *   D4DEM.1312
                                *                   R!CHAN = Channel                                    *   D4DEM.1313
                                *                   R!DCB = DCB address                                 *   D4DEM.1314
                                *                                                                       *   D4DEM.1315
                                *        Exit:      A = 0 if OK                                         *   D4DEM.1316
                                *                                                                       *   D4DEM.1317
                                ***********************************************************************   D4DEM.1318
                                                                                                          D4DEM.1319
6022                            CLEAR    *                                                                D4DEM.1320
6022   020432  054000                    B = R!CHAN                                                       D4DEM.1321
6024   160000                            IOB: 0                          .Clear                           D4DEM.1322
6025   000000                            PASS                                                             D4DEM.1323
6026    <macro>                          $IF (IOB = DN),OR,(IOB = BZ)    .If error                        D4DEM.1324
                                                                                        .>>>>> 0               $IF      1
6033    <macro>                            STORE S$NCLR,DK@STS,R!DCB     .Set status                      D4DEM.1325
6040    <macro>                            STORE 1,DKS@CR,R!DCB          .Set create flag                 D4DEM.1326
6050   077000 /010061                      R = ACTDEM                    .Put DCB back on queue           D4DEM.1327
6052   010001                              A = 1                         .Bad status                      D4DEM.1328
6053    <macro>                          $ELSE                                                            D4DEM.1329
6054   010000                              A = 0                         .Good status                     D4DEM.1330
6055    <macro>                          $ENDIF                                                           D4DEM.1331
                                                                                        .<<<<< 0               $ENDIF   1
6055   001000                           EXIT                                                             D4DEM.1332
                                                                                                          D4DEM.1333
                                ***********************************************************************   D4DEM.1334
                                *                                                                       *   D4DEM.1335
                                *        TIMER:     Put DCB on timeout queue                            *   D4DEM.1336
                                *                                                                       *   D4DEM.1337
                                *        Entry:     R!DCB = DCB address                                 *   D4DEM.1338
                                *                   R!%W5 = Time to wait                                *   D4DEM.1339
                                *                   B = Channel                                         *   D4DEM.1340
                                *                                                                       *   D4DEM.1341
                                ***********************************************************************   D4DEM.1342
                                                                                                          D4DEM.1343
6056                            TIMER    *                                                                D4DEM.1344
6056   002000  000000                    I = 0                                                            D4DEM.1345
6060    <macro>                          ADDRESS R!EH,DK@TMO,R!DCB       .Timeout cell                    D4DEM.1346
6063   020427  024000                    R!EG = R!%W5                    .Time to wait                    D4DEM.1347
6065    <macro>                          RKERN QTIME                     .Put on timer queue              D4DEM.1348
6067   003000                            I = 1                                                            D4DEM.1349
6070   167000                            IOB: 7                          .Enable channel interrupts       D4DEM.1350
```

```
6071   001000                    EXIT                                                    D4DEM.1351
                                                                                          D4DEM.1352
                                 ************************************************************D4DEM.1353
                                 *                                                        *D4DEM.1354
                                 *        STARTIT:  Initiates a read/write to DD39/49 disk.*D4DEM.1355
                                 *                                                        *D4DEM.1356
                                 *        Entry:    R!DCB = DCB address                   *D4DEM.1357
                                 *                  R!FNC = Read/write code               *D4DEM.1358
                                 *                  R!TYPE = Device type                  *R13739AP.111
                                 *                  R!CYL = Cylinder                      *D4DEM.1359
                                 *                  R!HED = Head group                    *D4DEM.1360
                                 *                  R!SEC = Sector                        *D4DEM.1361
                                 *                                                        *D4DEM.1362
                                 ************************************************************D4DEM.1363
                                                                                          D4DEM.1364
6072                             STARTIT  *                                               D4DEM.1365
6072    <macro>                           GET A,DK@ACT,R!DCB          .Channel activity   D4DEM.1366
6073   102002  001000                     EXIT, A # 0                 .If active          D4DEM.1367
6075   073053                             R = CLEAR                   .Clear channel      D4DEM.1368
6076   102002  001000                     EXIT, A # 0                 .If error           D4DEM.1369
6100    <macro>                           GET R!ENTRY,DK@CLB,R!DCB    .Current LMA entry  D4DEM.1370
6105   024423                             R!%W1 = A                   .Save               D4DEM.1371
6106    <macro>                           GET R!%W2,DK@LLB,R!DCB      .Last buffer entry  D4DEM.1372
6113    <macro>                           GET R!%W3,DK@FLB,R!DCB      .First buffer entry D4DEM.1373
6120                             CHKCP     *                                              D4DEM.1374
6120    <macro>                           STORE 0,LB@CP,R!%W1         .Clear continue pending flag D4DEM.1375
6127   010006  025423                     R!%W1 = R!%W1 + LB@LE       .Next               D4DEM.1376
6131   041000  023424  100004             R!%W1 = R!%W3, A > R!%W2    .Check wrap         D4DEM.1377
       102003  020425  024423
6137   020423  023442  107021             P = CHKCP, R!%W1 # R!ENTRY                      D4DEM.1378
6142    <macro>                           STORE 0,DK@LMA,R!DCB        .Reset LMA register to use D4DEM.1379
6153    <macro>                           STORE 0,DK@ADD,R!DCB        .Reset LMA check register D4DEM.1380
6162   072423                             R = LMA                     .Enter LMA and get parameter D4DEM.1381
6163   020451  011001  024423             R!%W1 = R!FNC & 1           .Decode function    D4DEM.1382
6166    <macro>                           GET A,LB@FUL,R!ENTRY        .Buffer full flag   D4DEM.1383
6173    <macro>                           $IF (A = R!%W1)             .If buffer not ready D4DEM.1384
                                                                                .>>>>> 0        $IF      1
6176    <macro>                             STORE 1,DKS@RS,R!DCB      .Waiting for resource D4DEM.1385
6207   001000                               EXIT                                          D4DEM.1386
6210    <macro>                           $ENDIF                                         D4DEM.1387
                                                                                .<<<<< 0        $ENDIF   1
6210   002000  000000                     I = 0                                         R13739AP.112
```

```
6212   <macro>                    $IF (R!%W1 # 0)                  .If read                                  D4DEM.1388
                                                                                                .>>>>> 0          $IF      1
6215   <macro>                      $IF (R!TYPE = DSK$DD40)                                    R13739AP.113
                                                                                                .>>>>> 1          $IF      1
6221   <macro>                        STORE 1,DK@SYN,R!DCB       .Set read mode                R13739AP.114
6232   <macro>                      $ENDIF                                                     R13739AP.115
                                                                                                .<<<<< 1          $ENDIF   1
6232  020446                        A = R!SCO                    .Parameter                      D4DEM.1389
6233  162000                        IOB: 2                       .Read                           D4DEM.1390
6234   <macro>                      STORE 1,DKF@RD,R!DCB         .Set read flag                  D4DEM.1391
6241   <macro>                    $ELSE                                                          D4DEM.1392
6242   <macro>                      $IF (R!TYPE = DSK$DD40)                                    R13739AP.116
                                                                                                .>>>>> 1          $IF      1
6246   <macro>                        ADDRESS R!%W1,DK@OWC,R!DCB                               R13739AP.117
6251  030423  013060                  A = (R!%W1) - D40$SPT      .Advance outstanding write count R13739AP.118
6253  102002  036423                  (R!%W1) = (R!%W1) + 1, A # 0                             R13739AP.119
6255   <macro>                        STORE 2,DK@SYN,R!DCB       .Set write mode               R13739AP.120
6266   <macro>                      $ENDIF                                                     R13739AP.121
                                                                                                .<<<<< 1          $ENDIF   1
6266  020446                        A = R!SCO                    .Parameter                      D4DEM.1393
6267  163000                        IOB: 3                       .Write                          D4DEM.1394
6270   <macro>                      STORE 1,DKF@WR,R!DCB         .Set write flag                 D4DEM.1395
6276   <macro>                    $ENDIF                                                         D4DEM.1396
                                                                                                .<<<<< 0          $ENDIF   1
6276   <macro>                    PUT R!SCO,DK@ACC,R!DCB         .Save accumulator parameter     D4DEM.1397
6303   <macro>                    PUT R!SEC,DK@SEC,R!DCB         .Save current sector            D4DEM.1398
6316   <macro>                    PUT R!HED,DK@HED,R!DCB         .Save current head group        R13739AP.122
6332  010002  024427              R!%W5 = D4IOLIM                .Timeout limit                  D4DEM.1399
6334  073256                      R = TIMER                      .Put on timer queue             D4DEM.1400
6335  072220                      R = TRACE                      .Trace I/O                      D4DEM.1401
6336  020450  023452  024510      R!NUM = R!TOTAL - R!DONE       .Number sectors left            D4DEM.1402
6341   <macro>                    $UNTIL (R!NUM >= 2)            .Find DAL 2 sectors ahead       D4DEM.1403
6344  030433  024433                R!DAL = (R!DAL)              .Next                           D4DEM.1404
6346  102162                        P = NODAL, A = 0             .If no more DAL's               D4DEM.1405
6347   <macro>                      GET A,DL@CON,R!DAL           .Continue flag                  D4DEM.1406
6355  102153                        P = NODAL, A = 0             .If not contiguous              D4DEM.1407
6356   <macro>                      GET A,DL@TOT,R!DAL           .Total sectors                  D4DEM.1408
6362  025510                        R!NUM = A + R!NUM            .Sum                            D4DEM.1409
6363   <macro>                    $ENDTIL                                                        D4DEM.1410
6364   <macro>                    $IF (R!%HISP = 0),AND,(0 = R!FNC & 1) .If write, no HISP       D4DEM.1411
                                                                                                .>>>>> 0          $IF      1
6376   <macro>                      GET A,DL@TDL,R!DAL           .Transfer data location         D4DEM.1412
```

```
6404   013002  103123                P = NODAL, A # BP$DONE      .If data not available yet          D4DEM.1413
6406    <macro>                    $ENDIF                                                          D4DEM.1414
                                                                                  .<<<<< 0          $ENDIF    1
6406                        CONTINUE *                                                             D4DEM.1415
6406   002000  000000              I = 0                                                           D4DEM.1416
6410    <macro>                    $IF (IOB # BZ),OR,(IOB = DN)  .If not active                    D4DEM.1417
                                                                                  .>>>>> 0          $IF       1
6415   070111                        P = CONT1                   .Missed continue                  D4DEM.1418
6416    <macro>                    $ELSE                                                           D4DEM.1419
6417    <macro>                      GET R!%W1,DK@ACT,R!DCB      .Channel activity                 D4DEM.1420
6421    <macro>                      GET R!%W2,DKS@CR,R!DCB      .Create error rec. flag           D4DEM.1421
6427    <macro>                      $IF (R!%W2 # 0),OR,(0 # R!%W1 & O'177770)                     D4DEM.1422
                                                                                  .>>>>> 1          $IF       1
6441   070065                          P = CONT1                 .Not a continue situation         D4DEM.1423
6442    <macro>                      $ENDIF                                                          D4DEM.1424
                                                                                  .<<<<< 1          $ENDIF    1
6442    <macro>                    $ENDIF                                                          D4DEM.1425
                                                                                  .<<<<< 0          $ENDIF    1
6442   077000 /007565              R = NEXTBUF                 .Next LMA entry                   D4DEM.1426
6444   026445                      R!SEC = R!SEC + 1           .Next sector                      D4DEM.1427
6445    <macro>                    $IF (A = R!SPT)             .If end of track                  D4DEM.1428
                                                                                  .>>>>> 0          $IF       1
6450   026444                        R!HED = R!HED + 1         .Next track                       D4DEM.1429
6451   023472  102054                P = CONT1, A = R!TPC      .If seek needed                   D4DEM.1430
6453   010000  024445                R!SEC = 0                 .Reset                            D4DEM.1431
6455    <macro>                    $ENDIF                                                          D4DEM.1432
                                                                                  .<<<<< 0          $ENDIF    1
6455    <macro>                    GET R!%W1,LB@FUL,R!ENTRY    .Buffer full flag                 D4DEM.1433
6463    <macro>                    $IF (R!%W1 = R!FNC & 1)     .If buffer not ready              D4DEM.1434
                                                                                  .>>>>> 0          $IF       1
6472    <macro>                      STORE 1,LB@CP,R!ENTRY     .Set continue pending             D4DEM.1435
6503   070023                        P = CONT1                                                     D4DEM.1436
6504    <macro>                    $ENDIF                                                          D4DEM.1437
                                                                                  .<<<<< 0          $ENDIF    1
6504   072101                      R = LMA                     .Enter LMA and get parameter      D4DEM.1438
6505   020446                      A = R!SCO                   .Enter for continue               D4DEM.1439
6506   176000                      IOB:16                                                        D4DEM.1440
6507    <macro>                    PUT R!SCO,DK@ACC,R!DCB      .Save parameter                   D4DEM.1441
6514    <macro>                    STORE 1,DK@CNG,R!DCB        .Set chaining on                  D4DEM.1442
6525   072030                      R = TRACE                   .Trace event                      D4DEM.1443
6526                        CONT1  *                                                              D4DEM.1444
6526   003000                      I = 1                                                         D4DEM.1445
```

```
6527   001000                          EXIT                                                          D4DEM.1446
                                                                                                     D4DEM.1447
6530                           NODAL   *                                                             D4DEM.1448
6530    <macro>                        $IF (0 # R!FNC & 1),AND,(R!DAL = 0) .If time for R.A.         D4DEM.1449
                                                                                .>>>>> 0                $IF      1
6542    <macro>                          GET A,DK@RCC,R!DCB          .Read ahead limit                D4DEM.1450
6547    <macro>                          $IF (A # 0)                 .If some to do                   D4DEM.1451
                                                                                .>>>>> 1                $IF      1
6551   010001  024451                      R!FNC = DAF$READ          .Set function code               D4DEM.1452
6553   071145                              P = CONTINUE              .Continue I/O                    D4DEM.1453
6554    <macro>                          $ENDIF                                                        D4DEM.1454
                                                                                .<<<<< 1                $ENDIF   1
6554    <macro>                        $ENDIF                                                          D4DEM.1455
                                                                                .<<<<< 0                $ENDIF   1
6554   001000                          EXIT                                                          D4DEM.1456
                                                                                                     D4DEM.1457
                               ***********************************************************************D4DEM.1458
                               *                                                                     *D4DEM.1459
                               *       TRACE:     Trace disk I/O.                                    *D4DEM.1460
                               *                                                                     *D4DEM.1461
                               ***********************************************************************D4DEM.1462
                                                                                                     D4DEM.1463
6555                           TRACE   *                                                             D4DEM.1464
6555   020451  005006  052000          R!%W4 = R!FNC < 6 + B                                         D4DEM.1465
       024426
6561   020466  005014  022443          R!%W3 = R!UNIT < D'12 + R!CYL                                 D4DEM.1466
       024425
6565    <macro>                        KRECORD TF$D410,(R!%W4,R!%W3,R!HED,R!SEC,R!SC1)               D4DEM.1467
                                                 *  Function (10)/Channel (6)                        D4DEM.1468
                                                 *  Unit (4)/Cylinder (12)                           D4DEM.1469
                                                 *  Head group                                       D4DEM.1470
                                                 *  Sector                                           D4DEM.1471
                                                 *  Local memory address                             D4DEM.1472
6604   001000                          EXIT                                                          D4DEM.1473
                                                                                                     D4DEM.1474
                               ***********************************************************************D4DEM.1475
                               *                                                                     *D4DEM.1476
                               *       LMA:       Set up Local Memory address register and get       *D4DEM.1477
                               *                  accumulator parameter for next read/write to disk. *D4DEM.1478
                               *                                                                     *D4DEM.1479
                               *       Entry:     R!ENTRY = Local buffer descriptor                  *D4DEM.1480
                               *                  R!DCB = DCB address                                *D4DEM.1481
```

```
                                *                B = Channel                                        *   D4DEM.1482
                                *                R!SEC = Desired sector                             *   D4DEM.1483
                                *                R!HED = Desired head                               *   D4DEM.1484
                                *                R!SUB = Subfunction code                           *   D4DEM.1485
                                *                R!SPT = Sectors per track                          *   D4DEM.1486
                                *                R!TPC = Tracks per cylinder                        *   D4DEM.1487
                                *                R!DCB = DCB address                                *   D4DEM.1488
                                *                R!TYPE = Device type                               *R13739AP.123
                                *                                                                   *   D4DEM.1489
                                *        Exit:   R!SC0 = Parameter                                  *   D4DEM.1490
                                *                R!SC1 = Local buffer address                       *   D4DEM.1491
                                *                                                                   *   D4DEM.1492
                                ***********************************************************************   D4DEM.1493
                                                                                                          D4DEM.1494
6605                            LMA      *                                                                D4DEM.1495
6605    <macro>                          GET R!SC1,LB@LOC,R!ENTRY      .Local memory address              D4DEM.1496
6612    <macro>                          $PUNTIF (A = 0),CODE=PT$DD49  .If buffer not allocated           D4DEM.1497
6614    <macro>                          FLDADD 1,Dk@LMA,R!DCB         .Complement register to use        D4DEM.1498
6637    <macro>                          $IF (A # 0)                   .If LMA register 0                 D4DEM.1499
                                                                                        .>>>>> 0              $IF      1
6641    020447                             A = R!SC1                   .Address                           D4DEM.1500
6642    174000                             IOB: 14                     .Enter                             D4DEM.1501
6643    <macro>                          $ELSE                                                            D4DEM.1502
6644    020447                             A = R!SC1                   .Address                           D4DEM.1503
6645    175000                             IOB: 15                     .Enter                             D4DEM.1504
6646    <macro>                          $ENDIF                                                           D4DEM.1505
                                                                                        .<<<<< 0              $ENDIF   1
6646    <macro>                          $IF (R!TYPE = DSK$DD40)       .If DD40                       R13739AP.124
                                                                                        .>>>>> 0              $IF      1
6652    020444  005007  024446             R!SC0 = R!HED < 7           .Current head                  R13739AP.125
6655    020445  013044                     A = R!SEC - D'36            .Convert sector to 2/drive, 4/secR13739AP.126
6657    101007                             P = LMA1, C = 1             .If greater than 35            R13739AP.127
6660    012014                             A = A + D'12                                               R13739AP.128
6661    101006                             P = LMA2, C = 1             .If greater than 23            R13739AP.129
6662    012014                             A = A + D'12                                               R13739AP.130
6663    101005                             P = LMA3, C = 1             .If greater than 11            R13739AP.131
6664    012014                             A = A + D'12                .Make 0 .. 11                  R13739AP.132
6665    070004                             P = LMA4                                                   R13739AP.133
                                                                                                       R13739AP.134
6666    012020                  LMA1       A = A + 1S4                 .[36 .. 47] -> 3S4 + [0 .. 11] R13739AP.135
6667    012020                  LMA2       A = A + 1S4                 .[24 .. 35] -> 2S4 + [0 .. 11] R13739AP.136
6670    012020                  LMA3       A = A + 1S4                 .[12 .. 23] -> 1S4 + [0 .. 11] R13739AP.137
```

```
6671   025446                    LMA4      R!SCO = A + R!SCO              .[ 0 .. 11] -> 0S4 + [0 .. 11]   R13739AP.138
6672    <macro>                           $ELSE                          .Not DD40                        R13739AP.139
6673   020444  024446                      R!SCO = R!HED                 .Current head                    R13739AP.140
6675    <macro>                            $IF (R!SPT = R!SEC + 1)       .If end of track                 R13739AP.141
                                                                                          .>>>>> 1              $IF      1
6704   026446                               R!SCO = R!SCO + 1            .Next head                       R13739AP.142
6705   023472                               A = A - R!TPC                                                 R13739AP.143
6706   103002  024446                       R!SCO = A, A = 0             .If end of cylinder              R13739AP.144
6710    <macro>                            $ENDIF                                                        R13739AP.145
                                                                                          .<<<<< 1              $ENDIF   1
6710    <macro>                            GET A,DK@SPO,R!DCB            .Get spiral offset               R13739AP.146
6714   022445  024423                      R!%W1 = A + R!SEC             .Physical sector                 R13739AP.147
6716   023473                              A = A - R!SPT                                                  R13739AP.148
6717   100002  024423                      R!%W1 = A, C = 1              .Check wrap                      R13739AP.149
6721   020446  005010  022423              R!SCO = R!SCO < D'8 + R!%W1 .Combine head and sector          R13739AP.150
       024446
6725    <macro>                           $ENDIF                                                         R13739AP.151
                                                                                          .<<<<< 0              $ENDIF   1
6725   020475  005014  022446             R!SCO = R!SUB < D'12 + R!SCO  .Include subfunction code          D4DEM.1516
       024446
6731   001000                             EXIT                                                            D4DEM.1517
                                                                                                          D4DEM.1518
                                 ********************************************************************     D4DEM.1519
                                 *                                                                  *     D4DEM.1520
                                 *        FILL:     Attempts to keep all dedicated local buffers full *     D4DEM.1521
                                 *                  on write requests.  Enter with interrupts off.    *     D4DEM.1522
                                 *                                                                  *     D4DEM.1523
                                 *        Entry:    R!DCB = DCB address                               *     D4DEM.1524
                                 *                  R!DAL = Starting DAL address                      *     D4DEM.1525
                                 *                  R!SEQ = MEMIO sequence number                     *     D4DEM.1526
                                 *                  R!TOTAL = Total sectors for request               *     D4DEM.1527
                                 *                  R!DONE = Sectors done for request                 *     D4DEM.1528
                                 *                                                                  *     D4DEM.1529
                                 ********************************************************************     D4DEM.1530
                                                                                                          D4DEM.1531
6732                             FILL      *                                                              D4DEM.1532
                                                                                                          D4DEM.1533
                                 *         Locate first write DAL which needs data.                       D4DEM.1534
                                                                                                          D4DEM.1535
6732   020451  011001  024410             EXIT, 0 # R!FNC & 1           .If read request                 D4DEM.1536
       010000  023410  102002
       001000
```

```
6741   010000  024446                    R!SCO = 0                          .Initialize                          D4DEM.1537
6743    <macro>                          $UNTIL (R!SEQ # R!TOTAL)           .Find first incomplete DAL           D4DEM.1538
6746   020446  022453  023452             R!SCO = R!SCO + R!SEQ - R!DONE .Sum of buffers in use                   D4DEM.1539
       024446
6752   030433  024433                     R!DAL = (R!DAL)                    .Next                                D4DEM.1540
6754   103002  001000                     EXIT, A = 0                        .If no more                          D4DEM.1541
6756   077000 /007274                     R = DALPAR                         .Get its parameters                  D4DEM.1542
6760   020451  011001  024410             EXIT, 0 # R!FNC & 1                .If read request                     D4DEM.1543
       010000  023410  102002
       001000
6767    <macro>                          $ENDTIL                                                                 D4DEM.1544
6770   020446  022453  023452             R!SCO = R!SCO + R!SEQ - R!DONE                                          D4DEM.1545
       024446
6774    <macro>                          GET A,DK@NLB,R!DCB                 .Number of local buffers             D4DEM.1546
7001   041000  023446  100003             EXIT, A <= R!SCO                   .If all full                         D4DEM.1547
       102002  070002  001000
                                                                                                                   D4DEM.1548
                                *        Fill remaining empty buffers.                                           D4DEM.1549
                                                                                                                   D4DEM.1550
7007    <macro>                          $IF (R!TMP # R!%MYID)              .Path not on this IOP                D4DEM.1551
                                                                                                  .>>>>> 0         $IF      1
7014    <macro>                            GET R!%W1,DA@IFC,R!DAL           .Internal function                   D4DEM.1552
7021    <macro>                            $IF (R!%W1 = IM$RESP),OR,(R!%W1 = IM$STAT) .Data in MOS               D4DEM.1553
                                                                                                  .>>>>> 1         $IF      1
7030    <macro>                              STORE BP$DONE,DL@TDL,R!DAL .Set DTR done status                      D4DEM.1554
7041    <macro>                            $ELSE                                                                 D4DEM.1555
7042   001000                                EXIT                                                                 D4DEM.1556
7043    <macro>                            $ENDIF                                                                D4DEM.1557
                                                                                                  .<<<<< 1         $ENDIF   1
7043    <macro>                          $ENDIF                                                                  D4DEM.1558
                                                                                                  .<<<<< 0         $ENDIF   1
7043    <macro>                          GET R!ENTRY,DK@CLB,R!DCB           .Current local buffer                D4DEM.1559
7050   024427                            R!%W5 = A                          .Initialize                          D4DEM.1560
7051    <macro>                          GET A,LB@LOC,R!ENTRY               .Local buffer address                D4DEM.1561
7055   103002  001000                    EXIT, A = 0                        .If buffers not allocated            D4DEM.1562
7057                            FILLUP   *                                                                        D4DEM.1563
7057    <macro>                          $IF (R!SEQ = R!TOTAL)              .If this DAL complete                D4DEM.1564
                                                                                                  .>>>>> 0         $IF      1
7063   030433  024433                     R!DAL = (R!DAL)                    .Next                                D4DEM.1565
7065   103002  001000                     EXIT, A = 0                        .If no more                          D4DEM.1566
7067   072205                             R = DALPAR                         .Get its parameters                  D4DEM.1567
7070   020451  011001  024410             EXIT, 0 # R!FNC & 1                .If read request                     D4DEM.1568
```

```
         010000  023410  102002
         001000
7077     <macro>                    $IF (R!IMP # R!%MYID)        .Path not on this IOP                D4DEM.1569
                                                                                   .>>>>> 1              $IF      1
7103     <macro>                      GET R!%W1,DA@IFC,R!DAL     .Internal function                   D4DEM.1570
7110     <macro>                      $IF (R!%W1 = IM$RESP),OR,(R!%W1 = IM$STAT) .Data in MOS         D4DEM.1571
                                                                                   .>>>>> 2              $IF      1
7117     <macro>                        STORE BP$DONE,DL@TDL,R!DAL .Set DTR done status                D4DEM.1572
7130     <macro>                      $ELSE                                                            D4DEM.1573
7131     001000                         EXIT                                                           D4DEM.1574
7132     <macro>                      $ENDIF                                                           D4DEM.1575
                                                                                   .<<<<< 2              $ENDIF   1
7132     <macro>                    $ENDIF                                                             D4DEM.1576
                                                                                   .<<<<< 1              $ENDIF   1
7132     <macro>                  $ENDIF                                                               D4DEM.1577
                                                                                   .<<<<< 0              $ENDIF   1
7132     <macro>                  $IF (R!SCO = 0)              .If buffer empty                        D4DEM.1578
                                                                                   .>>>>> 0              $IF      1
7135     026453                     R!SEQ = R!SEQ + 1          .Increment sequence number              D4DEM.1579
7136     <macro>                    PUT R!SEQ,DL@SEQ,R!DAL     .Save in MDAL                           D4DEM.1580
7143     <macro>                    PUT R!SEQ,LB@SEQ,R!ENTRY   .Save in LMA entry                      D4DEM.1581
7150     <macro>                    PUT R!DAL,LB@DAL,R!ENTRY   .Originating MDAL                       D4DEM.1582
7155     <macro>                    ADDRESS R!ED,MEM@BM,R!%MEMIO .Assume MOS queue                     D4DEM.1583
7160     <macro>                    $IF (R!FNC # DAF$INIW),AND,(R!TMP = R!%MYID) .Path present         D4DEM.1584
                                                                                   .>>>>> 1              $IF      1
7170     <macro>                      $IF (R!SEQ <= R!TOTAL - R!WBC) .If data not in MOS               D4DEM.1585
                                                                                   .>>>>> 2              $IF      1
7201     <macro>                        $IF (R!TM = TM$CMEM)       .Target is central                  D4DEM.1586
                                                                                   .>>>>> 3              $IF      1
7204     <macro>                          ADDRESS R!ED,MEM@HI,R!%MEMIO .High speed in queue            D4DEM.1587
7207     <macro>                        $ELSEIF (R!TM = TM$SSD)    .Target is SSD                      D4DEM.1588
7214     <macro>                          ADDRESS R!ED,MEM@SI,R!%MEMIO .SSD in queue                   D4DEM.1589
7217     <macro>                        $ENDIF                                                         D4DEM.1590
                                                                                   .<<<<< 3              $ENDIF   1
7217     <macro>                      $ENDIF                                                           D4DEM.1591
                                                                                   .<<<<< 2              $ENDIF   1
7217     <macro>                    $ENDIF                                                             D4DEM.1592
                                                                                   .<<<<< 1              $ENDIF   1
7217     <macro>                    $IF (R!FNC = DAF$PTRN)       .If patterning                        D4DEM.1593
                                                                                   .>>>>> 1              $IF      1
7223     <macro>                      STORE 1,LB@FUL,R!ENTRY     .Set Buffer Full flag                 D4DEM.1594
7233     <macro>                    $ELSE                                                              D4DEM.1595
```

```
7234   020442  024000                    R!EH = R!ENTRY          .Local descriptor                D4DEM.1596
7236    <macro>                          RKERN ENQU              .Put on queue                    D4DEM.1597
7240    <macro>                        $ENDIF                                                     D4DEM.1598
                                                                                .<<<<< 1              $ENDIF    1
7240    <macro>                      $ELSE                                                        D4DEM.1599
7241   027446                          R!SCO = R!SCO - 1       .Keep count                      D4DEM.1600
7242    <macro>                      $ENDIF                                                       D4DEM.1601
                                                                                .<<<<< 0              $ENDIF    1
7242   072323                        R = NEXTBUF               .Next buffer                     D4DEM.1602
7243   020442  023427  107166        P = FILLUP, R!ENTRY # R!%W5  .If more to do                D4DEM.1603
7246   001000                        EXIT                                                         D4DEM.1604
                                                                                                   D4DEM.1605
                                 ***********************************************************       D4DEM.1606
                                 *                                                         *       D4DEM.1607
                                 *       STRIP:    Get parameters from specified MDAL.     *       D4DEM.1608
                                 *                                                         *       D4DEM.1609
                                 *       Entry:    R!DAL   = MDAL address                 *       D4DEM.1610
                                 *                 R!DCB = DCB address                     *       D4DEM.1611
                                 *                                                         *       D4DEM.1612
                                 ***********************************************************       D4DEM.1613
                                                                                                   D4DEM.1614
7247                             STRIP    *                                                       D4DEM.1615
7247   072104                        R = DALADD                .Get disk address from DAL       D4DEM.1616
7250   072024                        R = DALPAR                                                   D4DEM.1617
7251   072220                        R = LIMITS                .Get device limits               D4DEM.1618
7252   020445  022452  024445        R!SEC = R!SEC + R!DONE    .Calculate current address       D4DEM.1619
7255                             CALC     *                                                       D4DEM.1620
7255    <macro>                      $UNTIL (R!SEC < R!SPT)                                       D4DEM.1621
7260   020445  023473  024445          R!SEC = R!SEC - R!SPT   .Decrement                       D4DEM.1622
7263   026444                          R!HED = R!HED + 1       .Next track                      D4DEM.1623
7264    <macro>                        $IF (A = R!TPC)         .If end of cylinder              D4DEM.1624
                                                                                .>>>>> 0              $IF       1
7267   010000  024444                    R!HED = 0             .First track                     D4DEM.1625
7271   026443                            R!CYL = R!CYL + 1     .Next cylinder                   D4DEM.1626
7272    <macro>                        $ENDIF                                                     D4DEM.1627
                                                                                .<<<<< 0              $ENDIF    1
7272    <macro>                      $ENDTIL                                                      D4DEM.1628
7273   001000                        EXIT                                                         D4DEM.1629
                                                                                                   D4DEM.1630
                                 ***********************************************************       D4DEM.1631
                                 *                                                         *       D4DEM.1632
                                 *       DALPAR:   Get function and counts from specified DAL.     D4DEM.1633
```

```
                                *                                                       *   D4DEM.1634
                                *       Entry:    R!DAL = DAL address                   *   D4DEM.1635
                                *                                                       *   D4DEM.1636
                                ***********************************************************   D4DEM.1637
                                                                                              D4DEM.1638
7274                            DALPAR  *                                                     D4DEM.1639
7274    <macro>                         GET R!FNC,DL@FC,R!DAL          .Function code         D4DEM.1640
7302    <macro>                         GET R!TOTAL,DL@TOT,R!DAL       .Total sectors         D4DEM.1641
7307    <macro>                         GET R!DONE,DL@DON,R!DAL        .Sectors done          D4DEM.1642
7314    <macro>                         GET R!WBC,DL@RAC,R!DAL         .Write behinds done    D4DEM.1643
7322    <macro>                         GET R!SEQ,DL@SEQ,R!DAL         .MEMIO sequence number D4DEM.1644
7327    <macro>                         GET R!TM,DL@TM,R!DAL           .Target memory type    D4DEM.1645
7335    020511  016000 /000000          R!TMP = (R!TM + TMCB)          .Pointer to control table D4DEM.1646
        024410  030410  024512
7343    <macro>                         $PUNTIF (A = 0),CODE=PT$BPATH .If path not defined    D4DEM.1647
7345    <macro>                         GET R!TMP,TM@IOP,R!TMP         .IOP with high speed channel D4DEM.1648
7352    001000                          EXIT                                                  D4DEM.1649
                                                                                              D4DEM.1650
                                ***********************************************************   D4DEM.1651
                                *                                                       *   D4DEM.1652
                                *       DALADD:   Get starting disk address from DAL.         *   D4DEM.1653
                                *                                                       *   D4DEM.1654
                                *       Entry:    R!DAL = DAL address                   *   D4DEM.1655
                                *                                                       *   D4DEM.1656
                                ***********************************************************   D4DEM.1657
                                                                                              D4DEM.1658
7353                            DALADD  *                                                     D4DEM.1659
7353    <macro>                         GET R!UNIT,DL@UNT,R!DAL        .Unit number           D4DEM.1660
7362    <macro>                         GET R!CYL,DL@CYL,R!DAL         .Cylinder              D4DEM.1661
7370    <macro>                         GET R!HED,DL@HED,R!DAL         .Head group            D4DEM.1662
7376    <macro>                         GET R!SEC,DL@SEC,R!DAL         .Sector                D4DEM.1663
7404    001000                          EXIT                                                  D4DEM.1664
                                                                                              D4DEM.1665
                                ***********************************************************   D4DEM.1666
                                *                                                       *   D4DEM.1667
                                *       LBPAR:    Get parameters from specified Local Buffer entry.  *   D4DEM.1668
                                *                                                       *   D4DEM.1669
                                *       Entry:    R!LOCAL = Local Buffer entry address  *   D4DEM.1670
                                *                                                       *   D4DEM.1671
                                ***********************************************************   D4DEM.1672
                                                                                              D4DEM.1673
7405                            LBPAR   *                                                     D4DEM.1674
```

```
7405    <macro>                 GET R!SEQ,LB@SEQ,R!LOCAL      .Sequence number                    D4DEM.1675
7412    <macro>                 GET R!DAL,LB@DAL,R!LOCAL      .MDAL address                       D4DEM.1676
7417    <macro>                 GET R!DCB,LB@DCB,R!LOCAL      .DCB address                        D4DEM.1677
7424    <macro>                 $IF (R!DAL # 0)                                                     D4DEM.1678
                                                                          .>>>>> 0                     $IF      1
7427    <macro>                   GET R!TM,DL@TM,R!DAL        .Target memory from DAL             D4DEM.1679
7435    <macro>                 $ELSE                                                               D4DEM.1680
7436    <macro>                   GET R!TM,DK@TM,R!DCB        .Target memory from last I/O        D4DEM.1681
7444    <macro>                 $ENDIF                                                              D4DEM.1682
                                                                          .<<<<< 0                     $ENDIF   1
7444   020511  016000 /000000   R!TMP = (R!TM + TMCB)         .Point at target memory table       D4DEM.1683
       024410  030410  024512
7452    <macro>                 $PUNTIF (A = 0),CODE=PT$BPATH .If path not defined                D4DEM.1684
7454    <macro>                 GET R!TMP,TM@IOP,R!TMP        .Get IOP with channel               D4DEM.1685
7461    <macro>                 GET R!FLG,LB@IOI,R!LOCAL      .I/O initiated flag                 D4DEM.1686
7470   001000                   EXIT                                                                D4DEM.1687
                                                                                                     D4DEM.1688
                        ***********************************************************************  D4DEM.1689
                        *                                                                     *  D4DEM.1690
                        *       LIMITS:   Get device limits for this request.                 *  D4DEM.1691
                        *                                                                     *  D4DEM.1692
                        *       Entry:    R!DAL   = MDAL address                              *  D4DEM.1693
                        *                 R!DCB = DCB address                                 *  D4DEM.1694
                        *                                                                     *  D4DEM.1695
                        ***********************************************************************  D4DEM.1696
                                                                                                     D4DEM.1697
7471                    LIMITS  *                                                                    D4DEM.1698
7471    <macro>                 GET R!DPT,DK@DPT,R!DCB        .Device parameter table             D4DEM.1699
7476    <macro>                 GET R!SUB,DL@SUB,R!DAL        .Subfunction code                   D4DEM.1700
7505    <macro>                 $IF (R!SUB = DAS$ID),OR,(R!SUB = DAS$ABS) .If absolute mode       D4DEM.1701
                                                                          .>>>>> 0                     $IF      1
7514    <macro>                   GET R!SPT,DP@ASE,R!DPT      .Absolute sectors/track             D4DEM.1702
7522    <macro>                 $ELSE                                                               D4DEM.1703
7523    <macro>                   GET R!SPT,DP@DSE,R!DPT      .Default sectors/track              D4DEM.1704
7531    <macro>                 $ENDIF                                                              D4DEM.1705
                                                                          .<<<<< 0                     $ENDIF   1
7531    <macro>                 GET R!TPC,DK@TPC,R!DCB        .Tracks/cylinder                    D4DEM.1706
7540    <macro>                 GET R!DIA,DL@DIA,R!DAL        .Diagnostic mode flag               D4DEM.1707
7546    <macro>                 $IF (A = 1)                   .If diagnostic request              D4DEM.1708
                                                                          .>>>>> 0                     $IF      1
7551    <macro>                   GET R!CPD,DP@ACY,R!DPT      .Absolute cylinders/disk            D4DEM.1709
7556    <macro>                 $ELSE                                                               D4DEM.1710
```

```
7557    <macro>                 GET R!CPD,DP@DCY,R!DPT      .Default cylinders/disk          D4DEM.1711
7564    <macro>               $ENDIF                                                          D4DEM.1712
                                                                          .<<<<< 0                $ENDIF   1
7564   001000                 EXIT                                                             D4DEM.1713
                                                                                                D4DEM.1714
                        ********************************************************************    D4DEM.1715
                        *                                                                  *    D4DEM.1716
                        *      NEXTBUF:  Locates the local buffer entry following the      *    D4DEM.1717
                        *                current entry.                                    *    D4DEM.1718
                        *                                                                  *    D4DEM.1719
                        *      Entry:    R!DCB   = DCB address                             *    D4DEM.1720
                        *                R!ENTRY = Current entry                           *    D4DEM.1721
                        *                                                                  *    D4DEM.1722
                        *      Exit:     R!ENTRY = Next entry                              *    D4DEM.1723
                        *                                                                  *    D4DEM.1724
                        ********************************************************************    D4DEM.1725
                                                                                                D4DEM.1726
7565                    NEXTBUF *                                                               D4DEM.1727
7565    <macro>               GET A,DK@LLB,R!DCB          .Last buffer entry                    D4DEM.1728
7571    <macro>               $IF (A = R!ENTRY)           .If at end of table                   D4DEM.1729
                                                                          .>>>>> 0                $IF      1
7574    <macro>                 GET R!ENTRY,DK@FLB,R!DCB    .Wrap to top                      D4DEM.1730
7601    <macro>               $ELSE                                                             D4DEM.1731
7602   010006  025442           R!ENTRY = R!ENTRY + LB@LE   .Next buffer entry                D4DEM.1732
7604    <macro>               $ENDIF                                                            D4DEM.1733
                                                                          .<<<<< 0                $ENDIF   1
7604   001000                 EXIT                                                             D4DEM.1734
                                                                                                D4DEM.1735
                        ********************************************************************    D4DEM.1736
                        *                                                                  *    D4DEM.1737
                        *      SENDTR:   Builds and sends Data Transfer Request to TRANSFR *    D4DEM.1738
                        *                in IOP with hi speed channel.                     *    D4DEM.1739
                        *                                                                  *    D4DEM.1740
                        *      Entry:    R!DAL   = Original disk request                   *    D4DEM.1741
                        *                R!DCB   = DCB address                             *    D4DEM.1742
                        *                R!FNC   = MDAL function code                      *    D4DEM.1743
                        *                R!IMP = Target memory IOP                         *    D4DEM.1744
                        *                                                                  *    D4DEM.1745
                        *      Exit:     R!STATUS = status of data availability            *    D4DEM.1746
                        *                                                                  *    D4DEM.1747
                        ********************************************************************    D4DEM.1748
                                                                                                D4DEM.1749
```

```
 7605                          SENDTR   *                                                          D4DEM.1750
 7605   010000  024434                  R!STATUS = 0                  .Initial status, data available  D4DEM.1751
 7607   020512  023000  103002          EXIT, R!TMP = R!%MYID         .If channel present              D4DEM.1752
        001000
 7613   020451  011020  024410          EXIT, 0 # R!FNC & O'20        .If internal request             D4DEM.1753
        010000  023410  102002
        001000
 7622    <macro>                        $IF (R!FNC = DAF$PIRN)        .If pattern request              D4DEM.1754
                                                                                     .>>>>> 0          $IF      1
 7626    <macro>                          STORE IM$RESP,DA@IFC,R!DAL  .Indicate data available         D4DEM.1755
 7633   001000                            EXIT                                                        D4DEM.1756
 7634    <macro>                        $ENDIF                                                       D4DEM.1757
                                                                                     .<<<<< 0          $ENDIF   1
 7634    <macro>                        GET R!%W1,DL@TDL,R!DAL        .Transfer data location          D4DEM.1758
 7643   013002  103002  001000          EXIT, A = BP$DONE             .If data available               D4DEM.1759
 7646    <macro>                        GET R!%W2,DA@IFC,R!DAL        .Internal function               D4DEM.1760
 7653    <macro>                        $IF (R!%W2 = IM$RESP),OR,(R!%W2 = IM$STAT) .Data in MOS        D4DEM.1761
                                                                                     .>>>>> 0          $IF      1
 7662    <macro>                          STORE BP$DONE,DL@TDL,R!DAL  .Set DTR done status             D4DEM.1762
 7673   001000                            EXIT                                                        D4DEM.1763
 7674    <macro>                        $ENDIF                                                       D4DEM.1764
                                                                                     .<<<<< 0          $ENDIF   1
 7674   010001  024434                  R!STATUS = 1                  .Set data unavailable status     D4DEM.1765
 7676    <macro>                        $IF (R!%W1 = BP$NONE)         .If no DTR sent yet              D4DEM.1766
                                                                                     .>>>>> 0          $IF      1
 7702    <macro>                          GET R!%W2,DK@DLO,R!DCB      .First DAL on queue              D4DEM.1767
 7707    <macro>                          $IF (A # R!DAL)             .If not the only one             D4DEM.1768
                                                                                     .>>>>> 1          $IF      1
 7712    <macro>                            $UNTIL ((R!%W2) = R!DAL)  .Search for preceding DAL        D4DEM.1769
 7715   030424  024424                      R!%W2 = (R!%W2)           .Next                            D4DEM.1770
 7717    <macro>                            $PUNTIF (A = 0),CODE=PT$DD49 .Not found                    D4DEM.1771
 7721    <macro>                          $ENDTIL                                                     D4DEM.1772
 7722    <macro>                          GET A,DA@IFC,R!%W2          .Internal fnc code               D4DEM.1773
 7726   013005  102002  001000            EXIT, A # IM$STAT           .If request not done             D4DEM.1774
 7731    <macro>                          $ENDIF                                                      D4DEM.1775
                                                                                     .<<<<< 1          $ENDIF   1
 7731   014000 /000000  024440            R!DTR = IMDAL               .Use immediate DAL               D4DEM.1776
 7734    <macro>                          CLEAR START=R!DTR,COUNT=DA@@LH .Clear header                D4DEM.1777
 7747    <macro>                          COPY R!DAL+DA@@LH,R!DTR+DA@@LH,DA@@LE .Copy DAL            D4DEM.1778
 7767    <macro>                          STORE IM$TRAN,DA@IFC,R!DTR  .Internal fnc. code              D4DEM.1779
 7774    <macro>                          PUT R!DAL,TR@MDL,R!DTR      .Originating DAL                 D4DEM.1780
10001    <macro>                          STORE 0,TR@DON,R!DTR        .Clear sectors done              D4DEM.1781
```

```
10006    <macro>                    GET R!%W1,DK@MC0,R!DCB       .MCB address (upper)                D4DEM.1782
10013    <macro>                    GET R!%W2,DK@MC1,R!DCB       .(lower)                            D4DEM.1783
10020    <macro>                    PUT R!%W1,TR@MC0,R!DIR                                           D4DEM.1784
10025    <macro>                    PUT R!%W2,TR@MC1,R!DIR                                           D4DEM.1785
10032    <macro>                    STORE BP$SENT,DL@TDL,R!DAL   .Set 'DIR sent' code                D4DEM.1786
10043   002000  000000              I = 0                                                            D4DEM.1787
10045   020440  024000              R!DS = R!DIR                 .DIR address                        D4DEM.1788
10047    <macro>                    RKERN DDALGETA               .Get an MOS DAL                     D4DEM.1789
10051    <macro>                    RKERN  DSUM                  .Write DAL to MOS                   D4DEM.1790
10053   020512  024000              R!EI = R!IMP                 .Target memory IOP                  D4DEM.1791
10055    <macro>                    RKERN  EMSGIOP               .Send request to IOP                D4DEM.1792
10057   003000                      I = 1                                                            D4DEM.1793
10060    <macro>                  $ENDIF                                                             D4DEM.1794
                                                                                  .<<<<< 0                $ENDIF    1
10060   001000                    EXIT                                                               D4DEM.1795
                                                                                                     D4DEM.1796
                             ***********************************************************************  D4DEM.1797
                             *                                                                     *  D4DEM.1798
                             *        ACTDEM:    Puts DCB on queue and activates D4DEM.             *  D4DEM.1799
                             *                                                                     *  D4DEM.1800
                             *        Entry:     R!DCB   = DCB address                              *  D4DEM.1801
                             *                                                                     *  D4DEM.1802
                             ***********************************************************************  D4DEM.1803
                                                                                                     D4DEM.1804
10061                        ACTDEM   *                                                             D4DEM.1805
10061   002000  000000               I = 0                                                           D4DEM.1806
10063   020431  024000               R!DT = R!DCB                 .DCB address                       D4DEM.1807
10065    <macro>                     RKERN D4ACT                  .Put back on queue                 D4DEM.1808
10067   003000                       I = 1                                                           D4DEM.1809
10070   001000                       EXIT                                                            D4DEM.1810
                                                                                                     D4DEM.1811
                             ***********************************************************************  D4DEM.1812
                             *                                                                     *  D4DEM.1813
                             *        CHANCHK:   Waits for a DN on an MOS or HSP channel.  Upon    *  D4DEM.1814
                             *                   completion of I/O, dequeues LMA entry and sets    *  D4DEM.1815
                             *                   buffer ready.  Also checks for a disabled channel.*  D4DEM.1816
                             *                                                                     *  D4DEM.1817
                             *        Entry:     B = Channel number                                 *  D4DEM.1818
                             *                   R!QUE = Queue address for LMA entry                *  D4DEM.1819
                             *                                                                     *  D4DEM.1820
                             *        Exit:      R!LOCAL = LMA entry address                        *  D4DEM.1821
                             *                   R!DAL = MDAL address (0 if read ahead)             *  D4DEM.1822
```

```
                                 *                R!DCB = DCB address                                  *   D4DEM.1823
                                 *                R!SEQ = Sequence number                              *   D4DEM.1824
                                 *                                                                     *   D4DEM.1825
                                 ***********************************************************************   D4DEM.1826
                                                                                                           D4DEM.1827
10071                            CHANCHK  *                                                               D4DEM.1828
10071   002000  000000                    I = 0                                                           D4DEM.1829
10073   020461  024000                    R!ED = R!QUE                    .Remove entry from queue        D4DEM.1830
10075    <macro>                          RKERN EDQU                                                      D4DEM.1831
10077   020000  024506                    R!LOCAL = R!EH                  .Save address                   D4DEM.1832
10101    <macro>                          GET R!%W2,LB@LOC,R!LOCAL        .Buffer being checked           D4DEM.1833
10106    <macro>                          $UNTIL (IOB # BZ)               .Until channel quiet            D4DEM.1834
10110    <macro>                            $IF (B = [MOS])                                               D4DEM.1835
                                                                                          .>>>>> 0            $IF      1
10114   020000  024423                       R!%W1 = R!BMLOCL                                             D4DEM.1836
10116    <macro>                            $ELSEIF (B = R!%HISP)                                         D4DEM.1837
10123   020000  024423                       R!%W1 = R!HILOCL                                             D4DEM.1838
10125    <macro>                            $ELSEIF (B = R!%HISP + 1)                                     D4DEM.1839
10135   020000  024423                       R!%W1 = R!HOLOCL                                             D4DEM.1840
10137    <macro>                            $ELSEIF (B = R!%SSD)                                          D4DEM.1841
10144   020000  024423                       R!%W1 = R!SILOCL                                             D4DEM.1842
10146    <macro>                            $ELSEIF (B = R!%SSD + 1)                                      D4DEM.1843
10156   020000  024423                       R!%W1 = R!SOLOCL                                             D4DEM.1844
10160    <macro>                            $ENDIF                                                        D4DEM.1845
                                                                                          .<<<<< 0            $ENDIF   1
10160   020423  023424  103017              P = CHKDONE, R!%W1 # R!%W2                                   D4DEM.1846
10163   003000                              I = 1                                                        D4DEM.1847
10164    <macro>                            RKERN CHNWTDN                   .Wait for DN                 D4DEM.1848
10166   002000  000000                      I = 0                                                        D4DEM.1849
10170    <macro>                            $IF (IOB = DN),AND,(IOB = BZ) .If error                      D4DEM.1850
                                                                                          .>>>>> 0            $IF      1
10176    <macro>                              RKERN   CEREC               .If error, try to recover      D4DEM.1851
10200    <macro>                            $ENDIF                                                        D4DEM.1852
                                                                                          .<<<<< 0            $ENDIF   1
10200    <macro>                          $ENDTIL                                                         D4DEM.1853
10201                            CHKDONE  *                                                               D4DEM.1854
10201   003000                            I = 1                                                           D4DEM.1855
10202   012002                            A = A + 2                       .Allow interrupts               D4DEM.1856
10203   002000  000000                    I = 0                                                           D4DEM.1857
10205   073600                            R = LBPAR                       .Get descriptor parameters      D4DEM.1858
10206    <macro>                          $IF (R!FLG = 0),AND,(B = [MOS]) .If MCB pointers in danger      D4DEM.1859
                                                                                          .>>>>> 0            $IF      1
```

```
10215   <macro>                 STORE 1,DK@LCK,R!DCB        .Lock the MCB                        D4DEM.1860
10225   <macro>               $ENDIF                                                             D4DEM.1861
                                                                          .<<<<< 0                  $ENDIF    1
10225   <macro>               FLDADD 1,LB@FUL,R!LOCAL     .Complement buffer full flag         D4DEM.1862
10250   <macro>               STORE 0,LB@IOI,R!LOCAL      .Clear I/O initiated                 D4DEM.1863
10257   <macro>               GET A,LB@CP,R!LOCAL         .Check for continue pending          D4DEM.1864
10265   <macro>               $IF (A # 0)                 .If set                              D4DEM.1865
                                                                          .>>>>> 0                  $IF       1
10267   <macro>                 STORE 0,LB@CP,R!LOCAL     .Clear flag                          D4DEM.1866
10276   <macro>                 GET B,DK@CHN,R!DCB        .Disk channel                        D4DEM.1867
10304   <macro>                 GET R!ENTRY,DK@CLB,R!DCB  .Get the current LMA entry           D4DEM.1868
10311   <macro>                 GET R!UNIT,DK@UNI,R!DCB   .Unit for trace                      D4DEM.1869
10317   <macro>                 GET R!SEC,DK@SEC,R!DCB    .Current sector                      D4DEM.1870
10325   <macro>                 GET R!HED,DK@HED,R!DCB    .Current head                        D4DEM.1871
10334   <macro>                 GET R!CYL,DK@CYL,R!DCB    .Current cylinder                    D4DEM.1872
10341   <macro>                 GET R!SPT,DK@SPT,R!DCB    .Sectors per track                   D4DEM.1873
10347   <macro>                 GET R!TPC,DK@TPC,R!DCB    .Tracks per cylinder                 D4DEM.1874
10356   <macro>                 GET R!SUB,DK@SUB,R!DCB    .Subfunction code                    D4DEM.1875
10364   <macro>                 GET R!TYPE,DK@TYP,R!DCB   .Device type                       R13739AP.152
10373   <macro>                 $IF (R!DAL = 0)           .Set function code                   D4DEM.1876
                                                                          .>>>>> 1                  $IF       1
10376   010001  024451            R!FNC = DAF$READ        .Read-ahead                          D4DEM.1877
10400   <macro>                 $ELSE                                                          D4DEM.1878
10401   <macro>                   GET R!FNC,DL@FC,R!DAL   .From DAL                            D4DEM.1879
10407   <macro>                 $ENDIF                                                         D4DEM.1880
                                                                          .<<<<< 1                  $ENDIF    1
10407   077000 /006406          R = CONTINUE              .Issue continue                      D4DEM.1881
10411   <macro>               $ENDIF                                                             D4DEM.1882
                                                                          .<<<<< 0                  $ENDIF    1
10411   <macro>               GET A,DKS@RS,R!DCB          .Waiting for resource flag           D4DEM.1883
10417   117336                R = ACTDEM, A # 0           .If set                              D4DEM.1884
10420   003000                I = 1                                                              D4DEM.1885
10421   001000                EXIT                                                               D4DEM.1886
                                                                                                 D4DEM.1887
                      ************************************************************************   D4DEM.1888
                      *                                                                      *   D4DEM.1889
                      *       BLCK:     Block number validation routine.                       *   D4DEM.1890
                      *                 This routine checks the block number in a sector       *   D4DEM.1891
                      *                 of data against an expected value and punts if it      *   D4DEM.1892
                      *                 is incorrect.  Validation is enabled only if           *   D4DEM.1893
                      *                 BNFLG is nonzero and the validation request flag       *   D4DEM.1894
                      *                 is set in the MDAL.                                    *   D4DEM.1895
```

```
                                 *                                                                  *   D4DEM.1896
                                 *        Entry:    R!LOCAL = LMA entry address                     *   D4DEM.1897
                                 *                  R!DAL = MDAL address                            *   D4DEM.1898
                                 *                  R!SEQ = Sequence number                         *   D4DEM.1899
                                 *                                                                  *   D4DEM.1900
                                 ********************************************************************   D4DEM.1901
                                                                                                        D4DEM.1902
10422                            BLCK     *                                                             D4DEM.1903
10422    <macro>                          $IF ((BNFLG) # 0)              .If validation enabled          D4DEM.1904
                                                                                        .>>>>> 0            $IF      1
10430    <macro>                            GET R!%W1,DL@BNC,R!DAL       .Block number check parcel      D4DEM.1905
10435    <macro>                            $IF (0 # A > D'15)           .If validation requested        D4DEM.1906
                                                                                        .>>>>> 1            $IF      1
10443    <macro>                              GET R!%W2,LB@LOC,R!LOCAL   .Local buffer address           D4DEM.1907
10450    020453  013001  024425               R!%W3 = R!SEQ - 1          .Adjust (start with block 0)    D4DEM.1908
10453                            BLCK1    *                                                             D4DEM.1909
10453    <macro>                              $IF (R!%W3 # 0)            .If not first block             D4DEM.1910
                                                                                        .>>>>> 2            $IF      1
10456    <macro>                                GET R!%W5,DL@UNS,R!DAL   .Number physical units          D4DEM.1911
10464    020425  024426                         R!%W4 = R!%W3            .Save sequence                  D4DEM.1912
10466    <macro>                                $UNTIL  (R!%W5 = 1)      .Multiply by physical units     D4DEM.1913
10471    020425  022426  024425                   R!%W3 = R!%W3 + R!%W4                                 D4DEM.1914
10474    027427                                   R!%W5 = R!%W5 - 1                                     D4DEM.1915
10475    <macro>                                $ENDTIL                                                 D4DEM.1916
10476    <macro>                              $ENDIF                                                    D4DEM.1917
                                                                                        .<<<<< 2            $ENDIF   1
10476    020425  022423  011177               R!%W4 = R!%W3 + R!%W1 & O'177   .Expected block          D4DEM.1918
         024426
10502    020424  012003  024410               R!%W5 = (R!%W2 + 3) > D'9       .Actual block           D4DEM.1919
         030410  004011  024427
10510    <macro>                              $PUNTIF (R!%W4 # R!%W5),CODE=PT$BLOCK                     D4DEM.1920
10514    <macro>                            $ENDIF                                                      D4DEM.1921
                                                                                        .<<<<< 1            $ENDIF   1
10514    <macro>                          $ENDIF                                                        D4DEM.1922
                                                                                        .<<<<< 0            $ENDIF   1
10514    001000                           EXIT                                                          D4DEM.1923
                                                                                                        D4DEM.1924
                                 ********************************************************************   D4DEM.1925
                                 *                                                                  *   D4DEM.1926
                                 *        DATAMOVE: Starts I/O on the specified memory channel.     *   D4DEM.1927
                                 *                                                                  *   D4DEM.1928
                                 *        Entry:    R!FNC   = I/O function code                     *   D4DEM.1929
```

```
                                 *                R!BUFU  = CPU or MOS address (upper)                 *   D4DEM.1930
                                 *                R!BUFL  = CPU or MOS address (lower)                 *   D4DEM.1931
                                 *                R!LOCAL = LMA entry address                          *   D4DEM.1932
                                 *                                                                     *   D4DEM.1933
                                 ***********************************************************************   D4DEM.1934
                                                                                                           D4DEM.1935
10515                            DATAMOVE *                                                                D4DEM.1936
10515   002000  000000                   1 = 0                                                             D4DEM.1937
10517   020451  024000                   R!CA = R!FNC                  .Function code                      D4DEM.1938
10521   020454  024000                   R!CB = R!BUFU                 .Address (upper)                    D4DEM.1939
10523   020455  024000                   R!CC = R!BUFL                 .(lower)                            D4DEM.1940
10525    <macro>                         GET R!CD,LB@LOC,R!LOCAL       .Local address                      D4DEM.1941
10532   014000 /001000  024000           R!CE = D4$WPS                 .Transfer length                    D4DEM.1942
10535    <macro>                         RKERN COMIO                   .Start it                           D4DEM.1943
10537   003000                           1 = 1                                                             D4DEM.1944
10540   001000                           EXIT                                                              D4DEM.1945
                                                                                                           D4DEM.1946
                                 ***********************************************************************   D4DEM.1947
                                 *                                                                     *   D4DEM.1948
                                 *        GETCPU:   Gets CPU address from the MDAL.                    *   D4DEM.1949
                                 *                                                                     *   D4DEM.1950
                                 *        Entry:    R!DAL = DAL address                                *   D4DEM.1951
                                 *                  R!SEQ = Sector sequence number                     *   D4DEM.1952
                                 *                                                                     *   D4DEM.1953
                                 *        Exit:     R!BUFU = CPU address (upper)                       *   D4DEM.1954
                                 *                  R!BUFL = CPU address (lower)                       *   D4DEM.1955
                                 *                                                                     *   D4DEM.1956
                                 ***********************************************************************   D4DEM.1957
                                                                                                           D4DEM.1958
10541                            GETCPU   *                                                                D4DEM.1959
10541   020453  013001  024447           R!SC1 = R!SEQ - 1             .Adjust to start at sector 0        D4DEM.1960
10544   077000 /005153                   R = CPUCALC                   .Calucate CPU address               D4DEM.1961
10546   020456  024454                   R!BUFU = R!CPU                .CPU address                        D4DEM.1962
10550   020457  024455                   R!BUFL = R!CPL                                                    D4DEM.1963
10552   001000                           EXIT                                                              D4DEM.1964
                                                                                                           D4DEM.1965
                                 ***********************************************************************   D4DEM.1966
                                 *                                                                     *   D4DEM.1967
                                 *        RDDONE:   Activates D4DEM when MDAL is completed.            *   D4DEM.1968
                                 *                                                                     *   D4DEM.1969
                                 *        Entry:    R!DAL = DAL address                                *   D4DEM.1970
                                 *                  R!DCB = DCB address                                *   D4DEM.1971
```

```
                                 *                R!SEQ = Sequence number                                  *   D4DEM.1972
                                 *                                                                         *   D4DEM.1973
                                 ***************************************************************************   D4DEM.1974
                                                                                                               D4DEM.1975
10553                            RDDONE  *                                                                     D4DEM.1976
10553    <macro>                          GET A,DL@TOT,R!DAL              .Total sectors                      D4DEM.1977
10557    <macro>                          $IF (A = R!SEQ)                 .If last sector                     D4DEM.1978
                                                                                          .>>>>> 0                $IF      1
10562   002000  000000                      I = 0                                                              D4DEM.1979
10564    <macro>                          GET A,DL@RAC,R!DAL              .Read ahead count                   D4DEM.1980
10571    <macro>                          $IF (A # 0)                     .If some left to transfer           D4DEM.1981
                                                                                          .>>>>> 1                $IF      1
10573    <macro>                            STORE 1,DK@EWB,R!DCB          .Signal to send status              D4DEM.1982
10604    <macro>                          $ELSE                                                              D4DEM.1983
10605    <macro>                            ADDRESS R!ED,DK@DLO,R!DCB     .DAL queue                          D4DEM.1984
10610    <macro>                            RKERN EDQU                    .Take off queue                     D4DEM.1985
10612    <macro>                            GET A,DL@FC,R!DAL             .Function code                      D4DEM.1986
10617    <macro>                            $IF (A # DAF$INTR)            .Not internal                       D4DEM.1987
                                                                                          .>>>>> 2                $IF      1
10622   077000 /004501                       R = STAT                       .Send status                       D4DEM.1988
10624    <macro>                             RELDAL R!DAL                   .Release DAL                       D4DEM.1989
10634    <macro>                            $ELSE                                                              D4DEM.1990
10635    <macro>                              ADDRESS R!ED,DCB@DN,R!%DCBQ .DAL done queue                      D4DEM.1991
10640    <macro>                              RKERN ENQU                  .Put on queue                        D4DEM.1992
10642    <macro>                            $ENDIF                                                             D4DEM.1993
                                                                                          .<<<<< 2                $ENDIF   1
10642    <macro>                          $ENDIF                                                             D4DEM.1994
                                                                                          .<<<<< 1                $ENDIF   1
10642   073561                              R = ACTDEM                    .Activate the demon                 D4DEM.1995
10643    <macro>                          $ENDIF                                                             D4DEM.1996
                                                                                          .<<<<< 0                $ENDIF   1
10643   001000                           EXIT                                                                  D4DEM.1997
                                         END                                                                   D4DEM.1998
```

0X	%B		1: 4 D	5:12 B	5:13 B	5:17 B	5:30 B	6:33 B	7:21 B	9:41 B
			10: 5 B	10: 6 B	10:12 B	10:13 B	10:16 B	10:20 B	10:24 B	11:16 B
			11:31 B	11:41 B	13:33 B	13:34 B	14: 5 B	14: 6 B	14:14 B	14:20 B
			14:24 B	14:39 B	15: 8 B	15:11 B	15:12 B	15:15 B	15:21 B	15:27 B
			15:38 B	15:41 B	16: 3 B	16:17 B	16:22 B	16:32 B	16:42 B	17: 7 B
			17:14 B	17:21 B	17:26 B	17:29 B	17:34 B	17:36 B	17:38 B	17:40 B
			18: 4 B	18: 7 B	18:11 B	18:22 B	18:23 B	18:26 B	18:32 B	18:38 B
			18:41 B	19:26 B	19:33 B	19:36 B	19:38 B	20:12 B	20:13 B	20:21 B
			20:29 B	20:32 B	20:34 B	21: 8 B	21:15 B	21:18 B	21:20 B	21:36 B
			21:37 B	21:41 B	22: 3 B	22: 4 B	22: 7 B	22:17 B	22:20 B	22:22 B
			22:38 B	23: 1 B	23: 9 B	23:22 B	23:25 B	23:27 B	24:12 B	24:23 B
			24:28 B	24:32 B	25:21 B	25:26 B	25:37 B	25:42 B	26:22 B	27:17 B
			27:18 B	27:27 B	27:28 B	27:33 B	29: 8 B	29:32 B	29:34 B	30: 3 B
			30: 6 B	30:17 B	30:18 B	30:27 B	30:40 B	31:10 B	31:13 B	31:23 B
			31:24 B	31:26 B	31:28 B	31:32 B	31:34 B	31:35 B	32:15 B	32:17 B
			35:31 B	38:15 B	38:35 B	39:11 B	39:28 B	40:11 B	40:16 B	40:42 B
			41:19 B	41:37 B	42:23 B	43:18 B	46:19 B	50: 7 B	50:21 B	56:25 B
			59:27 B	61:32 B	62:20 B					
0X	%DCBQ		3:38 D	5:24	5:26	5:28	6: 9	7: 9	12:23	12:27
			12:30	18:34	62:23					
0X	%EX		6:31 D	6:31	6:35 D	6:35	13:15 D	13:15	16:33 D	16:33
			18:37 D	18:37	36:42 D	36:42	38:11 D	38:11	40:12 D	40:12
			42:36 D	42:36	62:21 D	62:21				
10	%GBLNUM	$APTEXT	2:10 F	2:10						
400	%GBLREG	$APTEXT	2:10							
0X	%HISP		3:39 D	21: 6	21:34	45:41	58:16	58:18		
0X	%HISPBUF		3:40 D	16: 6						
0	%KERNEL	$APTEXT	5:20 D	5:20 B	5:32 D	5:32 B	6:26 D	6:26 B	7:13 D	7:13 B
			9:38 D	9:38 B	16:31 D	16:31 B	32:28 D	32:28 B	32:29 D	32:29 B
			32:36 D	32:36 B	32:41 D	32:41 B	34: 1 D	34: 1 B	34:11 D	34:11 B
			36:16 D	36:16 B	36:21 D	36:21 B	43:40 D	43:40 B	52: 2 D	52: 2 B
			57: 8 D	57: 8 B	57: 9 D	57: 9 B	57:11 D	57:11 B	57:28 D	57:28 B
			58: 9 D	58: 9 B	58:28 D	58:28 B	58:32 D	58:32 B	61:14 D	61:14 B
			62:16 D	62:16 B	62:24 D	62:24 B				
0X	%MEMIO		3:41 D	19:25	19:30	20:11	20:18	21: 7	21:12	21:35
			22: 1	22:37	24: 8	28: 3	28: 5	28: 7	28: 9	28:11
			51:23	51:30	51:32					
0X	%MYID		3:42 D	8:27	14:11	15:37	17:28	23:14	25: 1	33:33
			35:29	36: 3	36:14	50:21	51: 4	51:25	56: 3	
0X	%PUNT		13:12 D	13:12	13:16 D	13:16	41:11 D	41:11	42: 8 D	42: 8
			42:37 D	42:37	48:17 D	48:17	53:15 D	53:15	54:13 D	54:13
			56:31 D	56:31	60:31 D	60:31				

0X	%SSD		4: 1 D	19:24	20:10	58:20	58:22				
0X	%SSDBUF		4: 2 D	16: 8							
0X	%TR1		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TR2		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TR3		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TR4		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TR5		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TRAC		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
0X	%TVNT		30:28 D	30:28	39:29 D	39:29	47:28 D	47:28			
177777	AD$NODEM	$APTEXT	13:15	42:36							
20	BB@PTR@N	$APTEXT	34:17	34:17 F	35:32 F						
3	BB@PTR@P	$APTEXT	34:17 F	34:17	35:32 F	35:32					
0	BB@PTR@S	$APTEXT	34:17	34:17 F	35:32 F						
0X	BMLOCL		4: 3 D	58:15							
0X	BNFLG		4: 4 D	16:12	60:10						
2	BP$DONE	$APTEXT	46: 1	50:25	51: 8	56:15	56:19				
0	BP$NONE	$APTEXT	56:25								
1	BP$SENT	$APTEXT	57: 5								
0X	CA		4: 5 D	34: 6	36:15	61: 9					
0X	CB		4: 6 D	34: 7	35:33	36: 7	61:10				
1	CB$BBSZ	$APTEXT	35:35								
1	CB$BIOP	$APTEXT	34:15	35:33							
1	CB$DDSZ	$APTEXT	36: 9								
0	CB$DIOP	$APTEXT	34:12	36: 7							
2	CB@LH	$APTEXT	34:18								
3	CB@SZ	$APTEXT	34:10								
14	CB@SZ@P	$APTEXT	33:42	36:19							
0X	CC		4: 7 D	34: 8	35:33	36: 7	61:11				
0X	CD		4: 8 D	34: 9	35:34	36: 8	61:12				
0X	CE		4: 9 D	34:10	35:35	36: 9	61:13				
0X	CEREC		4:10 D	58:32 D	58:32						
0X	CHNWTDN		4:11 D	58:28 D	58:28						
0X	COMIO		4:12 D	34:11 D	34:11	36:16 D	36:16	61:14 D	61:14		
0X	DSD4DEM		4:13 D	5:19							
1000	D4$WPS	$APTEXT	36:42	38:25	61:13						
60	D40$SPT	$APTEXT	45:15								
0X	D4ACT		4:20 D	57:28 D	57:28						
2	D4DSLIM	$APTEXT	12:18								
2	D4HDLIM	$APTEXT	39:27	41:18	42:22						
2	D4IOLIM	$APTEXT	45:28								
24	D4SKLIM	$APTEXT	30:26								
30	DA@@LF	$APTEXT	56:39								

10	DA@@LH	$APTEXT	56:38	56:39						
10	DA@DID@P	$APTEXT	6:30							
20	DA@IFC@N	$APTEXT	6:15	6:15 F	32:24	32:24 F	32:31 F	50:22	50:22 F	51: 5
			51: 5 F	56:10 F	56:16	56:16 F	56:33	56:33 F	56:40 F	
1	DA@IFC@P	$APTEXT	6:15 F	6:15	32:24 F	32:24	32:31 F	32:31	50:22 F	50:22
			51: 5 F	51: 5	56:10 F	56:10	56:16 F	56:16	56:33 F	56:33
			56:40 F	56:40						
0	DA@IFC@S	$APTEXT	6:15	6:15 F	32:24	32:24 F	32:31 F	50:22	50:22 F	51: 5
			51: 5 F	56:10 F	56:16	56:16 F	56:33	56:33 F	56:40 F	
20	DA@MLS@N	$APTEXT	32:39	32:39 F						
5	DA@MLS@P	$APTEXT	32:39 F	32:39						
0	DA@MLS@S	$APTEXT	32:39	32:39 F						
21	DAI$INTR	$APTEXT	23:27	31:22	62:19					
22	DAI$INTW	$APTEXT	25:29	51:25						
12	DAI$PTRN	$APTEXT	38:23	51:40	56: 9					
1	DAI$READ	$APTEXT	14:11	24:17	31:22	47: 9	59:22			
6	DAI$WBFR	$APTEXT	6:14	17:20	27:22					
14	DAI$WRBH	$APTEXT	6:14	6:33	17:20	27:22				
0	DAR$OK	$APTEXT	32:34							
2	DAS$ABS	$APTEXT	54:31							
1	DAS$ID	$APTEXT	54:31							
20	DCB@DA@N	$APTEXT	5:24	5:24 F						
2	DCB@DA@P	$APTEXT	5:24 F	5:24	7: 9					
0	DCB@DA@S	$APTEXT	5:24	5:24 F						
20	DCB@DN@N	$APTEXT	5:26	5:26 F						
4	DCB@DN@P	$APTEXT	5:26 F	5:26	6: 9	62:23				
0	DCB@DN@S	$APTEXT	5:26	5:26 F						
20	DCB@QL@N	$APTEXT	12:30 F							
1	DCB@QL@P	$APTEXT	12:30 F	12:30						
0	DCB@QL@S	$APTEXT	12:30 F							
20	DCB@QU@N	$APTEXT	5:28	5:28 F	12:23	12:23 F	12:27 F	18:34	18:34 F	
0	DCB@QU@P	$APTEXT	5:28 F	12:23 F	12:27 F	18:34 F				
0	DCB@QU@S	$APTEXT	5:28	5:28 F	12:23	12:23 F	12:27 F	18:34	18:34 F	
0X	DCCB		4:14 D	7:18						
20	DD@PTR@N	$APTEXT	34:14	34:14 F	36: 6 F					
3	DD@PTR@P	$APTEXT	34:14 F	34:14	36: 6 F	36: 6				
0	DD@PTR@S	$APTEXT	34:14	34:14 F	36: 6 F					
0X	DDALGETA		4:15 D	32:28 D	32:28	57: 8 D	57: 8			
0X	DGIVEUP		4:16 D	5:20 D	5:20					
4	DK$RAHD	$APTEXT	10:36	11:14	11:22	14:20				
1	DK$READ	$APTEXT	15:31							
100	DK$RLSE	$APTEXT	15:29							

2	DK$WRIT	$APTEXT	10: 6								
1	DK@ABI@M	$APTEXT	11:38 F	14:33 F	15:19 F	26:19 F					
1	DK@ABI@N	$APTEXT	11:38 F	11:38	14:33 F	14:33	15:19 F	26:19 F	26:19		
2	DK@ABI@P	$APTEXT	11:38 F	11:38	14:33 F	14:33	15:19 F	15:19	26:19 F	26:19	
1	DK@ABI@S	$APTEXT	11:38 F	11:38	14:33 F	14:33	15:19 F	26:19 F	26:19		
137777	DK@ABI@X	$APTEXT	15:19								
20	DK@ACC@N	$APTEXT	30:22 F	39:25 F	41:16 F	42:12 F	45:25 F	46:38 F			
53	DK@ACC@P	$APTEXT	30:22 F	30:22	39:25 F	39:25	41:16 F	41:16	42:12 F	42:12	
			45:25 F	45:25	46:38 F	46:38					
0	DK@ACC@S	$APTEXT	30:22 F	39:25 F	41:16 F	42:12 F	45:25 F	46:38 F			
20	DK@ACI@N	$APTEXT	9:39	9:39 F	14: 3	14: 3 F	44:17	44:17 F	46:10	46:10 F	
0	DK@ACI@P	$APTEXT	9:39 F	14: 3 F	44:17 F	46:10 F					
0	DK@ACI@S	$APTEXT	9:39	9:39 F	14: 3	14: 3 F	44:17	44:17 F	46:10	46:10 F	
1	DK@ADD@M	$APTEXT	44:32 F								
1	DK@ADD@N	$APTEXT	44:32 F								
2	DK@ADD@P	$APTEXT	44:32 F	44:32							
2	DK@ADD@S	$APTEXT	44:32 F								
157777	DK@ADD@X	$APTEXT	44:32								
20	DK@BFO@N	$APTEXT	33:29	33:29 F							
63	DK@BFO@P	$APTEXT	33:29 F	33:29							
0	DK@BFO@S	$APTEXT	33:29	33:29 F							
20	DK@BF1@N	$APTEXT	33:30	33:30 F							
64	DK@BF1@P	$APTEXT	33:30 F	33:30							
0	DK@BF1@S	$APTEXT	33:30	33:30 F							
377	DK@BIO@M	$APTEXT	9:25	35:30							
10	DK@BIO@N	$APTEXT	9:21	9:21 F	9:25 F	9:25	33:34	33:34 F	35:30 F	35:30	
17	DK@BIO@P	$APTEXT	9:21 F	9:21	9:25 F	9:25	33:34 F	33:34	35:30 F	35:30	
0	DK@BIO@S	$APTEXT	9:21	9:21 F	9:25 F	9:25	33:34	33:34 F	35:30 F	35:30	
1	DK@CE@M	$APTEXT	29:40								
1	DK@CE@N	$APTEXT	29:40	29:40 F							
2	DK@CE@P	$APTEXT	29:40 F	29:40							
6	DK@CE@S	$APTEXT	29:40	29:40 F							
777	DK@CHN@M	$APTEXT	12:38	59:10							
11	DK@CHN@N	$APTEXT	12:38	59:10							
7	DK@CHN@P	$APTEXT	12:38 F	12:38	59:10 F	59:10					
7	DK@CHN@S	$APTEXT	12:38	12:38 F	59:10	59:10 F					
20	DK@CLB@N	$APTEXT	10:23	10:23 F	14:40	14:40 F	38: 5	38: 5 F	40: 4	40: 4 F	
			44:21	44:21 F	50:32	50:32 F	59:11	59:11 F			
13	DK@CLB@P	$APTEXT	10:23 F	10:23	14:40 F	14:40	38: 5 F	38: 5	40: 4 F	40: 4	
			44:21 F	44:21	50:32 F	50:32	59:11 F	59:11			
0	DK@CLB@S	$APTEXT	10:23	10:23 F	14:40	14:40 F	38: 5	38: 5 F	40: 4	40: 4 F	
			44:21	44:21 F	50:32	50:32 F	59:11	59:11 F			

1	DK@CNG@M	$APTEXT	46:39 F									
1	DK@CNG@N	$APTEXT	46:39 F	46:39								
2	DK@CNG@P	$APTEXT	46:39 F	46:39								
4	DK@CNG@S	$APTEXT	46:39 F	46:39								
377	DK@CNI@M	$APTEXT	9:20	33:31								
10	DK@CNI@N	$APTEXT	9:20	33:31								
16	DK@CNI@P	$APTEXT	9:20 F	9:20	33:31 F	33:31						
10	DK@CNI@S	$APTEXT	9:20	9:20 F	33:31	33:31 F						
20	DK@CPD@N	$APTEXT	28:33 F									
57	DK@CPD@P	$APTEXT	28:33 F	28:33								
0	DK@CPD@S	$APTEXT	28:33 F									
20	DK@CYL@N	$APTEXT	30:13	30:13 F	30:24 F	42:17 F	59:15	59:15 F				
3	DK@CYL@P	$APTEXT	30:13 F	30:13	30:24 F	30:24	42:17 F	42:17	59:15 F	59:15		
0	DK@CYL@S	$APTEXT	30:13	30:13 F	30:24 F	42:17 F	59:15	59:15 F				
1	DK@DIA@M	$APTEXT	8:14	28:31								
1	DK@DIA@N	$APTEXT	8:14	8:14 F	28:31 F	28:31						
2	DK@DIA@P	$APTEXT	8:14 F	8:14	28:31 F	28:31						
5	DK@DIA@S	$APTEXT	8:14	8:14 F	28:31 F	28:31						
377	DK@DIO@M	$APTEXT	33:35	36: 4								
10	DK@DIO@N	$APTEXT	33:35	36: 4 F								
17	DK@DIO@P	$APTEXT	33:35 F	33:35	36: 4 F	36: 4						
10	DK@DIO@S	$APTEXT	33:35	33:35 F	36: 4 F							
177400	DK@DIO@X	$APTEXT	36: 4									
20	DK@DLO@N	$APTEXT	10:10	10:10 F	13:31	13:31 F	14: 2	14: 2 F	15:23	15:23 F		
			17:11	17:11 F	20:22	20:22 F	22: 8	22: 8 F	31:33	31:33 F		
			56:26	56:26 F								
14	DK@DLO@P	$APTEXT	9:36	10:10 F	10:10	13:31 F	13:31	14: 2 F	14: 2	15:23 F		
			15:23	16:30	17:11 F	17:11	20:22 F	20:22	22: 8 F	22: 8		
			31:33 F	31:33	56:26 F	56:26	62:15					
0	DK@DLO@S	$APTEXT	10:10	10:10 F	13:31	13:31 F	14: 2	14: 2 F	15:23	15:23 F		
			17:11	17:11 F	20:22	20:22 F	22: 8	22: 8 F	31:33	31:33 F		
			56:26	56:26 F								
20	DK@DL1@N	$APTEXT	7:27	7:27 F	10:14	10:14 F	32:13	32:13 F				
15	DK@DL1@P	$APTEXT	7:27 F	7:27	10:14 F	10:14	32:13 F	32:13				
0	DK@DL1@S	$APTEXT	7:27	7:27 F	10:14	10:14 F	32:13	32:13 F				
20	DK@DPI@N	$APTEXT	54:28	54:28 F								
61	DK@DPI@P	$APTEXT	54:28 F	54:28								
0	DK@DPI@S	$APTEXT	54:28	54:28 F								
10	DK@DRC@N	$APTEXT	28:42	28:42 F								
54	DK@DRC@P	$APTEXT	28:42 F	28:42								
0	DK@DRC@S	$APTEXT	28:42	28:42 F								
10	DK@DWC@N	$APTEXT	29: 1	29: 1 F								

55	DK@DWC@P	$APTEXT	29: 1 F	29: 1							
0	DK@DWC@S	$APTEXT	29: 1	29: 1 F							
1	DK@EWB@M	$APTEXT	16:26	16:29 F	62:13 F						
1	DK@EWB@N	$APTEXT	16:26	16:26 F	16:29 F	62:13 F	62:13				
2	DK@EWB@P	$APTEXT	16:26 F	16:26	16:29 F	16:29	62:13 F	62:13			
3	DK@EWB@S	$APTEXT	16:26	16:26 F	16:29 F	62:13 F	62:13				
167777	DK@EWB@X	$APTEXT	16:29								
20	DK@FLB@N	$APTEXT	14:41	14:41 F	44:24	44:24 F	55:22	55:22 F			
11	DK@FLB@P	$APTEXT	14:41 F	14:41	44:24 F	44:24	55:22 F	55:22			
0	DK@FLB@S	$APTEXT	14:41	14:41 F	44:24	44:24 F	55:22	55:22 F			
37	DK@HED@M	$APTEXT	30:34	39:13	39:14	42:18 F	45:27	59:14			
5	DK@HED@N	$APTEXT	30:34	30:34 F	39:13	39:13 F	39:14 F	39:14	42:18 F	42:18	
			45:27 F	45:27	59:14	59:14 F					
4	DK@HED@P	$APTEXT	30:34 F	30:34	39:13 F	39:13	39:14 F	39:14	42:18 F	42:18	
			45:27 F	45:27	59:14 F	59:14					
2	DK@HED@S	$APTEXT	30:34	30:34 F	39:13	39:13 F	39:14 F	39:14	42:18 F	42:18	
			45:27 F	45:27	59:14	59:14 F					
1	DK@INV@M	$APTEXT	10:37	10:42 F	15:20 F						
1	DK@INV@N	$APTEXT	10:37	10:37 F	10:42 F	10:42	15:20 F				
16	DK@INV@P	$APTEXT	10:37 F	10:37	10:42 F	10:42	15:20 F	15:20			
1	DK@INV@S	$APTEXT	10:37	10:37 F	10:42 F	10:42	15:20 F				
137777	DK@INV@X	$APTEXT	15:20								
1	DK@LCK@M	$APTEXT	23:32 F	59: 1 F							
1	DK@LCK@N	$APTEXT	23:32 F	23:32	59: 1 F	59: 1					
16	DK@LCK@P	$APTEXT	23:32 F	23:32	59: 1 F	59: 1					
0	DK@LCK@S	$APTEXT	23:32 F	23:32	59: 1 F	59: 1					
20	DK@LLB@N	$APTEXT	15: 2	15: 2 F	44:23	44:23 F	55:19	55:19 F			
12	DK@LLB@P	$APTEXT	15: 2 F	15: 2	44:23 F	44:23	55:19 F	55:19			
0	DK@LLB@S	$APTEXT	15: 2	15: 2 F	44:23	44:23 F	55:19	55:19 F			
1	DK@LMA@M	$APTEXT	44:31 F	48:18							
1	DK@LMA@N	$APTEXT	44:31 F	44:31	48:18 F	48:18					
2	DK@LMA@P	$APTEXT	44:31 F	44:31	48:18 F	48:18					
0	DK@LMA@S	$APTEXT	44:31 F	44:31	48:18	48:18 F					
20	DK@LNK@N	$APTEXT	12:26	12:26 F	12:33 F						
37	DK@LNK@P	$APTEXT	12:26 F	12:26	12:33 F	12:33					
0	DK@LNK@S	$APTEXT	12:26	12:26 F	12:33 F						
20	DK@MC0@N	$APTEXT	33:37	33:37 F	57: 1	57: 1 F					
32	DK@MC0@P	$APTEXT	33:37 F	33:37	57: 1 F	57: 1					
0	DK@MC0@S	$APTEXT	33:37	33:37 F	57: 1	57: 1 F					
20	DK@MC1@N	$APTEXT	33:38	33:38 F	57: 2	57: 2 F					
33	DK@MC1@P	$APTEXT	33:38 F	33:38	57: 2 F	57: 2					
0	DK@MC1@S	$APTEXT	33:38	33:38 F	57: 2	57: 2 F					

37	DK@NHD@M	$APTEXT	30:25								
5	DK@NHD@N	$APTEXT	30:25 F								
51	DK@NHD@P	$APTEXT	30:25 F	30:25							
13	DK@NHD@S	$APTEXT	30:25 F								
177740	DK@NHD@X	$APTEXT	30:25								
4	DK@NLB@N	$APTEXT	50:14	50:14 F							
56	DK@NLB@P	$APTEXT	50:14 F	50:14							
0	DK@NLB@S	$APTEXT	50:14	50:14 F							
20	DK@OWC@N	$APTEXT	30:16	30:16 F	30:41	30:41 F	40:35	40:35 F			
65	DK@OWC@P	$APTEXT	30:16 F	30:16	30:41 F	30:41	40:35 F	40:35	45:14		
0	DK@OWC@S	$APTEXT	30:16	30:16 F	30:41	30:41 F	40:35	40:35 F			
20	DK@RAC@N	$APTEXT	8:21	8:21 F	11:19 F	32:20 F					
5	DK@RAC@P	$APTEXT	8:21 F	8:21	11:19 F	11:19	32:20 F	32:20			
0	DK@RAC@S	$APTEXT	8:21	8:21 F	11:19 F	32:20 F					
377	DK@RAD@M	$APTEXT	8:30	9:19	10:34	11: 6	14:37	15:13 F			
10	DK@RAD@N	$APTEXT	8:30	9:19 F	9:19	10:34	11: 6 F	11: 6	14:37	15:13 F	
10	DK@RAD@P	$APTEXT	8:30 F	8:30	9:19 F	9:19	10:34 F	10:34	11: 6 F	11: 6	
			14:37 F	14:37	15:13 F	15:13					
10	DK@RAD@S	$APTEXT	8:30	8:30 F	9:19	9:19 F	10:34	10:34 F	11: 6	11: 6 F	
			14:37	14:37 F	15:13 F						
177400	DK@RAD@X	$APTEXT	9:19	11: 6	15:13						
37	DK@RAH@M	$APTEXT	8:22	9:17	11:18	32:19					
5	DK@RAH@N	$APTEXT	8:22	8:22 F	9:17 F	9:17	11:18 F	11:18	32:19 F	32:19	
6	DK@RAH@P	$APTEXT	8:22 F	8:22	9:17 F	9:17	11:18 F	11:18	32:19 F	32:19	
2	DK@RAH@S	$APTEXT	8:22	8:22 F	9:17 F	9:17	11:18 F	11:18	32:19 F	32:19	
777	DK@RAS@M	$APTEXT	8:23	9:18	11:17	32:18					
11	DK@RAS@N	$APTEXT	8:23	9:18 F	11:17 F	32:18 F					
6	DK@RAS@P	$APTEXT	8:23 F	8:23	9:18 F	9:18	11:17 F	11:17	32:18 F	32:18	
7	DK@RAS@S	$APTEXT	8:23	8:23 F	9:18 F	11:17 F	32:18 F				
177000	DK@RAS@X	$APTEXT	9:18	11:17	32:18						
3	DK@RAU@M	$APTEXT	32:21								
2	DK@RAU@N	$APTEXT	8:20	8:20 F	32:21 F	32:21					
6	DK@RAU@P	$APTEXT	8:20 F	8:20	32:21 F	32:21					
0	DK@RAU@S	$APTEXT	8:20	8:20 F	32:21 F	32:21					
377	DK@RCC@M	$APTEXT	29: 4	47: 6							
10	DK@RCC@N	$APTEXT	29: 4 F	47: 6							
54	DK@RCC@P	$APTEXT	29: 4 F	29: 4	47: 6 F	47: 6					
10	DK@RCC@S	$APTEXT	29: 4 F	47: 6	47: 6 F						
177400	DK@RCC@X	$APTEXT	29: 4								
777	DK@SEC@M	$APTEXT	31: 1	45:26	59:13						
11	DK@SEC@N	$APTEXT	31: 1	45:26 F	59:13						
4	DK@SEC@P	$APTEXT	31: 1 F	31: 1	45:26 F	45:26	59:13 F	59:13			

7	DK@SEC@S	$APTEXT	31: 1	31: 1 F	45:26 F	59:13	59:13 F		
177000	DK@SEC@X	$APTEXT	45:26						
1	DK@SEI@N	$APTEXT	29:31	29:31 F	40:33	40:33 F	41:35	41:35 F	
7	DK@SEI@P	$APTEXT	29:31 F	29:31	40:33 F	40:33	41:35 F	41:35	
0	DK@SEL@S	$APTEXT	29:31	29:31 F	40:33	40:33 F	41:35	41:35 F	
1	DK@SPF@M	$APTEXT	29:27						
1	DK@SPF@N	$APTEXT	29:27 F	29:27					
2	DK@SPF@P	$APTEXT	29:27 F	29:27					
7	DK@SPF@S	$APTEXT	29:27 F	29:27					
20	DK@SPO@N	$APTEXT	29:10 F	29:24 F	49:11	49:11 F			
62	DK@SPO@P	$APTEXT	29:10 F	29:10	29:24 F	29:24	49:11 F	49:11	
0	DK@SPO@S	$APTEXT	29:10 F	29:24 F	49:11	49:11 F			
777	DK@SPT@M	$APTEXT	28:32	59:16					
11	DK@SPT@N	$APTEXT	28:32 F	59:16					
60	DK@SPT@P	$APTEXT	28:32 F	28:32	59:16 F	59:16			
7	DK@SPT@S	$APTEXT	28:32 F	59:16	59:16 F				
177000	DK@SPT@X	$APTEXT	28:32						
20	DK@SRA@N	$APTEXT	11: 5 F						
27	DK@SRA@P	$APTEXT	11: 5 F	11: 5					
77	DK@SRV@M	$APTEXT	13:42						
6	DK@SRV@N	$APTEXT	13:42						
2	DK@SRV@P	$APTEXT	13:42 F	13:42					
12	DK@SRV@S	$APTEXT	13:42	13:42 F					
20	DK@STS@N	$APTEXT	43:16 F						
36	DK@STS@P	$APTEXT	43:16 F	43:16					
0	DK@STS@S	$APTEXT	43:16 F						
37	DK@SUB@M	$APTEXT	8:16	29: 6	59:18				
5	DK@SUB@N	$APTEXT	8:16	29: 6 F	59:18				
56	DK@SUB@P	$APTEXT	8:16 F	8:16	29: 6 F	29: 6	59:18 F	59:18	
13	DK@SUB@S	$APTEXT	8:16	8:16 F	29: 6 F	59:18	59:18 F		
177740	DK@SUB@X	$APTEXT	29: 6						
3	DK@SYN@M	$APTEXT	30:36	45: 5 F	45:17 F				
2	DK@SYN@N	$APTEXT	30:36	30:36 F	45: 5 F	45: 5	45:17 F	45:17	
2	DK@SYN@P	$APTEXT	30:36 F	30:36	45: 5 F	45: 5	45:17 F	45:17	
10	DK@SYN@S	$APTEXT	30:36	30:36 F	45: 5 F	45: 5	45:17 F	45:17	
17	DK@TM@M	$APTEXT	28:34						
4	DK@TM@N	$APTEXT	8:18	8:18 F	28:34 F	28:34	54: 8	54: 8 F	
51	DK@TM@P	$APTEXT	8:18 F	8:18	28:34 F	28:34	54: 8 F	54: 8	
0	DK@TM@S	$APTEXT	8:18	8:18 F	28:34 F	28:34	54: 8	54: 8 F	
44	DK@TMO@P	$APTEXT	43:38						
7	DK@TMP@M	$APTEXT	15:10	28:35					
3	DK@TMP@N	$APTEXT	15:10	15:10 F	28:35 F	28:35			

51	DK@IMP@P	$APIEXT	15:10 F	15:10	28:35 F	28:35					
4	DK@IMP@S	$APIEXT	15:10	15:10 F	28:35 F	28:35					
37	DK@IPC@M	$APIEXT	54:37	59:17							
5	DK@IPC@N	$APIEXT	54:37	54:37 F	59:17	59:17 F					
60	DK@IPC@P	$APIEXT	54:37 F	54:37	59:17 F	59:17					
2	DK@IPC@S	$APIEXT	54:37	54:37 F	59:17	59:17 F					
37	DK@IYP@M	$APIEXT	12:39	59:19							
5	DK@IYP@N	$APIEXT	12:39	12:39 F	59:19	59:19 F					
7	DK@IYP@P	$APIEXT	12:39 F	12:39	59:19 F	59:19					
2	DK@IYP@S	$APIEXT	12:39	12:39 F	59:19	59:19 F					
3	DK@UNI@M	$APIEXT	42:19								
2	DK@UNI@N	$APIEXT	29:33	29:33 F	41: 2	41: 2 F	42:14	42:14 F	42:19 F	42:19	
			59:12	59:12 F							
4	DK@UNI@P	$APIEXT	29:33 F	29:33	41: 2 F	41: 2	42:14 F	42:14	42:19 F	42:19	
			59:12 F	59:12							
0	DK@UNI@S	$APIEXT	29:33	29:33 F	41: 2	41: 2 F	42:14	42:14 F	42:19 F	42:19	
			59:12	59:12 F							
377	DK@WBD@M	$APIEXT	18: 2	23:21							
10	DK@WBD@N	$APIEXT	17:32	17:32 F	18: 2 F	18: 2	23:21 F	23:21			
10	DK@WBD@P	$APIEXT	17:32 F	17:32	18: 2 F	18: 2	23:21 F	23:21			
0	DK@WBD@S	$APIEXT	17:32	17:32 F	18: 2	18: 2 F	23:21	23:21 F			
377	DK@WCC@M	$APIEXT	17:33	29: 5							
10	DK@WCC@N	$APIEXT	17:33	29: 5 F							
55	DK@WCC@P	$APIEXT	17:33 F	17:33	29: 5 F	29: 5					
10	DK@WCC@S	$APIEXT	17:33	17:33 F	29: 5 F						
177400	DK@WCC@X	$APIEXT	29: 5								
1	DKF@ER@M	$APIEXT	13:17 F	13:22	42:38 F						
1	DKF@ER@N	$APIEXT	13:17 F	13:17	13:22	13:22 F	42:38 F	42:38			
0	DKF@ER@P	$APIEXT	13:17 F	13:22 F	42:38 F						
10	DKF@ER@S	$APIEXT	13:17 F	13:17	13:22	13:22 F	42:38 F	42:38			
1	DKF@HD@M	$APIEXT	39:26 F								
1	DKF@HD@N	$APIEXT	39:26 F	39:26							
0	DKF@HD@P	$APIEXT	39:26 F								
13	DKF@HD@S	$APIEXT	39:26 F	39:26							
1	DKF@RA@M	$APIEXT	11:23 F	26:16							
1	DKF@RA@N	$APIEXT	11:23 F	26:16	26:16 F						
0	DKF@RA@P	$APIEXT	11:23 F	26:16 F							
15	DKF@RA@S	$APIEXT	11:23 F	26:16	26:16 F						
177773	DKF@RA@X	$APIEXT	11:23								
1	DKF@RD@M	$APIEXT	11:24 F	45:10 F							
1	DKF@RD@N	$APIEXT	11:24 F	45:10 F							
0	DKF@RD@P	$APIEXT	11:24 F	45:10 F							

17	DKF@RD@S	$APTEXT	11:24 F	45:10 F								
177776	DKF@RD@X	$APTEXT	11:24	45:10								
1	DKF@RL@M	$APTEXT	41:17 F									
1	DKF@RL@N	$APTEXT	41:17 F	41:17								
0	DKF@RL@P	$APTEXT	41:17 F									
11	DKF@RL@S	$APTEXT	41:17 F	41:17								
1	DKF@SK@M	$APTEXT	30:23 F									
1	DKF@SK@N	$APTEXT	30:23 F	30:23								
0	DKF@SK@P	$APTEXT	30:23 F									
14	DKF@SK@S	$APTEXT	30:23 F	30:23								
1	DKF@SL@M	$APTEXT	42:13 F									
1	DKF@SL@N	$APTEXT	42:13 F	42:13								
0	DKF@SL@P	$APTEXT	42:13 F									
12	DKF@SL@S	$APTEXT	42:13 F	42:13								
1	DKF@WR@M	$APTEXT	45:22 F									
1	DKF@WR@N	$APTEXT	45:22 F	45:22								
0	DKF@WR@P	$APTEXT	45:22 F									
16	DKF@WR@S	$APTEXT	45:22 F	45:22								
1	DKS@CR@M	$APTEXT	10: 8	13: 1	13:18 F	43:17 F	46:11					
1	DKS@CR@N	$APTEXT	10: 8	13: 1	13:18 F	43:17 F	46:11					
2	DKS@CR@P	$APTEXT	10: 8 F	10: 8	13: 1 F	13: 1	13:18 F	13:18	43:17 F	43:17		
			46:11 F	46:11								
17	DKS@CR@S	$APTEXT	10: 8	10: 8 F	13: 1	13: 1 F	13:18 F	43:17 F	46:11	46:11 F		
177776	DKS@CR@X	$APTEXT	13:18	43:17								
1	DKS@RS@M	$APTEXT	13:27	13:30 F	38:14 F	44:38 F	59:30					
1	DKS@RS@N	$APTEXT	13:27	13:27 F	13:30 F	38:14 F	38:14	44:38 F	44:38	59:30		
			59:30 F									
2	DKS@RS@P	$APTEXT	13:27 F	13:27	13:30 F	13:30	38:14 F	38:14	44:38 F	44:38		
			59:30 F	59:30								
16	DKS@RS@S	$APTEXT	13:27	13:27 F	13:30 F	38:14 F	38:14	44:38 F	44:38	59:30		
			59:30 F									
177775	DKS@RS@X	$APTEXT	13:30									
1	DKS@UR@M	$APTEXT	12:34	12:35 F								
1	DKS@UR@N	$APTEXT	12:34	12:34 F	12:35 F							
2	DKS@UR@P	$APTEXT	12:34 F	12:34	12:35 F	12:35						
15	DKS@UR@S	$APTEXT	12:34	12:34 F	12:35 F							
177773	DKS@UR@X	$APTEXT	12:35									
20	DL@BM0@N	$APTEXT	25:31	25:31 F								
34	DL@BM0@P	$APTEXT	25:31 F	25:31								
0	DL@BM0@S	$APTEXT	25:31	25:31 F								
20	DL@BM1@N	$APTEXT	25:32	25:32 F								
35	DL@BM1@P	$APTEXT	25:32 F	25:32								

0	DL@BM1@S	$APTEXT	25:32	25:32 F								
20	DL@BNC@N	$APTEXT	16:13	16:13 F	60:11	60:11 F						
16	DL@BNC@P	$APTEXT	16:13 F	16:13	60:11 F	60:11						
0	DL@BNC@S	$APTEXT	16:13	16:13 F	60:11	60:11 F						
177	DL@CHN@M	$APTEXT	7:17									
7	DL@CHN@N	$APTEXT	7:17									
27	DL@CHN@P	$APTEXT	7:17 F	7:17								
11	DL@CHN@S	$APTEXT	7:17	7:17 F								
1	DL@CON@M	$APTEXT	9:33 F	10:21	10:39	45:35						
1	DL@CON@N	$APTEXT	9:33 F	9:33	10:21	10:21 F	10:39	10:39 F	45:35	45:35 F		
21	DL@CON@P	$APTEXT	9:33 F	9:33	10:21 F	10:21	10:39 F	10:39	45:35 F	45:35		
6	DL@CON@S	$APTEXT	9:33 F	9:33	10:21	10:21 F	10:39	10:39 F	45:35	45:35 F		
13	DL@CYL@N	$APTEXT	7:39	7:39 F	53:29	53:29 F						
30	DL@CYL@P	$APTEXT	7:39 F	7:39	53:29 F	53:29						
0	DL@CYL@S	$APTEXT	7:39	7:39 F	53:29	53:29 F						
1	DL@DIA@N	$APTEXT	7:32	7:32 F	54:38	54:38 F						
15	DL@DIA@P	$APTEXT	7:32 F	7:32	54:38 F	54:38						
0	DL@DIA@S	$APTEXT	7:32	7:32 F	54:38	54:38 F						
20	DL@DON@N	$APTEXT	10:18	10:18 F	11: 9 F	53: 9	53: 9 F					
13	DL@DON@P	$APTEXT	10:18 F	10:18	11: 9 F	11: 9	53: 9 F	53: 9				
0	DL@DON@S	$APTEXT	10:18	10:18 F	11: 9 F	53: 9	53: 9 F					
20	DL@ERC@N	$APTEXT	32:37 F									
17	DL@ERC@P	$APTEXT	32:37 F	32:37								
0	DL@ERC@S	$APTEXT	32:37 F									
10	DL@FC@N	$APTEXT	6:12	6:12 F	7:30	7:30 F	24:15	24:15 F	27:20	27:20 F		
			53: 7	53: 7 F	59:24	59:24 F	62:17	62:17 F				
26	DL@FC@P	$APTEXT	6:12 F	6:12	7:30 F	7:30	24:15 F	24:15	27:20 F	27:20		
			53: 7 F	53: 7	59:24 F	59:24	62:17 F	62:17				
0	DL@FC@S	$APTEXT	6:12	6:12 F	7:30	7:30 F	24:15	24:15 F	27:20	27:20 F		
			53: 7	53: 7 F	59:24	59:24 F	62:17	62:17 F				
37	DL@HED@M	$APTEXT	7:40	53:30								
5	DL@HED@N	$APTEXT	7:40	53:30								
30	DL@HED@P	$APTEXT	7:40 F	7:40	53:30 F	53:30						
13	DL@HED@S	$APTEXT	7:40	7:40 F	53:30	53:30 F						
20	DL@PAT@N	$APTEXT	38:24	38:24 F								
25	DL@PAT@P	$APTEXT	38:24 F	38:24								
0	DL@PAT@S	$APTEXT	38:24	38:24 F								
377	DL@RAC@M	$APTEXT	11: 7	16:23	18: 1							
10	DL@RAC@N	$APTEXT	11: 7 F	11: 7	16:23 F	16:23	18: 1 F	18: 1	23:20	23:20 F		
			27:23	27:23 F	53:10	53:10 F	62:10	62:10 F				
37	DL@RAC@P	$APTEXT	11: 7 F	11: 7	16:23 F	16:23	18: 1 F	18: 1	23:20 F	23:20		
			27:23 F	27:23	53:10 F	53:10	62:10 F	62:10				

0	DL@RAC@S	$APTEXT	11: 7 F	11: 7	16:23	16:23 F	18: 1 F	18: 1	23:20	23:20 F
			27:23	27:23 F	53:10	53:10 F	62:10	62:10 F		
377	DL@RAI@M	$APTEXT	11: 8	15:42						
10	DL@RAI@N	$APTEXT	11: 8 F	15:42						
37	DL@RAI@P	$APTEXT	11: 8 F	11: 8	15:42 F	15:42				
10	DL@RAI@S	$APTEXT	11: 8 F	15:42	15:42 F					
177400	DL@RAI@X	$APTEXT	11: 8							
377	DL@RC@M	$APTEXT	32:32							
10	DL@RC@N	$APTEXT	32:32							
26	DL@RC@P	$APTEXT	32:32 F	32:32						
10	DL@RC@S	$APTEXT	32:32	32:32 F						
20	DL@RCC@N	$APTEXT	28:36	28:36 F						
36	DL@RCC@P	$APTEXT	28:36 F	28:36						
0	DL@RCC@S	$APTEXT	28:36	28:36 F						
7	DL@SEC@N	$APTEXT	7:41	7:41 F	53:31	53:31 F				
31	DL@SEC@P	$APTEXT	7:41 F	7:41	53:31 F	53:31				
0	DL@SEC@S	$APTEXT	7:41	7:41 F	53:31	53:31 F				
20	DL@SEQ@N	$APTEXT	11:10 F	51:20 F	53:11	53:11 F				
12	DL@SEQ@P	$APTEXT	11:10 F	11:10	51:20 F	51:20	53:11 F	53:11		
0	DL@SEQ@S	$APTEXT	11:10 F	51:20 F	53:11	53:11 F				
37	DL@SUB@M	$APTEXT	7:34	54:29						
5	DL@SUB@N	$APTEXT	7:34	7:34 F	54:29	54:29 F				
15	DL@SUB@P	$APTEXT	7:34 F	7:34	54:29 F	54:29				
1	DL@SUB@S	$APTEXT	7:34	7:34 F	54:29	54:29 F				
3	DL@TDL@M	$APTEXT	45:42	50:25 F	51: 8 F	56:14	56:19 F	57: 5 F		
2	DL@TDL@N	$APTEXT	45:42	45:42 F	50:25 F	50:25	51: 8 F	51: 8	56:14	56:14 F
			56:19 F	56:19	57: 5 F	57: 5				
21	DL@TDL@P	$APTEXT	45:42 F	45:42	50:25 F	50:25	51: 8 F	51: 8	56:14 F	56:14
			56:19 F	56:19	57: 5 F	57: 5				
7	DL@TDL@S	$APTEXT	45:42	45:42 F	50:25 F	50:25	51: 8 F	51: 8	56:14	56:14 F
			56:19 F	56:19	57: 5 F	57: 5				
37777	DL@TM0@M	$APTEXT	37:17							
16	DL@TM0@N	$APTEXT	37:17							
24	DL@TM0@P	$APTEXT	37:17 F	37:17						
2	DL@TM0@S	$APTEXT	37:17	37:17 F						
20	DL@TM1@N	$APTEXT	37:18	37:18 F						
25	DL@TM1@P	$APTEXT	37:18 F	37:18						
0	DL@TM1@S	$APTEXT	37:18	37:18 F						
2	DL@TM@N	$APTEXT	7:36	7:36 F	17:12	17:12 F	53:12	53:12 F	54: 6	54: 6 F
24	DL@TM@P	$APTEXT	7:36 F	7:36	17:12 F	17:12	53:12 F	53:12	54: 6 F	54: 6
0	DL@TM@S	$APTEXT	7:36	7:36 F	17:12	17:12 F	53:12	53:12 F	54: 6	54: 6 F
20	DL@TOI@N	$APTEXT	7:42	7:42 F	10:17	10:17 F	22: 5	22: 5 F	23:17	23:17 F

			25: 2	25: 2 F	27:24	27:24 F	45:37	45:37 F	53: 8	53: 8 F		
			62: 6	62: 6 F								
14	DL@IOI@P	$APIEXI	7:42 F	7:42	10:17 F	10:17	22: 5 F	22: 5	23:17 F	23:17		
			25: 2 F	25: 2	27:24 F	27:24	45:37 F	45:37	53: 8 F	53: 8		
			62: 6 F	62: 6								
0	DL@IOI@S	$APIEXI	7:42	7:42 F	10:17	10:17 F	22: 5	22: 5 F	23:17	23:17 F		
			25: 2	25: 2 F	27:24	27:24 F	45:37	45:37 F	53: 8	53: 8 F		
			62: 6	62: 6 F								
4	DL@UNS@N	$APIEXI	37:21	37:21 F	60:19	60:19 F						
20	DL@UNS@P	$APIEXI	37:21 F	37:21	60:19 F	60:19						
0	DL@UNS@S	$APIEXI	37:21	37:21 F	60:19	60:19 F						
3	DL@UNI@M	$APIEXI	7:38	53:28								
2	DL@UNI@N	$APIEXI	7:38	7:38 F	53:28	53:28 F						
27	DL@UNI@P	$APIEXI	7:38 F	7:38	53:28 F	53:28						
7	DL@UNI@S	$APIEXI	7:38	7:38 F	53:28	53:28 F						
0X	DMGA		4:17 D	32:29 D	32:29	32:36 D	32:36					
20	DP@ACY@N	$APIEXI	54:41	54:41 F								
10	DP@ACY@P	$APIEXI	54:41 F	54:41								
0	DP@ACY@S	$APIEXI	54:41	54:41 F								
777	DP@ASE@M	$APIEXI	54:32									
11	DP@ASE@N	$APIEXI	54:32									
7	DP@ASE@P	$APIEXI	54:32 F	54:32								
7	DP@ASE@S	$APIEXI	54:32	54:32 F								
20	DP@DCY@N	$APIEXI	29:41	29:41 F	55: 1	55: 1 F						
6	DP@DCY@P	$APIEXI	29:41 F	29:41	55: 1 F	55: 1						
0	DP@DCY@S	$APIEXI	29:41	29:41 F	55: 1	55: 1 F						
777	DP@DSE@M	$APIEXI	54:34									
11	DP@DSE@N	$APIEXI	54:34									
5	DP@DSE@P	$APIEXI	54:34 F	54:34								
7	DP@DSE@S	$APIEXI	54:34	54:34 F								
17	DP@IDX@M	$APIEXI	29:15									
4	DP@IDX@N	$APIEXI	29:15									
4	DP@IDX@P	$APIEXI	29:15 F	29:15								
14	DP@IDX@S	$APIEXI	29:15	29:15 F								
1	DP@SPI@N	$APIEXI	29:12	29:12 F								
4	DP@SPI@P	$APIEXI	29:12 F	29:12								
0	DP@SPI@S	$APIEXI	29:12	29:12 F								
0	DP@SPI@P	$APIEXI	29:17									
0X	DS		4:18 D	32:27	32:35	57: 7						
4	DSK$DD39	$APIEXI	13: 5	41: 4	41:40							
6	DSK$DD40	$APIEXI	13: 7	30:35	39:16	41: 4	41:40	45: 4	45:13	48:29		
3	DSK$DD49	$APIEXI	13: 9	29:39	41: 6	42: 1						

5	DSK$DD50	$APTEXT	13: 9	29:39	41: 8	42: 4				
0X	DSUM		57: 9 D	57: 9						
0X	DT		4:19 D	57:27						
0X	ED		4:21 D	6:25	7:12	9:36	16:30	51:23	51:30	51:32
			58: 8	62:15	62:23					
0X	EDQU		6:26 D	6:26	7:13 D	7:13	16:31 D	16:31	58: 9 D	58: 9
			62:16 D	62:16						
0X	EG		4:24 D	43:39						
0X	EH		4:22 D	7:14	9:37	32:39	43:38	52: 1	58:10	
0X	EI		4:23 D	32:40	57:10					
0X	EMSGIOP		4:25 D	32:41 D	32:41	57:11 D	57:11			
0X	ENQU		4:26 D	9:38 D	9:38	52: 2 D	52: 2	62:24 D	62:24	
0X	ESWP		4:27 D	5:32 D	5:32					
32	F$BGET	$APTEXT	38:11							
33	F$BREI	$APTEXT	40:12							
55	F$CREATE	$APTEXT	13:15	42:36						
7	F$PAUSE	$APTEXT	18:37							
2	F$POP	$APTEXT	6:31							
27	F$RDAL	$APTEXT	6:35	16:33	62:21					
45	F$TRANSF	$APTEXT	36:42							
0	FS$IN	$APTEXT	18: 3							
1	FS$OUT	$APTEXT	16: 2							
0	FS$WAIT	$APTEXT	36:42							
0X	HB		4:28 D	33:42	36:19					
0X	HC		4:29 D	34: 2	36:20					
0X	HILOCL		4:30 D	58:17						
0X	HMA		4:31 D	34: 1 D	34: 1					
0X	HMD		4:32 D	36:21 D	36:21					
0X	HOLOCL		4:33 D	58:19						
4	IM$RESP	$APTEXT	50:24	51: 7	56:10	56:18				
5	IM$STAT	$APTEXT	6:17	32:26	32:31	50:24	51: 7	56:18	56:34	
3	IMSTRAN	$APTEXT	56:40							
0X	IMDAL		4:34 D	56:37						
1	LB@CP@M	$APTEXT	10:25 F	44:26 F	46:31 F	59: 6	59: 9 F			
1	LB@CP@N	$APTEXT	10:25 F	10:25	44:26 F	46:31 F	46:31	59: 6	59: 6 F	59: 9 F
2	LB@CP@P	$APTEXT	10:25 F	10:25	44:26 F	44:26	46:31 F	46:31	59: 6 F	59: 6
			59: 9 F	59: 9						
1	LB@CP@S	$APTEXT	10:25 F	10:25	44:26 F	46:31 F	46:31	59: 6	59: 6 F	59: 9 F
137777	LB@CP@X	$APTEXT	44:26	59: 9						
20	LB@DAL@N	$APTEXT	51:22 F	54: 2	54: 2 F					
4	LB@DAL@P	$APTEXT	51:22 F	51:22	54: 2 F	54: 2				
0	LB@DAL@S	$APTEXT	51:22 F	54: 2	54: 2 F					

20	LB@DCB@N	$APIEXT	54: 3	54: 3 F								
5	LB@DCB@P	$APIEXT	54: 3 F	54: 3								
0	LB@DCB@S	$APIEXT	54: 3	54: 3 F								
1	LB@FUL@M	$APIEXT	51:41 F	59: 4								
1	LB@FUL@N	$APIEXT	15: 7	15: 7 F	40:10	40:10 F	44:35	44:35 F	46:28	46:28 F		
			51:41 F	51:41	59: 4 F	59: 4						
2	LB@FUL@P	$APIEXT	15: 7 F	15: 7	40:10 F	40:10	44:35 F	44:35	46:28 F	46:28		
			51:41 F	51:41	59: 4 F	59: 4						
0	LB@FUL@S	$APIEXT	15: 7	15: 7 F	40:10	40:10 F	44:35	44:35 F	46:28	46:28 F		
			51:41 F	51:41	59: 4	59: 4 F						
1	LB@IOI@M	$APIEXT	54:15	59: 5 F								
1	LB@IOI@N	$APIEXT	54:15	54:15 F	59: 5 F							
2	LB@IOI@P	$APIEXT	54:15 F	54:15	59: 5 F	59: 5						
2	LB@IOI@S	$APIEXT	54:15	54:15 F	59: 5 F							
157777	LB@IOI@X	$APIEXT	59: 5									
6	LB@LF	$APIEXT	15: 4	44:27	55:24							
20	LB@LOC@N	$APIEXT	38: 8	38: 8 F	38:19 F	40: 7	40: 7 F	40:13 F	48:16	48:16 F		
			50:34	50:34 F	58:11	58:11 F	60:14	60:14 F	61:12	61:12 F		
1	LB@LOC@P	$APIEXT	38: 8 F	38: 8	38:19 F	38:19	40: 7 F	40: 7	40:13 F	40:13		
			48:16 F	48:16	50:34 F	50:34	58:11 F	58:11	60:14 F	60:14		
			61:12 F	61:12								
0	LB@LOC@S	$APIEXT	38: 8	38: 8 F	38:19 F	40: 7	40: 7 F	40:13 F	48:16	48:16 F		
			50:34	50:34 F	58:11	58:11 F	60:14	60:14 F	61:12	61:12 F		
20	LB@SEQ@N	$APIEXT	51:21 F	54: 1	54: 1 F							
3	LB@SEQ@P	$APIEXT	51:21 F	51:21	54: 1 F	54: 1						
0	LB@SEQ@S	$APIEXT	51:21 F	54: 1	54: 1 F							
777	MEMSLIM	$APIEXT	19:29	20:17	20:25	21:11	21:41	22:11	23:37			
20	MEM@BM@N	$APIEXT	24: 8	24: 8 F	28:11	28:11 F						
4	MEM@BM@P	$APIEXT	22:37	24: 8 F	24: 8	28:11 F	28:11	51:23				
0	MEM@BM@S	$APIEXT	24: 8	24: 8 F	28:11	28:11 F						
20	MEM@HI@N	$APIEXT	21:12	21:12 F	28: 3	28: 3 F						
0	MEM@HI@P	$APIEXT	21: 7	21:12 F	28: 3 F	51:30						
0	MEM@HI@S	$APIEXT	21:12	21:12 F	28: 3	28: 3 F						
20	MEM@HO@N	$APIEXT	22: 1	22: 1 F	28: 5	28: 5 F						
2	MEM@HO@P	$APIEXT	21:35	22: 1 F	22: 1	28: 5 F	28: 5					
0	MEM@HO@S	$APIEXT	22: 1	22: 1 F	28: 5	28: 5 F						
20	MEM@SI@N	$APIEXT	19:30	19:30 F	28: 7	28: 7 F						
6	MEM@SI@P	$APIEXT	19:25	19:30 F	19:30	28: 7 F	28: 7	51:32				
0	MEM@SI@S	$APIEXT	19:30	19:30 F	28: 7	28: 7 F						
20	MEM@SO@N	$APIEXT	20:18	20:18 F	28: 9	28: 9 F						
10	MEM@SO@P	$APIEXT	20:11	20:18 F	20:18	28: 9 F	28: 9					
0	MEM@SO@S	$APIEXT	20:18	20:18 F	28: 9	28: 9 F						

1	MIO$BMR	$APTEXT	34: 6								
2	MIO$BMRI	$APTEXT	24:27		24:41		25:29				
3	MIO$BMW	$APTEXT	36:15								
4	MIO$BMWI	$APTEXT	25:22		25:28						
6	MIO$HSII	$APTEXT	21:19								
10	MIO$HSOI	$APTEXT	22:21								
12	MIO$SSII	$APTEXT	19:37								
14	MIO$SSOI	$APTEXT	20:33								
0	MIOP	$APTEXT	32:40								
5	MOS	$APTEXT	22:36		58:14		58:42				
0X	O$D3ERR		4:35	D	13: 6						
0X	O$D40ERR		4:36	D	13: 8						
0X	O$D4DEM		1: 4	D	1: 4						
0X	O$D4ERR		4:37	D	13:10						
0X	O$DDIAG		42:36	D	42:36						
1	OL$ECE	$APTEXT	30: 2								
2	OL$PCE	$APTEXT	30: 5								
17	OV@NUM@N	$APTEXT	1: 4								
1	OV@TYP@N	$APTEXT	1: 4								
26	PT$BLOCK	$APTEXT	60:31								
112	PT$BPATH	$APTEXT	53:15		54:13						
50	PT$DD49	$APTEXT	48:17		56:31						
107	PT$DEV	$APTEXT	13:12		41:11		42: 8				
0	PT$HUH	$APTEXT	13:16		42:37						
0X	QTIME		4:38	D	43:40	D	43:40				
5	S$NCLR	$APTEXT	43:16								
0X	SILOCL		4:39	D	58:21						
11	SM@FST@N	$APTEXT	1: 4								
7	SM@NUM@N	$APTEXT	1: 4								
0X	SOLOCL		4:40	D	58:23						
53	TF$D4HD	$APTEXT	39:29								
54	TF$D4IO	$APTEXT	47:28								
52	TF$D4SK	$APTEXT	30:28								
1	TM$BMR	$APTEXT	8:13		17:28		23: 3		23: 8	24:21	28:38
0	TM$CMEM	$APTEXT	16: 5		51:29						
2	TM$SSD	$APTEXT	51:31								
20	TM@IOP@N	$APTEXT	53:16		53:16	F	54:14		54:14	F	
5	TM@IOP@P	$APTEXT	53:16	F	53:16		54:14	F	54:14		
0	TM@IOP@S	$APTEXT	53:16		53:16	F	54:14		54:14	F	
0X	TMCB		4:41	D	53:13		54:11				
20	TR@DON@N	$APTEXT	56:42	F							
13	TR@DON@P	$APTEXT	56:42	F	56:42						

0	IR@DON@S	$APIEXT	56:42 F	
20	IR@MCO@N	$APIEXT	57: 3 F	
34	IR@MCO@P	$APIEXT	57: 3 F	57: 3
0	IR@MCO@S	$APIEXT	57: 3 F	
20	IR@MC1@N	$APIEXT	57: 4 F	
35	IR@MC1@P	$APIEXT	57: 4 F	57: 4
0	IR@MC1@S	$APIEXT	57: 4 F	
20	IR@MDL@N	$APIEXT	56:41 F	
11	IR@MDL@P	$APIEXT	56:41 F	56:41
0	IR@MDL@S	$APIEXT	56:41 F	

SYMBOL QUALIFIER IS D4DEM

1	$CONST	9:33	D	9:33	F	9:33		10:25	D	10:25	F	10:25		10:42	D	10:42	F						
		10:42		11:23	D	11:23	F	11:24	D	11:24	F	11:24		11:38	D	11:38	F						
		11:38		12:30	D	12:30	F	12:33	D	12:33	F	12:35	D	12:35	F	13:17	D						
		13:17	F	13:17		13:18	D	13:18	F	13:30	D	13:30	F	14:33	D	14:33	F						
		14:33		15:13	D	15:13	F	15:19	D	15:19	F	15:20	D	15:20	F	16:29	D						
		16:29	F	23:32	D	23:32	F	23:32		26:19	D	26:19	F	26:19		29:10	D						
		29:10	F	30:23	D	30:23	F	30:23		32:31	D	32:31	F	32:31		32:37	D						
		32:37	F	38:14	D	38:14	F	38:14		39:26	D	39:26	F	39:26		40:13	D						
		40:13	F	41:17	D	41:17	F	41:17		42:13	D	42:13	F	42:13		42:17	D						
		42:17	F	42:17		42:18	D	42:18	F	42:18		42:38	D	42:38	F	42:38							
		43:16	D	43:16	F	43:16		43:17	D	43:17	F	43:17		44:26	D	44:26	F						
		44:31	D	44:31	F	44:31		44:32	D	44:32	F	44:38	D	44:38	F	44:38							
		45: 5	D	45: 5	F	45: 5		45:10	D	45:10	F	45:10		45:17	D	45:17	F						
		45:17		45:22	D	45:22	F	45:22		46:31	D	46:31	F	46:31		46:39	D						
		46:39	F	46:39		50:25	D	50:25	F	50:25		51: 8	D	51: 8	F	51: 8							
		51:41	D	51:41	F	51:41		56:10	D	56:10	F	56:10		56:19	D	56:19	F						
		56:19		56:40	D	56:40	F	56:40		56:42	D	56:42	F	57: 5	D	57: 5	F						
		57: 5		59: 1	D	59: 1	F	59: 1		59: 5	D	59: 5	F	59: 9	D	59: 9	F						
		62:13	D	62:13	F	62:13																	
7	$FIRST	11: 4	D	11: 4																			
514	$REGORG	2:10	D	2:10	F	2:11		2:11	D	2:12		2:12	D	2:13		2:13	D						
		2:14		2:14	D	2:15		2:15	D	2:15	F	2:16		2:16	D	2:17							
		2:17	D	2:18		2:18	D	2:19		2:19	D	2:20		2:20	D	2:21							
		2:21	D	2:21	F	2:22		2:22	D	2:23		2:23	D	2:24		2:24	D						
		2:25		2:25	D	2:26		2:26	D	2:26	F	2:27		2:27	D	2:28							
		2:28	D	2:29		2:29	D	2:30		2:30	D	2:31		2:31	D	2:32							
		2:32	D	2:33		2:33	D	2:34		2:34	D	2:35		2:35	D	2:36							
		2:36	D	2:37		2:37	D	2:38		2:38	D	2:39		2:39	D	2:40							
		2:40	D	2:41		2:41	D	2:42		2:42	D	3: 1		3: 1	D	3: 2							

		3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6
		3: 6 D	3: 7	3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3:10
		3:10 D	3:11	3:11 D	3:12	3:12 D	3:13	3:13 D	3:14
		3:14 D	3:15	3:15 D	3:16	3:16 D	3:17	3:17 D	3:18
		3:18 D	3:19	3:19 D	3:20	3:20 D	3:21	3:21 D	3:22
		3:22 D	3:23	3:23 D	3:24	3:24 D	3:25	3:25 D	3:26
		3:26 D	3:27	3:27 D	3:28	3:28 D	3:29	3:29 D	3:30
		3:30 D	3:31	3:31 D	3:32	3:32 D	3:33	3:33 D	3:34
		3:34 D	3:35	3:35 D	3:36	3:36 D	3:36 F	6:31	6:35
		13:15	16:33	18:37	36:42	36:42 F	38:11	40:12	42:36
		62:21							
0	%NP	1: 4	2:26 D						
430	%P	1: 4	2:26 D	2:26	36:42 F	36:42			
410	%S1	2:10	2:11 D	5:24	5:26	6:12	6:15	6:29	7:17
		7:25	7:26	7:27	7:30	7:32	7:34	7:36	7:38
		7:39	7:40	7:41	7:42	8:14	8:16	8:18	8:20
		8:21	8:22	8:23	8:28	8:29	8:30	9:17	9:18
		9:20	9:21	9:25	9:33	9:40	9:41	10: 5	10: 8
		10:10	10:14	10:17	10:18	10:20	10:21	10:23	10:25
		10:33	10:34	10:37	10:39	10:42	11: 5	11: 7	11: 8
		11: 9	11:10	11:17	11:18	11:19	11:37	11:38	12:26
		12:30	12:33	12:34	12:35	12:38	12:39	13: 1	13:18
		13:27	13:30	13:31	13:42	14: 2	14:33	14:37	14:40
		14:41	15: 2	15: 7	15:10	15:13	15:19	15:20	15:23
		15:42	16:12	16:13	16:15	16:26	16:29	17: 2	17: 3
		17: 4	17:10	17:11	17:12	17:24	17:25	17:32	17:33
		18: 1	18: 6	19:30	20:18	20:22	22: 1	22: 5	22: 8
		22:40	22:42	23:14	23:17	23:20	23:32	24: 8	24:15
		24:22	24:23	24:31	24:35	25: 2	25:13	25:15	25:31
		25:32	26:19	27:20	27:23	27:24	27:26	28: 5	28: 7
		28: 9	28:11	28:31	28:32	28:33	28:34	28:35	28:36
		28:42	29: 1	29: 4	29: 5	29: 6	29:10	29:12	29:15
		29:24	29:27	29:31	29:33	29:40	29:41	30:13	30:16
		30:22	30:24	30:25	30:34	30:36	30:41	31: 1	31:33
		32:13	32:18	32:19	32:20	32:21	32:24	32:31	32:32
		32:37	32:39	33:29	33:30	33:31	33:34	33:35	33:37
		33:38	34:14	34:17	35:30	35:32	36: 4	36: 6	37:17
		37:18	37:21	38: 5	38: 8	38:14	38:19	38:24	39:13
		39:14	39:25	40: 4	40: 7	40:10	40:13	40:33	40:35
		41: 2	41:16	41:35	42:12	42:14	42:17	42:18	42:19
		43:16	43:17	44:21	44:23	44:24	44:26	44:31	44:32
		44:35	44:38	45: 5	45:17	45:25	45:26	45:27	45:35

		45:37	45:41	45:42	46:11	46:13	46:28	46:30	46:31
		46:38	46:39	47: 5	47: 6	48:16	49: 5	49:11	49:40
		49:41	50: 8	50: 9	50:14	50:22	50:25	50:32	50:34
		50:42	51: 1	51: 5	51: 8	51:20	51:21	51:22	51:27
		51:41	53: 7	53: 8	53: 9	53:10	53:11	53:12	53:13
		53:14	53:16	53:28	53:29	53:30	53:31	54: 1	54: 2
		54: 3	54: 6	54: 8	54:11	54:12	54:14	54:15	54:28
		54:29	54:32	54:34	54:37	54:38	54:41	55: 1	55:19
		55:22	56: 5	56: 6	56:10	56:14	56:16	56:19	56:26
		56:33	56:40	56:41	56:42	57: 1	57: 2	57: 3	57: 4
		57: 5	58:11	58:18	58:22	59: 1	59: 5	59: 6	59: 9
		59:10	59:11	59:12	59:13	59:14	59:15	59:16	59:17
		59:18	59:19	59:24	59:30	60:10	60:11	60:13	60:14
		60:19	60:29	61:12	62: 6	62:10	62:13	62:17	
411	%S2	2:10	2:12 D						
412	%S3	2:10	2:13 D						
413	%S4	2:10	2:14 D						
414	%S5	2:10	2:15 D						
415	%I1	2:16 D	9:19	11: 4	11: 6	16:23	18: 2	23:21	34:41
		37:28	48:18	56:38	56:39	59: 4			
416	%I2	2:17 D	9:19	11: 6	16:23	18: 2	23:21	48:18	56:38
		56:39	59: 4						
417	%I3	2:18 D	9:19	11: 6	16:23	18: 2	23:21	48:18	56:39
		59: 4							
420	%I4	2:19 D							
421	%I5	2:20 D							
422	%I6	2:21 D							
423	%W1	2:22 D	7:38	8:20	8:32	9:29	12:26	12:27	12:29
		14:37	14:39	15:40	16:13	16:26	16:28	17:11	17:12
		17:32	17:34	17:39	23:15	23:20	23:21	25:25	27:20
		27:22	28:39	28:42	29: 4	29:15	29:16	29:18	29:40
		30: 1	30: 4	30:13	30:28	30:34	31: 7	31:13	34:18
		34:41	34:42	35:40	38:24	38:29	39:13	39:29	41:14
		41:16	41:41	42: 2	42: 5	42:12	44:22	44:26	44:27
		44:29	44:30	44:34	44:37	45: 2	45:14	45:15	45:16
		46:10	46:13	46:28	46:30	49:12	49:14	49:15	50:22
		50:24	51: 5	51: 7	56:14	56:25	57: 1	57: 3	58:15
		58:17	58:19	58:21	58:23	58:26	60:11	60:27	
424	%W2	2:23 D	6:30	6:31	7:39	8: 8	8:21	8:33	9:30
		10:17	10:20	13: 6	13: 8	13:10	13:15	16: 6	16: 8
		16:12	28:40	29: 1	29: 5	29:17	29:18	29:19	29:21
		29:24	30: 2	30: 5	30:38	30:40	31: 1	31: 2	31: 6

		31:10	34:40	34:41	34:42	37:27	37:28	37:29	38:25
		38:28	38:31	39:22	39:25	42:36	44:23	44:28	46:11
		46:13	56:16	56:18	56:26	56:29	56:30	56:33	57: 2
		57: 4	58:11	58:26	60:14	60:29			
425	%W3	2:24 D	7:40	8: 4	8: 7	8:22	8:34	8:36	9: 5
		9: 9	9:31	10:18	10:20	16:16	29: 9	29:12	29:27
		29:41	30: 1	30: 4	38:27	38:29	38:30	44:24	44:29
		47:26	47:28	60:15	60:18	60:20	60:22	60:27	
426	%W4	2:25 D	7:41	8: 1	8: 2	8: 3	8:23	8:36	9: 1
		9: 4	9: 9	9:32	37:22	37:24	47:24	47:28	60:20
		60:22	60:27	60:31					
427	%W5	2:26 D	7:42	8: 1	8:30	8:42	30:26	37:21	37:23
		37:25	39:27	41:18	42:22	43:39	45:28	50:33	52:10
		60:19	60:21	60:23	60:30	60:31			
1321	ABORTRA	9:41	10: 6	11: 1	11:35 D				
10061	ACTDEM	16:22	18:11	18:32	23:22	27:33	38:15	40:11	43:18
		57:25 D	59:31	62:29					
2552	AGAIN	19:15 D	24: 3	24:33	24:38	26: 8	26:22		
462	BINFO	3:11 D	34:15	34:17	35:32	35:34			
10422	BLCK	20:29	22:17	23: 1	24:23	25:21	27:18	60: 8 D	
10453	BLCK1	16:17	60:16 D						
463	BPTR	3:12 D	9:22	9:23	9:24	9:25	15:14	15:39	15:40
		17:39	24:37	25:13	25:15	33:34	34:17	35:23	35:24
		35:30	35:32						
455	BUFL	3: 6 D	25:32	33:30	34:42	36:42	61:11	61:34	
454	BUFU	3: 5 D	25:31	33:29	34:42	36:42	61:10	61:33	
7255	CALC	11:16	32:17	52:27 D					
432	CHAN	2:29 D	7:17	7:18	12:38	13:15	43:11		
10071	CHANCHK	20:12	21:36	22:38	27:17	58: 6 D			
6120	CHKCP	44:25 D	44:30						
10201	CHKDONE	58:26	58:36 D						
6022	CLEAR	30:18	39:11	40:42	41:37	43:10 D	44:19		
470	CODE	3:17 D	16: 2	18: 3	36:42				
6526	CONT1	46: 8	46:14	46:24	46:32	46:41 D			
6406	CONTINUE	46: 4 D	47:10	59:27					
471	CPD	3:18 D	28:33	54:41	55: 1				
457	CPL	3: 8 D	36:42	37:18	37:29	61:34			
456	CPU	3: 7 D	36:42	37:17	37:29	61:33			
5153	CPUCALC	36:41	37:16 D	61:32					
5767	CRODIAG	30: 3	30: 6	42:35 D					
443	CYI	2:38 D	8:33	9:30	11:19	29:16	30: 1	30: 4	30:15
		30:20	30:22	30:24	30:28	32:20	47:26	52:34	53:29

		59:15							
433	DAL	2:30 D	6:11	6:12	6:15	6:30	6:35	7:14	7:17
		9:33	9:37	10:10	10:14	10:16	10:21	10:39	11: 7
		11: 8	11: 9	11:10	13:31	14: 2	14: 5	15:23	15:42
		16:13	16:23	16:33	16:41	17: 1	17: 5	18: 1	18: 8
		20:24	22: 5	22:10	23:17	23:20	24:14	24:15	25: 2
		25:21	25:31	25:32	27:20	27:23	27:24	28:36	31:33
		32:14	32:24	32:27	32:31	32:32	32:35	32:37	32:39
		37:17	37:18	37:21	38:24	45:33	45:35	45:37	45:42
		47: 5	50: 5	50:22	50:25	50:39	51: 5	51: 8	51:20
		51:22	53: 7	53: 8	53: 9	53:10	53:11	53:12	53:28
		53:29	53:30	53:31	54: 2	54: 5	54: 6	54:29	54:38
		56:10	56:14	56:16	56:19	56:28	56:29	56:39	56:41
		57: 5	59:21	59:24	60:11	60:19	62: 6	62:10	62:17
		62:21							
7353	DALADD	32:15	52:23	53:27 D					
7274	DALPAR	10:12	15:27	17: 7	31:34	50: 7	50:41	52:24	53: 6 D
167	DALQ	5:11	7: 8 D						
10515	DATAMOVE	19:38	20:34	21:20	22:22	25:37	61: 7 D		
431	DCB	2:28 D	7:19	7:20	7:27	8:14	8:16	8:18	8:20
		8:21	8:22	8:23	8:30	9:17	9:18	9:19	9:20
		9:21	9:25	9:36	9:39	10: 8	10:10	10:14	10:23
		10:34	10:37	10:42	11: 5	11: 6	11:17	11:18	11:19
		11:23	11:24	11:38	12:23	12:26	12:33	12:34	12:35
		12:38	12:39	13: 1	13:15	13:17	13:18	13:22	13:27
		13:30	13:31	13:42	14: 2	14: 3	14:33	14:37	14:40
		14:41	15: 2	15:10	15:13	15:19	15:20	15:23	16:26
		16:29	16:30	17:11	17:32	17:33	18: 2	18:36	20:22
		22: 8	23:21	23:32	26:16	26:19	28:31	28:32	28:33
		28:34	28:35	28:42	29: 1	29: 4	29: 5	29: 6	29:10
		29:24	29:27	29:31	29:33	29:40	30:13	30:16	30:22
		30:23	30:24	30:25	30:28	30:34	30:36	30:41	31: 1
		31:33	32:13	32:18	32:19	32:20	32:21	33:29	33:30
		33:31	33:34	33:35	33:37	33:38	35:30	36: 4	38: 5
		38:14	39:13	39:14	39:25	39:26	39:29	40: 4	40:33
		40:35	41: 2	41:16	41:17	41:35	42:12	42:13	42:14
		42:17	42:18	42:19	42:36	42:38	43:16	43:17	43:38
		44:17	44:21	44:23	44:24	44:31	44:32	44:38	45: 5
		45:10	45:14	45:17	45:22	45:25	45:26	45:27	46:10
		46:11	46:38	46:39	47: 6	48:18	49:11	50:14	50:32
		54: 3	54: 8	54:28	54:37	55:19	55:22	56:26	57: 1
		57: 2	57:27	59: 1	59:10	59:11	59:12	59:13	59:14

		59:15	59:16	59:17	59:18	59:19	59:30	62:13	62:15
1345	DCBQ	5:13	12:11 D						
474	DIA	3:21 D	7:33	8:15	28:31	29: 8	54:38		
464	DINFO	3:13 D	34:12	34:14	36: 6	36: 8			
452	DONE	3: 3 D	14:24	45:31	50: 3	50:12	52:26	53: 9	
61	DONEQ	5:14	6: 8 D						
476	DPI	3:23 D	29:12	29:15	29:17	29:41	54:28	54:32	54:34
		54:41	55: 1						
465	DPTR	3:14 D	15:14	15:39	24:37	25: 4	25: 5	25: 6	25:13
		25:15	25:25	33:35	34:14	35:40	35:41	35:42	36: 4
		36: 6							
440	DTR	2:35 D	15: 9	15:16	56:37	56:38	56:39	56:40	56:41
		56:42	57: 3	57: 4	57: 7				
442	ENTRY	2:37 D	10:23	10:25	38: 5	38: 8	38:19	38:36	40: 4
		40: 7	40:10	40:13	40:17	44:21	44:30	44:35	46:28
		46:31	48:16	50:32	50:34	51:21	51:22	51:41	52: 1
		52:10	55:21	55:22	55:24	59:11			
6732	FILL	10:13	16:42	31:32	49:36 D				
7057	FILLUP	50:36 D	52:10						
441	FLG	2:36 D	8:40	8:42	9:10	9:15	9:39	10: 6	10:33
		10:36	11:14	11:22	11:37	14: 3	14:20	14:32	15:29
		15:31	18:21	19:35	20:31	21:17	22:19	23:12	24:26
		24:37	25:11	25:36	54:15	58:42			
451	FNC	3: 2 D	6:12	6:14	6:29	6:33	7:25	7:31	8:28
		9:40	10: 5	14:11	14:21	15:25	17: 2	17:10	17:20
		19:37	20:33	21:19	22:21	24:27	24:41	25:22	25:28
		25:30	29:35	30:39	31:22	38:23	44:34	45:41	46:30
		47: 5	47: 9	47:24	49:40	50: 8	50:42	51:25	51:40
		53: 7	56: 5	56: 9	59:22	59:24	61: 9		
5244	GETBUFS	31:23	31:28	38: 4 D					
5252	GETBUFSA	38: 7 D	38:36						
10541	GETCPU	19:36	20:32	21:18	22:20	24:28	61:30 D		
4716	GETMOS	15:41	17:40	25:26	34:39 D				
477	GIVE	3:24 D	5:10	5:16	6:18	12:19	23:42		
444	HED	2:39 D	8:34	8:36	9: 9	9:17	9:31	11:18	30:25
		31:13	32:19	39:14	39:17	39:19	39:29	40:38	45:27
		46:23	47:28	48:30	49: 3	52:30	52:33	53:30	59:14
5350	HEDSET	30:17	30:40	31:10	31:13	39:10 D	40:39		
501	HIFLG	3:26 D	19:10	21: 4	21: 5	21:23	23:38		
502	HOFLG	3:27 D	19:11	21:32	21:33	22:25	23:38		
2702	HSPIN	20:17	20:26	21: 2 D					
2762	HSPONXT	21:40 D							

2743	HSPOUT	21:11	21:30 D						
3515	INHS	23: 4	27:19 D						
3511	INHSCH	19:26	21: 8	27:16 D					
4344	INITI	30:35	31:12 D						
3615	INITA	15:21	28:30 D						
4350	INITB	13:34	30:37	30:42	31:20 D				
1607	IOGO	13:32	13:40 D						
7405	LBPAR	19:33	20:21	21:15	22: 4	24:12	53:42 D	58:40	
507	LEN	3:32 D	8:41	8:42	9:13	9:19	9:22		
7471	LIMITS	27:27	52:25	54:27 D					
6605	LMA	44:33	46:35	48:15 D					
6666	LMA1	48:32	48:40 D						
6667	LMA2	48:34	48:41 D						
6670	LMA3	48:36	48:42 D						
6671	LMA4	48:38	49: 1 D						
506	LOCAL	3:31 D	14:40	15: 1	15: 2	15: 4	15: 7	19:30	20:18
		21:12	22: 1	24: 8	38: 6	38:36	40: 5	40:17	54: 1
		54: 2	54: 3	54:15	58:10	58:11	59: 4	59: 5	59: 6
		59: 9	60:14	61:12					
435	MCB	2:32 D	5: 9	5:17	25:41	33:41	34: 2	34: 9	34:12
		34:15	34:18	36:20	36:22				
3467	MCBFULL	25:14	25:16	26:14 D					
2543	MEMIO	5:12	19: 8 D						
3204	MOSDN	23: 5	23:19	23:31 D					
3041	MOSIO	21:41	22:12	22:32 D					
437	MOSI	2:34 D	33:38	34: 8	35:33	36: 7			
3215	MOSNXT	23:12	23:35 D						
436	MOSU	2:33 D	33:37	34: 7	35:33	36: 7			
5120	MOVEIT	16: 3	18: 4	36:40 D					
505	MSFLG	3:30 D	19:14	22:34	22:35	23:38	25:40		
1347	NEXT	12:16 D	13:19	13:23	14: 1	14:34	15:24	15:30	15:36
		15:37	17: 1	17: 3	17: 6	17:13	17:21	17:26	17:29
		17:34	18:26	18:41					
7565	NEXTBUF	10:24	38:35	40:16	46:19	52: 9	55:18 D		
172	NEXTDAT	7:10 D	10: 9	10:22	10:38	11:31	11:41		
6530	NODAL	45:34	45:36	46: 1	47: 3 D				
654	NODICE	8:27	8:29	8:31	8:32	8:33	8:34	8:37	9:28 D
510	NUM	3:33 D	9:20	9:23	9:24	11: 4	11: 5	11: 6	11: 7
		11: 8	11: 9	11:10	11:12	25: 5	25: 6	25:15	33:31
		35:24	35:41	35:42	45:31	45:32	45:38		
430	NUMDONE	2:27 D	12:12	12:18	12:37	19: 9	19:29	19:42	20:17
		20:25	20:38	21:11	21:24	21:41	22:11	22:26	23:37

		24:11							
3563	QCHECK	5:30	17:36	26: 7	28: 2 D				
701	QDAL	7:26	7:31	7:33	7:35	7:37	8:13	8:15	8:17
		8:19	9:15	9:29	9:30	9:31	9:32	9:34 D	
461	QUE	3:10 D	6: 9	6:10	6:11	6:25	7: 9	7:12	19:25
		20:11	21: 7	21:35	22:37	58: 8			
10553	RDDONE	20:13	21:37	23: 9	23:27	62: 5 D			
2062	READAH	14:24	15:26	15:35 D					
4562	READMCB	15:11	15:38	17:38	24:32	33:28 D			
513	REI	3:36 D	12:34	18:23					
5454	RELBUFS	18:22	38:16	40: 3 D					
5462	RELBUFSA	40: 6 D	40:17						
5523	RELEASE	18:23	40:32 D	41:36					
5104	RELMEM	5:17	25:42	36:17 D					
5631	RELX	40:34	41: 1	41:20 D					
2502	REQUEUE	15: 8	15:39	17:37	18: 8	18:10 D			
446	SC0	2:41 D	7:27	7:30	7:32	7:34	7:36	7:38	7:39
		7:40	7:41	7:42	10:34	10:36	11: 4	11:14	15:42
		16: 1	16:16	16:28	40: 7	40:12	45: 8	45:20	45:25
		46:36	46:38	48:30	49: 1	49: 3	49: 6	49: 8	49:15
		49:19	50: 1	50: 3	50:12	50:15	51:18	52: 6	
447	SC1	2:42 D	10:11	10:16	10:17	10:18	16: 1	17:42	37:20
		37:22	37:24	37:28	37:29	47:28	48:16	48:21	48:24
		61:31							
445	SEC	2:40 D	8:36	9: 9	9:18	9:32	11:15	11:17	31:10
		32:16	32:18	45:26	46:20	46:25	47:28	48:31	49: 5
		49:12	52:26	52:28	52:29	53:31	59:13		
5632	SELECT	29:32	29:34	41:34 D					
7605	SENDTR	14:14	17:14	31:26	56: 1 D				
453	SEQ	3: 4 D	14:13	17: 4	17:24	18: 6	22: 7	23:18	25: 3
		27:26	50: 2	50: 3	50:12	50:38	51:19	51:20	51:21
		51:27	53:11	54: 1	60:15	61:31	62: 8		
503	SIFLG	3:28 D	19:12	19:22	19:23	19:41	23:38		
504	SOFLG	3:29 D	19:13	20: 8	20: 9	20:37	23:39		
473	SPT	3:20 D	8: 2	8: 3	9: 3	28:32	31: 3	46:22	49: 5
		49:13	52:28	52:29	54:32	54:34	59:16		
2553	SSDIN	19:20 D							
2636	SSDONXT	20:16 D							
2617	SSDOUT	19:29	20: 6 D						
6	START	5: 7 D	5:21	5:25	5:27	5:29	5:31	18:38	
6072	STARTIT	31:24	31:35	44:16 D					
4501	STAT	6:33	16:32	32:14	32:22 D	62:20			

434	STATUS	2:31 D	31:24	31:27	31:30	38: 8	38:11	38:19	38:27
		56: 2	56:23						
7247	STRIP	7:21	13:33	14: 6	52:22 D				
475	SUB	3:22 D	7:35	8:17	29: 6	49:19	54:29	54:31	59:18
500	TDIR	3:25 D	22:40	22:42	23:14	23:27	24:15	24:17	24:22
		24:31	24:35	25:29					
6056	TIMER	30:27	39:28	41:19	42:23	43:36 D	45:29		
511	TM	3:34 D	7:37	8:13	8:19	16: 5	17:13	17:28	23: 3
		23: 8	24:21	28:34	28:38	36:42	51:29	51:31	53:12
		53:13	54: 6	54: 8	54:11				
512	TMP	3:35 D	8:27	14:11	15: 9	15:10	15:16	15:37	17:28
		23:14	25: 1	28:35	33:33	35:29	36: 3	36:14	50:21
		51: 4	51:25	53:14	53:16	54:12	54:14	56: 3	57:10
450	TOTAL	3: 1 D	11: 4	11:12	11:14	11:15	14:24	16:28	17: 4
		17:24	17:42	18: 6	27:24	27:26	32:16	45:31	50: 2
		50:38	51:27	53: 8					
472	TPC	3:19 D	8: 6	46:24	49: 7	52:32	54:37	59:17	
6555	TRACE	45:30	46:40	47:23 D					
467	TYPE	3:16 D	12:39	13: 5	13: 7	13: 9	29:39	30:35	39:16
		41: 4	41: 6	41: 8	41:40	42: 1	42: 4	45: 4	45:13
		48:29	59:19						
466	UNIT	3:15 D	8:32	9:29	29:34	32:21	41: 2	41: 5	41: 7
		41: 9	41:41	42: 2	42: 5	42:16	42:19	47:26	53:28
		59:12							
460	WBC	3: 9 D	14:39	15:36	16: 1	16:16	17: 4	17:24	17:41
		17:42	18: 1	18: 6	27:23	27:26	51:27	53:10	
4415	WBSTAT	18: 7	27:28	32:12 D					
2517	WHOA	15:12	18:31 D						
4755	WRITB	15:15	35:26 D						
4750	WRITBIOP	35:22 D	37: 1						
5074	WRITD	35:38	36:12 D						
5021	WRITDIOP	23:25	35:39 D						
2253	WRITEBH	15:28	16:40 D						
2333	WRITEBH1	16:41	17:18 D						

```
0     <macro>                    OVERLAY DD49,TYPE=DATA                                    DD49.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'          OVERLAY  1
```

```
                                  ***********************************************************************   DD49.5
                                  *                                                                     *   DD49.6
                                  *    The DD49 overlay is the interrupt handler for DD49 and DD39 type *   DD49.7
                                  *    disks. It is a data type overlay which is dynamically loaded in  *   DD49.8
                                  *    any IOP which has the afore mentioned disk types configured. The *   DD49.9
                                  *    global register %DD49 points to its base address in memory.      *   DD49.10
                                  *                                                                     *   DD49.11
                                  ***********************************************************************   DD49.12
                                                                                                            DD49.13
                                          QUAL                                                              DD49.14
                                          EXT       %DD49                                                   DD49.15
                                          BASEREG   %DD49                                                   DD49.16
  6     <macro>                   %INT    REGDEFS   ,,(DDCB,DDAL,DDAL1,DDAL2,DDAL3,DCYL,DHED,DSEC,DFRST,   DD49.17
                                  ,                 DCUR,DLAST,DNUM,DFNC,DSTAT,SEQ,DDN,DBZ,SPT,TPC,BREAK,   DD49.18
                                  ,                 QU,TOT,CONT,DFLG,DLOC,DNXT,MDAL,DKOVR,STAT0,STAT1,      DD49.19
                                  ,                 TM,TMP,DTYPE)                                       R13739AU.1
                610               %S1     EQUALS    $REGORG                                                 $REG    3
                611               %S2     EQUALS    $REGORG                                                 $REG    3
                612               %S3     EQUALS    $REGORG                                                 $REG    3
                613               %S4     EQUALS    $REGORG                                                 $REG    3
                614               %S5     EQUALS    $REGORG                                                 $REG    3
                615               %T1     EQUALS    $REGORG                                                 $REG    3
                616               %T2     EQUALS    $REGORG                                                 $REG    3
                617               %T3     EQUALS    $REGORG                                                 $REG    3
                620               %T4     EQUALS    $REGORG                                                 $REG    3
                621               %T5     EQUALS    $REGORG                                                 $REG    3
                622               %T6     EQUALS    $REGORG                                                 $REG    3
                623               %W1     EQUALS    $REGORG                                                 $REG    3
                624               %W2     EQUALS    $REGORG                                                 $REG    3
                625               %W3     EQUALS    $REGORG                                                 $REG    3
                626               %W4     EQUALS    $REGORG                                                 $REG    3
                627               %W5     EQUALS    $REGORG                                                 $REG    3
                630               DDCB    EQUALS    $REGORG                                                 $REG    3
                631               DDAL    EQUALS    $REGORG                                                 $REG    3
                632               DDAL1   EQUALS    $REGORG                                                 $REG    3
                633               DDAL2   EQUALS    $REGORG                                                 $REG    3
                634               DDAL3   EQUALS    $REGORG                                                 $REG    3
                635               DCYL    EQUALS    $REGORG                                                 $REG    3
                636               DHED    EQUALS    $REGORG                                                 $REG    3
                637               DSEC    EQUALS    $REGORG                                                 $REG    3
                640               DFRST   EQUALS    $REGORG                                                 $REG    3
                641               DCUR    EQUALS    $REGORG                                                 $REG    3
```

```
642     DLAST   EQUALS  $REGORG                                         $REG    3
643     DNUM    EQUALS  $REGORG                                         $REG    3
644     DFNC    EQUALS  $REGORG                                         $REG    3
645     DSTAT   EQUALS  $REGORG                                         $REG    3
646     SEQ     EQUALS  $REGORG                                         $REG    3
647     DDN     EQUALS  $REGORG                                         $REG    3
650     DBZ     EQUALS  $REGORG                                         $REG    3
651     SPT     EQUALS  $REGORG                                         $REG    3
652     TPC     EQUALS  $REGORG                                         $REG    3
653     BREAK   EQUALS  $REGORG                                         $REG    3
654     QU      EQUALS  $REGORG                                         $REG    3
655     TOT     EQUALS  $REGORG                                         $REG    3
656     CONT    EQUALS  $REGORG                                         $REG    3
657     DFLG    EQUALS  $REGORG                                         $REG    3
660     DLOC    EQUALS  $REGORG                                         $REG    3
661     DNXT    EQUALS  $REGORG                                         $REG    3
662     MDAL    EQUALS  $REGORG                                         $REG    3
663     DKOVR   EQUALS  $REGORG                                         $REG    3
664     STAT0   EQUALS  $REGORG                                         $REG    3
665     STAT1   EQUALS  $REGORG                                         $REG    3
666     TM      EQUALS  $REGORG                                         $REG    3
667     TMP     EQUALS  $REGORG                                         $REG    3
670     DTYPE   EQUALS  $REGORG                                         $REG    3
                                                                  DD49.21
                EXT     %DCBQ                                     DD49.22
                EXT     %ICHAN                                    DD49.23
                EXT     %HISP                                     DD49.24
                EXT     %MEMIO                                    DD49.25
                EXT     %MYID                                     DD49.26
                EXT     %SSD                                      DD49.27
                EXT     CA                                        DD49.28
                EXT     CB                                        DD49.29
                EXT     CC                                        DD49.30
                EXT     CD                                        DD49.31
                EXT     CE                                        DD49.32
                EXT     COMIO                                     DD49.33
                EXT     D$D4DEM                                   DD49.34
                EXT     DCCB                                      DD49.35
                EXT     DT                                        DD49.36
                EXT     DY                                        DD49.37
                EXT     ED                                        DD49.38
                EXT     EQCP                                      DD49.39
```

```
EXT       EDQU                                          DD49.40
EXT       EG                                            DD49.41
EXT       LH                                            DD49.42
EXT       HONN                                          DD49.43
EXT       ICHK                                          DD49.44
EXT       IMCB                                          DD49.45
                                                        DD49.46
ENTRY     D4TOUT                                        DD49.47
ENTRY     DD49                                          DD49.48
```

```
                                 ********************************************************************   DD49.50
                                 *        Common entry for all interrupts                           *   DD49.51
                                 ********************************************************************   DD49.52
                                                                                                        DD49.53
 6                               DD49     *                                                            DD49.54
 6   042000                               C = 1, IOB = DN                .Check done and busy        R13739IA.1
 7   004020                               A = A > D'16                                               R13739IA.2
10   043000                               C = 1, IOB = BZ                                            R13739IA.3
11   103002  040000                       C = 1, A = 0                   .Assume busy if not done    R13739IA.4
13   024647                               R!DDN = A                                                  R13739IA.5
14   004020                               A = A > D'16                                               R13739IA.6
15   024650                               R!DBZ = A                                                  R13739IA.7
16   010000  024645                       R!DSTAT = S$OKAY               .Initial status                DD49.68
20   010000  024663                       R!DkOVR = 0                    .Initialize overrun flag       DD49.69
22   024656                               R!CONT = A                     .Initialize continue flag      DD49.70
23   024653                               R!BREAK = A                    .Initialize break chaining flag DD49.71
24   050000  013020  016000               R!DDCB = B - O'20 + DCCB       .Locate DCB for this channel   DD49.72
    /000000  024630
31   030630  024630                       R!DDCB = (R!DDCB)                                             DD49.73
33    <macro>                             STORE 0,DK@IDL,R!DDCB          .Clear idle flag            R13739AU.2
42    <macro>                             GET R!DTYPE,DK@TYP,R!DDCB      .Get device type            R13739AU.3
51    <macro>                             ADDRESS R!EH,DK@TMO,R!DDCB     .Remove from timer queue       DD49.74
54    <macro>                             RKERN DQTIME                                                  DD49.75
                                                                                                       R13739TA.8
                                 *        The following code deals with a DCU-5 problem that causes the R13739TA.9
                                 *        data transfer to stop prematurely if an IOB:6 is issued when  R13739TA.10
                                 *        the last two words of the next sector are being transferred   R13739TA.11
                                 *        into local memory on a read.  Thus, the following code checks R13739TA.12
                                 *        to see if the data transfer for the next (chained) sector is  R13739TA.13
                                 *        close to the end, and if it is, wait for the transfer to      R13739TA.14
                                 *        complete before issuing the IOB:6.                            R13739TA.15
                                                                                                       R13739TA.16
56    <macro>                             GET R!DFLG,DK@ACT,R!DDCB       .Current activity           R13739TA.17
60   011005                               A = A & 5                                                  R13739TA.18
61    <macro>                             $IF (A # 0) AND (R!DDN = 0)    .If chained read or read-ahead R13739TA.19
                                                                                        .>>>>> 0          $IF      1
66    <macro>                               GET R!%W1,DK@LMA,R!DDCB                                  R13739TA.20
                                                                                                       R13739TA.21
                                 *        keep the following lines of code together to avoid screwing up R13739TA.22
                                 *        the timing with too many lines of code or an out-of-stack jump.R13739TA.23
                                 *        The IOB:6 MUST be within 6 words transfer time (76 CPs) of the R13739TA.24
                                 *        IOB:10/11 in order for this code to work correctly.           R13739TA.25
```

```
                                                                                R13739TA.26
  74    <macro>                 $IF (A # 0)              .LMA 0                 R13739TA.27
                                                                   .>>>>> 1          $IF      1
  76    170000                    IOB: 10                                        R13739TA.28
  77    <macro>                 $ELSE                    .LMA 1                 R13739TA.29
 100    171000                    IOB: 11                                        R13739TA.30
 101    <macro>                 $ENDIF                                          R13739TA.31
                                                                   .<<<<< 1          $ENDIF   1
 101    015000 /003777          A = A & O'3777                                  R13739TA.32
 103    102035                  P = DD49A, A = 0         .If transfer complete  R13739TA.33
 104    017000 /003740          A = A - O'3740                                  R13739TA.34
 106    <macro>                 $IF (C = 0)              .If not close to done  R13739TA.35
                                                                   .>>>>> 1          $IF      1
 110    166000                    IOB: 6                                         R13739TA.36
                                                                                R13739TA.37
                        *       If we didn't issue the IOB:6 before the last two words of the  R13739TA.38
                        *       transfer, the next sector may not have transferred completely. R13739TA.39
                        *       If this has happened, done will have set, so change the flags  R13739TA.40
                        *       DDN and DBZ accordingly.  This will cause a soft overrun to be R13739TA.41
                        *       detected, which will result in the read for the next sector    R13739TA.42
                        *       being re-issued.                                               R13739TA.43
                                                                                R13739TA.44
 111    <macro>                   $IF (IOB = DN)                                R13739TA.45
                                                                   .>>>>> 2          $IF      1
 114    026647                      R!DDN = R!DDN + 1                           R13739TA.46
 115    027650                      R!DBZ = R!DBZ - 1                           R13739TA.47
 116    <macro>                   $ENDIF                                        R13739TA.48
                                                                   .<<<<< 2          $ENDIF   1
                                                                                R13739TA.49
                        *       The data transfer of the chained sector is almost done - wait  R13739TA.50
                        *       for it to complete before issuing the IOB:6.                   R13739TA.51
                                                                                R13739TA.52
 116    <macro>                 $ELSE                    .Close to done         R13739TA.53
 117    010062  024624            R!%W2 = D'50           .Loop count for timeout R13739TA.54
 121    <macro>                   $UNTIL (A = 0)         .Until transfer complete R13739TA.55
 122    027624                      R!%W2 = R!%W2 - 1                           R13739TA.56
 123    126000 /003422              P = D4ERROR, A = 0     .If timed out       R13739TA.57
 125    <macro>                     $IF (R!%W1 # 0)        .LMA 0              R13739TA.58
                                                                   .>>>>> 2          $IF      1
 130    170000                        IOB: 10                                   R13739TA.59
 131    <macro>                     $ELSE                  .LMA 1              R13739TA.60
 132    171000                        IOB: 11                                   R13739TA.61
```

```
133    <macro>                          $ENDIF                                                    R13739TA.62
                                                                                  .<<<<< 2              $ENDIF    1
133   015000 /003777                    A = A & O'3777                                            R13739TA.63
135    <macro>                         $ENDTIL                                                    R13739TA.64
136   166000                           IOB: 6                                                     R13739TA.65
137    <macro>                       $ENDIF                                                        R13739TA.66
                                                                                  .<<<<< 1              $ENDIF    1
                                                                                                    R13739TA.67
                             *       Not a chained read - just issue the IOB:6 and continue.        R13739TA.68
                                                                                                    R13739TA.69
137    <macro>                       $ELSE                          .Not chained read               R13739TA.70
140                          DD49A   *                                                              R13739TA.71
140   166000                           IOB: 6                                                     R13739TA.72
141    <macro>                       $ENDIF                                                        R13739TA.73
                                                                                  .<<<<< 0              $ENDIF    1
141    <macro>                       GET A,DKF@IM,R!DDCB            .Check for immediate return        DD49.76
144   127000 /003533                 P = ERRPOP, A # 0                                                 DD49.77
146    <macro>                       $IF (R!DDN # 0)                .If channel done                   DD49.79
                                                                                  .>>>>> 0              $IF       1
151   172000                           IOB:12                       .Read status 0                     DD49.80
152   024664                           R!STAT0 = A                  .Save                              DD49.81
153   010002  025627                   R!%W5 = R!%W5 + 2            .Delay ~6 clocks                   DD49.82
155   173000                           IOB:13                       .Read status 1                     DD49.83
156   024665                           R!STAT1 = A                  .Save                              DD49.84
157    <macro>                       $ENDIF                                                            DD49.85
                                                                                  .<<<<< 0              $ENDIF    1
157   020647  023650  126000         P = D4ERROR, R!DDN = R!DBZ     .If error detected                 DD49.86
     /003422
163    <macro>                       $IF (R!DDN # 0)                .If channel done                   DD49.87
                                                                                  .>>>>> 0              $IF       1
166   077000 /003226                   R = D4DONE                   .Channel done checks               DD49.88
170   020645  127000 /003507           P = ERROR, R!DSTAT # S$OKAY  .If error detected                 DD49.89
173    <macro>                         STORE 0,DK@ACT,R!DDCB        .Clear activity flags              DD49.90
175    <macro>                       $ELSE                                                             DD49.91
176   010002  024000                   R!EG = D4IOLIM               .Re-queue for timeout of current   DD49.92
200    <macro>                         RKERN QTIME                                                     DD49.93
202   167000                           IOB:7                        .Enable interrupts                 DD49.94
203    <macro>                       $ENDIF                                                            DD49.95
                                                                                  .<<<<< 0              $ENDIF    1
203    <macro>                       STORE 0,DK@CNG,R!DDCB          .Clear chain flag                  DD49.96
                                                                                                    R13739AU.4
212    <macro>                       $IF (R!DFLG # DK$WRIT)         .If this function is not a write R13739AU.5
```

```
                                                                  .>>>>> 0          $IF      1
216   <macro>                  STORE 0,DK@OWC,R!DDCB      .Clear outstanding write count   R13739AU.6
223   <macro>                $ENDIF                                                        R13739AU.7
                                                                  .<<<<< 0          $ENDIF   1
223  020657  011007  024610    P = D4CNTL, 0 = R!DFLG & 7    .If not read or write          DD49.97
     010000  023610  126000
    /002775
                                                                                            DD49.98
                        ************************************************************        DD49.99
                        *                                                          *        DD49.100
                        * Read/write interrupt processing. The primary intent is to sustain *  DD49.101
                        * the current stream by chaining the next contiguous sector if it is *  DD49.102
                        * requested. Also attempt to initiate the data transfer to the target *  DD49.103
                        * memory to maximize overlap with the disk i/o.            *        DD49.104
                        *                                                          *        DD49.105
                        ************************************************************        DD49.106
                                                                                            DD49.107
                          * Adjust current disk address                                    DD49.108
                                                                                            DD49.109
232   <macro>            GET R!DCYL,DK@CYL,R!DDCB      .Current cylinder                    DD49.110
237   <macro>            GET R!DHED,DK@HED,R!DDCB      .Current head group                  DD49.111
246   <macro>            GET R!DSEC,DK@SEC,R!DDCB      .Current sector                      DD49.112
254   <macro>            GET R!SPT,DK@SPT,R!DDCB       .Sectors per track                   DD49.113
262   <macro>            GET R!TPC,DK@TPC,R!DDCB       .Tracks per cylinder                 DD49.114
271  026637              R!DSEC = R!DSEC + 1           .Next sector                         DD49.115
272   <macro>            $IF (A = R!SPT)               .Next head?                          DD49.116
                                                                  .>>>>> 0          $IF      1
275  010000  024637        R!DSEC = 0                  .First sector                        DD49.117
277  026636                R!DHED = R!DHED + 1         .Next head                           DD49.118
300   <macro>              $IF (A = R!TPC)             .Next cylinder?                      DD49.119
                                                                  .>>>>> 1          $IF      1
303  010000  024636          R!DHED = 0                .First head                          DD49.120
305  026635                  R!DCYL = R!DCYL + 1       .Next cylinder                       DD49.121
306   <macro>                $PUNTIF (R!DDN = 0),CODE=PT$DD49 .Cant continue across cyl     DD49.122
311   <macro>              $ENDIF                                                          DD49.123
                                                                  .<<<<< 1          $ENDIF   1
311   <macro>              $IF (R!DKOVR = 0)           .If not overrun                      DD49.124
                                                                  .>>>>> 1          $IF      1
314   <macro>                $IF (R!DTYPE # DSK$DD40) OR (R!DDN = 0)                       R13739AU.8
                                                                  .>>>>> 2          $IF      1
322   <macro>                  PUT R!DHED,DK@HED,R!DDCB .Set new head                      R13739AU.9
336   <macro>                $ENDIF                                                          R13739AU.10
```

```
                                                                                .<<<<< 2          $ENDIF    1
336   <macro>                    $ENDIF                                                        DD49.126
                                                                                .<<<<< 1          $ENDIF    1
336   <macro>                  $ENDIF                                                          DD49.127
                                                                                .<<<<< 0          $ENDIF    1
336   <macro>                  $IF (R!DDN = 0)                .If next in progress             DD49.128
                                                                                .>>>>> 0          $IF       1
341   <macro>                    PUT R!DSEC,DK@SEC,R!DDCB     .Set new current sector          DD49.129
354   <macro>                  $ENDIF                                                          DD49.130
                                                                                .<<<<< 0          $ENDIF    1
354   <macro>                  GET R!TMP,DK@TMP,R!DDCB        .Target memory IOP               DD49.131
363   <macro>                  GET R!TM,DK@TM,R!DDCB          .Target memory type              DD49.132
371   <macro>                  $PUNTIF (A >= TM$MAX),CODE=PT$BADTM .Illegal target memory      DD49.133
                                                                                                 DD49.134
                               * Locate DAL for current sector.                                  DD49.135
                               * If chaining, locate DALS for sectors 2 ahead                    DD49.136
                               * and, if writing DK@NLB ahead.                                   DD49.137
                                                                                                 DD49.138
375   010000  024631           R!DDAL = 0                     .Preset to 0                     DD49.139
377   024632                   R!DDAL1 = A                                                     DD49.140
400   024633                   R!DDAL2 = A                                                     DD49.141
401   024634                   R!DDAL3 = A                                                     DD49.142
402   <macro>                  $IF (R!DFLG # DK$RAHD)         .If not read ahead               DD49.143
                                                                                .>>>>> 0          $IF       1
                                                                                                 DD49.144
                                 * Locate DAL for sector completed                               DD49.145
                                                                                                 DD49.146
406   <macro>                    ADDRESS R!DDAL1,DK@DL0,R!DDCB .DAL queue address              DD49.147
411   010000  024643             R!DNUM = 0                   .Clear number of sectors to do     DD49.148
413   <macro>                    $UNTIL (R!DDAL1 = 0) OR (R!DNUM # 0)                          DD49.149
417   030632  024632               R!DDAL1 = (R!DDAL1)                                         DD49.150
421   <macro>                      $IF (A # 0)                                                 DD49.151
                                                                                .>>>>> 1          $IF       1
423   <macro>                        GET R!%W1,DL@DON,R!DDAL1 .Number of sectors done          DD49.152
430   <macro>                        GET R!%W2,DL@TOT,R!DDAL1 .Total to do                     DD49.153
435   020624  023623  024643         R!DNUM = R!%W2 - R!%W1   .Compute number left to do       DD49.154
440   <macro>                      $ENDIF                                                      DD49.155
                                                                                .<<<<< 1          $ENDIF    1
440   <macro>                    $ENDTIL                                                       DD49.156
441   020632  024631             R!DDAL = R!DDAL1             .Current working DAL               DD49.157
443   <macro>                    $PUNTIF (A = 0),CODE=PT$DD49                                  DD49.158
445   <macro>                    GET R!DFNC,DL@FC,R!DDAL      .Function code                    DD49.159
```

```
453    <macro>                    FLDADD 1,DL@DON,R!DDAL                                          DD49.160
457    027643                     R!DNUM = R!DNUM - 1          .Compensate for completed sector   DD49.161
460    030631  024662             R!MDAL = (R!DDAL)            .Keep track whether more DALS      DD49.162
                                                                                                   DD49.163
                                  * If chaining, locate DAL containing next continue sector      DD49.164
                                                                                                   DD49.165
462    <macro>                    $IF (R!DDN = 0)              .If chaining                       DD49.166
                                                                                  .>>>>> 1            $IF      1
465    020632  024633               R!DDAL2 = R!DDAL1          .Preset to current                 DD49.167
467    020643  024623               R!%W1 = R!DNUM             .Number left to do in this DAL     DD49.168
471    <macro>                      $UNTIL (R!%W1 >= 2) OR (R!DDAL2 = 0)                          DD49.169
476    030633  024633                 R!DDAL2 = (R!DDAL2)      .Next DAL                          DD49.170
500    <macro>                        $IF (A # 0)                                                  DD49.171
                                                                                  .>>>>> 2            $IF      1
502    <macro>                          GET A,DL@CON,R!DDAL2   .Check for connected               DD49.172
510    <macro>                          $IF (A # 0)            .If it is                          DD49.173
                                                                                  .>>>>> 3            $IF      1
512    <macro>                            GET A,DL@TOT,R!DDAL2 .Add to total to do                DD49.174
516    025623                             R!%W1 = A + R!%W1                                       DD49.175
517    <macro>                          $ELSE                                                     DD49.176
520    010000  024633                     R!DDAL2 = 0          .End search                        DD49.177
522    010001  024653                     R!BREAK = 1          .Break chaining                    DD49.178
524    <macro>                          $ENDIF                                                    DD49.179
                                                                                  .<<<<< 3            $ENDIF   1
524    <macro>                        $ENDIF                                                      DD49.180
                                                                                  .<<<<< 2            $ENDIF   1
524    <macro>                      $ENDTIL                                                       DD49.181
                                                                                                   DD49.182
                                    * Locate next write behind sector                              DD49.183
                                                                                                   DD49.184
525    <macro>                      $IF (R!DFLG = DK$WRIT) AND (R!DDAL2 # 0)                      DD49.185
                                                                                  .>>>>> 2            $IF      1
534    020633  024634                 R!DDAL3 = R!DDAL2        .Preset to current                 DD49.186
536    <macro>                        GET R!%W2,DK@NLB,R!DDCB .Number of local buffers            DD49.187
544    <macro>                        $UNTIL (R!%W1 >= R!%W2) OR (R!DDAL3 = 0)                    DD49.188
551    030634  024634                   R!DDAL3 = (R!DDAL3)    .Next DAL                          DD49.189
553    <macro>                          $IF (A # 0)            .If one                            DD49.190
                                                                                  .>>>>> 3            $IF      1
555    <macro>                            GET A,DL@CON,R!DDAL3 .Check if connected                DD49.191
563    <macro>                            $IF (A # 0)          .If it is                          DD49.192
                                                                                  .>>>>> 4            $IF      1
565    <macro>                              GET A,DL@TOT,R!DDAL3 .Add to total left to do         DD49.193
```

```
571   025623                          R!%W1 = A + R!%W1                                   DD49.194
572    <macro>                      $ELSE                                                 DD49.195
573   010000  024634                  R!DDAL3 = 0        .End search                      DD49.196
575    <macro>                      $ENDIF                                                DD49.197
                                                                        .<<<<< 4              $ENDIF    1
575    <macro>                    $ENDIF                                                  DD49.198
                                                                        .<<<<< 3              $ENDIF    1
575    <macro>                  $ENDIIL                                                   DD49.199
576    <macro>                  $IF (R!TMP # R!%MYID),AND,(R!DDAL3 # 0)                   DD49.200
                                                                        .>>>>> 3              $IF       1
605    <macro>                    GET A,DL@TDL,R!DDAL3                                    DD49.201
613   013002  102003  010000      R!DDAL3 = 0, A # BP$DONE                                DD49.202
      024634
617    <macro>                  $ENDIF                                                    DD49.203
                                                                        .<<<<< 3              $ENDIF    1
617    <macro>                $ENDIF                                                      DD49.204
                                                                        .<<<<< 2              $ENDIF    1
617    <macro>              $ENDIF                                                        DD49.205
                                                                        .<<<<< 1              $ENDIF    1
617    <macro>            $ELSE                                                           DD49.206
                                                                                           DD49.207
                            * Check for read-ahead abort                                   DD49.208
                                                                                           DD49.209
620   010001  024644        R!DFNC = DAF$READ          .Set current function                DD49.210
622    <macro>              GET A,DK@ABT,R!DDCB        .Abort flag                          DD49.211
630    <macro>              $IF (A # 0)                                                     DD49.212
                                                                        .>>>>> 1              $IF       1
632    <macro>                $IF (R!DDN # 0)          .If complete                         DD49.213
                                                                        .>>>>> 2              $IF       1
635   077000 /003415            R = ACTDEM             .Activate demon                      DD49.214
637    <macro>                $ENDIF                                                        DD49.215
                                                                        .<<<<< 2              $ENDIF    1
637    <macro>                PKERN ICHK               .Leave                               DD49.216
641    <macro>              $ENDIF                                                          DD49.217
                                                                        .<<<<< 1              $ENDIF    1
641    <macro>            $ENDIF                                                          DD49.218
                                                                        .<<<<< 0              $ENDIF    1
                                                                                           DD49.219
                            * Get local buffer descriptors                                 DD49.220
                                                                                           DD49.221
641    <macro>            GET R!DCUR,DK@CLB,R!DDCB   .Current descriptor                    DD49.222
646    <macro>            GET R!DFRST,DK@FLB,R!DDCB  .First descriptor                      DD49.223
```

```
653     <macro>                     GET R!DLAST,DK@LLB,R!DDCB      .Last descriptor                    DD49.224
660     <macro>                     $IF (A = R!DCUR)               .Check for wrap                     DD49.225
                                                                                   .>>>>> 0              $IF      1
663    020640  024661                 R!DNXT = R!DFRST             .Next is first                      DD49.226
665     <macro>                     $ELSE                                                              DD49.227
666    020641  012006  024661         R!DNXT = R!DCUR + LB@LE      .Next is next                       DD49.228
671     <macro>                     $ENDIF                                                             DD49.229
                                                                                   .<<<<< 0              $ENDIF   1
                                                                                                       DD49.230
                                    * Validate ending address for sector completed                     DD49.231
                                                                                                       DD49.232
671     <macro>                     GET A,DK@ADD,R!DDCB            .Address pointer                    DD49.233
677     <macro>                     $IF (A = 0)                                                        DD49.234
                                                                                   .>>>>> 0              $IF      1
701    170000                         IOB:10                       .LMA register 0                     DD49.235
702     <macro>                     $ELSE                                                              DD49.236
703    171000                         IOB:11                       .LMA register 1                     DD49.237
704     <macro>                     $ENDIF                                                             DD49.238
                                                                                   .<<<<< 0              $ENDIF   1
704    024623                       R!%W1 = A                      .Save result                        DD49.239
705    014000 /001000  005002       R!%W2 = D4$WPS < 2             .Parcels in sector                  DD49.240
       024624
711     <macro>                     GET R!%W3,DK@SUB,R!DDCB        .Subfunction                        DD49.241
717     <macro>                     $IF (0 # R!%W3 & 1)            .If not full buffer I/O             DD49.242
                                                                                   .>>>>> 0              $IF      1
726    010020  024624                 R!%W2 = D'16                 .16-parcel transfer                 DD49.243
730     <macro>                     $ELSEIF (R!DFLG = DK$WRIT) AND (R!%W3 = DAS$ABS)                   DD49.244
741    010020  024624                 R!%W2 = D'16                 .16-parcel transfer                 DD49.245
743     <macro>                     $ENDIF                                                             DD49.246
                                                                                   .<<<<< 0              $ENDIF   1
743     <macro>                     GET A,LB@LOC,R!DCUR            .Start address for sector           DD49.247
747    022624  024626               R!%W4 = A + R!%W2              .Expected ending address            DD49.248
751    020657  013002  103003       R!%W4 = R!%W4 + D'8, R!DFLG = DK$WRIT .If writing                  DD49.249
       010010  025626
756    020626  023623  127000       P = BADADDR, R!%W4 # R!%W1     .If address error                   DD49.250
      /003454
762     <macro>                     FLDADD 1,DK@ADD,R!DDCB         .Update pointer                     DD49.251
                                                                                                       DD49.252
                                    * Queue the local descriptor for demon                             DD49.253
                                    * Start data transfer if possible                                  DD49.254
                                                                                                       DD49.255
1007    <macro>                     PUT R!DNXT,DK@CLB,R!DDCB       .Update current                     DD49.256
```

```
1014    <macro>                          FLDADD 1,LB@FUL,R!DCUR          .Indicate current done                DD49.257
1037   010000  024654                    R!QU = 0                        .Initialize queue pointer             DD49.258
1041    <macro>                          $IF (R!DDAL = 0)                .If read-ahead                        DD49.259
                                                                                                .>>>>> 0            $IF      1
1044    <macro>                            ADDRESS R!QU,MEM@BM,R!%MEMIO .For buffer memory                     DD49.260
1047   010005  024627                      R!%W5 = [MOS]                 .Save channel number                  DD49.261
1051    <macro>                            FLDADD 1,DK@RAD,R!DDCB        .Update read-aheads done              DD49.262
1074   024646                              R!SEQ = A                     .Next to do                           DD49.263
1075   020667  023000  127000              P = QMEMIO, R!TMP # R!%MYID .If no path on this IOP                 DD49.264
      /002027
1101    <macro>                          $ELSEIF (0 # R!DFNC & 1) OR (R!DDAL3 # 0) .Read or W.B.             DD49.265
1113    <macro>                            $IF (0 = R!DFNC & 1)          .If write,                            DD49.266
                                                                                                .>>>>> 1            $IF      1
1122   020634  024631                        R!DDAL = R!DDAL3            .DAL with next W.B. sector            DD49.267
1124    <macro>                            $ENDIF                                                              DD49.268
                                                                                                .<<<<< 1            $ENDIF   1
1124    <macro>                            GET R!SEQ,DL@SEQ,R!DDAL       .Number done                          DD49.269
1131    <macro>                            GET R!TOT,DL@TOT,R!DDAL       .Total to do                          DD49.270
1136    <macro>                            $IF (R!SEQ < R!TOT)           .If more to do                        DD49.271
                                                                                                .>>>>> 1            $IF      1
1142   026646                                R!SEQ = R!SEQ + 1                                                 DD49.272
1143    <macro>                              ADDRESS R!QU,MEM@BM,R!%MEMIO .Buffer memory                       DD49.273
1146   010005  024627                        R!%W5 = [MOS]               .Save channel                         DD49.274
1150    <macro>                              $IF (R!TMP = R!%MYID),AND,(0 = R!DFNC & 0'20)                     DD49.275
                                                                                                .>>>>> 2            $IF      1
1163    <macro>                                $IF (R!TM # TM$BMR)       .If target not BMR                    DD49.276
                                                                                                .>>>>> 3            $IF      1
1167    <macro>                                  GET R!%W4,DL@RAC,R!DDAL .Sectors in MOS                         DD49.277
1175    <macro>                                  $IF (0 # R!DFNC & 1)  .If reading                               DD49.278
                                                                                                .>>>>> 4            $IF      1
1204    <macro>                                    $IF (R!TM = TM$CMEM) .If target Central memory                DD49.279
                                                                                                .>>>>> 5            $IF      1
1207    <macro>                                      ADDRESS R!QU,MEM@HO,R!%MEMIO .High speed out                DD49.280
1212   020000  012001  024627                        R!%W5 = R!%HISP + 1 .Channel                              DD49.281
1215    <macro>                                    $ELSE                 .If target is SSD                     DD49.282
1216    <macro>                                      ADDRESS R!QU,MEM@SO,R!%MEMIO .SSD out                       DD49.283
1221   020000  012001  024627                        R!%W5 = R!%SSD + 1 .Channel                               DD49.284
1224    <macro>                                    $ENDIF                                                      DD49.285
                                                                                                .<<<<< 5            $ENDIF   1
1224    <macro>                                  $ELSEIF (R!SEQ <= R!TOT - R!%W4) .Next sector not W.B.        DD49.286
1236    <macro>                                    $IF (R!TM = TM$CMEM) .If target Central memory              DD49.287
                                                                                                .>>>>> 5            $IF      1
```

```
1241    <macro>                          ADDRESS R!QU,MEM@HI,R!%MEMIO .High speed in          DD49.288
1244   020000  024627                    R!%W5 = R!%HISP    .Channel                          DD49.289
1246    <macro>                         $ELSE               .If target is SSD                  DD49.290
1247    <macro>                          ADDRESS R!QU,MEM@SI,R!%MEMIO .SSD in                 DD49.291
1252   020000  024627                    R!%W5 = R!%SSD     .Channel                          DD49.292
1254    <macro>                         $ENDIF                                                 DD49.293
                                                                          .<<<<< 5                $ENDIF   1
1254    <macro>                           $ENDIF                                               DD49.294
                                                                          .<<<<< 4                $ENDIF   1
1254    <macro>                         $ENDIF                                                 DD49.295
                                                                          .<<<<< 3                $ENDIF   1
1254    <macro>                       $ELSE                 .If no path or internal            DD49.296
1255    <macro>                         $IF (0 # R!DFNC & 1) AND (R!SEQ = 1) .First read       DD49.297
                                                                          .>>>>> 3                $IF      1
1270   077000 /003415                     R = ACIDEM        .Start the transfer activity       DD49.298
1272    <macro>                         $ENDIF                                                 DD49.299
                                                                          .<<<<< 3                $ENDIF   1
1272   075000 /002027                    P = QMEMIO         .Bypass memio start                DD49.300
1274    <macro>                       $ENDIF                                                   DD49.301
                                                                          .<<<<< 2                $ENDIF   1
1274    <macro>                     $ENDIF                                                     DD49.302
                                                                          .<<<<< 1                $ENDIF   1
1274    <macro>                   $ENDIF                                                       DD49.303
                                                                          .<<<<< 0                $ENDIF   1
                                                                                                 DD49.304
                                  * Start memory transfer.                                       DD49.305
                                                                                                 DD49.306
1274    <macro>                   $IF (R!QU # 0)            .If something to start               DD49.307
                                                                          .>>>>> 0                $IF      1
1300   020644  013012  126000       P = QMEMIO, R!DFNC = DAF$PTRN .If patterning                DD49.308
      /002027
1304   020627  054000               B = R!%W5                 .Channel to use                    DD49.309
1306    <macro>                     GET R!CD,LB@LOC,R!DCUR    .Buffer address                    DD49.310
1313   014000 /001000  024000       R!CE = D4$WPS             .Length                            DD49.311
1316    <macro>                     $IF (R!%W5 = [MOS]),AND,(R!TM # TM$BMR) .If buffer memory  DD49.312
                                                                          .>>>>> 1                $IF      1
1326    <macro>                       GET A,DK@LCK,R!DDCB     .MCB lock                          DD49.313
1333   127000 /002027                 P = QMEMIO, A # 0       .If locked                         DD49.314
                                                                                                 DD49.315
                                      * If I/O was not initiated on previous interrupt,          DD49.316
                                      * make sure demon got in to adjust DIO pointer.            DD49.317
                                                                                                 DD49.318
```

```
1335   020641  013006  024623          R!%W1 = R!DCUR - LB@LE     .Last local descriptor                DD49.319
1340   020623  023640  101003          R!%W1 = R!DLAST, R!%W1 < R!DFRST                                 DD49.320
       020642  024623
1345    <macro>                        GET R!%W2,LB@FUL,R!%W1     .Buffer full state                    DD49.321
1353    <macro>                        GET R!%W3,LB@IOI,R!%W1     .I/O initiated flag                   DD49.322
1362    <macro>                        GET R!%W4,LB@FUL,R!DCUR    .Full state for current               DD49.323
1370    <macro>                        $IF (R!%W3 = 0) AND (R!%W2 = R!%W4) .If DIO ptr in doubt         DD49.324
                                                                                  .>>>>> 2                $IF      1
1377   075000 /002027                   P = QMEMIO                .Let demon handle transfer            DD49.325
1401    <macro>                        $ENDIF                                                            DD49.326
                                                                                  .<<<<< 2                $ENDIF   1
1401    <macro>                        GET R!CB,DK@BFO,R!DDCB     .Base of MOS buffers                  DD49.327
1406    <macro>                        GET R!CC,DK@BF1,R!DDCB                                           DD49.328
1413    <macro>                        GET R!%W3,DK@BIO,R!DDCB    .Check for overflow                   DD49.329
1421    <macro>                        GET R!%W4,DK@CNT,R!DDCB    .Read-ahead buffer length             DD49.330
1427    <macro>                        GET R!%W5,DK@DIO,R!DDCB    .Current pointer                      DD49.331
1435    <macro>                        $IF (R!DDAL = 0)           .If read-ahead                        DD49.332
                                                                                  .>>>>> 2                $IF      1
1440   020627  012001  024623            R!%W1 = R!%W5 + 1        .Bump pointer                         DD49.333
1443   023626  103003  010000            R!%W1 = 0, A = R!%W4     .Check for wrap                       DD49.334
       024623
1447    <macro>                          $IF (R!%W1 = R!%W3)      .If buffer not ready                  DD49.335
                                                                                  .>>>>> 3                $IF      1
1453    <macro>                            STORE 1,DK@ABT,R!DDCB .Abort read aheads                      DD49.336
1464    <macro>                            FLDSUB 1,DK@RAD,R!DDCB .Adjust count                          DD49.337
1507    <macro>                            STORE 0,LB@FUL,R!DCUR .Set buffer empty                       DD49.338
1520    <macro>                            PKERN ICHK                                                    DD49.339
1522    <macro>                          $ENDIF                                                          DD49.340
                                                                                  .<<<<< 3                $ENDIF   1
1522    <macro>                          PUT R!%W1,DK@DIO,R!DDCB .Update pointer                         DD49.341
1535   020627  024623                    R!%W1 = R!%W5            .Current read-ahead pointer           DD49.342
1537   010004  024000                    R!CA = MIO$BMWI          .Write to MOS                         DD49.343
1541    <macro>                        $ELSE                                                             DD49.344
1542   020655  023646  024623            R!%W1 = R!TOT - R!SEQ    .For write behind                     DD49.345
1545   020627  022623  024623            R!%W1 = R!%W5 + R!%W1    .Add to pointer                       DD49.346
1550   020623  023626  100004            R!%W1 = R!%W1 - R!%W4, R!%W1 >= R!%W4  .Check for wrap         DD49.347
       020623  023626  024623
1556   020623  023625  126000            P = QMEMIO, R!%W1 = R!%W3 .If data not ready                   DD49.348
      /002027
1562    <macro>                          $IF (R!SEQ = R!TOT)      .If at end of this DAL                DD49.349
                                                                                  .>>>>> 3                $IF      1
1566    <macro>                            GET R!%W2,DL@RAC,R!DDAL .Total write-behinds done             DD49.350
```

```
1574    <macro>                                  FLDSUB R!%W2,DK@WBD,R!DDCB .Decrement channel count        DD49.351
1617    020627  022624  024627                   R!%W5 = R!%W5 + R!%W2 .Update DIO pointer                DD49.352
1622    020627  023626  100004                   R!%W5 = R!%W5 - R!%W4, R!%W5 >= R!%W4 .Check for wrap    DD49.353
        020627  023626  024627
1630    <macro>                                  PUT R!%W5,DK@DIO,R!DDCB                                  DD49.354
1643    077000  /003415                          R = ACTDEM            .Activate demon for next wbh       DD49.355
1645    <macro>                                $ENDIF                                                     DD49.356
                                                                                   .<<<<< 3                 $ENDIF    1
1645    010002  024000                           R!CA = MIO$BMRI       .Read from MOS                     DD49.357
1647    <macro>                                $ENDIF                                                     DD49.358
                                                                                   .<<<<< 2                 $ENDIF    1
1647    <macro>                              $ELSE                                                        DD49.359
1650    <macro>                                GET R!CB,DL@IM0,R!DDAL     .Starting address               DD49.360
1657    <macro>                                GET R!CC,DL@IM1,R!DDAL     .                               DD49.361
1664    <macro>                                $IF (R!SEQ # 1)            .If not first                   DD49.362
                                                                                   .>>>>> 2                 $IF       1
1670    020646  013001  024624                   R!%W2 = R!SEQ - 1          .Sequence to transfer           DD49.363
1673    024623                                   R!%W1 = A                  .Save                           DD49.364
1674    <macro>                                  GET R!%W4,DL@UNS,R!DDAL .stripe factor                     DD49.365
1702    <macro>                                  $UNTIL (R!%W4 = 1)                                         DD49.366
1705    020623  022624  024623                     R!%W1 = R!%W1 + R!%W2                                    DD49.367
1710    027626                                     R!%W4 = R!%W4 - 1                                        DD49.368
1711    <macro>                                  $ENDTIL                                                    DD49.369
1712    <macro>                                $ELSE                                                      DD49.370
1713    010000  024623                           R!%W1 = 0                  .Sequence number                DD49.371
1715    <macro>                                $ENDIF                                                     DD49.372
                                                                                   .<<<<< 2                 $ENDIF    1
1715    <macro>                                $IF (0 = R!DFNC & 1)       .If write                       DD49.373
                                                                                   .>>>>> 2                 $IF       1
1724    <macro>                                  $IF (R!TM = TM$CMEM)       .If target Central memory       DD49.374
                                                                                   .>>>>> 3                 $IF       1
1727    010006  024000                             R!CA = MIO$HSII          .High speed in                  DD49.375
1731    <macro>                                  $ELSEIF (R!TM = TM$SSD) .If SSD                            DD49.376
1736    010012  024000                             R!CA = MIO$SSII          .SSD in                         DD49.377
1740    <macro>                                  $ELSE                      .If BMR                         DD49.378
1741    010002  024000                             R!CA = MIO$BMRI          .BMR read                       DD49.379
1743    <macro>                                  $ENDIF                                                     DD49.380
                                                                                   .<<<<< 3                 $ENDIF    1
1743    <macro>                                $ELSE                      .If read                        DD49.381
1744    <macro>                                  $IF (R!TM = TM$CMEM)       .If target Central memory       DD49.382
                                                                                   .>>>>> 3                 $IF       1
1747    010010  024000                             R!CA = MIO$HSOI          .High speed out                 DD49.383
```

```
1751    <macro>                          $ELSEIF (R!IM = IM$SSD) .If SSD                          DD49.384
1756   010014  024000                      R!CA = MIO$SSOI        .SSD out                          DD49.385
1760    <macro>                          $ELSE                   .If BMR                           DD49.386
1761   010004  024000                      R!CA = MIO$BMWI        .BMR write                        DD49.387
1763    <macro>                          $ENDIF                                                    DD49.388
                                                                                .<<<<< 3            $ENDIF    1
1763    <macro>                            $ENDIF                                                  DD49.389
                                                                                .<<<<< 2            $ENDIF    1
1763    <macro>                          $ENDIF                                                    DD49.390
                                                                                .<<<<< 1            $ENDIF    1
1763   010000  024625                    R!%W3 = 0                                                 DD49.391
1765    <macro>                          LSHIFT R!%W3,R!%W1,D'9                                    DD49.392
2004    <macro>                          DADD R!CB,R!CC,R!%W3,R!%W1  .Final address                DD49.393
2014    <macro>                          RKERN COMIO                 .Initiate transfer            DD49.394
2016    <macro>                          STORE 1,LB@IOI,R!DCUR       .Set i/o in progress          DD49.395
                                                                                                   DD49.396
                                       * Activate demon if something to do                         DD49.397
                                                                                                   DD49.398
2027                             QMEMIO  *                                                         DD49.399
2027    <macro>                          $IF (R!DDAL # 0)            .If not read ahead            DD49.400
                                                                                .>>>>> 1            $IF       1
2032    <macro>                            PUT R!SEQ,DL@SEQ,R!DDAL   .Update sequence number       DD49.401
2037    <macro>                          $ENDIF                                                    DD49.402
                                                                                .<<<<< 1            $ENDIF    1
2037    <macro>                          PUT R!SEQ,LB@SEQ,R!DCUR     .Into descriptor              DD49.403
2044    <macro>                          PUT R!DDAL,LB@DAL,R!DCUR    .Also DAL address             DD49.404
2051    <macro>                          $IF (R!DFNC = DAF$PTRN)     .If patterning                DD49.405
                                                                                .>>>>> 1            $IF       1
2055    <macro>                            STORE 1,LB@FUL,R!DCUR     .Set buffer full flag         DD49.406
2065    <macro>                          $ELSE                                                     DD49.407
2066   020641  024000                      R!EH = R!DCUR             .Queue local descriptor       DD49.408
2070   020654  024000                      R!ED = R!QU               .Queue address                DD49.409
2072    <macro>                            RKERN ENQU                                              DD49.410
2074   014000 /000000                      A = D$D4DEM                                             DD49.411
2076    <macro>                            RKERN DACT                .Activate demon               DD49.412
2100    <macro>                          $ENDIF                                                    DD49.413
                                                                                .<<<<< 1            $ENDIF    1
2100    <macro>                        $ENDIF                                                      DD49.414
                                                                                .<<<<< 0            $ENDIF    1
2100   020632  024631                  R!DDAL = R!DDAL1              .Reset to first DAL           DD49.415
2102   020000  054000                  B = R!%ICHAN                  .Restore B                    DD49.416
                                                                                                   DD49.417
```

```
                                  * Read interrupt processing                                    DD49.418
                                                                                                 DD49.419
2104    <macro>                   $IF (R!DFLG = DK$READ)                                         DD49.420
                                                                          .>>>>> 0                $IF      1
2110    <macro>                     $IF (R!DDN = 0)                                              DD49.421
                                                                          .>>>>> 1                $IF      1
2113    <macro>                       $IF (R!DDAL2 = 0)        .Time for read-ahead?               DD49.422
                                                                          .>>>>> 2                $IF      1
2116    <macro>                         GET R!%W1,DK@RCC,R!DDCB .Read-ahead limit                 DD49.423
2124    <macro>                         $IF (R!DNUM = 0)        .Done with current DAL            DD49.424
                                                                          .>>>>> 3                $IF      1
                                                                                                 DD49.425
                                          * If another DAL exists, check for                     DD49.426
                                          * contiguous with current.                             DD49.427
                                                                                                 DD49.428
2127    <macro>                           $IF (R!MDAL # 0)                                       DD49.429
                                                                          .>>>>> 4                $IF      1
2132    <macro>                             GET A,DL@CON,R!MDAL .See if connected                DD49.430
2140    <macro>                             $IF (A # 0)         .If so                           DD49.431
                                                                          .>>>>> 5                $IF      1
2142    <macro>                               $IF (R!BREAK = 0),AND,(R!%W1 # 0)              R16410DB.1
                                                                          .>>>>> 6                $IF      1
2150  010001  024656                              R!CONT = 1      .If no break indicated       R16410DB.2
2152    <macro>                               $ENDIF                                         R16410DB.3
                                                                          .<<<<< 6                $ENDIF   1
2152    <macro>                             $ELSE                                                DD49.433
2153    <macro>                               STORE 1,DK@ABT,R!DDCB .Set read-ahead abort        DD49.434
2164    <macro>                               STORE 1,DK@INV,R!DDCB .Set data invalid            DD49.435
2175  010000  024662                          R!MDAL = 0                                         DD49.436
2177    <macro>                             $ENDIF                                               DD49.437
                                                                          .<<<<< 5                $ENDIF   1
2177    <macro>                           $ENDIF                                                 DD49.438
                                                                          .<<<<< 4                $ENDIF   1
                                                                                                 DD49.439
                                          * If no more contiguous requests, set read-ahead       DD49.440
                                                                                                 DD49.441
2177    <macro>                           $IF (R!MDAL = 0)        .If no more DALS               DD49.442
                                                                          .>>>>> 4                $IF      1
2202    <macro>                             STORE 0,DKF@RD,R!DDCB .Clear read flag               DD49.443
2206    <macro>                             STORE 1,DKF@RA,R!DDCB .Set read-ahead flag           DD49.444
2214    <macro>                             PUT R!DCYL,DK@RAC,R!DDCB .save starting R.A. address DD49.445
2221    <macro>                             PUT R!DHED,DK@RAH,R!DDCB                             DD49.446
```

```
2235    <macro>                          PUT R!DSEC,DK@RAS,R!DDCB                                  DD49.447
2250    <macro>                          GET R!%W5,DK@UNT,R!DDCB .Current unit                     DD49.448
2256    <macro>                          PUT R!%W5,DK@RAU,R!DDCB .Save                             DD49.449
2271    <macro>                          $IF (R!%W1 # 1),AND,(R!BREAK = 0) .No break indicated     DD49.450
                                                                                  .>>>>> 5              $IF      1
2300   010001  024656                      R!CONT = 1          .Set continue flag                   DD49.451
2302    <macro>                          $ENDIF                                                    DD49.452
                                                                                  .<<<<< 5              $ENDIF   1
2302    <macro>                        $ENDIF                                                      DD49.453
                                                                                  .<<<<< 4              $ENDIF   1
2302    <macro>                      $ELSEIF (R!DNUM = 1) AND (R!%W1 # 0)                          DD49.454
2312   020662  103003  010001          R!CONT = 1, R!MDAL = 0 .Set continue if no more DALs        DD49.455
       024656
2316    <macro>                      $ENDIF                                                        DD49.456
                                                                                  .<<<<< 3              $ENDIF   1
2316    <macro>                    $ELSE                                                           DD49.457
2317   010001  024656                R!CONT = 1                .Set continue                       DD49.458
2321    <macro>                    $ENDIF                                                          DD49.459
                                                                                  .<<<<< 2              $ENDIF   1
2321    <macro>                  $ELSE                                                             DD49.460
2322   077000 /003415                R = ACTDEM                .Activate demon to restart          DD49.461
2324    <macro>                  $ENDIF                                                            DD49.462
                                                                                  .<<<<< 1              $ENDIF   1
2324    <macro>                  FLDADD 1,DK@RD1,R!DDCB      .Update reads done                  DD49.463
2330    <macro>                  $IF (A = 0)                 .Check wrap                         DD49.464
                                                                                  .>>>>> 1              $IF      1
2332    <macro>                    FLDADD 1,DK@RDO,R!DDCB    .Increment upper                    DD49.465
2336    <macro>                  $ENDIF                                                            DD49.466
                                                                                  .<<<<< 1              $ENDIF   1
2336    <macro>                $ENDIF                                                              DD49.467
                                                                                  .<<<<< 0              $ENDIF   1
                                                                                                      DD49.468
                               * Read-ahead interrupt processing                                     DD49.469
                                                                                                      DD49.470
2336    <macro>                $IF (R!DFLG = DK$RAHD)                                                DD49.471
                                                                                  .>>>>> 0              $IF      1
2342    <macro>                  $IF (R!DDN = 0)             .If chaining                        DD49.472
                                                                                  .>>>>> 1              $IF      1
2345    <macro>                    GET R!%W1,DK@RCC,R!DDCB   .Read-ahead limit                   DD49.473
2353    <macro>                    GET R!%W2,DK@RAD,R!DDCB   .Read-aheads done                   DD49.474
2361    <macro>                    $IF (1 < R!%W1 - R!%W2)   .If more to do                       DD49.475
                                                                                  .>>>>> 2              $IF      1
```

```
2370   010001  024656          R!CONT = 1                .Set continue                    DD49.476
2372   <macro>               $ENDIF                                                       DD49.477
                                                                          .<<<<< 2            $ENDIF   1
2372   <macro>               $ELSE                                                        DD49.478
2373   077000 /003415          R = ACTDEM                .Let demon release channel       DD49.479
2375   <macro>               $ENDIF                                                       DD49.480
                                                                          .<<<<< 1            $ENDIF   1
2375   <macro>               FLDADD 1,DK@RA0,R!DDCB      .Update counter                  DD49.481
2401   <macro>             $ENDIF                                                         DD49.482
                                                                          .<<<<< 0            $ENDIF   1
                                                                                             DD49.483
                           * Write interrupt                                                 DD49.484
                                                                                             DD49.485
2401   <macro>             $IF (R!DFLG = DK$WRIT)                                           DD49.486
                                                                          .>>>>> 0            $IF      1
2405   <macro>               $IF (R!DNUM = 0)            .If current request complete     DD49.487
                                                                          .>>>>> 1            $IF      1
2410   <macro>                 ADDRESS R!ED,DK@DLO,R!DDCB .Remove from request queue      DD49.488
2413   <macro>                 RKERN EDQU                                                   DD49.489
2415   <macro>                 $IF (R!DFNC # DAF$WRBH),OR,(R!TMP # R!%MYID) .Not W.B.     DD49.490
                                                                          .>>>>> 2            $IF      1
2424   <macro>                   ADDRESS R!ED,DCB@DN,R!%DCBQ .Place on done queue         DD49.491
2427   <macro>                   RKERN ENQU                                                 DD49.492
2431   077000 /003415            R = ACTDEM                .Activate demon to process       DD49.493
2433   <macro>                 $ELSE                                                        DD49.494
2434   020000  024000            R!DY = R!EH               .Release DAL                     DD49.495
2436   <macro>                   RKERN HONN                                                 DD49.496
2440   <macro>                 $ENDIF                                                       DD49.497
                                                                          .<<<<< 2            $ENDIF   1
2440   <macro>               $ENDIF                                                       DD49.498
                                                                          .<<<<< 1            $ENDIF   1
2440   <macro>               $IF  (R!DDN = 0)            .If chaining                     DD49.499
                                                                          .>>>>> 1            $IF      1
2443   020633  102003  010001  R!CONT = 1, R!DDAL2 # 0   .Set continue if more to do        DD49.500
       024656
2447   <macro>               $ELSE                       .Need demons help                DD49.501
2450   077000 /003415          R = ACTDEM                .Activate                        DD49.502
2452   <macro>               $ENDIF                                                       DD49.503
                                                                          .<<<<< 1            $ENDIF   1
2452   <macro>               FLDADD 1,DK@WR1,R!DDCB      .Update writes done              DD49.504
2456   <macro>               $IF (A = 0)                 .Check wrap                      DD49.505
                                                                          .>>>>> 1            $IF      1
```

```
2460    <macro>                     FLDADD 1,DK@WRO,R!DDCB     .Increment upper                    DD49.506
2464    <macro>                   $ENDIF                                                            DD49.507
                                                                                .<<<<< 1                $ENDIF   1
2464    <macro>                 $ENDIF                                                              DD49.508
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                    DD49.509
                                * Issue continue                                                    DD49.510
                                                                                                    DD49.511
2464    <macro>                 $IF (R!CONT # 0)                                                    DD49.512
                                                                                .>>>>> 0                $IF      1
2467   026637                     R!DSEC = R!DSEC + 1        .Next sector                         DD49.513
2470    <macro>                   $IF (A = R!SPT)            .Next head?                          DD49.514
                                                                                .>>>>> 1                $IF      1
2473   010000  024637               R!DSEC = 0               .First sector                        DD49.515
2475   026636                       R!DHED = R!DHED + 1      .Next head                           DD49.516
2476    <macro>                     $IF (A = R!TPC)          .Next cylinder?                      DD49.517
                                                                                .>>>>> 2                $IF      1
2501    <macro>                       PKERN ICHK             .Leave                               DD49.518
2503    <macro>                     $ENDIF                                                          DD49.519
                                                                                .<<<<< 2                $ENDIF   1
2503    <macro>                   $ENDIF                                                            DD49.520
                                                                                .<<<<< 1                $ENDIF   1
2503   020661  024641             R!DCUR = R!DNXT            .Move to next local descriptor       DD49.521
2505   012006  024661             R!DNXT = A + LB@LE                                              DD49.522
2507   041000  023642  100004     R!DNXT = R!DFRST, A > R!DLAST .Check for wrap                   DD49.523
       102003  020640  024661
2515    <macro>                   GET R!DLOC,LB@LOC,R!DNXT   .Local buffer                        DD49.524
2522    <macro>                   GET R!%W1,LB@FUL,R!DCUR    .Check for buffer ready              DD49.525
2530    <macro>                   GET R!%W2,LB@FUL,R!DNXT                                         DD49.526
2536    <macro>                   $IF (R!%W1 # R!%W2)        .If not                              DD49.527
                                                                                .>>>>> 1                $IF      1
2542    <macro>                     STORE 1,LB@CP,R!DNXT     .Let demon issue continue            DD49.528
2553    <macro>                     PKERN ICHK                                                      DD49.529
2555    <macro>                   $ENDIF                                                            DD49.530
                                                                                .<<<<< 1                $ENDIF   1
                                                                                                    DD49.531
                                  * Enter the LMA address                                           DD49.532
                                                                                                    DD49.533
2555    <macro>                   FLDADD 1,DK@LMA,R!DDCB     .LMA pointer                         DD49.534
2600    <macro>                   $IF (A # 0)                .LMA 0                               DD49.535
                                                                                .>>>>> 1                $IF      1
2602   020660                       A = R!DLOC                                                      DD49.536
```

```
2603   174000                          IOB:14                  .Enter                                  DD49.537
2604    <macro>                       $ELSE                                                            DD49.538
2605   020660                          A = R!DLOC                                                      DD49.539
2606   175000                          IOB:15                  .Enter                                  DD49.540
2607    <macro>                       $ENDIF                                                           DD49.541
                                                                                .<<<<< 1                $ENDIF   1
                                                                                                        DD49.542
                                      * Compute physical address.                                      DD49.543
                                                                                                        DD49.544
2607    <macro>                       GET A,DK@TYP,R!DDCB     .Check disk type                R13739AU.11
2615    <macro>                       $IF (A = DSK$DD40)      .If DD40                        R13739AU.12
                                                                                .>>>>> 1                $IF      1
2620    <macro>                         ADDRESS R!%W2,DK@OWC,R!DDCB                           R13739AU.13
2623   030624                           A = (R!%W2)             .Advance outstanding write count R13739AU.14
2624    <macro>                         $IF (A # 0) AND (A # D40$SPT) .Less than full track   R13739AU.15
                                                                                .>>>>> 2                $IF      1
2631   036624                             (R!%W2) = (R!%W2) + 1 .Advance count                R13739AU.16
2632    <macro>                         $ENDIF                                                R13739AU.17
                                                                                .<<<<< 2                $ENDIF   1
2632   020636  005007  024624           R!%W2 = R!DHED < 7      .Current head                   R13739AU.18
2635   020637  013044                   A = R!DSEC - D'36       .Convert sector to 2/drive, 4/secR13739AU.19
2637   101007                           P = D40A, C = 1         .If greater than 35             R13739AU.20
2640   012014                           A = A + D'12                                            R13739AU.21
2641   101006                           P = D40B, C = 1         .If greater than 23             R13739AU.22
2642   012014                           A = A + D'12                                            R13739AU.23
2643   101005                           P = D40C, C = 1         .If greater than 11             R13739AU.24
2644   012014                           A = A + D'12            .Make 0 .. 11                   R13739AU.25
2645   070004                           P = D40D                                                R13739AU.26
                                                                                                R13739AU.27
2646   012020                   D40A    A = A + 1S4             .[36 .. 47] -> 3S4 + [0 .. 11]  R13739AU.28
2647   012020                   D40B    A = A + 1S4             .[24 .. 35] -> 2S4 + [0 .. 11]  R13739AU.29
2650   012020                   D40C    A = A + 1S4             .[12 .. 23] -> 1S4 + [0 .. 11]  R13739AU.30
2651   025624                   D40D    R!%W2 = A + R!%W2       .[ 0 .. 11] -> 0S4 + [0 .. 11]  R13739AU.31
2652    <macro>                       $ELSE                   .Not DD40                       R13739AU.32
2653   020636  024623                   R!%W1 = R!DHED          .Current head                   R13739AU.33
2655    <macro>                         $IF (R!SPT = R!DSEC + 1) .If end of track               R13739AU.34
                                                                                .>>>>> 2                $IF      1
2664   026623                             R!%W1 = R!%W1 + 1     .Next head                    R13739AU.35
2665   023652                             A = A - R!TPC                                         R13739AU.36
2666   103002  024623                     R!%W1 = A, A = 0      .If end of cylinder           R13739AU.37
2670    <macro>                         $ENDIF                                                R13739AU.38
                                                                                .<<<<< 2                $ENDIF   1
```

```
2670   <macro>                          GET A,DK@SPO,R!DDCB        .Get spiral offset                R13739AU.39
2674   022637  024624                   R!%W2 = A + R!DSEC         .Physical sector                  R13739AU.40
2676   023651                           A = A - R!SPT                                                R13739AU.41
2677   100002  024624                   R!%W2 = A, C = 1           .Check wrap                       R13739AU.42
2701   020623  005010  022624           R!%W2 = R!%W1 < D'8 + R!%W2 .Combine head and sector        R13739AU.43
       024624
2705   <macro>                        $ENDIF                                                         R13739AU.44
                                                                                   .<<<<< 1              $ENDIF    1
2705   <macro>                        GET R!%W1,DK@SUB,R!DDCB    .Get subfunction code                DD49.554
2713   020623  005014  022624         R!%W2 = R!%W1 < D'12 + R!%W2 .combine                           DD49.555
       024624
2717   176000                         IOB:16                     .Enter                               DD49.556
2720   <macro>                        PUT R!%W2,DK@ACC,R!DDCB    .Save accummulator code              DD49.557
2725   <macro>                        STORE 1,DK@CNG,R!DDCB      .Indicate chaining in progress       DD49.558
2736   020644  005006  052000         R!%W1 = R!DFNC < 6 + B     .Record function and channel         DD49.559
       024623
2742   <macro>                        GET R!%W2,DK@UNT,R!DDCB    .Device unit                         DD49.560
2750   020624  005014  022635         R!%W2 = R!%W2 < D'12 + R!DCYL .Record with cylinder             DD49.561
       024624
2754   <macro>                        KRECORD TF$D4IO,(R!%W1,R!%W2,R!DHED,R!DSEC,R!DLOC)             DD49.562
                                            * Read/write (10)/Channel (6)                             DD49.563
                                            * Unit (4)/Cylinder (12)                                  DD49.564
                                            * Head group                                              DD49.565
                                            * Sector                                                  DD49.566
                                            * Local memory address                                    DD49.567
2773   <macro>                      $ENDIF                                                            DD49.568
                                                                                   .<<<<< 0              $ENDIF    1
2773   <macro>                      PKERN ICHK                                                        DD49.569
                                                                                                       DD49.570
                                 ***********************************************************************  DD49.571
                                 *         Handle interrupts for non-data functions                  *  DD49.572
                                 ***********************************************************************  DD49.573
2775                             D4CNTL   *                                                            DD49.574
2775   <macro>                      STORE 0,DK@SYN,R!DDCB      .Clear sync info                     R13739AU.45
                                                                                                       DD49.575
                                 ***********************************************************************  DD49.576
                                 *         Select interrupt processing.                              *  DD49.577
                                 ***********************************************************************  DD49.578
3004   <macro>                      $IF (R!DFLG = DK$SLCT)                                            DD49.579
                                                                                   .>>>>> 0              $IF       1
3010   <macro>                        STORE 0,DK@SEL,R!DDCB      .Set unit selected                   R13739AU.46
3021   <macro>                        STORE 0,DK@CE,R!DDCB       .Clear CE cylinder enable            DD49.581
```

```
3030   072365                          R = ACTDEM                  .Activate demon                    DD49.582
3031    <macro>                        FLDADD 1,DK@SLO,R!DDCB      .Count selects                     DD49.583
                                                                                                      DD49.584
                                 ******************************************************************   DD49.585
                                 *         Release interrupt processing                           *   DD49.586
                                 ******************************************************************   DD49.587
3035    <macro>                        $ELSEIF (R!DFLG = DK$RLSE)                                     DD49.588
3042    <macro>                        STORE 1,DK@SEL,R!DDCB       .Set unit released             R13739AU.47
3052   072343                          R = ACTDEM                  .Activate demon                    DD49.590
3053    <macro>                        FLDADD 1,DK@RLO,R!DDCB      .Count releases                    DD49.591
                                                                                                      DD49.592
                                 ******************************************************************   DD49.593
                                 *         Seek interrupt processing                              *   DD49.594
                                 ******************************************************************   DD49.595
3057    <macro>                        $ELSEIF (R!DFLG = DK$SEEK)                                     DD49.596
3064    <macro>                        GET R!DHED,DK@NHD,R!DDCB    .Desired head group                DD49.597
3072    <macro>                        GET R!%W1,DK@HED,R!DDCB     .Current head group                DD49.598
3101    <macro>                        $IF (A # R!DHED)            .If head select needed             DD49.599
                                                                                        .>>>>> 1          $IF      1
3104    <macro>                          PUT R!DHED,DK@HED,R!DDCB  .Set new head group            R13739AU.48
3120   020670  013006  102052           P = D4CNTL1, R!DTYPE = DSK$DD40                          R13739AU.49
3123   020636  005010  024624           R!%W2 = R!DHED < D'8      .Save for function echo            DD49.600
3126   016000 /040000                   A = A + O'40000                                              DD49.601
3130   161000                           IOB:1                     .Issue                             DD49.602
3131    <macro>                         PUT R!%W2,DK@ACC,R!DDCB   .Save function                     DD49.603
3136    <macro>                         STORE 1,DKF@HD,R!DDCB     .Set head select flag              DD49.604
3144    <macro>                         ADDRESS R!EH,DK@TMO,R!DDCB .Timeout cell                     DD49.605
3147   010002  024000                   R!EG = D4HDLIM            .Timer value                       DD49.606
3151    <macro>                         RKERN QTIME               .Put on timer queue                DD49.607
3153   167000                           IOB:7                     .Enable interrupts                 DD49.608
3154    <macro>                         KRECORD TF$D4HD,(B,R!%W1,R!DHED,R!DDCB)                      DD49.610
                                            * Channel                                                 DD49.611
                                            * Old head group                                          DD49.612
                                            * New head group                                          DD49.613
                                            * DCB address                                             DD49.614
3173    <macro>                        $ELSE                                                          DD49.615
3174                             D4CNTL1  *                                                       R13739AU.50
3174   072221                          R = ACTDEM                  .Activate demon                    DD49.616
3175    <macro>                        $ENDIF                                                         DD49.617
                                                                                        .<<<<< 1          $ENDIF   1
3175    <macro>                        FLDADD 1,DK@SK1,R!DDCB      .Count seeks                       DD49.618
3201    <macro>                        $IF (A = 0)                                                    DD49.619
```

```
                                                                    .>>>>> 1          $IF      1
3203    <macro>                   FLDADD 1,DK@SKO,R!DDCB                            DD49.620
3207    <macro>                 $ENDIF                                              DD49.621
                                                                    .<<<<< 1          $ENDIF   1
                                                                                      DD49.622
                        ************************************************************  DD49.623
                        *        Head select interrupt processing                 *  DD49.624
                        ************************************************************  DD49.625
3207    <macro>                 $ELSEIF (R!DFLG = DK$HEAD)                          DD49.626
3214   072201                     R = ACTDEM                 .Activate demon         DD49.627
3215    <macro>                   FLDADD 1,DK@HDO,R!DDCB     .Count head selects     DD49.628
                                                                                      DD49.629
                        ************************************************************  DD49.630
                        *        Unknown reason for interrupt                     *  DD49.631
                        ************************************************************  DD49.632
3221    <macro>                 $ELSE                                               DD49.633
3222    <macro>                   $PUNTIF CODE=PT$DD49       .Unexpected interrupt   DD49.634
3224    <macro>                 $ENDIF                                              DD49.635
                                                                    .<<<<< 0          $ENDIF   1
3224    <macro>                 PKERN ICHK                 .Leave                   DD49.636
                                                                                      DD49.637
                        ************************************************************  DD49.638
                        ************************************************************  DD49.639
                                                                                      DD49.640
                        ************************************************************  DD49.641
                        *        Subroutines and error detection                  *  DD49.642
                        ************************************************************  DD49.643
                                                                                      DD49.644
                        *        Special handling for channel done                    DD49.645
                                                                                      DD49.646
3226                    D4DONE   *                                                    DD49.647
                                                                                      DD49.648
                                 * Verify the status registers agree with             DD49.649
                                 * expected values.                                   DD49.650
                                                                                      DD49.651
3226    <macro>                  GET R!%W1,DK@ACC,R!DDCB    .Last function sent      DD49.652
3233    <macro>                  $IF (R!DFLG = DK$SLCT) OR (R!DFLG = DK$RLSE)        DD49.653
                                                                    .>>>>> 0          $IF      1
3242    <macro>                    GET R!%W2,DK@TYP,R!DDCB  .Device type              DD49.654
3251    <macro>                    $IF (R!STATO # R!%W1 < D'8 + 3) .Expected status?  DD49.655
                                                                    .>>>>> 1          $IF      1
3261   010003  024645                R!DSTAT = S$STATO      .Set error code          DD49.656
```

```
3263    <macro>                          $ELSEIF (R!%W2 = DSK$DD39)  .If DD39 type                          DD49.657
3270    020665  017000 /001400             R!DSTAT = S$STAT1, R!STAT1 # O'1400                             DD49.658
        102003  010004  024645
3276    <macro>                          $ELSEIF (R!%W2 = DSK$DD40)  .DD40                            R13739AU.51
3303    020665  015000 /007700             A = R!STAT1 & O'7700 - O'1400                              R13739AU.52
        017000 /001400
3310    102003  010004  024645             R!DSTAT = S$STAT1, A # 0                                   R13739AU.53
3313    <macro>                          $ELSEIF (R!STAT1 # 0)       .Bad general status                     DD49.659
3317    010004  024645                     R!DSTAT = S$STAT1         .Error code                           DD49.660
3321    <macro>                          $ENDIF                                                            DD49.661
                                                                                   .<<<<< 1                  $ENDIF    1
3321    <macro>                        $ELSE                                                               DD49.662
3322    <macro>                          GET A,DK@CNG,R!DDCB         .Check chain flag                     DD49.663
3330    <macro>                          $IF (A # 0)                 .Overrun?                             DD49.664
                                                                                   .>>>>> 1                  $IF       1
3332    010001  024663                     R!DKOVR = 1               .Set overrun flag                     DD49.665
3334    <macro>                            FLDADD 1,DK@OVO,R!DDCB    .Count it                             DD49.666
3340    <macro>                            $IF (R!DTYPE # DSK$DD40)  .Don't sync DD40                   R13739AU.54
                                                                                   .>>>>> 2                  $IF       1
3344    <macro>                              STORE O'37,DK@HED,R!DDCB .Force head select                R13739AU.55
3355    <macro>                            $ENDIF                                                     R13739AU.56
                                                                                   .<<<<< 2                  $ENDIF    1
3355    <macro>                          $ELSE                       .If not overrun case                  DD49.668
3356    <macro>                            $IF (R!STATO # R!%W1 < D'8 + 1) .Expected status?               DD49.669
                                                                                   .>>>>> 2                  $IF       1
3366    010003  024645                       R!DSTAT = S$STATO         .Error code                           DD49.670
3370    <macro>                            $ELSEIF (R!STAT1 # R!%W1) .If bad echo                          DD49.671
3375    010004  024645                       R!DSTAT = S$STAT1                                               DD49.672
3377    <macro>                            $ENDIF                                                            DD49.673
                                                                                   .<<<<< 2                  $ENDIF    1
3377    <macro>                          $ENDIF                                                            DD49.674
                                                                                   .<<<<< 1                  $ENDIF    1
3377    <macro>                        $ENDIF                                                              DD49.675
                                                                                   .<<<<< 0                  $ENDIF    1
3377    <macro>                        $IF (R!DSTAT # S$OKAY)        .If error                             DD49.676
                                                                                   .>>>>> 0                  $IF       1
3402    <macro>                          PUT R!STATO,DK@ST1,R!DDCB   .Save status 0                        DD49.677
3407    <macro>                          PUT R!STAT1,DK@ST2,R!DDCB   .Save status 1                        DD49.678
3414    <macro>                        $ENDIF                                                              DD49.679
                                                                                   .<<<<< 0                  $ENDIF    1
3414    001000                         EXIT                                                                DD49.680
                                                                                                           DD49.681
```

```
                        *        Activate demon                                                  DD49.682
                                                                                                 DD49.683
3415                    ACTDEM   *                                                               DD49.684
3415   020630  024000           R!DT = R!DDCB                 .DCB address                      DD49.685
3417    <macro>                 RKERN D4ACT                                                     DD49.686
3421   001000                   EXIT                                                            DD49.687
                                                                                                 DD49.688
                        *        Handle errors detected by done/busy                             DD49.689
                                                                                                 DD49.690
3422                    D4ERROR  *                                                               DD49.691
3422    <macro>                 STORE 0,DK@CNG,R!DDCB         .Clear chaining                   DD49.692
3431    <macro>                 $IF (R!DDN = 0)               .Done/busy both clear             DD49.693
                                                                                .>>>>> 0            $IF      1
3434   010001  024645             R!DSTAT = S$DONE                                                DD49.694
3436    <macro>                 $ELSE                         .Done/busy both set               DD49.695
3437   010002  024645             R!DSTAT = S$BUSY                                                DD49.696
3441    <macro>                   PUT R!STAT0,DK@ST1,R!DDCB   .Save status 0                    DD49.697
3446    <macro>                   PUT R!STAT1,DK@ST2,R!DDCB   .Save status 1                    DD49.698
3453    <macro>                 $ENDIF                                                          DD49.699
                                                                                .<<<<< 0            $ENDIF   1
3453   070034                   P = ERROR                     .Activate error recovery          DD49.700
                                                                                                 DD49.701
                        *        Bad LMA register value                                          DD49.702
                                                                                                 DD49.703
3454                    BADADDR  *                                                               DD49.704
3454    <macro>                 PUT R!DFLG,DK@ACT,R!DDCB      .Restore activity flags           DD49.705
3456    <macro>                 PUT R!%W1,DK@ST1,R!DDCB       .Address in error                 DD49.706
3463    <macro>                 PUT R!%W4,DK@ST2,R!DDCB       .Expected address                 DD49.707
3470    <macro>                 $IF (R!DDAL # 0)              .If not read ahead                DD49.708
                                                                                .>>>>> 0            $IF      1
3473    <macro>                   FLDSUB 1,DL@DON,R!DDAL      .Take back last sector done       DD49.709
3477    <macro>                 $ENDIF                                                          DD49.710
                                                                                .<<<<< 0            $ENDIF   1
3477   010007  024645           R!DSTAT = S$FLMA              .Status                           DD49.711
3501   166000                   IOB:6                         .Disable interrupts               DD49.712
3502    <macro>                 ADDRESS R!EH,DK@TMO,R!DDCB    .Remove from timer queue          DD49.713
3505    <macro>                 RKERN DQTIME                                                    DD49.714
                                                                                                 DD49.715
                        *        Activate error recovery                                         DD49.716
                                                                                                 DD49.717
3507                    ERROR    *                                                               DD49.718
3507    <macro>                 GET A,DKF@ER,R!DDCB           .Error recovery active flag       DD49.719
```

```
3512    <macro>                         $IF (A = 0)                     .If not active                          DD49.720
                                                                                        .>>>>> 0                    $IF      1
3514    <macro>                           STORE 1,DKS@CR,R!DDCB         .Set create flag                        DD49.721
3524    <macro>                           PUT R!DSTAT,DK@STS,R!DDCB     .Save error status                      DD49.722
3531   073114                             R = ACTDEM                    .Activate demon                         DD49.723
3532    <macro>                         $ELSE                                                                   DD49.724
3533                            ERRPOP    *                                                                     DD49.725
3533    <macro>                           ADDRESS R!ED,DK@PP0,R!DDCB    .Error recovery pop cell                DD49.726
3536    <macro>                           RKERN EDQU                    .Remove from queue                      DD49.727
3540    <macro>                           $PUNTIF (R!EH = 0),CODE=PT$DD49 .If not there                         DD49.728
3543    <macro>                           PUT R!DSTAT,AD@P1,R!EH        .return error status                    DD49.729
3550    <macro>                           RKERN EQCP                    .Activate                               DD49.730
3552    <macro>                         $ENDIF                                                                  DD49.731
                                                                                        .<<<<< 0                    $ENDIF   1
3552    <macro>                         $IF (R!DSTAT # S$TOUT)          .If not timeout                         DD49.732
                                                                                        .>>>>> 0                    $IF      1
3556    <macro>                           PKERN TCHK                                                            DD49.733
3560    <macro>                         $ELSE                                                                   DD49.734
3561   001000                             EXIT                                                                  DD49.735
3562    <macro>                         $ENDIF                                                                  DD49.736
                                                                                        .<<<<< 0                    $ENDIF   1
                                                                                                                DD49.737
                                Handle device timeout - called by CLOCK                                         DD49.738
                                                                                                                DD49.739
3562                            D4TOUT    *                                                                     DD49.740
3562   020000  013044  024630           R!DDCB = R!EH - DK@TMO@P      .DCB address                            DD49.741
3565    <macro>                         GET B,DK@CHN,R!DDCB             .Channel                                DD49.742
3573   166000                           IOB:6                           .Disable interrupts                     DD49.743
3574    <macro>                         GET A,DK@ACT,R!DDCB             .Current activity flags                 DD49.744
3575    <macro>                         $IF (A = DK$RLSE)               .If release                             DD49.745
                                                                                        .>>>>> 0                    $IF      1
3600    <macro>                           STORE 0,DK@ACT,R!DDCB         .Clear                                  DD49.746
3602    <macro>                           STORE 1,DK@SEL,R!DDCB         .Show channel not selected        R13739AU.57
3612   073175                             R = ACTDEM                    .Activate demon                         DD49.748
3613   001000                             EXIT                                                                  DD49.749
3614    <macro>                         $ELSE                                                                   DD49.750
3615   010010  024645                     R!DSTAT = S$TOUT              .Error status                           DD49.751
3617   071110                             P = ERROR                     .Activate error recovery                DD49.752
3620    <macro>                         $ENDIF                                                                  DD49.753
                                                                                        .<<<<< 0                    $ENDIF   1
                                                                                                                DD49.754
                                QUAL *                                                                          DD49.755
```

END

1	$CONST		5:20 D	5:20 F	7:33 D	7:33 F	7:40 D	7:40 F	8: 2 D	8: 2 F
			15:24 D	15:24 F	15:24	15:26 D	15:26 F	15:26	17:15 D	17:15 F
			17:15	17:29 D	17:29 F	17:29	18:27 D	18:27 F	18:27	18:28 D
			18:28 F	18:28	18:39 D	18:39 F	18:40 D	18:40 F	18:40	21:32 D
			21:32 F	21:32	23:14 D	23:14 F	23:14	23:34 D	23:34 F	23:41 D
			23:41 F	23:41	23:42 D	23:42 F	24: 8 D	24: 8 F	24: 8	24:26 D
			24:26 F	24:26	26:20 D	26:20 F	26:20	27:11 D	27:11 F	28: 3 D
			28: 3 F	28: 3	28:32 D	28:32 F	28:33 D	28:33 F	28:33	
671	$REGORG		2:16 D	2:16 F	2:17	2:17 D	2:18	2:18 D	2:19	2:19 D
			2:20	2:20 D	2:21	2:21 D	2:21 F	2:22	2:22 D	2:23
			2:23 D	2:24	2:24 D	2:25	2:25 D	2:26	2:26 D	2:27
			2:27 D	2:27 F	2:28	2:28 D	2:29	2:29 D	2:30	2:30 D
			2:31	2:31 D	2:32	2:32 D	2:32 F	2:33	2:33 D	2:34
			2:34 D	2:35	2:35 D	2:36	2:36 D	2:37	2:37 D	2:38
			2:38 D	2:39	2:39 D	2:40	2:40 D	2:41	2:41 D	2:42
			2:42 D	3: 1	3: 1 D	3: 2	3: 2 D	3: 3	3: 3 D	3: 4
			3: 4 D	3: 5	3: 5 D	3: 6	3: 6 D	3: 7	3: 7 D	3: 8
			3: 8 D	3: 9	3: 9 D	3:10	3:10 D	3:11	3:11 D	3:12
			3:12 D	3:13	3:13 D	3:14	3:14 D	3:15	3:15 D	3:16
			3:16 D	3:17	3:17 D	3:18	3:18 D	3:19	3:19 D	3:20
			3:20 D	3:21	3:21 D	3:22	3:22 D	3:23	3:23 D	3:23 F
0X	%DCBQ		3:25 D	20:22						
0X	%DD49		2:11 D	2:12 D	6:37 B	7:17 B	7:27 B	7:31 B	7:32 B	8: 6 B
			11:30 B	12:35 B	13: 9 B	14:15 B	14:18 B	14:29 B	14:30 B	14:38 B
			15: 9 B	15:38 B	16: 6 B	19:21 B	20: 5 B	20:24 B	20:37 B	
10	%GBLNUM	$APTEXT	2:16 F	2:16						
400	%GBLREG	$APTEXT	2:16							
0X	%HISP		3:27 D	13:34	14: 2					
0X	%ICHAN		3:26 D	17:41						
600	%INT	$APTEXT	2:16							
0	%KERNEL	$APTEXT	5:23 D	5:23 B	7:36 D	7:36 B	11:33 D	11:33 B	15:27 D	15:27 B
			17:14 D	17:14 B	17:33 D	17:33 B	17:35 D	17:35 B	20:19 D	20:19 B
			20:23 D	20:23 B	20:27 D	20:27 B	21:18 D	21:18 B	21:33 D	21:33 B
			23:28 D	23:28 B	24:29 D	24:29 B	25:20 D	25:20 B	27: 5 D	27: 5 B
			27:37 D	27:37 B	28: 9 D	28: 9 B	28:12 D	28:12 B	28:17 D	28:17 B
0X	%MEMIO		3:28 D	13: 5	13:22	13:33	13:36	14: 1	14: 4	
0X	%MYID		3:29 D	11:10	13: 9	13:25	20:21			
0	%NP		2:32 D							
630	%P		2:32 D	2:32						
0X	%PUNT		8:34 D	8:34	9:13 D	9:13	9:41 D	9:41	25:17 D	25:17
			28:10 D	28:10						
610	%S1		2:16	2:17 D	5:20	5:21	5:37	7:40	8: 2	8: 5

		8: 6	8:20	8:21	8:22	8:23	8:24	8:41	9: 8
		9:11	9:12	9:34	9:35	9:42	10: 1	10:15	10:18
		10:34	10:39	10:42	11:11	11:25	11:41	11:42	12: 1
		12:12	12:23	12:25	12:31	12:42	13:11	13:13	13:17
		13:18	13:25	13:28	13:30	13:40	14:14	14:33	14:37
		15: 4	15: 5	15: 6	15:12	15:13	15:14	15:15	15:16
		15:24	15:26	15:30	15:42	16: 5	16:13	16:14	16:19
		16:29	17:15	17:22	17:25	17:26	17:29	18: 9	18:18
		18:27	18:28	18:41	18:42	19: 1	19: 2	19: 3	19:24
		19:27	19:39	19:40	19:42	20: 8	20:40	21: 1	21:27
		21:28	21:29	21:32	22:10	22:37	23: 1	23: 9	23:13
		23:14	23:17	23:34	23:41	23:42	24: 2	24: 8	24:10
		24:16	24:17	24:20	24:25	24:41	25: 2	25:11	25:36
		25:39	25:41	26:13	26:17	26:20	26:25	26:37	26:38
		27:11	27:17	27:18	27:27	27:28	27:31	28: 3	28: 4
		28:11	28:27	28:33					
611	%S2	2:16	2:18 D						
612	%S3	2:16	2:19 D						
613	%S4	2:16	2:20 D						
614	%S5	2:16	2:21 D						
0X	%SSD	3:30 D	13:37	14: 5					
615	%T1	2:22 D	12:37	13: 1	13: 7	15:25	16: 1	17:12	21:39
616	%T2	2:23 D	12:37	13: 1	13: 7	15:25	16: 1	21:39	
617	%T3	2:24 D	12:37	13: 1	13: 7	15:25	16: 1	21:39	
620	%T4	2:25 D							
621	%T5	2:26 D							
622	%T6	2:27 D							
0X	%TR1	23:20 D	23:20	24:31 D	24:31				
0X	%TR2	23:20 D	23:20	24:31 D	24:31				
0X	%TR3	23:20 D	23:20	24:31 D	24:31				
0X	%TR4	23:20 D	23:20	24:31 D	24:31				
0X	%TR5	23:20 D	23:20	24:31 D	24:31				
0X	%TRAC	23:20 D	23:20	24:31 D	24:31				
0X	%TVNT	23:20 D	23:20	24:31 D	24:31				
623	%W1	2:28 D	5:37	6:39	9:34	9:36	10:10	10:11	10:19
		10:35	11: 1	12:20	12:35	15: 1	15: 2	15: 3	15: 4
		15: 5	15:19	15:20	15:23	15:30	15:31	15:34	15:35
		15:36	15:37	15:38	16:18	16:21	16:25	17:12	17:13
		18: 9	18:22	19: 5	19:11	19:39	19:42	21:28	21:31
		22:35	22:38	22:40	23: 5	23: 9	23:10	23:15	23:20
		24:17	24:31	25:36	25:41	26:25	26:27	27:27	
624	%W2	2:29 D	6:34	6:36	9:35	9:36	10:34	10:35	12:21

			12:26	12:28	12:32	15: 4	15: 8	15:42	16: 1	16: 2
			16:17	16:21	19:40	19:42	21:29	21:31	22:13	22:14
			22:17	22:20	22:33	23: 2	23: 4	23: 5	23:10	23:13
			23:17	23:18	23:20	24:22	24:25	25:39	26: 1	26: 4
625	%W3		2:30 D	12:23	12:25	12:27	15: 5	15: 8	15:14	15:23
			15:38	17:11	17:12	17:13				
626	%W4		2:31 D	12:32	12:34	12:35	13:28	13:40	15: 6	15: 8
			15:15	15:20	15:36	15:37	16: 3	16: 4	16:19	16:20
			16:22	27:28						
627	%W5		2:32 D	7:22	13: 6	13:23	13:34	13:37	14: 2	14: 5
			14:32	14:36	15:16	15:19	15:31	15:35	16: 2	16: 3
			16: 4	16: 5	19: 2	19: 3				
3415	ACIDJM		11:30	14:15	16: 6	19:21	20: 5	20:24	20:37	24: 1
			24: 9	24:38	25:10	27: 3 D	28: 5	28:34		
20	AD@P1@N	$APTEXT	28:11 F							
14	AD@P1@P	$APTEXT	28:11 F	28:11						
0	AD@P1@S	$APTEXT	28:11 F							
3454	BADADDR		12:35	27:25 D						
2	BP$DONE	$APTEXT	11:12							
653	BREAK		3:10 D	5:16	10:22	18:22	19: 5			
0X	CA		3:31 D	15:32	16: 9	16:32	16:34	16:36	16:42	17: 2
			17: 4							
0X	CB		3:32 D	15:12	16:13	17:13				
0X	CC		3:33 D	15:13	16:14	17:13				
0X	CD		3:34 D	14:33						
0X	CL		3:35 D	14:34						
0X	COMIO		3:36 D	17:14 D	17:14					
656	CONT		3:13 D	5:15	18:23	19: 6	19:12	19:17	20: 1	20:34
			21:10							
0X	D$D4DEM		3:37 D	17:34						
1000	D4$WPS	$APTEXT	12:21	14:34						
60	D40$SPT	$APTEXT	22:16							
2646	D40A		22:22	22:30 D						
2647	D40B		22:24	22:31 D						
2650	D40C		22:26	22:32 D						
2651	D40D		22:28	22:33 D						
0X	D4ACT		27: 5 D	27: 5						
2775	D4CNTL		8: 6	23:33 D						
3174	D4CNTL1		24:21	24:37 D						
3226	D4DONE		7:31	25:31 D						
3422	D4ERROR		6:37	7:27	27:10 D					
2	D4HDLTM	$APTEXT	24:28							

2	D4IOLIM	$APIEXT	7:35						
3562	D4IOUT		4: 8 E	28:25 D					
0X	DACT		17:35 D	17:35					
12	DAE$PIRN	$APIEXT	14:30	17:28					
1	DAE$READ	$APIEXT	11:24						
14	DAE$WRBH	$APIEXT	20:21						
2	DAS$ABS	$APIEXT	12:27						
650	DBZ		3: 7 D	5:12	6:26	7:27			
4	DCB@DN@P	$APIEXT	20:22						
0X	DCCB		3:38 D	5:17					
641	DCUR		2:42 D	11:41	12: 3	12: 6	12:31	13: 1	14:33	15: 1
			15: 6	15:26	17:15	17:25	17:26	17:29	17:31	21:23
			21:28							
635	DCYL		2:38 D	8:20	8:33	18:41	23:18			
6	DD49		4: 9 E	5: 5 D						
140	DD49A		6:10	7:12 D						
631	DDAL		2:34 D	9:19	9:40	9:42	10: 1	10: 3	13: 4	13:14
			13:17	13:18	13:28	15:18	15:42	16:13	16:14	16:19
			17:21	17:22	17:26	17:40	27:30	27:31		
632	DDAL1		2:35 D	9:20	9:28	9:30	9:31	9:34	9:35	9:40
			10: 9	17:40						
633	DDAL2		2:36 D	9:21	10: 9	10:11	10:12	10:15	10:18	10:21
			10:32	10:33	18: 8	20:34				
634	DDAL3		2:37 D	9:22	10:33	10:35	10:36	10:39	10:42	11: 3
			11:10	11:11	11:12	13:11	13:14			
630	DDCB		2:33 D	5:18	5:19	5:20	5:21	5:22	5:33	5:37
			7:16	7:33	7:40	8: 2	8:20	8:21	8:22	8:23
			8:24	8:41	9: 8	9:11	9:12	9:28	10:34	11:25
			11:41	11:42	12: 1	12:12	12:23	12:37	12:42	13: 7
			14:37	15:12	15:13	15:14	15:15	15:16	15:24	15:25
			15:30	16: 1	16: 5	18: 9	18:27	18:28	18:39	18:40
			18:41	18:42	19: 1	19: 2	19: 3	19:24	19:27	19:39
			19:40	20: 8	20:18	20:40	21: 1	21:39	22:10	22:13
			23: 1	23: 9	23:13	23:14	23:17	23:34	23:41	23:42
			24: 2	24: 8	24:10	24:16	24:17	24:20	24:25	24:26
			24:27	24:31	24:41	25: 2	25:11	25:36	25:39	26:13
			26:17	26:20	26:37	26:38	27: 4	27:11	27:17	27:18
			27:26	27:27	27:28	27:36	27:42	28: 3	28: 4	28: 8
			28:26	28:27	28:29	28:32	28:33			
647	DDN		3: 6 D	5:10	5:36	6:25	7:19	7:27	7:30	8:34
			8:40	9: 7	10: 8	11:29	18: 6	19:38	20:33	27:13
657	DFLG		3:14 D	5:33	8: 1	8: 5	9:24	10:32	12:27	12:33

			18: 4	19:36	20:15	23:40	24: 7	24:15	25: 9	25:38
			27:26							
644	DENC		3: 3 D	9:42	11:24	13:11	13:13	13:25	13:30	14:14
			14:30	16:29	17:28	20:21	23:15			
640	DERST		2:41 D	11:42	12: 4	15: 2	21:26			
636	DHED		2:39 D	8:21	8:29	8:32	8:41	18:42	21:15	22:20
			22:35	23:20	24:16	24:19	24:20	24:22	24:31	
20	DK$HEAD	$APTEXT	25: 9							
4	DK$RAHD	$APTEXT	9:24	19:36						
1	DK$READ	$APTEXT	18: 4							
100	DK$RLSE	$APTEXT	24: 7	25:38	28:31					
10	DK$SEEK	$APTEXT	24:15							
40	DK$SLCT	$APTEXT	23:40	25:38						
2	DK$WRIT	$APTEXT	8: 1	10:32	12:27	12:33	20:15			
1	DK@ABI@M	$APTEXT	11:25	15:24 F	18:27 F					
1	DK@ABI@N	$APTEXT	11:25	11:25 F	15:24 F	15:24	18:27 F	18:27		
2	DK@ABI@P	$APTEXT	11:25 F	11:25	15:24 F	15:24	18:27 F	18:27		
1	DK@ABI@S	$APTEXT	11:25	11:25 F	15:24 F	15:24	18:27 F	18:27		
20	DK@ACC@N	$APTEXT	23:13 F	24:25 F	25:36	25:36 F				
53	DK@ACC@P	$APTEXT	23:13 F	23:13	24:25 F	24:25	25:36 F	25:36		
0	DK@ACC@S	$APTEXT	23:13 F	24:25 F	25:36	25:36 F				
20	DK@ACI@N	$APTEXT	5:33	5:33 F	7:33 F	27:26 F	28:29	28:29 F	28:32 F	
0	DK@ACI@P	$APTEXT	5:33 F	7:33 F	27:26 F	28:29 F	28:32 F			
0	DK@ACI@S	$APTEXT	5:33	5:33 F	7:33 F	27:26 F	28:29	28:29 F	28:32 F	
1	DK@ADD@M	$APTEXT	12:12	12:37						
1	DK@ADD@N	$APTEXT	12:12	12:12 F	12:37 F	12:37				
2	DK@ADD@P	$APTEXT	12:12 F	12:12	12:37 F	12:37				
2	DK@ADD@S	$APTEXT	12:12	12:12 F	12:37	12:37 F				
20	DK@BFO@N	$APTEXT	15:12	15:12 F						
63	DK@BFO@P	$APTEXT	15:12 F	15:12						
0	DK@BFO@S	$APTEXT	15:12	15:12 F						
20	DK@BFI@N	$APTEXT	15:13	15:13 F						
64	DK@BFI@P	$APTEXT	15:13 F	15:13						
0	DK@BFI@S	$APTEXT	15:13	15:13 F						
10	DK@BIO@N	$APTEXT	15:14	15:14 F						
17	DK@BIO@P	$APTEXT	15:14 F	15:14						
0	DK@BIO@S	$APTEXT	15:14	15:14 F						
1	DK@CE@M	$APTEXT	23:42 F							
1	DK@CE@N	$APTEXT	23:42 F							
2	DK@CE@P	$APTEXT	23:42 F	23:42						
6	DK@CE@S	$APTEXT	23:42 F							
176777	DK@CE@X	$APTEXT	23:42							

777	DK@CHN@M	$APTEXT	28:27								
11	DK@CHN@N	$APTEXT	28:27								
7	DK@CHN@P	$APTEXT	28:27 F	28:27							
7	DK@CHN@S	$APTEXT	28:27	28:27 F							
20	DK@CLB@N	$APTEXT	11:41	11:41 F	12:42 F						
13	DK@CLB@P	$APTEXT	11:41 F	11:41	12:42 F	12:42					
0	DK@CLB@S	$APTEXT	11:41	11:41 F	12:42 F						
1	DK@CNG@M	$APTEXT	7:40 F	23:14 F	26:13	27:11 F					
1	DK@CNG@N	$APTEXT	7:40 F	23:14 F	23:14	26:13	26:13 F	27:11 F			
2	DK@CNG@P	$APTEXT	7:40 F	7:40	23:14 F	23:14	26:13 F	26:13	27:11 F	27:11	
4	DK@CNG@S	$APTEXT	7:40 F	23:14 F	23:14	26:13	26:13 F	27:11 F			
173777	DK@CNG@X	$APTEXT	7:40	27:11							
377	DK@CNI@M	$APTEXT	15:15								
10	DK@CNI@N	$APTEXT	15:15								
16	DK@CNI@P	$APTEXT	15:15 F	15:15							
10	DK@CNI@S	$APTEXT	15:15	15:15 F							
20	DK@CYL@N	$APTEXT	8:20	8:20 F							
3	DK@CYL@P	$APTEXT	8:20 F	8:20							
0	DK@CYL@S	$APTEXT	8:20	8:20 F							
377	DK@DIO@M	$APTEXT	15:16	15:30	16: 5						
10	DK@DIO@N	$APTEXT	15:16	15:30 F	16: 5 F						
17	DK@DIO@P	$APTEXT	15:16 F	15:16	15:30 F	15:30	16: 5 F	16: 5			
10	DK@DIO@S	$APTEXT	15:16	15:16 F	15:30 F	16: 5 F					
177400	DK@DIO@X	$APTEXT	15:30	16: 5							
14	DK@DLO@P	$APTEXT	9:28	20:18							
20	DK@FLB@N	$APTEXT	11:42	11:42 F							
11	DK@FLB@P	$APTEXT	11:42 F	11:42							
0	DK@FLB@S	$APTEXT	11:42	11:42 F							
20	DK@HDO@N	$APTEXT	25:11 F								
43	DK@HDO@P	$APTEXT	25:11 F	25:11							
37	DK@HED@M	$APTEXT	8:21	8:41	24:17	24:20	26:20 F				
5	DK@HED@N	$APTEXT	8:21	8:21 F	8:41 F	8:41	24:17	24:17 F	24:20 F	24:20	
			26:20 F	26:20							
4	DK@HED@P	$APTEXT	8:21 F	8:21	8:41 F	8:41	24:17 F	24:17	24:20 F	24:20	
			26:20 F	26:20							
2	DK@HED@S	$APTEXT	8:21	8:21 F	8:41 F	8:41	24:17	24:17 F	24:20 F	24:20	
			26:20 F	26:20							
1	DK@IDI@M	$APTEXT	5:20 F								
1	DK@IDI@N	$APTEXT	5:20 F								
7	DK@IDI@P	$APTEXT	5:20 F	5:20							
1	DK@IDI@S	$APTEXT	5:20 F								
137777	DK@IDI@X	$APTEXT	5:20								

1	DK@INV@M	$APTEXT	18:28 F								
1	DK@INV@N	$APTEXT	18:28 F	18:28							
16	DK@INV@P	$APTEXT	18:28 F	18:28							
1	DK@INV@S	$APTEXT	18:28 F	18:28							
1	DK@LCK@N	$APTEXT	14:37	14:37 F							
16	DK@LCK@P	$APTEXT	14:37 F	14:37							
0	DK@LCK@S	$APTEXT	14:37	14:37 F							
20	DK@LLB@N	$APTEXT	12: 1	12: 1 F							
12	DK@LLB@P	$APTEXT	12: 1 F	12: 1							
0	DK@LLB@S	$APTEXT	12: 1	12: 1 F							
1	DK@LMA@M	$APTEXT	21:39								
1	DK@LMA@N	$APTEXT	5:37	5:37 F	21:39 F	21:39					
2	DK@LMA@P	$APTEXT	5:37 F	5:37	21:39 F	21:39					
0	DK@LMA@S	$APTEXT	5:37	5:37 F	21:39	21:39 F					
37	DK@NHD@M	$APTEXT	24:16								
5	DK@NHD@N	$APTEXT	24:16								
51	DK@NHD@P	$APTEXT	24:16 F	24:16							
13	DK@NHD@S	$APTEXT	24:16	24:16 F							
4	DK@NLB@N	$APTEXT	10:34	10:34 F							
56	DK@NLB@P	$APTEXT	10:34 F	10:34							
0	DK@NLB@S	$APTEXT	10:34	10:34 F							
20	DK@OVO@N	$APTEXT	26:17 F								
42	DK@OVO@P	$APTEXT	26:17 F	26:17							
20	DK@OWC@N	$APTEXT	8: 2 F								
65	DK@OWC@P	$APTEXT	8: 2 F	8: 2	22:13						
0	DK@OWC@S	$APTEXT	8: 2 F								
30	DK@PPO@P	$APTEXT	28: 8								
20	DK@RAO@N	$APTEXT	20: 8 F								
26	DK@RAO@P	$APTEXT	20: 8 F	20: 8							
20	DK@RAC@N	$APTEXT	18:41 F								
5	DK@RAC@P	$APTEXT	18:41 F	18:41							
0	DK@RAC@S	$APTEXT	18:41 F								
377	DK@RAD@M	$APTEXT	13: 7	15:25	19:40						
10	DK@RAD@N	$APTEXT	13: 7 F	13: 7	15:25 F	15:25	19:40				
10	DK@RAD@P	$APTEXT	13: 7 F	13: 7	15:25 F	15:25	19:40 F	19:40			
10	DK@RAD@S	$APTEXT	13: 7	13: 7 F	15:25	15:25 F	19:40	19:40 F			
177400	DK@RAD@X	$APTEXT	13: 7	15:25							
37	DK@RAH@M	$APTEXT	18:42								
5	DK@RAH@N	$APTEXT	18:42 F	18:42							
6	DK@RAH@P	$APTEXT	18:42 F	18:42							
2	DK@RAH@S	$APTEXT	18:42 F	18:42							
777	DK@RAS@M	$APTEXT	19: 1								

11	DK@RAS@N	$APTEXT	19: 1	F							
6	DK@RAS@P	$APTEXT	19: 1	F	19: 1						
7	DK@RAS@S	$APTEXT	19: 1	F							
177000	DK@RAS@X	$APTEXT	19: 1								
3	DK@RAU@M	$APTEXT	19: 3								
2	DK@RAU@N	$APTEXT	19: 3	F	19: 3						
6	DK@RAU@P	$APTEXT	19: 3	F	19: 3						
0	DK@RAU@S	$APTEXT	19: 3	F	19: 3						
377	DK@RCC@M	$APTEXT	18: 9		19:39						
10	DK@RCC@N	$APTEXT	18: 9		19:39						
54	DK@RCC@P	$APTEXT	18: 9	F	18: 9		19:39	F	19:39		
10	DK@RCC@S	$APTEXT	18: 9		18: 9	F	19:39		19:39	F	
20	DK@RD0@N	$APTEXT	19:27	F							
20	DK@RD0@P	$APTEXT	19:27	F	19:27						
20	DK@RD1@N	$APTEXT	19:24	F							
21	DK@RD1@P	$APTEXT	19:24	F	19:24						
20	DK@RL0@N	$APTEXT	24:10	F							
41	DK@RL0@P	$APTEXT	24:10	F	24:10						
777	DK@SEC@M	$APTEXT	8:22		9: 8						
11	DK@SEC@N	$APTEXT	8:22		9: 8	F					
4	DK@SEC@P	$APTEXT	8:22	F	8:22		9: 8	F	9: 8		
7	DK@SEC@S	$APTEXT	8:22		8:22	F	9: 8	F			
177000	DK@SEC@X	$APTEXT	9: 8								
1	DK@SEL@M	$APTEXT	23:41	F	24: 8	F	28:33	F			
1	DK@SEL@N	$APTEXT	23:41	F	23:41		24: 8	F	24: 8	28:33	F 28:33
7	DK@SEL@P	$APTEXT	23:41	F	23:41		24: 8	F	24: 8	28:33	F 28:33
0	DK@SEL@S	$APTEXT	23:41	F	23:41		24: 8	F	24: 8	28:33	F 28:33
20	DK@SK0@N	$APTEXT	25: 2	F							
24	DK@SK0@P	$APTEXT	25: 2	F	25: 2						
20	DK@SK1@N	$APTEXT	24:41	F							
25	DK@SK1@P	$APTEXT	24:41	F	24:41						
20	DK@SL0@N	$APTEXT	24: 2	F							
40	DK@SL0@P	$APTEXT	24: 2	F	24: 2						
20	DK@SP0@N	$APTEXT	23: 1		23: 1	F					
62	DK@SP0@P	$APTEXT	23: 1	F	23: 1						
0	DK@SP0@S	$APTEXT	23: 1		23: 1	F					
777	DK@SP1@M	$APTEXT	8:23								
11	DK@SP1@N	$APTEXT	8:23								
60	DK@SP1@P	$APTEXT	8:23	F	8:23						
7	DK@SP1@S	$APTEXT	8:23		8:23	F					
20	DK@ST1@N	$APTEXT	26:37	F	27:17	F	27:27	F			
50	DK@ST1@P	$APTEXT	26:37	F	26:37		27:17	F	27:17	27:27	F 27:27

0	DK@SI1@S	$APIEXT	26:37 F	27:17 F	27:27 F				
20	DK@SI2@N	$APIEXT	26:38 F	27:18 F	27:28 F				
52	DK@SI2@P	$APIEXT	26:38 F	26:38	27:18 F	27:18	27:28 F	27:28	
0	DK@SI2@S	$APIEXT	26:38 F	27:18 F	27:28 F				
20	DK@SIS@N	$APIEXT	28: 4 F						
36	DK@SIS@P	$APIEXT	28: 4 F	28: 4					
0	DK@SIS@S	$APIEXT	28: 4 F						
37	DK@SUB@M	$APIEXT	12:23	23: 9					
5	DK@SUB@N	$APIEXT	12:23	23: 9					
56	DK@SUB@P	$APIEXT	12:23 F	12:23	23: 9 F	23: 9			
13	DK@SUB@S	$APIEXT	12:23	12:23 F	23: 9	23: 9 F			
3	DK@SYN@M	$APIEXT	23:34 F						
2	DK@SYN@N	$APIEXT	23:34 F						
2	DK@SYN@P	$APIEXT	23:34 F	23:34					
10	DK@SYN@S	$APIEXT	23:34 F						
177477	DK@SYN@X	$APIEXT	23:34						
4	DK@TM@N	$APIEXT	9:12	9:12 F					
51	DK@TM@P	$APIEXT	9:12 F	9:12					
0	DK@TM@S	$APIEXT	9:12	9:12 F					
44	DK@TMO@P	$APIEXT	5:22	24:27	27:36	28:26			
7	DK@TMP@M	$APIEXT	9:11						
3	DK@TMP@N	$APIEXT	9:11	9:11 F					
51	DK@TMP@P	$APIEXT	9:11 F	9:11					
4	DK@TMP@S	$APIEXT	9:11	9:11 F					
37	DK@TPC@M	$APIEXT	8:24						
5	DK@TPC@N	$APIEXT	8:24	8:24 F					
60	DK@TPC@P	$APIEXT	8:24 F	8:24					
2	DK@TPC@S	$APIEXT	8:24	8:24 F					
37	DK@TYP@M	$APIEXT	5:21	22:10	25:39				
5	DK@TYP@N	$APIEXT	5:21	5:21 F	22:10	22:10 F	25:39	25:39 F	
7	DK@TYP@P	$APIEXT	5:21 F	5:21	22:10 F	22:10	25:39 F	25:39	
2	DK@TYP@S	$APIEXT	5:21	5:21 F	22:10	22:10 F	25:39	25:39 F	
2	DK@UNI@N	$APIEXT	19: 2	19: 2 F	23:17	23:17 F			
4	DK@UNI@P	$APIEXT	19: 2 F	19: 2	23:17 F	23:17			
0	DK@UNI@S	$APIEXT	19: 2	19: 2 F	23:17	23:17 F			
377	DK@WBD@M	$APIEXT	16: 1						
10	DK@WBD@N	$APIEXT	16: 1 F	16: 1					
10	DK@WBD@P	$APIEXT	16: 1 F	16: 1					
0	DK@WBD@S	$APIEXT	16: 1	16: 1 F					
20	DK@WRO@N	$APIEXT	21: 1 F						
22	DK@WRO@P	$APIEXT	21: 1 F	21: 1					
20	DK@WRI@N	$APIEXT	20:40 F						

23	DK@WR1@P	$APTEXT	20:40	F	20:40						
1	DKF@ER@M	$APTEXT	27:42								
1	DKF@ER@N	$APTEXT	27:42		27:42	F					
0	DKF@ER@P	$APTEXT	27:42	F							
10	DKF@ER@S	$APTEXT	27:42		27:42	F					
1	DKF@HD@M	$APTEXT	24:26	F							
1	DKF@HD@N	$APTEXT	24:26	F	24:26						
0	DKF@HD@P	$APTEXT	24:26	F							
13	DKF@HD@S	$APTEXT	24:26	F	24:26						
1	DKF@IM@M	$APTEXT	7:16								
1	DKF@IM@N	$APTEXT	7:16		7:16	F					
0	DKF@IM@P	$APTEXT	7:16	F							
7	DKF@IM@S	$APTEXT	7:16		7:16	F					
1	DKF@RA@M	$APTEXT	18:40	F							
1	DKF@RA@N	$APTEXT	18:40	F	18:40						
0	DKF@RA@P	$APTEXT	18:40	F							
15	DKF@RA@S	$APTEXT	18:40	F	18:40						
1	DKF@RD@M	$APTEXT	18:39	F							
1	DKF@RD@N	$APTEXT	18:39	F							
0	DKF@RD@P	$APTEXT	18:39	F							
17	DKF@RD@S	$APTEXT	18:39	F							
177776	DKF@RD@X	$APTEXT	18:39								
663	DKOVR		3:18	D	5:14		8:38		26:16		
1	DKS@CR@M	$APTEXT	28: 3	F							
1	DKS@CR@N	$APTEXT	28: 3	F							
2	DKS@CR@P	$APTEXT	28: 3	F	28: 3						
17	DKS@CR@S	$APTEXT	28: 3	F							
177776	DKS@CR@X	$APTEXT	28: 3								
1	DL@CON@M	$APTEXT	10:15		10:39		18:18				
1	DL@CON@N	$APTEXT	10:15		10:15	F	10:39		10:39	F	18:18		18:18	F
21	DL@CON@P	$APTEXT	10:15	F	10:15		10:39	F	10:39		18:18	F	18:18	
6	DL@CON@S	$APTEXT	10:15		10:15	F	10:39		10:39	F	18:18		18:18	F
20	DL@DON@N	$APTEXT	9:34		9:34	F	10: 1	F	27:31	F				
13	DL@DON@P	$APTEXT	9:34	F	9:34		10: 1	F	10: 1		27:31	F	27:31	
0	DL@DON@S	$APTEXT	9:34		9:34	F								
10	DL@FC@N	$APTEXT	9:42		9:42	F								
26	DL@FC@P	$APTEXT	9:42	F	9:42									
0	DL@FC@S	$APTEXT	9:42		9:42	F								
10	DL@RAC@N	$APTEXT	13:28		13:28	F	15:42		15:42	F				
37	DL@RAC@P	$APTEXT	13:28	F	13:28		15:42	F	15:42					
0	DL@RAC@S	$APTEXT	13:28		13:28	F	15:42		15:42	F				
20	DL@SEQ@N	$APTEXT	13:17		13:17	F	17:22	F						

12	DL@SEQ@P	$APTEXT	13:17 F	13:17	17:22 F	17:22					
0	DL@SEQ@S	$APTEXT	13:17	13:17 F	17:22 F						
3	DL@IDL@M	$APTEXT	11:11								
2	DL@IDL@N	$APTEXT	11:11	11:11 F							
21	DL@IDL@P	$APTEXT	11:11 F	11:11							
7	DL@IDL@S	$APTEXT	11:11	11:11 F							
37777	DL@IMO@M	$APTEXT	16:13								
16	DL@IMO@N	$APTEXT	16:13								
24	DL@IMO@P	$APTEXT	16:13 F	16:13							
2	DL@IMO@S	$APTEXT	16:13	16:13 F							
20	DL@IM1@N	$APTEXT	16:14	16:14 F							
25	DL@IM1@P	$APTEXT	16:14 F	16:14							
0	DL@IM1@S	$APTEXT	16:14	16:14 F							
20	DL@IOI@N	$APTEXT	9:35	9:35 F	10:18	10:18 F	10:42	10:42 F	13:18	13:18 F	
14	DL@IOI@P	$APTEXT	9:35 F	9:35	10:18 F	10:18	10:42 F	10:42	13:18 F	13:18	
0	DL@IOI@S	$APTEXT	9:35	9:35 F	10:18	10:18 F	10:42	10:42 F	13:18	13:18 F	
4	DL@UNS@N	$APTEXT	16:19	16:19 F							
20	DL@UNS@P	$APTEXT	16:19 F	16:19							
0	DL@UNS@S	$APTEXT	16:19	16:19 F							
642	DLAST		3: 1 D	12: 1	15: 2	21:25					
660	DLOC		3:15 D	21:27	21:42	22: 3	23:20				
643	DNUM		3: 2 D	9:29	9:30	9:36	10: 2	10:10	18:11	19:11	
			20:17								
661	DNXT		3:16 D	12: 4	12: 6	12:42	21:23	21:24	21:26	21:27	
			21:29	21:32							
0X	DQTIME		5:23 D	5:23	27:37 D	27:37					
637	DSEC		2:40 D	8:22	8:25	8:28	9: 8	19: 1	21:11	21:14	
			22:21	22:37	23: 2	23:20					
4	DSK$DD39	$APTEXT	26: 1								
6	DSK$DD40	$APTEXT	8:40	22:12	24:21	26: 4	26:19				
645	DSTAT		3: 4 D	5:13	7:32	25:42	26: 3	26: 7	26: 9	26:26	
			26:28	26:36	27:14	27:16	27:34	28: 4	28:11	28:16	
			28:37								
0X	DT		3:39 D	27: 4							
670	DTYPE		3:23 D	5:21	8:40	24:21	26:19				
0X	DY		3:40 D	20:26							
0X	ED		3:41 D	17:32	20:18	20:22	28: 8				
0X	EDQU		4: 1 D	20:19 D	20:19	28: 9 D	28: 9				
0X	EG		4: 2 D	7:35	24:28						
0X	EH		4: 3 D	5:22	17:31	20:26	24:27	27:36	28:10	28:11	
			28:26								
0X	ENQU		17:33 D	17:33	20:23 D	20:23					

0X	EQCP		3:42 D	28:12 D	28:12					
3507	ERROR		7:32	27:21	27:41 D	28:38				
3533	ERRPOP		7:17	28: 7 D						
0X	HONN		4: 4 D	20:27 D	20:27					
0X	ICHK		4: 5 D	11:33 D	11:33	15:27 D	15:27	21:18 D	21:18	21:33 D
			21:33	23:28 D	23:28	25:20 D	25:20	28:17 D	28:17	
1	LB@CP@M	$APTEXT	21:32 F							
1	LB@CP@N	$APTEXT	21:32 F	21:32						
2	LB@CP@P	$APTEXT	21:32 F	21:32						
1	LB@CP@S	$APTEXT	21:32 F	21:32						
20	LB@DAL@N	$APTEXT	17:26 F							
4	LB@DAL@P	$APTEXT	17:26 F	17:26						
0	LB@DAL@S	$APTEXT	17:26 F							
1	LB@FUL@M	$APTEXT	13: 1	15:26 F	17:29 F					
1	LB@FUL@N	$APTEXT	13: 1 F	13: 1	15: 4	15: 4 F	15: 6	15: 6 F	15:26 F	15:26
			17:29 F	17:29	21:28	21:28 F	21:29	21:29 F		
2	LB@FUL@P	$APTEXT	13: 1 F	13: 1	15: 4 F	15: 4	15: 6 F	15: 6	15:26 F	15:26
			17:29 F	17:29	21:28 F	21:28	21:29 F	21:29		
0	LB@FUL@S	$APTEXT	13: 1	13: 1 F	15: 4	15: 4 F	15: 6	15: 6 F	15:26 F	15:26
			17:29 F	17:29	21:28	21:28 F	21:29	21:29 F		
1	LB@IOI@M	$APTEXT	15: 5	17:15 F						
1	LB@IOI@N	$APTEXT	15: 5	15: 5 F	17:15 F	17:15				
2	LB@IOI@P	$APTEXT	15: 5 F	15: 5	17:15 F	17:15				
2	LB@IOI@S	$APTEXT	15: 5	15: 5 F	17:15 F	17:15				
6	LB@LE	$APTEXT	12: 6	15: 1	21:24					
20	LB@LOC@N	$APTEXT	12:31	12:31 F	14:33	14:33 F	21:27	21:27 F		
1	LB@LOC@P	$APTEXT	12:31 F	12:31	14:33 F	14:33	21:27 F	21:27		
0	LB@LOC@S	$APTEXT	12:31	12:31 F	14:33	14:33 F	21:27	21:27 F		
20	LB@SEQ@N	$APTEXT	17:25 F							
3	LB@SEQ@P	$APTEXT	17:25 F	17:25						
0	LB@SEQ@S	$APTEXT	17:25 F							
662	MDAL		3:17 D	10: 3	18:17	18:18	18:29	18:38	19:12	
4	MEM@BM@P	$APTEXT	13: 5	13:22						
0	MEM@HI@P	$APTEXT	14: 1							
2	MEM@HO@P	$APTEXT	13:33							
6	MEM@SI@P	$APTEXT	14: 4							
10	MEM@SO@P	$APTEXT	13:36							
2	MIO$BMRI	$APTEXT	16: 9	16:36						
4	MIO$BMWI	$APTEXT	15:32	17: 4						
6	MIO$HSII	$APTEXT	16:32							
10	MIO$HSOI	$APTEXT	16:42							
12	MIO$SSII	$APTEXT	16:34							

14	MIO$SSOI	$APTEXT	17: 2							
5	MOS	$APTEXT	13: 6	13:23	14:36					
0X	O$DD49		1: 4 D	1: 4						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
113	PI$BADTM	$APTEXT	9:13							
50	PI$DD49	$APTEXT	8:34	9:41	25:17	28:10				
2027	QMEMIO		13: 9	14:18	14:30	14:38	15: 9	15:38	17:19 D	
0X	QTIME		7:36 D	7:36	24:29 D	24:29				
654	QU		3:11 D	13: 2	13: 5	13:22	13:33	13:36	14: 1	14: 4
			14:29	17:32						
2	S$BUSY	$APTEXT	27:16							
1	S$DONE	$APTEXT	27:14							
7	S$ELMA	$APTEXT	27:34							
0	S$OKAY	$APTEXT	5:13	7:32	26:36					
3	S$STAIO	$APTEXT	25:42	26:26						
4	S$STAI1	$APTEXT	26: 3	26: 7	26: 9	26:28				
10	S$TOUT	$APTEXT	28:16	28:37						
646	SEQ		3: 5 D	13: 8	13:17	13:20	13:21	13:40	14:14	15:34
			15:41	16:16	16:17	17:22	17:25			
11	SM@EST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
651	SPT		3: 8 D	8:23	8:27	21:13	22:37	23: 3		
664	STAIO		3:19 D	7:21	25:41	26:25	26:37	27:17		
665	STAI1		3:20 D	7:24	26: 2	26: 5	26: 8	26:27	26:38	27:18
53	TF$D4HD	$APTEXT	24:31							
54	TF$D4IO	$APTEXT	23:20							
666	TM		3:21 D	9:12	13:27	13:32	13:42	14:36	16:31	16:33
			16:41	17: 1						
1	TM$BMR	$APTEXT	13:27	14:36						
0	TM$CMEM	$APTEXT	13:32	13:42	16:31	16:41				
3	TM$MAX	$APTEXT	9:13							
2	TM$SSD	$APTEXT	16:33	17: 1						
0X	TMCB		4: 6 D							
667	TMP		3:22 D	9:11	11:10	13: 9	13:25	20:21		
655	TOT		3:12 D	13:18	13:20	13:40	15:34	15:41		
652	TPC		3: 9 D	8:24	8:31	21:17	22:39			

SYMBOL QUALIFIER IS DD49

```
0     <macro>                    OVERLAY   DISK                                                    DISK.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                                 ************************************************************   DISK.5
                                 *                                                          *   DISK.6
                                 * Disk Demon Activity:                                     *   DISK.7
                                 *  Overlay processes DCB done queue, tries to keep local   *   DISK.8
                                 *  buffer available for disk I/O, keeps disk channel busy, *   DISK.9
                                 *  if a highspeed channel is available to the target memory *   DISK.10
                                 *  the channel is used for the data motion, otherwise MOS  *   DISK.11
                                 *  memory is used to pass the data between the IOP with the *   DISK.12
                                 *  disk and the IOP with the channel to the target memory. *   DISK.13
                                 *  Activity started by the disk interrupt routine          *   DISK.14
                                 *                                                          *   DISK.15
                                 ************************************************************   DISK.16
                                                                                                DISK.17
6    <macro>                     REGDEFS   ,,(LIM,DI,DJ,DK,DL,DM,DP,DR,MDAL,LAST,NXT,MSU,MSL,BPU   DISK.18
                                           ,BPL,PIR,TM,TMP)                                     DISK.19
                                 EQUALS    $REGORG                                                $REG   3
410      %S1                     EQUALS    $REGORG                                                $REG   3
411      %S2                     EQUALS    $REGORG                                                $REG   3
412      %S3                     EQUALS    $REGORG                                                $REG   3
413      %S4                     EQUALS    $REGORG                                                $REG   3
414      %S5                     EQUALS    $REGORG                                                $REG   3
415      %T1                     EQUALS    $REGORG                                                $REG   3
416      %T2                     EQUALS    $REGORG                                                $REG   3
417      %T3                     EQUALS    $REGORG                                                $REG   3
420      %T4                     EQUALS    $REGORG                                                $REG   3
421      %T5                     EQUALS    $REGORG                                                $REG   3
422      %T6                     EQUALS    $REGORG                                                $REG   3
423      %W1                     EQUALS    $REGORG                                                $REG   3
424      %W2                     EQUALS    $REGORG                                                $REG   3
425      %W3                     EQUALS    $REGORG                                                $REG   3
426      %W4                     EQUALS    $REGORG                                                $REG   3
427      %W5                     EQUALS    $REGORG                                                $REG   3
430      LIM                     EQUALS    $REGORG                                                $REG   3
431      DI                      EQUALS    $REGORG                                                $REG   3
432      DJ                      EQUALS    $REGORG                                                $REG   3
433      DK                      EQUALS    $REGORG                                                $REG   3
434      DL                      EQUALS    $REGORG                                                $REG   3
435      DM                      EQUALS    $REGORG                                                $REG   3
436      DP                      EQUALS    $REGORG                                                $REG   3
437      DR                      EQUALS    $REGORG                                                $REG   3
440      MDAL                    EQUALS    $REGORG                                                $REG   3
441      LAST                    EQUALS    $REGORG                                                $REG   3
442      NXT                     EQUALS    $REGORG                                                $REG   3
```

```
443     MSU     EQUALS    $REGORG                                          $REG     3
444     MSL     EQUALS    $REGORG                                          $REG     3
445     BPU     EQUALS    $REGORG                                          $REG     3
446     BPL     EQUALS    $REGORG                                          $REG     3
447     PTR     EQUALS    $REGORG                                          $REG     3
450     TM      EQUALS    $REGORG                                          $REG     3
451     TMP     EQUALS    $REGORG                                          $REG     3

                EXT       %HISP                                        DISK.21
                EXT       %MYID                                        DISK.22
                EXT       %SSD                                         DISK.23
                EXT       BMLOCL                                       DISK.24
                EXT       CA                                           DISK.25
                EXT       CB                                           DISK.26
                EXT       CC                                           DISK.27
                EXT       CD                                           DISK.28
                EXT       CE                                           DISK.29
                EXT       D$DISK                                       DISK.30
                EXT       DA                                           DISK.31
                EXT       DB                                           DISK.32
                EXT       DC                                           DISK.33
                EXT       DH                                           DISK.34
                EXT       DISKQ                                        DISK.35
                EXT       DISKQT                                       DISK.36
                EXT       DQ                                           DISK.37
                EXT       DS                                           DISK.38
                EXT       DT                                           DISK.39
                EXT       DY                                           DISK.40
                EXT       ED                                           DISK.41
                EXT       EH                                           DISK.42
                EXT       EI                                           DISK.43
                EXT       HOLOCL                                       DISK.44
                EXT       IMDAL                                        DISK.45
                EXT      DD                                            DISK.46
                EXT      DU                                            DISK.47
                EXT      EBNM                                          DISK.48
                EXT       SOLOCL                                       DISK.49
                EXT       TMCB                                         DISK.50
```

```
                                        * Take entries off Disk Queue to process.  Each entry on the      DISK.52
                                        * queue is a channel number of channel that needs processing.     DISK.53
                                                                                                          DISK.54
  6   010010  012001  024430            R!LIM = DISK$LIM + 1      .Number of DALs to process              DISK.55
 11                           DDSK      *                                                                 DISK.56
 11   002000  000000                    I = 0                     .lock out interrupts                    DISK.57
 13   014000 /000000  024410            DH = (DISKQ)                   .Get next DCB from queue           DISK.58
      030410  024000
 20   126000 /001242                    P = DIDL, A = 0                .Idle demon if queue is empty      DISK.59
                                                                                                          DISK.60
 22   003000                            I = 1                                                             DISK.61
 23   010000                            A = 0                          .For timing (allow interrupts)     DISK.62
 24   027430                            R!LIM = R!LIM - 1         .Done with max before give up           DISK.63
 25   002000  000000                    I = 0                                                             DISK.64
 27   103007                            P = DKDONET, A # 0                                                DISK.65
 30   014000 /000000                    A = D$DISK                     .DISK DESCRIPTOR                   DISK.66
 32    <macro>                          RKERN     DGIVEUP              .Relinquish control if necessary   DISK.67
 34   010010  024430                    R!LIM = DISK$LIM          .Reset limit                            DISK.68
 36                           DKDONET   *                                                                 DISK.69
 36    <macro>                          GET       EH,DD@DEM,DH                                            DISK.70
 43    <macro>                          STORE     0,DD@DEM,DH                                             DISK.71
 50   014000 /000000  024410            (DISKQ) = EH                   .Remove top entry from queue       DISK.72
      020000  034410
 55   103006  014000 /000000            (DISKQT) = 0, A = 0            .Clear last if queue is empty      DISK.73
      024410  010000  034410
                                                                                                          DISK.74
 63   030000  011120                    A = (DH) & 120            .Any ERRECK bits?                       DISK.75
 65   107054                            P = DDSK, A # 0           .Yes, let it control process            DISK.76
 66   014000 /000073  024437            R!DR = DDSQ                    .Return address                    DISK.77
 71   075000 /002431                    P = DRSC                  .Rescue Disabled Disk                   DISK.78
 73                           DDSQ      *                                                                 DISK.79
                                                                                                          DISK.80
                                                                                                          DISK.81
                                        ************************************************************      DISK.82
                                        * Channel Doing Read Ahead                                  *     DISK.83
                                        * Move data to MOS and free up local buffer if sector done  *     DISK.84
                                        ************************************************************      DISK.85
                                                                                                          DISK.86
 73    <macro>                          GET R!%W1,DB@INF,R!DH          .'FIRST'                           DISK.87
100    <macro>                          GET R!%W2,DB@UNF,R!DH          .'LAST'                            DISK.88
105    <macro>                          GET R!%W3,DB@IN,R!DH           .'IN'                              DISK.89
112    <macro>                          GET R!%W4,DB@OUT,R!DH          .'OUT'                             DISK.90
```

```
117    <macro>                          GET R!DI,DB@RAK,R!DH            .NUMBER OF ENTRIES                  DISK.91
124    <macro>                          $UNTIL (R!DI = 0)               .LOOK AT EACH ONE                   DISK.92
126    003000                             I = 1                         .Enable interrupts               R16410MB.1
127    010000                             A = 0                         .(let interrupts in)             R16410MB.2
130    002000  000000                     I = 0                         .Plenty interrupts               R16410MB.3
132    <macro>                            GET A,RA@DAT,R!%W4            .Data location                   R16410MB.4
133    <macro>                            $IF (A = 1)                   .If in local memory              R16410MB.5
                                                                                                  .>>>>> 0          $IF      1
136    <macro>                              ADDRESS R!DS,RA@BMO,R!%W4   .ALLOCATE MOS BUFFER                DISK.97
141    077000 /002602                       R = DGETBIG                 .                                   DISK.98
143    <macro>                              GET R!CD,RA@LOC,R!%W4       .LOCAL BUFFER ADDRESS               DISK.99
150    014000 /001000  024000               R!CE = D'512                .1 SECTOR                           DISK.100
153    010004  024000                       R!CA = MIO$BMWI             .WRITE TO MOS                       DISK.101
155    <macro>                              RKERN COMIO                 .                                   DISK.102
157    <macro>                              STORE 2,RA@DAT,R!%W4        .SET DATA LOCATION                  DISK.103
161    <macro>                              STORE 0,RA@LOC,R!%W4        .CLEAR LOCAL MEMORY ADDRESS         DISK.104
166    020000  024000                       R!DY = R!CD                 .RETURN BUFFER                      DISK.105
170    <macro>                              RKERN HPUTBUF               .                                   DISK.106
172    <macro>                            $ENDIF                                                            DISK.107
                                                                                                  .<<<<< 0          $ENDIF   1
172    <macro>                            $IF (R!%W4 = R!%W2)           .ADVANCE OUT                        DISK.108
                                                                                                  .>>>>> 0          $IF      1
176    020423  024426                       R!%W4 = R!%W1               .IF WRAPPED - SET TO FIRST          DISK.109
200    <macro>                            $ELSE                                                             DISK.110
201    010006  025426                       R!%W4 = R!%W4 + RA@LEN@P    .NEXT                               DISK.111
203    <macro>                            $ENDIF                                                            DISK.112
                                                                                                  .<<<<< 0          $ENDIF   1
203    027431                             R!DI = R!DI - 1               .DECREMENT COUNT                    DISK.113
204    <macro>                          $ENDTIL                                                             DISK.114
205    002000  000000                   I = 0                           .LOCK INTERRUPTS                    DISK.115
```

```
                                        ************************************************************      DISK.117
                                        *                                                          *      DISK.118
                                        * DDSX:                                                    *      DISK.119
                                        * Look at Channel's Done Queue; process its contents      *      DISK.120
                                        * Writes:  Deallocate DAL's and MOS                        *      DISK.121
                                        * Reads:  Send data to BIOP for Cray main memory transfer  *      DISK.122
                                        *                                                          *      DISK.123
                                        ************************************************************      DISK.124
                                                                                                          DISK.125
207   020000  012014  024410   DDSX     DI = (DH + DD@DNQ)        .Done Queue                           DISK.126
      030410  024431
214   126000 /001246                    P = DDKA, A = 0           .Anything?                            DISK.127
216   020000  012014  024000            ED = DH + DD@DNQ          .Yes, Take Off Queue                  DISK.128
221    <macro>                          RKERN    EDQU             .Remove                               DISK.129
223   003000                            I = 1                     .Allow Interrupts                     DISK.130
224    <macro>                          GET R!DJ,DA@MDL,R!DI            .M-DAL address                  DISK.131
231    <macro>                          GET R!DL,DA@FC,R!DI             .Function code                  DISK.132
237    <macro>                          GET R!TM,DA@TM,R!DI             .Target memory type             DISK.133
245   020450  016000 /000000            R!TMP = (R!TM + TMCB)           .IM control table               DISK.134
      024410  030410  024451
253    <macro>                          $PUNTIF (A = 0),CODE=PT$BPATH .If undefined path                DISK.135
255    <macro>                          GET R!TMP,TM@IOP,R!TMP          .Target memory processor        DISK.136
262   020434  011001  024433            DK = DL & 1               .Check if read                        DISK.137
265   103206                            P = DSRD, A # 0                                                 DISK.138
                                                                                                          DISK.139
                                        * Process write DAL from DCB done queue                         DISK.140
                                        *  Release DAL and buffer space for reads and writes            DISK.141
266   002000  000000                    I = 0                     .Lock                                 DISK.142
270                           DNDAL     *                                                               DISK.143
270   020431  012002  024000            DS = DI + DA@DLO@P              .DAL MOS area                 R15472BD.1
273   030000  024000                    CB = (DS)                       .Get the DAL address            DISK.145
275   010000  034000                    (DS) = 0                                                        DISK.146
277   026000                            DS = DS + 1                                                     DISK.147
300   030000  024000                    CC = (DS)                                                       DISK.148
302   010000  034000                    (DS) = 0                                                        DISK.149
                                                                                                          DISK.150
304    <macro>                          $IF       (CB # 0),OR,(CC # 0) .If a DAL allocated              DISK.151
                                                                                  .>>>>> 0                $IF      1
311   010001  024000                      R!CA = 1                      .Release 1 DAL                  DISK.152
313    <macro>                            RKERN   PUTDAL                .Release the DAL                DISK.153
315    <macro>                          $ENDIF                                                          DISK.154
                                                                                  .<<<<< 0                $ENDIF   1
```

```
315    <macro>                          GET R!DY,DA@LOC,R!DI          .CHECK FOR LOCAL BUFFER          DISK.155
322    <macro>                          $IF (A # 0)                   .IF THERE                        DISK.156
                                                                                        .>>>>> 0          $IF      1
324    <macro>                            RKERN HPUTBUF               .RETURN IT                       DISK.157
326    <macro>                            STORE 0,DA@LOC,R!DI         .CLEAR CELL                      DISK.158
333    <macro>                          $ENDIF                                                         DISK.159
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                        DISK.160
333    020434  013022  102122           P = DINTWR, DL = 22       .Internal write?                     DISK.161
336    020431  012034  024000           DS = DI + DA@BMO@P            .DAL buffer memory for disk      R15472BD.2
341    077000 /002635                   R = DRETBIG               .Put it back                         DISK.163
                                                                                                        DISK.164
343    003000                           I = 1                                                          DISK.165
344    020431  012005  024410           (DI + DA@MES@P) = 0           .Clear message cell in DAL       R15472BD.3
       010000  034410
351    002000  000000                   I = 0                     .Lock again                          DISK.167
353    020431  024000                   DY = DI                   .Link                                DISK.168
355    <macro>                          RKERN    HONN             .Put Back Slave                      DISK.169
357    020431  012011  024410           P = DDSX, (DI + DA@SID@P) = 0 .Last DAL??                      R15472BD.4
       030410  106154
364    <macro>                          GET A,DA@FC,R!DJ              .Function code                   DISK.171
371    013014  102053                   P = DALFREE, A = DAF$WRBH     .If write behind                 DISK.172
                                                                                                        DISK.173
                                                                                                        DISK.174
                                 *  Send a final reply to the mainframe when all I/O to disk is done.   DISK.175
                                                                                                        DISK.176
373    <macro>                          GET A,DA@RC,R!DJ              .Last error status               DISK.177
400    <macro>                          $IF (A # DAR$OK)              .If error                        DISK.178
                                                                                        .>>>>> 0          $IF      1
402    020432  012030  024410             (R!DJ + DA@CYL@P) = (R!DH + DB@ECY@P) .Error cyl/head         DISK.179
       020000  012027  024411
       030411  034410
412    020432  012031  024410             (R!DJ + DA@SEC@P) = (R!DH + DB@ESC@P) .Error sec/offset       DISK.180
       020000  012030  024411
       030411  034410
422    <macro>                          $ENDIF                                                         DISK.181
                                                                                        .<<<<< 0          $ENDIF   1
422    020432  024000                   R!DS = R!DJ                   .DAL to send                     DISK.182
424    <macro>                          GET A,DA@IFC,R!DJ             .Internal function code          DISK.183
430    <macro>                          $IF (A = MC$CSTAT)            .If early reply already sent     DISK.184
                                                                                        .>>>>> 0          $IF      1
433    <macro>                            RKERN DDALGETA              .Get an MOS DAL                  DISK.185
```

```
435    <macro>                          $ELSE                                                          DISK.186
436    <macro>                            STORE MC$CSTAT,DA@IFC,R!DJ   .Set internal function code     DISK.187
443    <macro>                          $ENDIF                                                         DISK.188
                                                                                  .<<<<< 0                $ENDIF   1
443    <macro>                          RKERN DMGA                        .Send to MIOP                DISK.189
                                                                                                        DISK.190
                                        * Put Back the Master DAL Space                                 DISK.191
445                            DALFREE  *                                                               DISK.192
445    020432  012005  024410           (DJ + DA@MES@P) = 0               .Clear code cell           R15472BD.5
       010000  034410
452    020432  024000                   DY = DJ                     .Master Address                     DISK.194
454    <macro>                          RKERN    HONN               .Deallocate                         DISK.195
456    071247                           P = DDSX                    .Check for More                     DISK.196
                                                                                                        DISK.197
                                        *  Process disk request from within IOP - reactivate activity   DISK.198
                                                                                                        DISK.199
457                            DINTWR   *                                                               DISK.200
457    020432  012010  024000           ED = DJ + 10                .Push/pop cell in master            DISK.201
462    <macro>                          RKERN    EDQU               .Take from queue                    DISK.202
464    <macro>                          RKERN    EQCP               .Put on CP queue                    DISK.203
466    020431  024000                   DY = DI                     .DAL                                DISK.204
470    <macro>                          RKERN    HONN               .Release                            DISK.205
472    071263                           P = DDSX                                                        DISK.206
                                                                                                        DISK.207
                                                                                                        DISK.208
                                        ************************************************************    DISK.209
                                        * Read DAL from done queue to process                     *    DISK.210
                                        *   IOP does not have channel to target memory:            *    DISK.211
                                        *          Move data from local memory to MOS buffer - then *    DISK.212
                                        *          send message to IOP to get data moved to memory. *    DISK.213
                                        *   IOP has highspeed channel to target memory:            *    DISK.214
                                        *          Move data directly from local buffer to memory.  *    DISK.215
                                        ************************************************************    DISK.216
473    002000  000000         DSRD      I = 0                       .Lock for MOS Get                   DISK.217
475    010004  024433                   DK = 4                      .Code to tell BIOP to move data     DISK.218
477    020431  012013  024435           DM = DI + DA@DAT@P                .Data location indicator      R15472BD.6
502    020434  013021  102023           P = DINTRD, DL = 21         .Internal read                      DISK.220
505    020000  023451  102257           P = DBYPAS, R!%MYID = R!TMP       .If path in this IOP          DISK.221
510    030435  013002  102212           P = DSTEAL, (DM) = 2        .Rd. Ahead steal; Data in MOS??     DISK.222
513    020431  012034  024000           DS = DI + DA@BMO@P                .Place to save MOS address    R15472BD.7
516    077000 /002602                   R = DGETBIG                 .Get And Store                      DISK.224
520    <macro>                          $IF (R!DL # 21)                                                 DISK.225
```

```
                                                                                  .>>>>> 0          $IF      1
524   020431  024423                  R!%W1 = R!DI                .E-DAL                          DISK.226
526   070073                          P = DWTMOS                  .                               DISK.227
527    <macro>                      $ENDIF                                                        DISK.228
                                                                                  .<<<<< 0          $ENDIF   1
527                           DINTRD  *                                                             DISK.229
527    <macro>                      GET A,DA@DAT,R!DI           .DATA LOCATION                    DISK.230
533    <macro>                      $IF (A = 2)                 .IF IN MOS FROM READ-AHEAD        DISK.231
                                                                                  .>>>>> 0          $IF      1
536    <macro>                        RKERN HGETBUF             .ALLOCATE A LOCAL BUFFER          DISK.232
540    <macro>                        $IF (R!DY = 0)            .IF NONE                          DISK.233
                                                                                  .>>>>> 1          $IF      1
543    <macro>                          ADDRESS R!ED,DB@DNQ,R!DH .PUT BACK ON DONE QUEUE           DISK.234
546   020431  024000                    R!EH = R!DI               .                                 DISK.235
550    <macro>                          RKERN ENQU                .                                 DISK.236
552   030000  023431  107345            P = DDSX, (R!ED) # R!DI   .IF OTHERS WAITING                DISK.237
555   075000 /001246                    P = DDKA                  .CHECK FOR E-DALS TO START        DISK.238
557    <macro>                        $ENDIF                                                        DISK.239
                                                                                  .<<<<< 1          $ENDIF   1
557    <macro>                        PUT R!DY,DA@LOC,R!DI      .SAVE ADDRESS                     DISK.240
564    <macro>                        GET R!CB,DA@BM0,R!DI      .MOS ADDRESS                      DISK.241
571    <macro>                        GET R!CC,DA@BM1,R!DI      .                                 DISK.242
576   020000  024000                  R!CD = R!DY               .LOCAL ADDRESS                    DISK.243
600   014000 /001000  024000          R!CE = D'512              .LENGTH                           DISK.244
603   010002  024000                  R!CA = MIO$BMRI           .READ FROM MOS                    DISK.245
605    <macro>                        RKERN COMIO               .                                 DISK.246
607    <macro>                        ADDRESS R!DS,DA@BM0,R!DI  .RELEASE MOS BUFFER               DISK.247
612   077000 /002635                  R = DRETBIG               .                                 DISK.248
614    <macro>                      $ENDIF                                                        DISK.249
                                                                                  .<<<<< 0          $ENDIF   1
614    <macro>                      GET R!%W1,DA@MDL,R!DI       .M-DAL address                    DISK.250
621                           DWTMOS  *                                                             DISK.251
621    <macro>                      GET R!CB,DA@BM0,R!%W1       .MOS ADDRESS                      DISK.252
626    <macro>                      GET R!CC,DA@BM1,R!%W1       .                                 DISK.253
633    <macro>                      GET R!DY,DA@LOC,R!DI        .GET LOCAL ADDRESS                DISK.254
640   020000  024000                R!CD = R!DY                 .ADDRESS                          DISK.255
642   014000 /001000  024000        R!CE = D'512                .LENGTH                           DISK.256
645   010004  024000                R!CA = MIO$BMWI             .WRITE TO MOS                     DISK.257
647    <macro>                      RKERN COMIO                 .                                 DISK.258
651   002000  000000                I = 0                       .LOCK                             DISK.259
653    <macro>                      GET R!DY,DA@LOC,R!DI        .ADDRESS                          DISK.260
```

```
660    <macro>                          RKERN HPUTBUF                  .RETURN LOCAL BUFFER                  DISK.262
662    <macro>                          $IF (R!DL = 21)                .IF INTERNAL READ                     DISK.263
                                                                                          .>>>>> 0              $IF      1
666    <macro>                            GET R!%W1,DA@BM0,R!DI        .MOS BUFFER ADDRESS                   DISK.264
673    <macro>                            GET R!%W2,DA@BM1,R!DI        .FROM E-DAL                           DISK.265
700    <macro>                            $IF (R!%W1 # R!CB) OR (R!%W2 # R!CC) .IF DIFFERENT                DISK.266
                                                                                          .>>>>> 1              $IF      1
707    <macro>                              ADDRESS R!DS,DA@BM0,R!DI   .MOS BUFFER ADDRESS                   DISK.267
712    077000 /002635                       R = DRETBIG                .RELEASE                              DISK.268
714    <macro>                            $ENDIF                                                             DISK.269
                                                                                          .<<<<< 1              $ENDIF   1
714    077000 /003141                     R = CHNCHK                 .WAIT FOR TRANSFER                    DISK.270
716    071237                             P = DINTWR                 .DONE                                 DISK.271
717    <macro>                          $ENDIF                                                             DISK.272
                                                                                          .<<<<< 0              $ENDIF   1
717    077000 /003141                   R = CHNCHK                   .WAIT FOR TRANSFER                    DISK.273
721    020431                           A = R!DI                     .E-DAL ADDRESS                        DISK.274
722    <macro>                          RKERN BNCHK                  .VALIDATE BLOCK NUMBER                DISK.275
724                             DSTEAL  *                                                                  DISK.276
                                                                                                           DISK.277
724    020431  012001  024410           (DI + DA@IFC@P) = DK         .Tell BIOP what to do with data  R15472BD.8
       020433  034410
731    020431  024000                   R!DS = R!DI                                                        DISK.279
733    <macro>                          GET R!CB,DA@DL0,R!DS                                               DISK.280
740    <macro>                          GET R!CC,DA@DL1,R!DS                                               DISK.281
745    <macro>                          $IF (R!CB = 0) AND (R!CC = 0) .IF NO MOS DAL ALLOCATED             DISK.282
                                                                                          .>>>>> 0              $IF      1
753    <macro>                            RKERN  DDALGETA                                                  DISK.283
755    <macro>                          $ENDIF                                                             DISK.284
                                                                                          .<<<<< 0              $ENDIF   1
755    <macro>                          RKERN DSUM                   .Write DAL to MOS                     DISK.285
757    020451  024000                   R!EI = R!TMP                 .Target memory IOP                    DISK.286
761    <macro>                          RKERN EMSGIOP                .Send request                         DISK.287
763    020431  024432                   DJ = DI                  .DAL Address                              DISK.288
765    071320                           P = DALFREE              .Release DAL space                        DISK.289
                                                                                                           DISK.290
                                                                                                           DISK.291
                                                                                                           DISK.292
                                        ************************************************************       DISK.293
                                        *   Bypass the write to MOS if executing in BIOP          *       DISK.294
                                        *         Instead write to HSP channel if it exists.      *       DISK.295
                                        ************************************************************       DISK.296
```

```
                                                                                                DISK.297
 766                          DBYPAS   *                                                        DISK.298
 766    <macro>                        $IF       ((R!DM) = 2)        .If data is in Buffer Memory  DISK.299
                                                                                     .>>>>> 0        $IF      1
 773    <macro>                          GET     R!%W1,DA@IMO,R!DI   .Target memory address       DISK.300
1002    <macro>                          GET     R!%W2,DA@IM1,R!DI                                DISK.301
1007    <macro>                          GET     R!%W3,DA@BMO,R!DI   .Buffer Memory address       DISK.302
1014    <macro>                          GET     R!%W4,DA@BM1,R!DI                                DISK.303
1021    <macro>                          GET     R!%W5,DA@OFF,R!DI                                DISK.304
1027    <macro>                          DADD    R!%W3,R!%W4,R!%W5   .Offset for partial sector I/O DISK.305
1034    <macro>                          GET     R!%W5,DA@LN1,R!DI   .Get transfer length         DISK.306
1041    <macro>                          TRANSFER FS$OUT,R!TM,R!%W1,R!%W2,R!%W3,R!%W4,R!%W5       DISK.307
                                                                                                 DISK.308
1070    <macro>                        $ELSE                         .If data is in Local Memory  DISK.309
1071    <macro>                        GET A,DA@IOA,R!DI             .SEE IF TRANSFER ALREADY STARTED DISK.310
1075   103035                          P = DMVMAINO, A # 0           .SKIP IF SO                  DISK.311
1076   020431  024000                  R!DQ = R!DI                   .EDAL address          R16410AA.1
1100   077000 /003073                  R = DHSETUP                                                DISK.313
1102    <macro>                        $IF (R!TM = TM$CMEM)          .Target is Central memory    DISK.314
                                                                                     .>>>>> 1        $IF      1
1105   010010  024000                    R!CA = MIO$HSOI             .Function code               DISK.315
1107    <macro>                        $ELSEIF (R!TM = TM$BMR)       .If BMR                      DISK.316
1114   010004  024000                    R!CA = MIO$BMWI                                            DISK.317
1116    <macro>                        $ELSEIF (R!TM = TM$SSD)       .If SSD                      DISK.318
1123   010014  024000                    R!CA = MIO$SSOI             .Function code               DISK.319
1125    <macro>                        $ELSE                                                      DISK.320
1126    <macro>                          $PUNTIF CODE=PT$BADTM       .Illegal target memory       DISK.321
1130    <macro>                        $ENDIF                                                     DISK.322
                                                                                     .<<<<< 1        $ENDIF   1
1130    <macro>                        RKERN COMIO                   .                            DISK.323
1132                          DMVMAINO *                                                           DISK.324
1132   020431                          A = R!DI                      .E-DAL ADDRESS               DISK.325
1133    <macro>                        RKERN  BNCHK                  .VALIDATE BLOCK NUMBER       DISK.326
1135    <macro>                        GET R!%W1,DA@LOC,R!DI         .Local buffer address        DISK.327
1142    <macro>                        $IF (R!TM = TM$CMEM)          .If target is central memory DISK.328
                                                                                     .>>>>> 1        $IF      1
1145   020000  024424                    R!%W2 = R!HOLOCL            .Last local buffer in use    DISK.329
1147   020000  012001  054000            B = R!%HISP + 1             .Channel                     DISK.330
1152    <macro>                        $ELSEIF (R!TM = TM$BMR)       .Target is BMR               DISK.331
1157   020000  024424                    R!%W2 = R!BMLOCL                                           DISK.332
1161   010005  054000                    B = [MOS]                                                  DISK.333
1163    <macro>                        $ELSEIF (R!TM = TM$SSD)       .Target is SSD               DISK.334
```

```
1170   020000  024424                R!%W2 = R!SOLOCL                                          DISK.335
1172   020000  012001  054000        B = R!%SSD + 1                                            DISK.336
1175    <macro>                     $ELSE                                                     DISK.337
1176    <macro>                       $PUNTIF CODE=PT$BADTM       .Illegal target memory      DISK.338
1200    <macro>                     $ENDIF                                                    DISK.339
                                                                              .<<<<< 1              $ENDIF    1
1200    <macro>                     $IF (R!%W1 = R!%W2)         .If transfer still going      DISK.340
                                                                              .>>>>> 1              $IF       1
1204   077000 /003141                R = CHNCHK                  .Wait for completion        DISK.341
1206    <macro>                     $ENDIF                                                    DISK.342
                                                                              .<<<<< 1              $ENDIF    1
1206    <macro>                     STORE 0,DA@IOA,R!DI         .CLEAR TRANSFER FLAG          DISK.343
1213    <macro>                     $ENDIF                                                    DISK.344
                                                                              .<<<<< 0              $ENDIF    1
                                                                                                  DISK.345
1213   002000  000000               I = 0                                                     DISK.346
1215    <macro>                     GET       R!DY,DA@LOC,R!DI  .Local Memory buffer address  DISK.347
                                                                                                  DISK.348
1222    <macro>                     $IF       (A # 0)           .If a buffer allocated        DISK.349
                                                                              .>>>>> 0              $IF       1
1224    <macro>                       RKERN   HPUIBUF           .Release the buffer           DISK.350
1226   003000                         I = 1                     .UP PERISCOPE                 DISK.351
1227   012001                         A = A + 1                 .INTERRUPT APPROACHING...     DISK.352
1230   012001                         A = A + 1                 .                             DISK.353
1231   002000  000000                 I = 0                     .DIVE! DIVE!                  DISK.354
1233    <macro>                       STORE   0,DA@LOC,R!DI                                   DISK.355
1240    <macro>                     $ENDIF                                                    DISK.356
                                                                              .<<<<< 0              $ENDIF    1
                                                                                                  DISK.357
1240   075000 /000270               P = DNDAL                                                 DISK.358


                                    ************************************************************  DISK.361
                                    * Idle the Disk Demon.                                    *  DISK.362
                                    ************************************************************  DISK.363
1242   002000  000000        DIDL   I = 0                       .Lock                          DISK.364
```

```
1244    <macro>                 PKERN    ESWP          .Idle                                DISK.365
                                                                                            DISK.366
                                                                                            DISK.367
```

```
                                        ************************************************************   DISK.369
                                        * Done Queue Processed, Now Try to Stay a Disk            *   DISK.370
                                        *   Buffer Ahead of Disk I-O                               *   DISK.371
                                        ************************************************************   DISK.372
1246                            DDKA    *                                                              DISK.373
1246    <macro>                         $IF       (0 # (R!DH) & O'1000) .If ERRECK create bit set      DISK.374
                                                                                .>>>>> 0                $IF      1
1256    <macro>                           GET     R!%W1,DB@CHN,R!DH   .Get the channel number          DISK.375
1264    <macro>                           CREATE ERRECK,(R!%W1,R!DH),PR1=0                             DISK.376
1306    <macro>                           $PUNTIF (A # 0)             .If CANT CREATE                  DISK.377
1310    <macro>                           STORE   0,DBF@CR,R!DH       .Clear 'create' flag             DISK.378
1314    <macro>                           STORE   1,DBF@ER,R!DH       .Set error recovery flag         DISK.379
1322   075000 /000011                     P = DDSK                                                     DISK.380
1324    <macro>                         $ENDIF                                                         DISK.381
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                        DISK.382
1324   014000 /001331  024437           R!DR = DDKD                 .Return address                  DISK.383
                                                                                                        DISK.384
                                                                                                        DISK.385
1327   075000 /002431                   P = DRSC                  .Check if Disabled                 DISK.386
1331                            DDKD    *                                                              DISK.387
1331   020436  102032                   P = DDOD, R!DP = 0                                             DISK.388
                                                                                                        DISK.389
                                        * All out of local disk buffers, try to continue other tasks   DISK.390
1333                            DPAUSE  *                                                              DISK.391
1333   020000  024000                   DI = DH                     .DCB address                     DISK.392
1335    <macro>                         RkERN     DISCO             .Activate the disk demon         DISK.393
1337   014000 /000000  024410           DI = (DISKQ)                                                 DISK.394
       030410  024431
1344   020431  023000  127000           P = DDSK, DI # DH           .If another DCB is queued.       DISK.395
      /000011
1350    <macro>                         PAUSE 1,A1=0,A2=-1                                             DISK.396
1360   010010  024430                   R!LIM = DISK$LIM            .Reset limit                     DISK.397
1362   075000 /000011                   P = DDSK                                                     DISK.398
                                                                                                        DISK.399
                                        ************************************************************   DISK.400
                                        *  Not read ahead, see if more E-DAL's should be built     *   DISK.401
                                        ************************************************************   DISK.402
                                                                                                        DISK.403
1364   020000  012004  024410   DDOD    DI = (DH + 4)             .Population of Slave Chain         DISK.404
       030410  024431
1371   103010                           P = DDEB, A # 0           .Empty?                            DISK.405
```

```
1372   020000  012005  024410          P = DDKB, (DH + DD@MDL) # 0  .Yes, Master Empty?                DISK.406
       030410  103010
1377   075000 /000011                  P = DDSK                     .Yes, Nothing to Do                DISK.407
1401                            DDEB   *                                                               DISK.408
1401   020431  013002  100003          P = DDKK, DI > DALFEW        .If more than min, don't build     DISK.409
       102002  070003
1406                            DDKB   *                            .Build Executables                 DISK.410
1406    <macro>                        RKERN    DBUD                                                  DISK.411
                                                                                                       DISK.412
                                       * Executables (Slaves) Exist,  Toil                             DISK.413
1410   020000  012002  024410   DDKK   DQ = (DH + DD@EDL)           .First E-DAL                       DISK.414
       030410  024000
1415   126000 /000011                  P = DDSK, A = 0                   .IF NO E-DAL                 DISK.415
1417   030000  024432                  DJ = (DQ)                    .Second                            DISK.416
1421   030000  011001  024431          DI = (DH) & 1                .Reading Bit                       DISK.417
1424   127000 /000011                  P = DDSK, A # 0                   .If a read request           DISK.418


                                       ************************************************************     DISK.420
                                       *                                                          *     DISK.421
                                       * Channel Doing Write or perhaps not busy                  *     DISK.422
                                       *                                                          *     DISK.423
                                       ************************************************************     DISK.424
                                                                                                       DISK.425
1426    <macro>                        GET R!TM,DA@TM,R!DQ              .Target memory type           DISK.426
1434   020450  016000 /000000          R!TMP = (R!TM + TMCB)            .TM control table             DISK.427
       024410  030410  024451
1442    <macro>                        $PUNTIF (A = 0),CODE=PT$BPATH .If undefined path               DISK.428
1444    <macro>                        GET R!TMP,TM@IOP,R!TMP           .Get IOP with channel         DISK.429
1451    <macro>                        GET R!DP,DA@FC,R!DQ              .Function code                DISK.430
1457   013022  102121                  P = DWRTM, A = DAF$INTW          .If internal write            DISK.431
1461   020000  023451  103125          P = DDWROTH, R!%MYID # R!TMP  .If no path on this IOP          DISK.432
1464   030000  024423                  R!%W1 = (R!DH)                   .FLAGS                        DISK.433
1466    <macro>                        $IF (R!%W1 = 2)                  .IF WRITE IN PROGRESS         DISK.434
                                                                                   .>>>>> 0              $IF      1
1472   070116                            P = DDWROTH                   .GET NEXT SECTOR READY        DISK.435
1473    <macro>                        $ELSEIF (R!%W1 # 0) AND (R!%W1 # 42) .IF BUSY - NOT WRITING   DISK.436
1503   070250                            P = DDWBH                     .CHECK FOR WRITE-BEHINDS      DISK.437
1504    <macro>                        $ENDIF                                                        DISK.438
                                                                                   .<<<<< 0              $ENDIF   1
```

```
1504   020000  012013  024410          P = DINITIO, (DQ + DA@DAI@P) = 1 .If data in local memory      R15472BD.9
       030410  013001  102063
                                                                                                      R15472BD.10
1512   020000  012037  024436          DP = DQ + DA@LOC@P            .Local cell                      R15472BD.11
1515   030436  024000                  DY = (DP)                     .Get local buffer address            DISK.442
1517    <macro>                        $IF       (A = 0)             .If no buffer allocated              DISK.443
                                                                                   .>>>>> 0                 $IF      1
1521   014000 /000000  024410            P = DPAUSE, (EBNM) < 2                                           DISK.444
       030410  013002  104173
1527    <macro>                          RKERN   HGETBUF             .Allocate local memory buffer        DISK.445
1531   020000  103004                    P = DKAVAL, DY # 0          .Available?                          DISK.446
1533   010042  034000                    (DH) = 42                   .No, disable chnl until one free     DISK.447
1535   071202                            P = DPAUSE                  .Pause until buffer available        DISK.448
1536                           DKAVAL   *                                                                 DISK.449
                                                                                                          DISK.450
1536   020000  034436                    (DP) = DY                                                        DISK.451
1540    <macro>                        $ENDIF                                                             DISK.452
                                                                                   .<<<<< 0                 $ENDIF   1
1540   077000 /002665                  R = DGETHSP              .Get data from bipolar                    DISK.453
1542    <macro>                        $IF (R!TM = TM$CMEM)        .If target is central memory          DISK.454
                                                                                   .>>>>> 0                 $IF      1
1545   020000  054000                    B = R!%HISP                 .Channel                             DISK.455
1547    <macro>                        $ELSEIF (R!TM = TM$BMR)     .Target is BMR                       DISK.456
1554   010005  054000                    B = [MOS]                                                        DISK.457
1556    <macro>                        $ELSEIF (R!TM = TM$SSD)     .Target is SSD                       DISK.458
1563   020000  054000                    B = R!%SSD                                                       DISK.459
1565    <macro>                        $ELSE                                                              DISK.460
1566    <macro>                          $PUNTIF CODE=PT$BADTM     .Illegal target memory               DISK.461
1570    <macro>                        $ENDIF                                                             DISK.462
                                                                                   .<<<<< 0                 $ENDIF   1
1570   077000 /003141                  R = CHNCHK                                                         DISK.463
1572   002000  000000                  I = 0                                                              DISK.464
1574                           DINITIO *                                                                  DISK.465
1574   014000 /001610  024437          R!DR = DDWROTH              .Return address                      DISK.466
1577   075000 /002545                  P = DRDLOK               .Do I/O                                   DISK.467
1601                           DWRTM   *                                                                  DISK.468
1601   030000  103006                  P = DDWROTH, (DH) # 0       .If busy, don't start write          DISK.469
1603   014000 /001610  024437          R!DR = DDWROTH              .Return address                      DISK.470
1606   075000 /002460                  P = DSTRTW               .Start write                              DISK.471
1610                           DDWROTH *                                                                  DISK.472
1610   020432  102142                  P = DDWBH, R!DJ = 0         .IF NO 2ND E-DAL                     DISK.473
1612   020432  012026  024410          DK = (DJ + DA@FC@P) > 10    .Get command code from DAL       R15472BD.12
```

```
        030410  004010  024433
1620    020433  011001  024431          DI = DK & 1                 .Read bit set?                          DISK.475
1623    127000 /000011                  P = DDSK, A # 0                 .If a read request                  DISK.476
1625    020432  012013  024410          DP = (DJ + DA@DAT@P)            .Data location                  R15472BD.13
        030410  024436
1632    013001  102120                  P = DDWBH, A = 1                .IF DATA IN LOCAL                   DISK.478
1634    020436  013002  102033          P = DUNDER, DP = 2          .Data in MOS, move to local             DISK.479
1637     <macro>                        GET R!TM,DA@TM,R!DJ             .Target memory type                 DISK.480
1645    020450  016000 /000000          R!TMP = (R!TM + TMCB)           .TM control table                   DISK.481
        024410  030410  024451
1653     <macro>                        $PUNTIF (A = 0),CODE=PT$BPATH .If undefined path                    DISK.482
1655     <macro>                        GET R!TMP,TM@IOP,R!TMP          .Get IOP with channel               DISK.483
1662    023000  127000 /002426          P = DDCOM, A # R!%MYID          .If path not on this IOP            DISK.484
1665    020433  013012  126000          P = DDCOM, DK = 12              .If patterning disk                 DISK.485
       /002426
1671                            DUNDER  *                                                                   DISK.486
1671     <macro>                        GET R!DY,DA@LOC,R!DJ            .LOCAL BUFFER ADDRESS               DISK.487
1676     <macro>                        $IF       (A = 0)               .If no buffer allocated             DISK.488
                                                                                        .>>>>> 0                $IF      1
1700     <macro>                          RKERN   HGETBUF               .Allocate local memory buffer       DISK.489
1702    020000  106350                    P = DPAUSE, DY = 0            .IF 2ND BUFFER NOT AVAILABLE        DISK.490
1704     <macro>                          PUT R!DY,DA@LOC,R!DJ          .SAVE ADDRESS                       DISK.491
1711     <macro>                        $ENDIF                                                              DISK.492
                                                                                        .<<<<< 0                $ENDIF   1
1711    020451  023000  103015          P = DBIOPR, R!TMP # R!%MYID     .Get data from MOS                  DISK.493
1714    020433  013022  102012          P = DBIOPR, DK = 22         .Internal write?                        DISK.494
1717    020000  024436                  R!DP = R!DQ                     .SAVE ADDRESS OF 1ST E-DAL          DISK.495
1721    020432  024000                  R!DQ = R!DJ                     .SET 2ND E-DAL PARAMETER            DISK.496
1723    077000 /002665                  R = DGETHSP                     .MOVE DATA TO LOCAL MEMORY          DISK.497
1725    020436  024000                  R!DQ = R!DP                     .RESTORE 1ST E-DAL ADDRESS          DISK.498
1727    070024                          P = DDWBH                       .SEE IF WRITE BEHIND TO DO          DISK.499
1730                            DBIOPR  *                                                                   DISK.500
1730    020432  024000                  DY = DJ                     .DAL address                            DISK.501
1732     <macro>                        RKERN    DDIO               .Set Up MOS Params                      DISK.502
1734    010002  024000                  CA = MIO$BMRI                   .READ - NO WAIT                     DISK.503
1736     <macro>                        RKERN COMIO                     .                                   DISK.504
1740    077000 /003141                  R = CHNCHK                  .Wait for Completion                    DISK.505
1742    002000  000000                  I = 0                                                               DISK.506
1744    020432  012013  024410          (DJ + DA@DAT@P) = 1             .Data is in local memory        R15472BD.14
        010001  034410
1751    075000 /002426                  P = DDCOM                       .DONE                               DISK.508
                                                                                                            DISK.509
```

```
                                        *                                                                   DISK.510
                                        * WRITE BEHIND PROCESSING                                          DISK.511
                                        *                                                                   DISK.512
                                                                                                            DISK.513
1753                            DDWBH   *                                                                   DISK.514
1753   030000  011004  024410           P = DDCOM, 0 # (R!DH) & 4      .IF READ-AHEAD IN PROGRESS           DISK.515
       010000  023410  127000
      /002426
1762    <macro>                         GET R!%W1,DB@RAF,R!DH          .NUMBER ALREADY DONE                 DISK.516
1767    <macro>                         GET R!%W2,DB@RAK,R!DH          .SIZE OF READ-AHEAD AREA             DISK.517
1774    <macro>                         $IF (R!%W1 < R!%W2)                                                 DISK.518
                                                                                       .>>>>> 0             $IF      1
                                                                                                            DISK.519
                                          * LOCATE NEXT WRITE BEHIND SECTOR                                 DISK.520
                                                                                                            DISK.521
2001   010000  024425                     R!%W3 = 0                      .CLEAR NEXT WRITE BEHIND             DISK.522
2003    <macro>                           ADDRESS R!MDAL,DB@MDL,R!DH     .MDAL QUEUE                          DISK.523
2006    <macro>                           $UNTIL (R!%W3 # 0)             .UNTIL NEXT WRITE BEHIND FOUND       DISK.524
2010   030440  024440                       R!MDAL = (R!MDAL)            .NEXT                                DISK.525
2012   102414                               P = DDCOM, A = 0             .IF NONE                             DISK.526
2013    <macro>                             GET R!%W4,DA@FC,R!MDAL       .Function code                       DISK.527
2021    <macro>                             $IF (R!%W4 # DAF$WRBH),AND,(R!%W4 # DAF$WBFR)                    DISK.528
                                                                                       .>>>>> 1             $IF      1
2031   070375                               P = DDCOM                    .Don't write behind                  DISK.529
2032    <macro>                             $ENDIF                                                           DISK.530
                                                                                       .<<<<< 1             $ENDIF   1
2032    <macro>                             GET R!TM,DA@TM,R!MDAL        .Target memory type                  DISK.531
2040   013001  102365                       P = DDCOM, A = TM$BMR        .If BMR                              DISK.532
2042   020450  016000 /000000               R!TMP = (R!TM + TMCB)        .TM control table                    DISK.533
       024410  030410  024451
2050    <macro>                             $PUNTIF (A = 0),CODE=PT$BPATH .If undefined path                  DISK.534
2052    <macro>                             GET R!TMP,TM@IOP,R!TMP       .Get IOP with channel                DISK.535
2057   023000  103346                       P = DDCOM, A # R!%MYID       .If path not on this IOP             DISK.536
2061    <macro>                             GET R!LAST,DA@IOT,R!MDAL     .TOTAL SECTORS FOR REQUEST           DISK.537
2066    <macro>                             GET R!NXT,DA@WBH,R!MDAL      .NEXT WRITE BEHIND                   DISK.538
2073    <macro>                             $IF (R!NXT < R!LAST)         .IF ONE TO DO                        DISK.539
                                                                                       .>>>>> 1             $IF      1
2077   010001  024425                       R!%W3 = 1                    .SHOW WRITE-BEHIND FOUND             DISK.540
2101    <macro>                             $ENDIF                                                           DISK.541
                                                                                       .<<<<< 1             $ENDIF   1
2101    <macro>                           $ENDTIL                                                          DISK.542
                                                                                                            DISK.543
```

```
                                        * ALLOCATE MOS                                                  DISK.544
                                                                                                        DISK.545
2102    <macro>                         GET R!PTR,DB@IN,R!DH          .POINTER TO NEXT ENTRY            DISK.546
2107    <macro>                         ADDRESS R!DS,RA@BMO,R!PTR     .ADDRESS OF MOS BUFFER POINTER    DISK.547
2112    072470                          R = DGETBIG                   .ALLOCATE MOS IF NOT THERE        DISK.548
2113    020000  024443                  R!MSU = R!CB                  .SAVE ADDRESS                     DISK.549
2115    020000  024444                  R!MSL = R!CC                  .                                 DISK.550
                                                                                                        DISK.551
                                        * COMPUTE CPU ADDRESS                                           DISK.552
                                                                                                        DISK.553
2117    <macro>                         GET R!%W1,DA@UNS,R!MDAL       .NUMBER OF UNITS IN GROUP         DISK.554
2125    010000  024424                  R!%W2 = 0                     .CLEAR                            DISK.555
2127    <macro>                         $UNTIL (R!%W1 = 0)            .MULTIPLY BY SEQUENCE NUMBER      DISK.556
2131    020424  022442  024424            R!%W2 = R!%W2 + R!NXT       .                                 DISK.557
2134    027423                            R!%W1 = R!%W1 - 1           .                                 DISK.558
2135    <macro>                         $ENDTIL                                                         DISK.559
2136    010000  024423                  R!%W1 = 0                     .TRANSLATE TO WORDS               DISK.560
2140    <macro>                         LSHIFT R!%W1,R!%W2,D'9        .                                 DISK.561
2157    <macro>                         GET R!BPU,DA@TMO,R!MDAL       .Target memory address            DISK.562
2166    <macro>                         GET R!BPL,DA@TM1,R!MDAL                                         DISK.563
2173    <macro>                         DADD R!BPU,R!BPL,R!%W1,R!%W2 .ADD OFFSET                        DISK.564
                                                                                                        DISK.565
2203    <macro>                         TRANSFER FS$IN,R!TM,R!BPU,R!BPL,R!MSU,R!MSL,D'512               DISK.566
2233    002000  000000                  I = 0                         .DISABLE INTERRUPTS               DISK.567
                                                                                                        DISK.568
                                        * IF E-DALS HAVE BEEN MADE BUT DATA HAS NOT                     DISK.569
                                        * BEEN MOVED, TRANSFER WRITE-BEHIND DIRECTLY TO                 DISK.570
                                        * E-DAL. ELSE, SAVE IN READ-AHEAD AREA.                         DISK.571
                                                                                                        DISK.572
2235    010000  024423                  R!%W1 = 0                     .SET DATA MOVED                   DISK.573
2237    <macro>                         GET A,DA@DAT,R!DQ             .1ST E-DAL                        DISK.574
2243    <macro>                         $IF (A = 0)                   .IF DATA NOT MOVED                DISK.575
                                                                                          .>>>>> 1          $IF      1
2245    020000  024423                    R!%W1 = R!DQ                .INDICATE                         DISK.576
2247    <macro>                         $ELSEIF (R!DJ # 0)            .IF 2ND E-DAL EXISTS              DISK.577
2253    <macro>                           GET A,DA@DAT,R!DJ           .CHECK FOR DATA MOVED             DISK.578
2257    103003  020432  024423            R!%W1 = R!DJ, A = 0         .IF NOT                           DISK.579
2262    <macro>                         $ENDIF                                                          DISK.580
                                                                                          .<<<<< 1          $ENDIF   1
                                                                                                        DISK.581
                                        * DIRECT WRITE-BEHIND TO APPROPRIATE LOCATION                   DISK.582
                                                                                                        DISK.583
```

```
2262   <macro>                  $IF (R!%W1 # 0)              .IF TO E-DAL                          DISK.584
                                                                                  .>>>>> 1              $IF      1
2265   <macro>                    PUT R!MSU,DA@BM0,R!%W1     .MOS ADDRESS                          DISK.585
2272   <macro>                    PUT R!MSL,DA@BM1,R!%W1     .                                     DISK.586
2277   <macro>                    STORE 2,DA@DAT,R!%W1       .SET DATA LOCATION                    DISK.587
2304   <macro>                    CLEAR START=R!PTR,COUNT=RA@LEN@P .CLEAR READ-AHEAD ENTRY         DISK.588
2317   <macro>                  $ELSE                                                               DISK.589
                                                                                                     DISK.590
                                * ENTER IN READ-AHEAD ENTRY UNTIL E-DAL MADE                         DISK.591
                                                                                                     DISK.592
2320   <macro>                    STORE 2,RA@DAT,R!PTR       .DATA LOCATION                        DISK.593
2322   020440  024000             R!DD = R!MDAL              .SET M-DAL ADDRESS                    DISK.594
2324   020442  024000             R!DU = R!NXT               .SECTOR TO FIND                       DISK.595
2326   <macro>                    RKERN DFRSTRA              .GET CYLINDER/HEAD/SECTOR             DISK.596
2330   <macro>                    ADDRESS R!%W1,RA@CYL,R!PTR  .ENTER IN ENTRY                      DISK.597
2333   020000  034423             (R!%W1) = R!DS             .                                     DISK.598
2335   020000  004011  024000     R!DT = R!DT > D'9          .ADJUST SECTOR                        DISK.599
2340   <macro>                    PUT R!DT,RA@SEC,R!PTR      .ENTER                                DISK.600
2353   <macro>                    FLDADD 1,DB@RAF,R!DH       .COUNT DONE                           DISK.601
2357   <macro>                    GET R!%W1,DB@UNF,R!DH      .LAST                                 DISK.602
2364   <macro>                    $IF (R!PTR = R!%W1)        .IF TIME TO WRAP                      DISK.603
                                                                                  .>>>>> 2              $IF      1
2370   <macro>                      GET R!PTR,DB@INF,R!DH    .SET TO FIRST                         DISK.604
2375   <macro>                    $ELSE                                                               DISK.605
2376   010006  025447               R!PTR = R!PTR + RA@LEN@P  .ADVANCE TO NEXT                      DISK.606
2400   <macro>                    $ENDIF                                                              DISK.607
                                                                                  .<<<<< 2              $ENDIF   1
2400   <macro>                    PUT R!PTR,DB@IN,R!DH       .SAVE                                 DISK.608
2405   <macro>                  $ENDIF                                                              DISK.609
                                                                                  .<<<<< 1              $ENDIF   1
2405   026442                   R!NXT = R!NXT + 1            .UPDATE WRITE-BEHIND SECTOR           DISK.610
2406   <macro>                  PUT R!NXT,DA@WBH,R!MDAL      .SAVE                                 DISK.611
                                                                                                     DISK.612
                                * SEND INITIAL REPLY IF LAST SECTOR MOVED TO MOS                     DISK.613
                                                                                                     DISK.614
2413   <macro>                  $IF (R!NXT = R!LAST)         .IF LAST TRANSFERRED                  DISK.615
                                                                                  .>>>>> 1              $IF      1
2417   077000 /003170             R = WBREPLY                .SEND REPLY TO CPU                    DISK.616
2421   <macro>                  $ELSE                                                               DISK.617
2422   020000  024000             R!DT = R!DH                .QUEUE FOR NEXT WRITE BEHIND          DISK.618
2424   <macro>                    RKERN DISCO                .                                     DISK.619
2426   <macro>                  $ENDIF                                                              DISK.620
```

```
                                                                                .<<<<< 1          $ENDIF    1
2426    <macro>                          $ENDIF                                                  DISK.621
                                                                                .<<<<< 0          $ENDIF    1
2426                             DDCOM   *                                                       DISK.622
2426   014000 /000011  024437            R!DR = DDSK                 .Return address             DISK.623
                                                                                                  DISK.624
                                         ****   ****   Fall through to DRSC    ****   ****       DISK.625
```

```
                                        ************************************************************   DISK.627
                                        *                                                          *   DISK.628
                                        * Subroutine: DRSC                                         *   DISK.629
                                        *   Reactivates Channel Waiting for local Buffer or for    *   DISK.630
                                        *   Data to be retrieved from Cray main memory             *   DISK.631
                                        * Entry:  DG = Channel                                     *   DISK.632
                                        *         DH = DCB                                         *   DISK.633
                                        *         DR = Return Point                                *   DISK.634
                                        * Exit:   DP = 0, Ok                                       *   DISK.635
                                        *         DP # 0, No Buffer Available, No Action           *   DISK.636
                                        *                                                          *   DISK.637
                                        ************************************************************   DISK.638
2431  030000  011040  024436  DRSC      DP = (DH) & 40            .Status of Channel                 DISK.639
2434  102142                            P = DREXIT, A = 0              .If not disabled              DISK.640
2435  020000  012002  024410            DQ = (DH + DD@EDL)        .First DAL                         DISK.641
      030410  024000
2442  030000  013042  103021            P = DRSCBY, (DH) # 42     .Disabled because data late?       DISK.642
2445  020000  012013  024410            DP = (DQ + DA@DAT@P)           .Data location            R15472BD.15
      030410  024436
2452  020436  013001  102073            P = DRDT, DP = 1          .In local memory, write it to disk DISK.644
2455  020436  013002  103112            P = DISH, DP # 2          .MOS. Read in before write         DISK.645
2460                          DSTRTW    *                                                            DISK.646
2460  020000  012037  024410            P = DRSWRT, (DQ + DA@LOC@P) # 0 .Disk buffer allocated?   R15472BD.16
      030410  103012
2465                          DRSCBY    *                                                            DISK.648
2465   <macro>                          RKERN    HGETBUF          .Yes, Get A Local Buffer           DISK.649
2467  020000  102104                    P = DNOL, DY = 0          .None Available                    DISK.650
2471  020000  012037  024410            (DQ + DA@LOC@P) = DY                                      R15472BD.17
      020000  034410
                                                                                                     DISK.652
2476                          DRSWRT    *                                                            DISK.653
2476  020000  012026  024410            R!%W1 = (DQ + DA@FC@P)                                    R15472BD.18
      030410  024423
2503  020423  011400  024436            DP = R!%W1 & 400                                             DISK.655
2506  103041                            P = DRDT, A # 0           .Read?                             DISK.656
                                                                                                     DISK.657
2507   <macro>                          $IF (0'12 = R!%W1 > D'8)                                     DISK.658
                                                                                  .>>>>> 0              $IF      1
2516   <macro>                            RKERN PAIRN                 .Pattern local buffer         DISK.659
2520  070025                              P = DRDLOK                                                 DISK.660
2521   <macro>                          $ENDIF                                                       DISK.661
                                                                                  .<<<<< 0              $ENDIF   1
```

```
                                                                                                  DISK.662
2521   020000  024000                    DY = DQ                    .First DAL address             DISK.663
2523    <macro>                          RKERN    DDIO              .Set Up MOS I/O Params         DISK.664
2525   020000  012031  024410            DA = (DQ + DA@SEC@P) & 777 < 2 .Adjust local buffer offset  R15472BD.19
       030410  011777  005002
       024000
2534   020000  022000  024000            CD = CD + DA                                              DISK.666
2537   010002  024000                    CA = MIOSBMRI                  .READ - NO WAIT            DISK.667
2541    <macro>                          RKERN COMIO                    .                          DISK.668
2543   077000 /003141                    R = CHNCHK                 .Wait for Completion           DISK.669
                                                                                                  DISK.670
                                         *  Start the disk I/O                                     DISK.671
2545                            DRDLOK   *                                                         DISK.672
2545   002000  000000                    I = 0                      .Lock                          DISK.673
2547   020000  024000           DRDT     DA = DH                    .DCB                           DISK.674
2551   020000  024000                    DC = DQ                    .DAL to Do I/O                 DISK.675
2553   020000  012030  024410            DB = (DC + DA@HED@P)           .Cyl and head              R15472BD.20
       030410  024000
2560   030000  011720  034000            (DH) = (DH) & 720          .Clear 'disabled' bit          DISK.677
2563    <macro>                          RKERN    DIOH              .Move Head?                    DISK.678
2565   020000  103003                    P = DISH, DS # 0           .If Zero, No Movement          DISK.679
2567    <macro>                          RKERN    DIOA              .Start I/O                     DISK.680
2571   010000  024436           DISH     DP = 0                     .Ok Status                     DISK.681
2573   070003                            P = DREXIT                                                DISK.682
                                                                                                  DISK.683
2574   010111  024436           DNOL     DP = 111                   .Flag for Failure              DISK.684
                                                                                                  DISK.685
2576                            DREXIT   *                                                         DISK.686
2576   020000  022437  024423            R!%W1 = R!%B + R!DR            .Calulate return address   DISK.687
2601   074423                            P = R!%W1                                                 DISK.688
                                                                                                  DISK.689
```

```
                                 ********************************************************************   DISK.691
                                 *                                                                  *   DISK.692
                                 *        DGETBIG   Allocate a disk buffer in buffer memory.        *   DISK.693
                                 *                                                                  *   DISK.694
                                 *        Entry:    DS - Pointer to address parcels                 *   DISK.695
                                 *                                                                  *   DISK.696
                                 ********************************************************************   DISK.697
                                                                                                        DISK.698
2602                             DGETBIG  *                                                             DISK.699
2602   030000  024000                     CB = (DS)                                                     DISK.700
2604   020000  012001  024410             CC = (DS + 1)                                                 DISK.701
       030410  024000
                                                                                                        DISK.702
2611    <macro>                          $IF       (CB = 0),AND,(CC = 0) .If buffer not yet allocated   DISK.703
                                                                                  .>>>>> 0                $IF      1
2617   010001  024000                       R!CA = 1                     .Allocate 1 buffer            DISK.704
2621    <macro>                             RKERN   GETDISK              .Allocate the buffer          DISK.705
2623    <macro>                             $PUNTIF (A # 0),CODE=PT$NOMOS .If no buffer available      DISK.706
2625   020000  034000                       (DS) = CB                    .Save the buffer address      DISK.707
2627   020000  012001  024410               (DS + 1) = CC                                              DISK.708
       020000  034410
2634    <macro>                          $ENDIF                                                        DISK.709
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                        DISK.710
2634   001000                           EXIT                                                           DISK.711


                                 ********************************************************************   DISK.713
                                 *                                                                  *   DISK.714
                                 *        DRETBIG   Release a buffer memory disk buffer.            *   DISK.715
                                 *                                                                  *   DISK.716
                                 *        Entry:    DS - Pointer to address parcels                 *   DISK.717
                                 *                                                                  *   DISK.718
                                 ********************************************************************   DISK.719
                                                                                                        DISK.720
2635                             DRETBIG  *                                                             DISK.721
2635   030000  024000                     CB = (DS)                                                     DISK.722
2637   020000  012001  024410             CC = (DS + 1)                                                 DISK.723
       030410  024000
                                                                                                        DISK.724
```

```
2644    <macro>                          $IF       (CB # 0),OR,(CC # 0) .If aA buffer is allocated          DISK.725
                                                                                          .>>>>> 0            $IF       1
2651    010001  024000                     R!CA = 1                     .Deallocate 1 buffer                DISK.726
2653    <macro>                            RKERN   PUTDISK              .Release the buffer                 DISK.727
2655    010000  034000                     (DS) = 0                     .Save the buffer address            DISK.728
2657    020000  012001  024410             (DS + 1) = 0                                                     DISK.729
        010000  034410
2664    <macro>                          $ENDIF                                                             DISK.730
                                                                                          .<<<<< 0            $ENDIF    1
                                                                                                             DISK.731
2664    001000                           EXIT                                                               DISK.732


                                         ************************************************************        DISK.734
                                         *  DGETHSP: Subroutine to get data from Cray instead of MOS  *     DISK.735
                                         *          ENTRY:  DQ = E-DAL being processed                *     DISK.736
                                         ************************************************************        DISK.737
                                                                                                             DISK.738
2665                             DGETHSP *                                                                  DISK.739
2665    <macro>                          GET A,DA@DAT,R!DQ            .DATA LOCATION                      DISK.740
2671    <macro>                          $IF (A = 2)                  .IF IN MOS                          DISK.741
                                                                                          .>>>>> 0            $IF       1
2674    010002  024000                     R!CA = MIO$BMRI              .BUFFER MEMORY READ                 DISK.742
2676    <macro>                            GET R!CB,DA@BM0,R!DQ         .ADDRESS                            DISK.743
2703    <macro>                            GET R!CC,DA@BM1,R!DQ         .                                   DISK.744
2710    <macro>                            GET R!CD,DA@LOC,R!DQ         .LOCAL BUFFER                       DISK.745
2715    <macro>                            GET R!CE,DA@LN1,R!DQ         .LENGTH                             DISK.746
2722    <macro>                            GET R!%W1,DA@OFF,R!DQ        .OFFSET                             DISK.747
2730    020423  005002  022000             R!CD = R!%W1 < 2 + R!CD      .ADD TO BUFFER ADDRESS              DISK.748
        024000
2734    <macro>                            RKERN COMIO                  .ISSUE                              DISK.749
2736    <macro>                          $ELSE                                                              DISK.750
2740    <macro>                            GET A,DB@RAF,R!DH            .WRITE-BEHINDS DONE                 DISK.751
2744    <macro>                            $PUNTIF (A # 0)              .IF ANY SKIPPED                     DISK.752
2746    077000 /003073                     R = DHSETUP                  .SET UP I/O PARAMETERS              DISK.753
2750    <macro>                            $IF (R!TM = TM$CMEM)         .Target is Central memory           DISK.754
                                                                                          .>>>>> 1            $IF       1
2753    010006  024000                       R!CA = MIO$HSII            .Function code                      DISK.755
2755    <macro>                            $ELSEIF (R!TM = TM$BMR)      .If BMR                             DISK.756
2762    010002  024000                       R!CA = MIO$BMRI                                                 DISK.757
```

```
2764    <macro>                    $ELSEIF (R!TM = TM$SSD)      .If SSD                                DISK.758
2771    010012  024000               R!CA = MIO$SSIF            .Function code                         DISK.759
2773    <macro>                    $ELSE                                                               DISK.760
2774    <macro>                      $PUNTIF CODE=PT$BADTM      .Illegal target memory                 DISK.761
2776    <macro>                    $ENDIF                                                              DISK.762
                                                                                .<<<<< 1                   $ENDIF    1
2776    <macro>                    RKERN COMIO                  .                                      DISK.763
3000    <macro>                    GET R!MDAL,DA@MDL,R!DQ       .M-DAL address                         DISK.764
3005    <macro>                    GET A,DA@SID,R!DQ            .CHECK FOR LAST E-DAL                  DISK.765
3011    <macro>                    $IF (A # 0)                  .IF SO,                                DISK.766
                                                                                .>>>>> 1                   $IF       1
3013    <macro>                      $IF (R!TM = TM$CMEM)       .If target is central memory           DISK.767
                                                                                .>>>>> 2                   $IF       1
3016    020000  054000               B = R!%HISP                .Channel                               DISK.768
3020    <macro>                      $ELSEIF (R!TM = TM$BMR)    .Target is BMR                         DISK.769
3025    010005  054000               B = [MOS]                                                         DISK.770
3027    <macro>                      $ELSEIF (R!TM = TM$SSD)    .Target is SSD                         DISK.771
3034    020000  054000               B = R!%SSD                                                        DISK.772
3036    <macro>                      $ELSE                                                             DISK.773
3037    <macro>                        $PUNTIF CODE=PT$BADTM    .Illegal target memory                 DISK.774
3041    <macro>                      $ENDIF                                                            DISK.775
                                                                                .<<<<< 2                   $ENDIF    1
3041    072100                       R = CHNCHK                   .                                      DISK.776
3042    072126                       R = WBREPLY                  .SEND EARLY RESPONSE                   DISK.777
3043    <macro>                    $ENDIF                                                              DISK.778
                                                                                .<<<<< 1                   $ENDIF    1
3043    <macro>                    GET R!%W1,DA@WBH,R!MDAL      .NEXT WRITE-BEHIND SECTOR              DISK.779
3050    <macro>                    GET R!%W2,DA@SEQ,R!DQ        .SECTOR JUST MOVED                     DISK.780
3055    <macro>                    $IF (R!%W1 = R!%W2)          .IF SAME                               DISK.781
                                                                                .>>>>> 1                   $IF       1
3061    <macro>                      FLDADD 1,DA@WBH,R!MDAL     .ADVANCE WRITE-BEHIND                  DISK.782
3065    <macro>                    $ENDIF                                                              DISK.783
                                                                                .<<<<< 1                   $ENDIF    1
3065    <macro>                  $ENDIF                                                                DISK.784
                                                                                .<<<<< 0                   $ENDIF    1
3065    <macro>                  STORE 1,DA@DAT,R!DQ            .SHOW DATA IN LOCAL                    DISK.785
3072    001000                   EXIT                                                                  DISK.786
                                                                                                       DISK.787
                                 ***************************************************************       DISK.788
                                 * DHSETUP:                                                    *       DISK.789
                                 *     Sets up HSP channel I/O from DAL                        *       DISK.790
                                 ***************************************************************       DISK.791
```

```
                                                                                                  DISK.792
3073                          DHSETUP  *                                                          DISK.793
3073    <macro>                        GET R!CB,DA@IM0,R!DQ          .Target memory address      R16410CB.1
3102    <macro>                        GET R!CC,DA@IM1,R!DQ                                      R16410CB.2
3107    020000  012031  024410         CE = (DQ + DA@SEC@P) & 777 < 2 .Local buffer offset       R15472BD.21
        030410  011777  005002
        024000
3116    020000  012037  024410         CD = (DQ + DA@LOC@P) + CE     .Local data address         R15472BD.22
        030410  022000  024000
3124    020000  012033  024410         CE = (DQ + DA@LN1@P)          .Size                       R15472BD.23
        030410  024000
3131    <macro>                        $PUNTIF  (CE > 1000)     .Check illegal size              DISK.799
3140    001000                         EXIT                                                      DISK.800
                                                                                                  DISK.801
                                       ************************************************************DISK.802
                                       *  MOS and HSP error check and recovery subroutine       *DISK.803
                                       ************************************************************DISK.804
                                                                                                  DISK.805
3141                          CHNCHK   *                                                          DISK.806
3141    <macro>                        $UNTIL (IOB # BZ)             .Until channel complete     DISK.807
3143    <macro>                          $IF ((EBNM) # 0)            .If more buffers available  DISK.808
                                                                                  .>>>>> 0          $IF      1
3151    003000                             I = 1                     .Unlock                     DISK.809
3152    <macro>                          $ENDIF                                                  DISK.810
                                                                                  .<<<<< 0          $ENDIF   1
3152    <macro>                          RKERN CHNWTDN               .Wait                       DISK.811
3154    002000  000000                   I = 0                       .Lock                       DISK.812
3156    <macro>                          $IF (IOB = DN) AND (IOB = BZ) .If error                 DISK.813
                                                                                  .>>>>> 0          $IF      1
3164    <macro>                            RKERN CEREC               .All done                   DISK.814
3166    <macro>                          $ENDIF                                                  DISK.815
                                                                                  .<<<<< 0          $ENDIF   1
3166    <macro>                        $ENDTIL                                                   DISK.816
3167    001000                         EXIT                                                      DISK.817
                                                                                                  DISK.818
                                       ************************************************************DISK.819
                                       *       SEND EARY REPLY FOR WRITE REQUEST - DATA MOVED    *DISK.820
                                       ************************************************************DISK.821
                                                                                                  DISK.822
3170                          WBREPLY  *                                                          DISK.823
3170    <macro>                        GET R!%W1,DA@FC,R!MDAL        .Function code              DISK.824
3176    <macro>                        $IF (R!%W1 = DAF$WRBH),OR,(R!%W1 = DAF$WBFR) .Write behind DISK.825
```

```
                                                                                .>>>>> 0          $IF      1
3205   003009                          I = 1                                                     DISK.826
3206    <macro>                        STORE MC$CSIAT,DA@IFC,R!MDAL .Internal function code      DISK.827
3213   014000 /000000  024423          R!%W1 = IMDAL                .Use immediate DAL           DISK.828
3216    <macro>                        COPY R!MDAL,R!%W1,DA@@LH+DA@@LF .Copy MDAL                DISK.829
                                                                                                  DISK.830
                                     *  Include error info if any errors ocurred.                 DISK.831
                                                                                                  DISK.832
3235    <macro>                        GET A,DA@RC,R!MDAL           .Last status                 DISK.833
3242    <macro>                        $IF (A # DAR$OK)             .If error                    DISK.834
                                                                                .>>>>> 1          $IF      1
3244   020423  012030  024410            (R!%W1 + DA@CYL@P) = (R!DH + DB@ECY@P) .Error cyl/head  DISK.835
       020000  012027  024411
       030411  034410
3254   020423  012031  024410            (R!%W1 + DA@SEC@P) = (R!DH + DB@ESC@P) .Error sec/offset DISK.836
       020000  012030  024411
       030411  034410
3264    <macro>                          STORE 0,DA@ERC,R!MDAL      .Clear errcount for next reply DISK.837
3271    <macro>                        $ENDIF                                                    DISK.838
                                                                                .<<<<< 1          $ENDIF   1
3271   002009  000000                  I = 0                                                     DISK.839
3273   020423  024000                  R!DS = R!%W1                 .DAL to send                 DISK.840
3275    <macro>                        RKERN DMGA                   .Send to MIOP                DISK.841
3277    <macro>                      $ENDIF                                                      DISK.842
                                                                                .<<<<< 0          $ENDIF   1
3277   001000                        EXIT                                                        DISK.843
                                                                                                  DISK.844
                                                                                                  DISK.845
                                     END                                                          DISK.846
```

0X	%B		1: 4 D	4: 9 B	4:30 B	5:10 B	6:12 B	7:11 B	8:41 B	9:18 B	
			9:29 B	10: 9 B	10:12 B	10:16 B	11: 4 B	11:18 B	12: 9 B	12:30 B	
			14:13 B	14:20 B	14:30 B	14:34 B	15: 3 B	15:13 B	15:16 B	16:19 B	
			16:31 B	16:35 B	16:39 B	17: 3 B	17:13 B	17:14 B	17:29 B	17:37 B	
			17:41 B	18: 7 B	18:12 B	20:38 B	23:10 B	23:29	25:34 B	25:37 B	
0X	%EX		11:12 D	11:12	14: 9 D	14: 9	14:32 D	14:32	19:23 D	19:23	
10	%GBLNUM	$APTEXT	2:15 F	2:15							
400	%GBLREG	$APTEXT	2:15								
0X	%HISP		3:10 D	11:38	16:22	26:14					
0	%KERNEL	$APTEXT	4:17 D	4:17 B	5:14 D	5:14 B	5:18 D	5:18 B	6:14 D	6:14 B	
			6:40 D	6:40 B	7: 4 D	7: 4 B	7:18 D	7:18 B	7:42 D	7:42 B	
			8: 5 D	8: 5 B	8:12 D	8:12 B	8:19 D	8:19 B	8:20 D	8:20 B	
			8:22 D	8:22 B	9:11 D	9:11 B	9:16 D	9:16 B	9:27 D	9:27 B	
			9:40 D	9:40 B	10: 1 D	10: 1 B	10:18 D	10:18 B	10:28 D	10:28 B	
			10:31 D	10:31 B	10:33 D	10:33 B	11:30 D	11:30 B	11:33 D	11:33 B	
			12:21 D	12:21 B	13: 1 D	13: 1 B	14:27 D	14:27 B	15: 8 D	15: 8 B	
			16:10 D	16:10 B	17:20 D	17:20 B	17:34 D	17:34 B	17:36 D	17:36 B	
			20:14 D	20:14 B	20:41 D	20:41 B	22:26 D	22:26 B	22:39 D	22:39 B	
			23: 3 D	23: 3 B	23: 9 D	23: 9 B	23:20 D	23:20 B	23:22 D	23:22 B	
			24:17 D	24:17 B	25: 4 D	25: 4 B	25:33 D	25:33 B	26: 7 D	26: 7 B	
			27:26 D	27:26 B	27:30 D	27:30 B	28:23 D	28:23 B			
0X	%MYID		3:11 D	8:38	15:34	17:13	17:25	18:33			
0X	%PUNT		6:21 D	6:21	11:27 D	11:27	12: 4 D	12: 4	14:10 D	14:10	
			15:30 D	15:30	16:28 D	16:28	17:11 D	17:11	18:31 D	18:31	
			24:18 D	24:18	25:36 D	25:36	26: 4 D	26: 4	26:20 D	26:20	
			27:12 D	27:12							
0X	%SSD		3:12 D	12: 2	16:26	26:18					
177777	AD$NODEM	$APTEXT	14: 9								
0X	BMLOCL		3:13 D	11:40							
0X	BNCHK		10:18 D	10:18	11:33 D	11:33					
0X	CA		3:14 D	5:13	6:39	9:26	9:39	11:21	11:23	11:25	
			17:35	23: 8	24:16	25: 3	25:25	25:40	25:42	26: 2	
0X	CB		3:15 D	6:31	6:38	9:22	9:34	10: 7	10:24	10:27	
			19: 6	24:10	24:15	24:19	24:39	25: 2	25:26	27: 3	
0X	CC		3:16 D	6:34	6:38	9:23	9:35	10: 7	10:25	10:27	
			19: 7	24:12	24:15	24:20	24:41	25: 2	25:27	27: 4	
0X	CD		3:17 D	5:11	5:17	9:24	9:37	23: 7	25:28	25:31	
			27: 9								
0X	CE		3:18 D	5:12	9:25	9:38	25:29	27: 6	27: 9	27:11	
			27:12								
0X	CEREC		27:30 D	27:30							
0X	CHNWTDN		27:26 D	27:26							

0X	COMIO		5:14 D	5:14	9:27 D	9:27	9:40 D	9:40	11:30 D	11:30		
			17:36 D	17:36	23: 9 D	23: 9	25:33 D	25:33	26: 7 D	26: 7		
0X	D$DISK		3:19 D	4:16								
0X	DA		3:20 D	23: 5	23: 7	23:15						
30	DA@@LE	$APTEXT	28: 5									
10	DA@@LH	$APTEXT	28: 5									
20	DA@BM0@N	$APTEXT	9:22	9:22 F	9:34	9:34 F	10: 4	10: 4 F	11: 7	11: 7 F		
			20: 3 F	25:26	25:26 F							
34	DA@BM0@P	$APTEXT	7:10	8:40	9:22 F	9:22	9:28	9:34 F	9:34	10: 4 F		
			10: 4	10: 8	11: 7 F	11: 7	20: 3 F	20: 3	25:26 F	25:26		
0	DA@BM0@S	$APTEXT	9:22	9:22 F	9:34	9:34 F	10: 4	10: 4 F	11: 7	11: 7 F		
			20: 3 F	25:26	25:26 F							
20	DA@BM1@N	$APTEXT	9:23	9:23 F	9:35	9:35 F	10: 5	10: 5 F	11: 8	11: 8 F		
			20: 4 F	25:27	25:27 F							
35	DA@BM1@P	$APTEXT	9:23 F	9:23	9:35 F	9:35	10: 5 F	10: 5	11: 8 F	11: 8		
			20: 4 F	20: 4	25:27 F	25:27						
0	DA@BM1@S	$APTEXT	9:23	9:23 F	9:35	9:35 F	10: 5	10: 5 F	11: 8	11: 8 F		
			20: 4 F	25:27	25:27 F							
30	DA@CYL@P	$APTEXT	7:30	28:12								
20	DA@DAT@N	$APTEXT	9: 8	9: 8 F	19:31	19:31 F	19:36	19:36 F	20: 5 F	25:22		
			25:22 F	26:36 F								
13	DA@DAT@P	$APTEXT	8:36	9: 8 F	9: 8	16: 1	17: 4	17:39	19:31 F	19:31		
			19:36 F	19:36	20: 5 F	20: 5	22:18	25:22 F	25:22	26:36 F		
			26:36									
0	DA@DAT@S	$APTEXT	9: 8	9: 8 F	19:31	19:31 F	19:36	19:36 F	20: 5 F	25:22		
			25:22 F	26:36 F								
20	DA@DL0@N	$APTEXT	10:24	10:24 F								
2	DA@DL0@P	$APTEXT	6:30	10:24 F	10:24							
0	DA@DL0@S	$APTEXT	10:24	10:24 F								
20	DA@DL1@N	$APTEXT	10:25	10:25 F								
3	DA@DL1@P	$APTEXT	10:25 F	10:25								
0	DA@DL1@S	$APTEXT	10:25	10:25 F								
20	DA@ERC@N	$APTEXT	28:18 F									
17	DA@ERC@P	$APTEXT	28:18 F	28:18								
0	DA@ERC@S	$APTEXT	28:18 F									
10	DA@FC@N	$APTEXT	6:17	6:17 F	7:21	7:21 F	15:32	15:32 F	18:21	18:21 F		
			27:41	27:41 F								
26	DA@FC@P	$APTEXT	6:17 F	6:17	7:21 F	7:21	15:32 F	15:32	16:42	18:21 F		
			18:21	22:32	27:41 F	27:41						
0	DA@FC@S	$APTEXT	6:17	6:17 F	7:21	7:21 F	15:32	15:32 F	18:21	18:21 F		
			27:41	27:41 F								
30	DA@HED@P	$APTEXT	23:17									

20	DA@IFC@N	$APTEXT	7:39	7:39 F	8: 2 F	28: 3 F							
1	DA@IFC@P	$APTEXT	7:39 F	7:39	8: 2 F	8: 2	10:21	28: 3 F	28: 3				
0	DA@IFC@S	$APTEXT	7:39	7:39 F	8: 2 F	28: 3 F							
20	DA@IOA@N	$APTEXT	11:15	11:15 F	12:12 F								
23	DA@IOA@P	$APTEXT	11:15 F	11:15	12:12 F	12:12							
0	DA@IOA@S	$APTEXT	11:15	11:15 F	12:12 F								
20	DA@LN1@N	$APTEXT	11:11	11:11 F	25:29	25:29 F							
33	DA@LN1@P	$APTEXT	11:11 F	11:11	25:29 F	25:29	27:10						
0	DA@LN1@S	$APTEXT	11:11	11:11 F	25:29	25:29 F							
20	DA@LOC@N	$APTEXT	7: 1	7: 1 F	7: 5 F	9:21 F	9:36	9:36 F	9:42	9:42 F			
			11:34	11:34 F	12:17	12:17 F	12:26 F	17:17	17:17 F	17:22 F			
			25:28	25:28 F									
37	DA@LOC@P	$APTEXT	7: 1 F	7: 1	7: 5 F	7: 5	9:21 F	9:21	9:36 F	9:36			
			9:42 F	9:42	11:34 F	11:34	12:17 F	12:17	12:26 F	12:26			
			16: 4	17:17 F	17:17	17:22 F	17:22	22:23	22:28	25:28 F			
			25:28	27: 8									
0	DA@LOC@S	$APTEXT	7: 1	7: 1 F	7: 5 F	9:21 F	9:36	9:36 F	9:42	9:42 F			
			11:34	11:34 F	12:17	12:17 F	12:26 F	17:17	17:17 F	17:22 F			
			25:28	25:28 F									
20	DA@MDL@N	$APTEXT	6:16	6:16 F	9:32	9:32 F	26: 8	26: 8 F					
17	DA@MDL@P	$APTEXT	6:16 F	6:16	9:32 F	9:32	26: 8 F	26: 8					
0	DA@MDL@S	$APTEXT	6:16	6:16 F	9:32	9:32 F	26: 8	26: 8 F					
5	DA@MES@P	$APTEXT	7:14	8: 9									
777	DA@OFF@M	$APTEXT	11: 9	25:30									
11	DA@OFF@N	$APTEXT	11: 9	25:30									
31	DA@OFF@P	$APTEXT	11: 9 F	11: 9	25:30 F	25:30							
7	DA@OFF@S	$APTEXT	11: 9	11: 9 F	25:30	25:30 F							
377	DA@RC@M	$APTEXT	7:27	28: 9									
10	DA@RC@N	$APTEXT	7:27	28: 9									
26	DA@RC@P	$APTEXT	7:27 F	7:27	28: 9 F	28: 9							
10	DA@RC@S	$APTEXT	7:27	7:27 F	28: 9	28: 9 F							
31	DA@SEC@P	$APTEXT	7:33	23: 4	27: 5	28:15							
20	DA@SEQ@N	$APTEXT	26:28	26:28 F									
12	DA@SEQ@P	$APTEXT	26:28 F	26:28									
0	DA@SEQ@S	$APTEXT	26:28	26:28 F									
20	DA@SID@N	$APTEXT	26: 9	26: 9 F									
11	DA@SID@P	$APTEXT	7:19	26: 9 F	26: 9								
0	DA@SID@S	$APTEXT	26: 9	26: 9 F									
37777	DA@TMO@M	$APTEXT	11: 5	19:19	27: 3								
16	DA@TMO@N	$APTEXT	11: 5	19:19	27: 3								
24	DA@TMO@P	$APTEXT	11: 5 F	11: 5	19:19 F	19:19	27: 3 F	27: 3					
2	DA@TMO@S	$APTEXT	11: 5	11: 5 F	19:19	19:19 F	27: 3	27: 3 F					

20	DA@IM1@N	$APIEXT	11: 6	11: 6 F	19:20	19:20 F	27: 4	27: 4 F		
25	DA@IM1@P	$APIEXT	11: 6 F	11: 6	19:20 F	19:20	27: 4 F	27: 4		
0	DA@IM1@S	$APIEXT	11: 6	11: 6 F	19:20	19:20 F	27: 4	27: 4 F		
2	DA@IM@N	$APIEXT	6:18	6:18 F	15:27	15:27 F	17: 8	17: 8 F	18:27	18:27 F
24	DA@IM@P	$APIEXT	6:18 F	6:18	15:27 F	15:27	17: 8 F	17: 8	18:27 F	18:27
0	DA@IM@S	$APIEXT	6:18	6:18 F	15:27	15:27 F	17: 8	17: 8 F	18:27	18:27 F
20	DA@IOI@N	$APIEXT	18:34	18:34 F						
14	DA@IOI@P	$APIEXT	18:34 F	18:34						
0	DA@IOI@S	$APIEXT	18:34	18:34 F						
4	DA@UNS@N	$APIEXT	19:11	19:11 F						
20	DA@UNS@P	$APIEXT	19:11 F	19:11						
0	DA@UNS@S	$APIEXT	19:11	19:11 F						
20	DA@WBH@N	$APIEXT	18:35	18:35 F	20:32 F	26:27	26:27 F	26:31 F		
37	DA@WBH@P	$APIEXT	18:35 F	18:35	20:32 F	20:32	26:27 F	26:27	26:31 F	26:31
0	DA@WBH@S	$APIEXT	18:35	18:35 F	20:32 F	26:27	26:27 F			
22	DAI$IN1W	$APIEXT	15:33							
6	DAI$WBFR	$APIEXT	18:23	28: 1						
14	DAI$WRBH	$APIEXT	7:22	18:23	28: 1					
2	DALEEW	$APIEXT	15: 5							
0	DARSOK	$APIEXT	7:29	28:11						
0X	DB		3:21 D	23:18						
777	DB@CHN@M	$APIEXT	14: 8							
11	DB@CHN@N	$APIEXT	14: 8							
7	DB@CHN@P	$APIEXT	14: 8 F	14: 8						
7	DB@CHN@S	$APIEXT	14: 8	14: 8 F						
14	DB@DNQ@P	$APIEXT	9:14							
27	DB@ECY@P	$APIEXT	7:31	28:13						
30	DB@ESC@P	$APIEXT	7:34	28:16						
20	DB@IN@N	$APIEXT	4:41	4:41 F	19: 3	19: 3 F	20:28 F			
44	DB@IN@P	$APIEXT	4:41 F	4:41	19: 3 F	19: 3	20:28 F	20:28		
0	DB@IN@S	$APIEXT	4:41	4:41 F	19: 3	19: 3 F	20:28 F			
20	DB@INI@N	$APIEXT	4:39	4:39 F	20:23	20:23 F				
12	DB@INI@P	$APIEXT	4:39 F	4:39	20:23 F	20:23				
0	DB@INI@S	$APIEXT	4:39	4:39 F	20:23	20:23 F				
5	DB@MDL@P	$APIEXT	18:17							
20	DB@OUI@N	$APIEXT	4:42	4:42 F						
45	DB@OUI@P	$APIEXT	4:42 F	4:42						
0	DB@OUI@S	$APIEXT	4:42	4:42 F						
20	DB@RAI@N	$APIEXT	18: 9	18: 9 F	20:19 F	25:35	25:35 F			
11	DB@RAI@P	$APIEXT	18: 9 F	18: 9	20:19 F	20:19	25:35 F	25:35		
0	DB@RAI@S	$APIEXT	18: 9	18: 9 F	25:35	25:35 F				
20	DB@RAE@N	$APIEXT	5: 1	5: 1 F	18:10	18:10 F				

10	DB@RAK@P	$APTEXT	5: 1 F	5: 1	18:10 F	18:10					
0	DB@RAK@S	$APTEXT	5: 1	5: 1 F	18:10	18:10 F					
20	DB@UNI@N	$APTEXT	4:40	4:40 F	20:20	20:20 F					
46	DB@UNI@P	$APTEXT	4:40 F	4:40	20:20 F	20:20					
0	DB@UNI@S	$APTEXT	4:40	4:40 F	20:20	20:20 F					
1	DBF@CR@M	$APTEXT	14:11 F								
1	DBF@CR@N	$APTEXT	14:11 F								
0	DBF@CR@P	$APTEXT	14:11 F								
6	DBF@CR@S	$APTEXT	14:11 F								
176777	DBF@CR@X	$APTEXT	14:11								
1	DBF@ER@M	$APTEXT	14:12 F								
1	DBF@ER@N	$APTEXT	14:12 F	14:12							
0	DBF@ER@P	$APTEXT	14:12 F								
13	DBF@ER@S	$APTEXT	14:12 F	14:12							
0X	DBUD		15: 8 D	15: 8							
0X	DC		3:22 D	23:16	23:17						
0X	DD		3:35 D	20:12							
20	DD@DEM@N	$APTEXT	4:20	4:20 F	4:21 F						
37	DD@DEM@P	$APTEXT	4:20 F	4:20	4:21 F	4:21					
0	DD@DEM@S	$APTEXT	4:20	4:20 F	4:21 F						
14	DD@DNQ	$APTEXT	6:10	6:13							
2	DD@EDI	$APTEXT	15:11	22:15							
5	DD@MDL	$APTEXT	15: 1								
0X	DDALGETA		7:42 D	7:42	10:28 D	10:28					
0X	DDIO		17:34 D	17:34	23: 3 D	23: 3					
0X	DFRSTRA		20:14 D	20:14							
0X	DGIVEUP		4:17 D	4:17							
0X	DH		3:23 D	4: 8	4:20	4:21	4:27	4:39	4:40	4:41	
			4:42	5: 1	6:10	6:13	7:30	7:33	9:14	14: 7	
			14: 8	14: 9	14:11	14:12	14:26	14:30	14:40	15: 1	
			15:11	15:15	15:35	16:12	16:37	18: 6	18: 9	18:10	
			18:17	19: 3	20:19	20:20	20:23	20:28	20:40	22:13	
			22:15	22:17	23:15	23:19	25:35	28:12	28:15		
0X	DIOA		23:22 D	23:22							
0X	DIOQ		23:20 D	23:20							
0X	DI[illegible]		14:27 D	14:27	20:41 D	20:41					
10	DI[illegible]LIM	$APTEXT	4: 4	4:18	14:33						
0X	DISKQ		3:24 D	4: 7	4:22	14:28					
0X	DISKQT		3:25 D	4:24							
0X	DMGA		8: 5 D	8: 5	28:23 D	28:23					
0X	DQ		3:26 D	11:17	15:12	15:14	15:27	15:32	16: 1	16: 4	
			17:27	17:28	17:30	19:31	19:34	22:16	22:18	22:23	

			22:28	22:32	23: 2	23: 4	23:16	25:22	25:26	25:27
			25:28	25:29	25:30	26: 8	26: 9	26:28	26:36	27: 3
			27: 4	27: 5	27: 8	27:10				
0X	DS		3:27 D	5: 9	6:30	6:31	6:32	6:33	6:34	6:35
			7:10	7:38	8:40	9:28	10: 8	10:23	10:24	10:25
			19: 4	20:16	23:21	24:10	24:11	24:19	24:20	24:39
			24:40	25: 5	25: 6	28:22				
0X	DSUM		10:31 D	10:31						
0X	DI		3:28 D	14:26	20:17	20:18	20:40			
0X	DU		3:36 D	20:13						
0X	DY		3:29 D	5:17	7: 1	7:17	8:11	8:21	9:13	9:21
			9:24	9:36	9:37	9:42	12:17	16: 5	16:11	16:16
			17:17	17:21	17:22	17:33	22:27	22:28	23: 2	
0X	EBNM		3:37 D	16: 8	27:22					
0X	ED		3:30 D	6:13	8:18	9:14	9:17			
0X	EDQU		6:14 D	6:14	8:19 D	8:19				
0X	EH		3:31 D	4:20	4:22	9:15				
0X	EI		3:32 D	10:32						
0X	EMSGTOP		10:33 D	10:33						
0X	ENQU		9:16 D	9:16						
0X	EQCP		8:20 D	8:20						
0X	ESWP		13: 1 D	13: 1						
55	ESCREATE	$APTEXT	14: 9							
7	ESPAUSE	$APTEXT	14:32							
45	ESTRANSI	$APTEXT	11:12	19:23						
0	ESSIN	$APTEXT	19:23							
1	ESSOUT	$APTEXT	11:12							
0	ESSWAIT	$APTEXT	11:12	19:23						
0X	GETDISK		24:17 D	24:17						
0X	HGETBUF		9:11 D	9:11	16:10 D	16:10	17:20 D	17:20	22:26 D	22:26
0X	HOLOCL		3:33 D	11:37						
0X	HONN		7:18 D	7:18	8:12 D	8:12	8:22 D	8:22		
0X	HPUTBUF		5:18 D	5:18	7: 4 D	7: 4	10: 1 D	10: 1	12:21 D	12:21
0X	IMDAL		3:34 D	28: 4						
5	MCSCSTAT	$APTEXT	7:41	8: 2	28: 3					
2	MIO$BMRT	$APTEXT	9:26	17:35	23: 8	25:25	25:42			
4	MIO$BMWT	$APTEXT	5:13	9:39	11:23					
6	MIO$HSIT	$APTEXT	25:40							
10	MIO$HSOT	$APTEXT	11:21							
12	MIO$SSIT	$APTEXT	26: 2							
14	MIO$SSOT	$APTEXT	11:25							
5	MOS	$APTEXT	11:41	16:24	26:16					

0X	O$DISK		1: 4 D	1: 4							
0X	O$ERRECK		14: 9 D	14: 9							
17	OV@NUM@N	$APTEXT	1: 4								
1	OV@TYP@N	$APTEXT	1: 4								
0X	PATRN		22:39 D	22:39							
113	PT$BADTM	$APTEXT	11:27	12: 4	16:28	26: 4	26:20				
112	PT$BPATH	$APTEXT	6:21	15:30	17:11	18:31					
0	PT$HUH	$APTEXT	14:10	25:36	27:12						
15	PT$NOMOS	$APTEXT	24:18								
0X	PUIDAL		6:40 D	6:40							
0X	PUIDISK		25: 4 D	25: 4							
3	RA@BMO@P	$APTEXT	5: 9	19: 4							
1	RA@CYL@P	$APTEXT	20:15								
20	RA@DAI@N	$APTEXT	5: 6	5: 6 F	5:15 F	20:11 F					
0	RA@DAI@P	$APTEXT	5: 6 F	5:15 F	20:11 F						
0	RA@DAI@S	$APTEXT	5: 6	5: 6 F	5:15 F	20:11 F					
6	RA@LEN@P	$APTEXT	5:25	20: 6	20:25						
20	RA@LOC@N	$APTEXT	5:11	5:11 F	5:16 F						
5	RA@LOC@P	$APTEXT	5:11 F	5:11	5:16 F	5:16					
0	RA@LOC@S	$APTEXT	5:11	5:11 F	5:16 F						
177	RA@SEC@M	$APTEXT	20:18								
7	RA@SEC@N	$APTEXT	20:18 F	20:18							
2	RA@SEC@P	$APTEXT	20:18 F	20:18							
0	RA@SEC@S	$APTEXT	20:18 F	20:18							
11	SM@FST@N	$APTEXT	1: 4								
7	SM@NUM@N	$APTEXT	1: 4								
0X	SOLOCL		3:38 D	12: 1							
1	TM$BMR	$APTEXT	11:22	11:39	16:23	18:28	25:41	26:15			
0	TM$CMEM	$APTEXT	11:20	11:36	16:21	25:39	26:13				
2	TM$SSD	$APTEXT	11:24	11:42	16:25	26: 1	26:17				
20	TM@IOP@N	$APTEXT	6:22	6:22 F	15:31	15:31 F	17:12	17:12 F	18:32	18:32 F	
5	TM@IOP@P	$APTEXT	6:22 F	6:22	15:31 F	15:31	17:12 F	17:12	18:32 F	18:32	
0	TM@IOP@S	$APTEXT	6:22	6:22 F	15:31	15:31 F	17:12	17:12 F	18:32	18:32 F	
0X	TMCB		3:39 D	6:19	15:28	17: 9	18:29				

SYMBOL QUALIFIER IS DISK

0	SCONST	4:21 D	4:21 F	5:15 D	5:15 F	5:15	5:16 D	5:16 F	7: 5 D		
		7: 5 F	8: 2 D	8: 2 F	8: 2	12:12 D	12:12 F	12:26 D	12:26 F		
		14:11 D	14:11 F	14:12 D	14:12 F	14:12	20: 5 D	20: 5 F	20: 5		
		20:11 D	20:11 F	20:11	26:36 D	26:36 F	26:36	28: 3 D	28: 3 F		
		28: 3	28:18 D	28:18 F							

452	$REGORG	2:15 D	2:15 F	2:16	2:16 D	2:17	2:17 D	2:18	2:18 D
		2:19	2:19 D	2:20	2:20 D	2:20 F	2:21	2:21 D	2:22
		2:22 D	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D	2:26
		2:26 D	2:26 F	2:27	2:27 D	2:28	2:28 D	2:29	2:29 D
		2:30	2:30 D	2:31	2:31 D	2:31 F	2:32	2:32 D	2:33
		2:33 D	2:34	2:34 D	2:35	2:35 D	2:36	2:36 D	2:37
		2:37 D	2:38	2:38 D	2:39	2:39 D	2:40	2:40 D	2:41
		2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2	3: 2 D	3: 3
		3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6	3: 6 D	3: 7
		3: 7 D	3: 7 F	11:12	11:12 F	14: 9	14:32	19:23	19:23 F
0	%NP	1: 4	2:31 D						
430	%P	1: 4	2:31 D	2:31	11:12 F	11:12	19:23 F	19:23	
410	%S1	2:15	2:16 D	4: 7	4:20	4:21	4:22	4:23	4:24
		4:25	4:39	4:40	4:41	4:42	5: 1	5:11	5:16
		6:10	6:16	6:17	6:18	6:19	6:20	6:22	7: 1
		7: 5	7:14	7:15	7:19	7:21	7:27	7:30	7:32
		7:33	7:35	7:39	8: 2	8: 9	8:10	9: 8	9:21
		9:22	9:23	9:32	9:34	9:35	9:36	9:42	10: 4
		10: 5	10:21	10:22	10:24	10:25	11: 5	11: 6	11: 7
		11: 8	11: 9	11:11	11:15	11:34	12:12	12:17	12:26
		14: 7	14: 8	14:28	14:40	15: 1	15:11	15:27	15:28
		15:29	15:31	15:32	16: 1	16: 8	16:42	17: 4	17: 8
		17: 9	17:10	17:12	17:17	17:22	17:39	17:40	18: 6
		18: 7	18: 9	18:10	18:21	18:27	18:29	18:30	18:32
		18:34	18:35	19: 3	19:11	19:19	19:20	19:31	19:36
		20: 3	20: 4	20: 5	20:18	20:19	20:20	20:23	20:28
		20:32	22:15	22:18	22:23	22:28	22:29	22:32	22:38
		23: 4	23:17	24:11	24:20	24:21	24:40	25: 6	25: 7
		25:22	25:26	25:27	25:28	25:29	25:30	25:35	26: 8
		26: 9	26:27	26:28	26:31	26:36	27: 3	27: 4	27: 5
		27: 8	27:10	27:22	27:41	28: 3	28: 9	28:12	28:14
		28:15	28:17	28:18					
411	%S2	2:15	2:17 D	7:31	7:34	28:13	28:16		
412	%S3	2:15	2:18 D						
413	%S4	2:15	2:19 D						
414	%S5	2:15	2:20 D						
415	%T1	2:21 D	19:18	20: 6	28: 5				
416	%T2	2:22 D	20: 6	28: 5					
417	%T3	2:23 D	28: 5						
420	%T4	2:24 D							
421	%T5	2:25 D							
422	%T6	2:26 D							

423	%W1	2:27 D	4:39	5:23	9: 2	9:32	9:34	9:35	10: 4
		10: 7	11: 5	11:12	11:34	12: 8	14: 8	14: 9	15:35
		15:37	15:39	18: 9	18:12	19:11	19:13	19:15	19:17
		19:18	19:21	19:30	19:34	19:37	20: 2	20: 3	20: 4
		20: 5	20:15	20:16	20:20	20:22	22:33	22:34	22:38
		23:29	23:30	25:30	25:31	26:27	26:30	27:41	28: 1
		28: 4	28: 5	28:12	28:15	28:22			
424	%W2	2:28 D	4:40	5:22	10: 5	10: 7	11: 6	11:12	11:37
		11:40	12: 1	12: 8	18:10	18:12	19:12	19:14	19:18
		19:21	26:28	26:30					
425	%W3	2:29 D	4:41	11: 7	11:10	11:12	18:16	18:18	18:38
426	%W4	2:30 D	4:42	5: 6	5: 9	5:11	5:15	5:16	5:22
		5:23	5:25	11: 8	11:10	11:12	18:21	18:23	
427	%W5	2:31 D	11: 9	11:10	11:11	11:12			
446	BPL	3: 4 D	19:20	19:21	19:23				
445	BPU	3: 3 D	19:19	19:21	19:23				
3141	CHNCHK	10:12	10:16	12: 9	16:31	17:37	23:10	26:23	27:19 D
445	DALFREE	7:22	8: 8 D	10:35					
1730	DBIOPR	17:25	17:26	17:32 D					
766	DBYPAS	8:38	11: 2 D						
2426	DDCOM	17:13	17:14	17:41	18: 7	18:20	18:24	18:28	18:33
		21: 4 D							
1401	DDEB	14:42	15: 4 D						
1246	DDKA	6:12	9:18	14: 5 D					
1406	DDKB	15: 2	15: 7 D						
1331	DDKD	14:17	14:21 D						
1410	DDKK	15: 6	15:11 D						
1364	DDOD	14:22	14:40 D						
11	DDSK	4: 5 D	4:28	14:13	14:30	14:34	15: 3	15:13	15:16
		17: 3	21: 5						
73	DDSQ	4:29	4:31 D						
207	DDSX	6:10 D	7:20	8:13	8:23	9:17			
1753	DDWBH	15:40	16:41	17: 6	17:31	18: 5 D			
1610	DDWROTH	15:34	15:38	16:34	16:37	16:38	16:40 D		
2602	DGETBIG	5:10	8:41	19: 5	24: 9 D				
2665	DGETHSP	16:19	17:29	25:21 D					
3073	DHSETUP	11:18	25:37	27: 2 D					
431	DI	2:33 D	5: 1	5: 2	5:28	6:11	6:16	6:17	6:18
		6:30	7: 1	7: 5	7:10	7:14	7:17	7:19	8:21
		8:36	8:40	9: 2	9: 8	9:15	9:17	9:21	9:22
		9:23	9:28	9:32	9:36	9:42	10: 4	10: 5	10: 8
		10:17	10:21	10:23	10:34	11: 5	11: 6	11: 7	11: 8

		11: 9	11:11	11:15	11:17	11:32	11:34	12:12	12:17
		12:26	14:29	14:30	14:41	15: 5	15:15	17: 2	
1242	DIDL	4: 9	12:42 D						
1574	DINITIO	16: 2	16:33 D						
527	DINIRD	8:37	9: 7 D						
457	DINIWR	7: 9	8:17 D	10:13					
2571	DISH	22:21	23:21	23:23 D					
432	DJ	2:34 D	6:16	7:21	7:27	7:30	7:33	7:38	7:39
		8: 2	8: 9	8:11	8:18	10:34	15:14	16:41	16:42
		17: 4	17: 8	17:17	17:22	17:28	17:33	17:39	19:35
		19:36	19:37						
433	DK	2:35 D	6:23	8:35	10:21	17: 1	17: 2	17:14	17:26
1536	DKAVAL	16:11	16:14 D						
36	DKDONEI	4:15	4:19 D						
434	DL	2:36 D	6:17	6:23	7: 9	8:37	9: 1	10: 3	
435	DM	2:37 D	8:36	8:39	11: 4				
1132	DMVMAINO	11:16	11:31 D						
270	DNDAI	6:29 D	12:30						
2574	DNOI	22:27	23:26 D						
436	DP	2:38 D	14:22	15:32	16: 4	16: 5	16:16	17: 5	17: 7
		17:27	17:30	22:13	22:19	22:20	22:21	22:34	23:23
		23:26							
1333	DPAUSE	14:25 D	16: 9	16:13	17:21				
437	DR	2:39 D	4:29	14:17	16:34	16:38	21: 5	23:29	
2545	DRDLOK	16:35	22:40	23:13 D					
2547	DRDI	22:20	22:35	23:15 D					
2635	DRETBIG	7:11	9:29	10: 9	24:38 D				
2576	DREXIT	22:14	23:24	23:28 D					
2431	DRSC	4:30	14:20	22:13 D					
2465	DRSCBY	22:17	22:25 D						
2476	DRSWRT	22:24	22:31 D						
473	DSRD	6:24	8:34 D						
724	DSTEAL	8:39	10:19 D						
2460	DSTRIW	16:39	22:22 D						
1671	DUNDER	17: 7	17:16 D						
1601	DWRTM	15:33	16:36 D						
621	DWTMOS	9: 3	9:33 D						
441	LAST	2:41 D	18:34	18:37	20:37				
430	LIM	2:32 D	4: 4	4:13	4:18	14:33			
440	MDAI	2:40 D	18:17	18:19	18:21	18:27	18:34	18:35	19:11
		19:19	19:20	20:12	20:32	26: 8	26:27	26:31	27:41
		28: 3	28: 5	28: 9	28:18				

444	MSL	3: 2 D	19: 7	19:23	20: 4				
443	MSU	3: 1 D	19: 6	19:23	20: 3				
442	NXT	2:42 D	18:35	18:37	19:14	20:13	20:31	20:32	20:37
447	PTR	3: 5 D	19: 3	19: 4	20: 6	20:11	20:15	20:18	20:22
		20:23	20:25	20:28					
450	TM	3: 6 D	6:18	6:19	11:12	11:20	11:22	11:24	11:36
		11:39	11:42	15:27	15:28	16:21	16:23	16:25	17: 8
		17: 9	18:27	18:29	19:23	25:39	25:41	26: 1	26:13
		26:15	26:17						
451	TMP	3: 7 D	6:20	6:22	8:38	10:32	15:29	15:31	15:34
		17:10	17:12	17:25	18:30	18:32			
3170	WBREPLY	20:38	26:24	27:40 D					

```
0     <macro>                    OVERLAY   ICOM                                                    ICOM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
                    ***********************************************************************   ICOM.5
                    *                                                                     *   ICOM.6
                    *         ICOM      Handles inter-processor communications for        *   ICOM.7
                    *                   DD39 and DD49 disks.                              *   ICOM.8
                    *                                                                     *   ICOM.9
                    ***********************************************************************   ICOM.10
                                                                                              ICOM.11
6     <macro>                   REGDEFS   ,,(QUE,NUMDONE,DAL,MSG,IOP,FNC,CYL,HED,SEC,DCB,DONE,L   ICOM.12
                    ,LNG,CHAN,LEN0,LEN1,SC0,SC1,SC2,SC3,OFF,DIR,TOTAL,IFC,MOSU,MOSL,UNIT,CPD   ICOM.13
                    ,,TPC,SPT,DIA,SUB,TM,TMP)                                                 ICOM.14
410                 %S1       EQUALS    $REGORG                                                 $REG    3
411                 %S2       EQUALS    $REGORG                                                 $REG    3
412                 %S3       EQUALS    $REGORG                                                 $REG    3
413                 %S4       EQUALS    $REGORG                                                 $REG    3
414                 %S5       EQUALS    $REGORG                                                 $REG    3
415                 %T1       EQUALS    $REGORG                                                 $REG    3
416                 %T2       EQUALS    $REGORG                                                 $REG    3
417                 %T3       EQUALS    $REGORG                                                 $REG    3
420                 %T4       EQUALS    $REGORG                                                 $REG    3
421                 %T5       EQUALS    $REGORG                                                 $REG    3
422                 %T6       EQUALS    $REGORG                                                 $REG    3
423                 %W1       EQUALS    $REGORG                                                 $REG    3
424                 %W2       EQUALS    $REGORG                                                 $REG    3
425                 %W3       EQUALS    $REGORG                                                 $REG    3
426                 %W4       EQUALS    $REGORG                                                 $REG    3
427                 %W5       EQUALS    $REGORG                                                 $REG    3
430                 QUE       EQUALS    $REGORG                                                 $REG    3
431                 NUMDONE   EQUALS    $REGORG                                                 $REG    3
432                 DAL       EQUALS    $REGORG                                                 $REG    3
433                 MSG       EQUALS    $REGORG                                                 $REG    3
434                 IOP       EQUALS    $REGORG                                                 $REG    3
435                 FNC       EQUALS    $REGORG                                                 $REG    3
436                 CYL       EQUALS    $REGORG                                                 $REG    3
437                 HED       EQUALS    $REGORG                                                 $REG    3
440                 SEC       EQUALS    $REGORG                                                 $REG    3
441                 DCB       EQUALS    $REGORG                                                 $REG    3
442                 DONE      EQUALS    $REGORG                                                 $REG    3
443                 LLNG      EQUALS    $REGORG                                                 $REG    3
444                 CHAN      EQUALS    $REGORG                                                 $REG    3
445                 LEN0      EQUALS    $REGORG                                                 $REG    3
446                 LEN1      EQUALS    $REGORG                                                 $REG    3
447                 SC0       EQUALS    $REGORG                                                 $REG    3
```

```
450     SC1     EQUALS  $REGORG                                          $REG    3
451     SC2     EQUALS  $REGORG                                          $REG    3
452     SC3     EQUALS  $REGORG                                          $REG    3
453     OFF     EQUALS  $REGORG                                          $REG    3
454     DTR     EQUALS  $REGORG                                          $REG    3
455     TOTAL   EQUALS  $REGORG                                          $REG    3
456     TFC     EQUALS  $REGORG                                          $REG    3
457     MOSU    EQUALS  $REGORG                                          $REG    3
460     MOSL    EQUALS  $REGORG                                          $REG    3
461     UNIT    EQUALS  $REGORG                                          $REG    3
462     CPD     EQUALS  $REGORG                                          $REG    3
463     TPC     EQUALS  $REGORG                                          $REG    3
464     SPT     EQUALS  $REGORG                                          $REG    3
465     DIA     EQUALS  $REGORG                                          $REG    3
466     SUB     EQUALS  $REGORG                                          $REG    3
467     TM      EQUALS  $REGORG                                          $REG    3
470     TMP     EQUALS  $REGORG                                          $REG    3
                                                                    ICOM.15
        EXT     %DCBQ                                               ICOM.16
        EXT     %MYID                                               ICOM.17
        EXT     %LSP1                                               ICOM.18
        EXT     %TRANQ                                              ICOM.19
        EXT     D$AMSG                                              ICOM.20
        EXT     D$ICOM                                              ICOM.21
        EXT     D$TRANSF                                            ICOM.22
        EXT     DACT                                                ICOM.23
        EXT     DALWRT                                              ICOM.24
        EXT     DCCB                                                ICOM.25
        EXT     DGIVEUP                                             ICOM.26
        EXT     DMGA                                                ICOM.27
        EXT     EMSGTOP                                             ICOM.28
        EXT     ENQU                                                ICOM.29
        EXT     ESWP                                                ICOM.30
        EXT     D4ACT                                               ICOM.31
        EXT     HPUT                                                ICOM.32
        EXT     IDALSND                                             ICOM.33
        EXT     IMDAL                                               ICOM.34
        EXT     MIAQ                                                ICOM.35
        EXT     MIMDALQ                                             ICOM.36
        EXT     PUTDAL                                              ICOM.37
        EXT     CA                                                  ICOM.38
        EXT     CB                                                  ICOM.39
```

```
EXT       CC                                                    ICOM.40
EXT       CE                                                    ICOM.41
EXT       DS                                                    ICOM.42
EXT       DT                                                    ICOM.43
EXT       DY                                                    ICOM.44
EXT       DZ                                                    ICOM.45
EXT       ED                                                    ICOM.46
EXT       EH                                                    ICOM.47
EXT       EI                                                    ICOM.48
EXT       TMCB                                                  ICOM.49
```

```
 6                                  START.   *                                                                    ICOM.51
 6   014000 /000000  024410                  R!QUE = (MIAQ)                  .Pointer to ICOM message queue        ICOM.52
     030410  024430
13    <macro>                                $PUNTIF   (A = 0),CODE=PT$MSGCH .If queue not established            ICOM.53
15   010000  024431                          R!NUMDONE = 0                                                         ICOM.54
                                                                                                                   ICOM.55
                                     *                                                                             ICOM.56
                                     *  Get next message to process.                                               ICOM.57
                                     *                                                                             ICOM.58
                                                                                                                   ICOM.59
17                                   NEXT     *                                                                    ICOM.60
17   002000  000000                          I = 0                                                                 ICOM.61
21    <macro>                                $IF       (R!NUMDONE = ICOM$LIM) .If time to giveup control           ICOM.62
                                                                                              .>>>>> 0              $IF      1
25   010000  024431                            R!NUMDONE = 0                 .Clear count of msg's processed       ICOM.63
27   014000 /000000                            A = D$ICOM                    .ICOM DEMON POINTER                   ICOM.64
31    <macro>                                  RKERN     DGIVEUP             .Give up control                      ICOM.65
33   071014                                    P = NEXT                                                            ICOM.66
34    <macro>                                $ENDIF                                                                ICOM.67
                                                                                              .<<<<< 0              $ENDIF   1
34   020430  012004  024410                  A = (R!QUE + QU$POP)            .Queue population                     ICOM.68
     030410
40    <macro>                                $IF      (A # 0)                .If message to process                ICOM.69
                                                                                              .>>>>> 0              $IF      1
42    <macro>                                  GETDAL  R!DAL                 .Get local DAL                        ICOM.70
52   002000  000000                            I = 0                                                               ICOM.71
54    <macro>                                  $IF   (A # 0)                 .If none available                    ICOM.72
                                                                                              .>>>>> 1              $IF      1
56    <macro>                                    PAUSE 1,A1=0,A2=-1          .Pause for 1/10 sec.                  ICOM.73
66   071060                                      P = START                   .Start over                           ICOM.74
67    <macro>                                  $ENDIF                                                              ICOM.75
                                                                                              .<<<<< 1              $ENDIF   1
67   020430  024000                            R!ED = R!QUE                  .Take message off queue               ICOM.76
71    <macro>                                  RKERN ETOQ                                                          ICOM.77
73    <macro>                                  $PUNTIF (R!EH = 0),CODE=PT$DD49 .If nothing there                   ICOM.78
76   020000  024433                            R!MSG = R!EH                  .Save message                         ICOM.79
100   <macro>                                $ELSE                                                                 ICOM.80
101   <macro>                                  PKERN   ESWP                  .End demon                            ICOM.81
103   <macro>                                $ENDIF                                                                ICOM.82
                                                                                              .<<<<< 0              $ENDIF   1
103  003000                                  I = 1                                                                 ICOM.83
104  026431                                  R!NUMDONE = R!NUMDONE + 1       .Increment number processed           ICOM.84
```

```
                                                                                                ICOM.85
                                *                                                               ICOM.86
                                *  Process the message.                                         ICOM.87
                                *                                                               ICOM.88
                                                                                                ICOM.89
105   020433  004012  011003          R!IOP = R!MSG > D'10 & 3       .IOP number of origination  ICOM.90
      024434
111   020434  005002  016000          R!%W5 = R!IOP < 2 + MIAQ       .Locate queue table         ICOM.91
     /000000  024427
116   026427                          R!%W5 = R!%W5 + 1                                          ICOM.92
117   030427  024423                  R!%W1 = (R!%W5)                .MOS base address (upper)   ICOM.93
121   026427                          R!%W5 = R!%W5 + 1                                          ICOM.94
122   030427  024424                  R!%W2 = (R!%W5)                .(lower)                    ICOM.95
124   020433  015000 /001777          R!%W3 = R!MSG & O'1777 < 3     .Offset from base           ICOM.96
      005003  024425
131    <macro>                        DADD      R!%W1,R!%W2,R!%W3    .Add offset                 ICOM.97
136    <macro>                        MOSR      R!%W1,R!%W2,R!DAL,DA@LEN@W .Read in from MOS      ICOM.98
156    <macro>                        GET       R!IFC,DA@IFC,R!DAL   .Get function code          ICOM.99
163    <macro>                        GET       R!CHAN,DL@CHN,R!DAL  .Channel                    ICOM.100
171    <macro>                        GET       R!FNC,DL@FC,R!DAL    .Function code              ICOM.101
177    <macro>                        GET R!DIA,DL@DIA,R!DAL         .Diagnostic flag            ICOM.102
205    <macro>                        GET R!SUB,DL@SUB,R!DAL         .Subfunction code           ICOM.103
214    <macro>                        GET R!TM,DL@TM,R!DAL           .Target memory              ICOM.104
222   020467  016000 /000000          R!TMP = (R!TM + TMCB)          .Pointer to TM control table ICOM.105
      024410  030410  024470
230    <macro>                        GET       R!%W5,DL@IOP,R!DAL   .IOP of disk                ICOM.106
237    <macro>                        GET R!UNIT,DL@UNT,R!DAL        .Requested unit             ICOM.107
246   020427  005003  022461          R!%W5 = R!%W5 < 3 + R!UNIT     .Trace IOP, unit,           ICOM.108
      024427
252   020427  005006  022444          R!%W5 = R!%W5 < 6 + R!CHAN     .and channel                ICOM.109
      024427
                                                                                                ICOM.110
256    <macro>                        KRECORD TF$ICOM,(R!IFC,R!DAL,R!MSG,R!%W5,R!FNC)            ICOM.111
                                                 *  Internal function code                      ICOM.112
                                                 *  MDAL address                                ICOM.113
                                                 *  Accumulator message                         ICOM.114
                                                 *  IOP, unit, and channel of request           ICOM.115
                                                 *  Read/write code                             ICOM.116
                                                                                                ICOM.117
                                *                                                               ICOM.118
                                *  Pocess the DAL.                                              ICOM.119
                                *                                                               ICOM.120
```

```
                                                                                                    ICOM.121
                                 ***********************************************************************  ICOM.122
                                 *        IM$STIO:  Start disk I/O from CPU request                  *  ICOM.123
                                 ***********************************************************************  ICOM.124
                                                                                                    ICOM.125
275    <macro>                          $IF       (R!IIC = IM$STIO)    .If disk request from CPU      ICOM.126
                                                                                          .>>>>> 0        $IF      1
302    020000  126000 /001535             P = STAT, R!%MYID = MIOP     .If status request             ICOM.127
                                                                                                    ICOM.128
                                 *  Validate the DAL parameters.                                      ICOM.129
                                                                                                    ICOM.130
305    <macro>                            $IF (R!CHAN > O'37),OR,(R!CHAN < O'20) .Illegal channel     ICOM.131
                                                                                          .>>>>> 1        $IF      1
316    075000 /001027                       P = PARAMERR               .Parameter error               ICOM.132
320    <macro>                            $ENDIF                                                      ICOM.133
                                                                                          .<<<<< 1        $ENDIF   1
                                                                                                    ICOM.134
320    020467  013003  125000             P = PARAMERR, R!TM >= IM$MAX .Too big, bad value           ICOM.135
       /001027
324    <macro>                            $IF (R!TMP = 0)              .Path not defined              ICOM.136
                                                                                          .>>>>> 1        $IF      1
327    <macro>                              $IF (R!TM = IM$SSD)        .If SSD                        ICOM.137
                                                                                          .>>>>> 2        $IF      1
333    <macro>                                STORE DAR$BKD,DL@RC,R!DAL .Set return code               ICOM.138
343    075000 /001037                         P = PARAM1               .Send to mainframe             ICOM.139
345    <macro>                              $ELSE                                                     ICOM.140
346    075000 /001027                         P = PARAMERR             .Handle illegal parameter      ICOM.141
350    <macro>                              $ENDIF                                                    ICOM.142
                                                                                          .<<<<< 2        $ENDIF   1
350    <macro>                            $ENDIF                                                      ICOM.143
                                                                                          .<<<<< 1        $ENDIF   1
350    014000 /000000  022444             R!DCB = DCCB + R!CHAN - O'20 .Channel table                ICOM.144
       013020  024441
355    030441  024441                     R!DCB = (R!DCB)              .DCB address                   ICOM.145
357    126000 /001027                     P = PARAMERR, A = 0          .Channel not configured        ICOM.146
361    <macro>                            GET R!%W1,DK@TYP,R!DCB       .Device type, DCB              ICOM.147
370    <macro>                            GET A,DL@TYP,R!DAL           .Device type, DAL              ICOM.148
375    023423  127000 /001027             P = PARAMERR, A # R!%W1      .Device type mismatch          ICOM.149
400    <macro>                            GET A,DK@UPC,R!DCB           .Maximum unit number           ICOM.150
405    041000  023461  100002             P = PARAMERR, A <= R!UNIT    .Illegal unit                  ICOM.151
       103003  075000 /001027
                                                                                                    ICOM.152
```

```
                                  *  Validate request length.                                    ICOM.153
                                                                                                 ICOM.154
413    <macro>                      GET     R!LEN0,DL@LN0,R!DAL .Request length (upper)         ICOM.155
420    <macro>                      GET     R!LEN1,DL@LN1,R!DAL .(lower)                        ICOM.156
425    <macro>                      $IF (R!LEN0 = 0),AND,(R!LEN1 = 0) .Zero length              ICOM.157
                                                                          .>>>>> 1                $IF      1
433   075000 /001027                  P = PARAMERR                .Parameter error              ICOM.158
435    <macro>                      $ENDIF                                                      ICOM.159
                                                                          .<<<<< 1                $ENDIF   1
435   020445  004011  024410         P = PARAMERR, 0 # R!LEN0 > D'9 .Limit transfers to 65K      ICOM.160
      010000  023410  127000
     /001027
444    <macro>                      $IF (R!TM = TM$SSD)         .If target is SSD             R164101B.1
                                                                          .>>>>> 1                $IF      1
450   020446  011077  024410           P = PARAMERR, 0 # R!LEN1 & MSK$SSD .Must be mod 64        R164101B.2
      010000  023410  127000
     /001027
457    <macro>                      $ENDIF                                                      R164101B.3
                                                                          .<<<<< 1                $ENDIF   1
457    <macro>                      GET R!%W1,DL@TM0,R!DAL      .Starting address                ICOM.161
466    <macro>                      GET R!%W2,DL@TM1,R!DAL      .                                ICOM.162
                                                                                                 ICOM.163
                                  *  Include the target memory base address.                     ICOM.164
                                                                                                 ICOM.165
473    <macro>                      GET R!%W3,TM@FW0,R!TMP      .Target memory base address      ICOM.166
500    <macro>                      GET R!%W4,TM@FW1,R!TMP                                       ICOM.167
505    <macro>                      DADD R!%W1,R!%W2,R!%W3,R!%W4 .Add base                       ICOM.168
515    <macro>                      $IF (R!TM = TM$SSD)         .If target is SSD             R164101B.4
                                                                          .>>>>> 1                $IF      1
521   020424  011077  024410           P = PARAMERR, 0 # R!%W2 & MSK$SSD .Must be mod 64         R164101B.5
      010000  023410  127000
     /001027
530    <macro>                      $ENDIF                                                      R164101B.6
                                                                          .<<<<< 1                $ENDIF   1
530    <macro>                      PUT R!%W1,DL@TM0,R!DAL      .Save in DAL                     ICOM.169
544    <macro>                      PUT R!%W2,DL@TM1,R!DAL      .                                ICOM.170
                                                                                                 ICOM.171
                                  *  Check ending address.                                       ICOM.172
                                                                                                 ICOM.173
551    <macro>                      DADD R!%W1,R!%W2,R!LEN0,R!LEN1 .Ending address               ICOM.174
561    <macro>                      GET R!%W3,TM@LM0,R!TMP      .Target memory limit address     ICOM.175
566    <macro>                      GET R!%W4,TM@LM1,R!TMP                                       ICOM.176
```

```
573   020423  023425  100004          P = PARAMERR, R!%W1 > R!%W3 .Too big                          ICOM.177
      102003  075000 /001027
601    <macro>              .         $IF (R!%W1 = R!%W3),AND,(R!%W2 > R!%W4)                       ICOM.178
                                                                          .>>>>> 1                     $IF      1
613   075000 /001027                    P = PARAMERR              .Too big                            ICOM.179
615    <macro>                         $ENDIF                                                        ICOM.180
                                                                          .<<<<< 1                   $ENDIF    1
                                                                                                      ICOM.181
                                     *  Check device limits.                                          ICOM.182
                                                                                                      ICOM.183
615   077000 /002352                  R = LIMITS                .Get device limits                    ICOM.184
617    <macro>                        GET     R!CYL,DL@CYL,R!DAL  .Cylinder                           ICOM.185
625   041000  023462  125000          P = PARAMERR, A >= R!CPD    .Illegal cylinder                   ICOM.186
     /001027
631    <macro>                        GET     R!HED,DL@HED,R!DAL  .Head group                         ICOM.187
637   041000  023463  125000          P = PARAMERR, A >= R!TPC    .Illegal head                       ICOM.188
     /001027
643    <macro>                        GET     R!SEC,DL@SEC,R!DAL  .Sector                             ICOM.189
651   041000  023464  125000          P = PARAMERR, A >= R!SPT    .Illegal sector                     ICOM.190
     /001027
                                                                                                      ICOM.191
                                  *  Calculate number of sectors to do and last sector length.       ICOM.192
                                                                                                      ICOM.193
655    <macro>                        GET     R!OFF,DL@OFF,R!DAL  .Offset into first sector           ICOM.194
663   020440  005011  022453          R!%W1 = R!SEC < D'9 + R!OFF                                     ICOM.195
      024423
667    <macro>                        DADD    R!LEN0,R!LEN1,R!%W1                                     ICOM.196
674   020446  011777  024443          R!LLNG = R!LEN1 & O'777     .Last sector length                 ICOM.197
677   027446                          R!LEN1 = R!LEN1 - 1         .Ending sector                      ICOM.198
700   101002  027445                  R!LEN0 = R!LEN0 - 1, C = 0  .If underflow                       ICOM.199
702    <macro>                        RSHIFT  R!LEN0,R!LEN1,D'9   .Double shift right                 ICOM.200
721   020446  023440  012001          R!TOTAL = R!LEN1 - R!SEC + 1 .Number of sectors to do           ICOM.201
      024455
725   020443  103004  014000          R!LLNG = D4$WPS, R!LLNG = 0  .If full last sector               ICOM.202
     /001000  024443
732   020455  013001  103004          R!LLNG = R!LLNG - R!OFF, R!TOTAL = 1 .If less than 1 sector     ICOM.203
      020443  023453  024443
740    <macro>                        PUT     R!LLNG,DL@LLN,R!DAL .Store last sector length          ICOM.204
754    <macro>                        PUT     R!TOTAL,DL@TOT,R!DAL .Number of sectors for request    ICOM.205
761    <macro>                        $IF (R!DIA = 1),AND,(R!FNC = DAF$WRIT) .Diagnostic write        ICOM.206
                                                                          .>>>>> 1                     $IF      1
773    <macro>                          GET A,DL@RCC,R!DAL        .Write behind count                 ICOM.207
```

```
 777    <macro>                          $IF (A = 0)                 .If not suppressed                          ICOM.208
                                                                                          .>>>>> 2                  $IF      1
1002    <macro>                            STORE DAF$WRBH,DL@FC,R!DAL .Adjust function code                         ICOM.209
1012    <macro>                          $ENDIF                                                                    ICOM.210
                                                                                          .<<<<< 2                  $ENDIF   1
1012    <macro>                        $ENDIF                                                                      ICOM.211
                                                                                          .<<<<< 1                  $ENDIF   1
1012   002000  000000                    I = 0                                                                     ICOM.212
1014    <macro>                          ADDRESS R!ED,DCB@DA,R!%DCBQ .Start DAL queue                              ICOM.213
1017   020432  024000                    R!EH = R!DAL                .DAL address                                  ICOM.214
1021    <macro>                          RKERN ENQU                  .Put on queue                                 ICOM.215
1023   077000 /002432                    R = ACTDEM                  .Activate D4DEM                               ICOM.216
1025   075000 /000017                    P = NEXT                    .Next request                                 ICOM.217
                                                                                                                   ICOM.218
                                 *  Handle illegal parameters in a request.                                        ICOM.219
                                                                                                                   ICOM.220
1027                             PARAMERR *                                                                        ICOM.221
1027    <macro>                          STORE DAR$PAR,DL@RC,R!DAL   .Illegal parameter                            ICOM.222
1037                             PARAM1   *                                                                        ICOM.223
1037   002000  000000                    I = 0                                                                     ICOM.224
1041   077000 /002333                    R = STATUS                  .Send status to MIOP                          ICOM.225
1043    <macro>                          RELDAL R!DAL                .Release local DAL                            ICOM.226
                                                                                                                   ICOM.227
                                 ************************************************************************          ICOM.228
                                 *        IM$TRAN, IM$ABRT:  Data Transfer or Abort request for TRANSFR *          ICOM.229
                                 ************************************************************************          ICOM.230
                                                                                                                   ICOM.231
1053    <macro>                          $ELSEIF   (R!IFC = IM$TRAN),OR,(R!IFC = IM$ABRT)                          ICOM.232
1064    <macro>                            GET R!IMP,TM@IOP,R!TMP      .Target memory IOP                            ICOM.233
1071    <macro>                            $PUNTIF (A # R!%MYID),CODE=PI$DD49                                    ICOM.234
1074   077000 /002442                    R = RLDAL                   .Release MOS DAL                              ICOM.235
1076   002000  000000                    I = 0                                                                     ICOM.236
1100    <macro>                          ADDRESS R!ED,TRA@QU,R!%TRANQ .TRANSFR request queue                       ICOM.237
1103   020432  024000                    R!EH = R!DAL                .Message to queue                             ICOM.238
1105    <macro>                          RKERN   ENQU                .Put on queue                                 ICOM.239
1107   014000 /000000                    A = D$TRANSF                .TRANSF DEMON FLAG                            ICOM.240
1111    <macro>                          RKERN   DACT                .Activate                                     ICOM.241
1113   003000                            I = 1                                                                     ICOM.242
                                                                                                                   ICOM.243
                                 ************************************************************************          ICOM.244
                                 *        IM$RESP, IM$ARSP: Data transfer or Abort transfer response    *          ICOM.245
                                 ************************************************************************          ICOM.246
```

```
                                                                                                ICOM.247
1114   <macro>                  $ELSEIF (R!IFC = IM$RESP),OR,(R!IFC = IM$ARSP)                  ICOM.248
1125   <macro>                    GET R!IMP,IM@IOP,R!IMP       .Target memory IOP                 ICOM.249
1132   <macro>                    $PUNTIF (A = R!%MYID),CODE=PI$DD49                              ICOM.250
1135   077000 /002442             R = RLDAL                    .Release MOS DAL                   ICOM.251
1137   020432  024454             R!DIR = R!DAL                .Save DIR address                  ICOM.252
1141   <macro>                    GET     R!DAL,IR@MDL,R!DIR   .Originating MDAL                  ICOM.253
1146   014000 /000000  022444     R!DCB = DCCB + R!CHAN - O'20 .Channel table                     ICOM.254
       013020  024441
1153   030441  024441             R!DCB = (R!DCB)              .DCB address                       ICOM.255
1155   <macro>                    GET     R!DONE,IR@DON,R!DIR .Get count of sectors moved         ICOM.256
1162   <macro>                    GET     R!TOTAL,IR@TOT,R!DTR .Total # sectors to do             ICOM.257
                                                                                                ICOM.258
                                * Abort Transfer response.                                      ICOM.259
                                                                                                ICOM.260
1167   <macro>                    $IF (R!IFC = IM$ARSP)        .If Abort response                 ICOM.261
                                                                                .>>>>> 1            $IF      1
1173   <macro>                      $IF (0 = R!FNC & 1)        .If writing                        ICOM.262
                                                                                .>>>>> 2            $IF      1
1202   002000  000000                I = 0                                                        ICOM.263
1204   <macro>                       ADDRESS R!ED,DK@DLO,R!DCB .DAL queue                       ICOM.264
1207   <macro>                       RKERN EDQU               .Take first entry                 ICOM.265
1211   <macro>                       $PUNTIF (R!EH # R!DAL),CODE=PI$DD49 .Must match            ICOM.266
1215   <macro>                       GET R!%W1,DA@IFC,R!DAL   .Internal function code           ICOM.267
1222   <macro>                       $IF (R!FNC = DAF$WBFR)   .Write behind with final reply    ICOM.268
                                                                                .>>>>> 3            $IF      1
1226   <macro>                         $IF (R!%W1 = IM$STAT) .If early status already sent      ICOM.269
                                                                                .>>>>> 4            $IF      1
1232   020432  024000                    R!DS = R!DAL         .Local DAL                         ICOM.270
1234   <macro>                           RKERN DDALGETA       .Get an MOS DAL                    ICOM.271
1236   <macro>                         $ENDIF                                                   ICOM.272
                                                                                .<<<<< 4            $ENDIF   1
1236   077000 /002333                  R = STATUS             .Send status                      ICOM.273
1240   <macro>                       $ELSEIF (R!FNC = DAF$WRBH) .Write behind                   ICOM.274
1245   <macro>                         $IF (R!%W1 # IM$STAT) .Status not sent yet               ICOM.275
                                                                                .>>>>> 4            $IF      1
1251   077000 /002333                    R = STATUS           .Send it                           ICOM.276
1253   <macro>                         $ENDIF                                                   ICOM.277
                                                                                .<<<<< 4            $ENDIF   1
1253   <macro>                       $ELSE                    .Any other write fnc.              ICOM.278
1254   077000 /002333                  R = STATUS             .Send status                       ICOM.279
1256   <macro>                       $ENDIF                                                     ICOM.280
```

```
                                                                    .<<<<< 3          $ENDIF    1
1256    <macro>                  RELDAL R!DAL            .Release local DAL                ICOM.281
1266    <macro>                $ENDIF                                                      ICOM.282
                                                                    .<<<<< 2          $ENDIF    1
1266    <macro>                STORE 0,DKF@ER,R!DCB      .Clear error recovery flag        ICOM.283
                                                                                            ICOM.284
                             *  Reading from disk.                                          ICOM.285
                                                                                            ICOM.286
1272    <macro>                $ELSEIF (R!FNC = DAF$READ)  .If reading from disk           ICOM.287
1277    <macro>                $IF   (R!DONE = R!TOTAL)  .If request complete              ICOM.288
                                                                    .>>>>> 2          $IF       1
1303   002000  000000            I = 0                                                     ICOM.289
1305    <macro>                  STORE BP$DONE,DL@TDL,R!DAL .Data in CPU                   ICOM.290
1316    <macro>                  ADDRESS R!ED,DK@DLO,R!DCB .DAL queue                      ICOM.291
1321    <macro>                  RKERN EDQU                .Take off queue                 ICOM.292
1323    <macro>                  ADDRESS R!ED,DCB@DN,R!%DCBQ .DAL done queue               ICOM.293
1326    <macro>                  RKERN ENQU                .Put on queue                   ICOM.294
1330    <macro>                $ENDIF                                                      ICOM.295
                                                                    .<<<<< 2          $ENDIF    1
                                                                                            ICOM.296
                             *  Writing to disk.                                            ICOM.297
                                                                                            ICOM.298
1330    <macro>                $ELSE                                                       ICOM.299
1331    <macro>                $IF   (R!DONE = 1)        .If first response                ICOM.300
                                                                    .>>>>> 2          $IF       1
1335    <macro>                  STORE IM$RESP,DA@IFC,R!DAL .Data in MOS                   ICOM.301
1342    <macro>                $ENDIF                                                      ICOM.302
                                                                    .<<<<< 2          $ENDIF    1
1342    <macro>                $IF (R!DONE = R!TOTAL)    .If all data in MOS               ICOM.303
                                                                    .>>>>> 2          $IF       1
1346    <macro>                  $IF (R!FNC = DAF$WRBH),OR,(R!FNC = DAF$WBFR)              ICOM.304
                                                                    .>>>>> 3          $IF       1
1355   002000  000000              I = 0                                                   ICOM.305
1357   077000 /002333              R = STATUS              .Status to mainframe            ICOM.306
1361    <macro>                    GET R!SEC,DL@SEC,R!DAL .Starting sector                 ICOM.307
1367    <macro>                    GET R!HED,DL@HED,R!DAL .Starting head group             ICOM.308
1375    <macro>                    GET R!CYL,DL@CYL,R!DAL .Starting cylinder               ICOM.309
1403    <macro>                    GET R!UNIT,DL@UNT,R!DAL .Unit number                    ICOM.310
1412   077000 /002352              R = LIMITS              .Get device limits              ICOM.311
1414    <macro>                    GET R!%W1,DL@TOT,R!DAL .Sectors total                   ICOM.312
1421   020440  022423  024440      R!SEC = R!SEC + R!%W1                                   ICOM.313
1424    <macro>                    $UNTIL (R!SEC < R!SPT) .Calculate disk address          ICOM.314
```

```
1427   020440  023464  024440                R!SEC = R!SEC - R!SPT .Decrement                          ICOM.315
1432   026437                                R!HED = R!HED + 1    .Next track                          ICOM.316
1433    <macro>                              $IF (A = R!TPC)      .If end of cylinder                  ICOM.317
                                                                                      .>>>>> 4            $IF      1
1436   010000  024437                          R!HED = 0          .Reset                               ICOM.318
1440   026436                                  R!CYL = R!CYL + 1 .Next cylinder                         ICOM.319
1441    <macro>                              $ENDIF                                                     ICOM.320
                                                                                      .<<<<< 4            $ENDIF   1
1441    <macro>                            $ENDTIL                                                      ICOM.321
1442    <macro>                            PUT R!SEC,DK@RAS,R!DCB .Save ending sector                   ICOM.322
1455    <macro>                            PUT R!HED,DK@RAH,R!DCB .Head group                           ICOM.323
1471    <macro>                            PUT R!CYL,DK@RAC,R!DCB .Cylinder                             ICOM.324
1476    <macro>                            PUT R!UNIT,DK@RAU,R!DCB .Unit number                         ICOM.325
1511   003000                              I = 1                                                        ICOM.326
1512    <macro>                          $ENDIF                                                         ICOM.327
                                                                                      .<<<<< 3            $ENDIF   1
1512    <macro>                            $ENDIF                                                       ICOM.328
                                                                                      .<<<<< 2            $ENDIF   1
1512    <macro>                          $ENDIF                                                         ICOM.329
                                                                                      .<<<<< 1            $ENDIF   1
1512    <macro>                          RELDAL R!DTR                .Release local DTR                 ICOM.330
1522   077000 /002432                    R = ACTDEM                  .Activate disk demon               ICOM.331
                                                                                                         ICOM.332
                                 ********************************************************************    ICOM.333
                                 *        IM$STAT:  Send status to mainframe                        *    ICOM.334
                                 ********************************************************************    ICOM.335
                                                                                                         ICOM.336
1524    <macro>                          $ELSEIF (R!IFC = IM$STAT)                                      ICOM.337
1532    <macro>                          $PUNTIF (R!%MYID # MIOP),CODE=PT$DD49                          ICOM.338
1535                             STAT     *                                                             ICOM.339
1535    <macro>                          $IF (R!IOP = R!%MYID)       .If MOS DAL belongs to this IOP    ICOM.340
                                                                                      .>>>>> 1            $IF      1
1541   077000 /002506                      R = RELBMDAL              .Release it                        ICOM.341
1543    <macro>                          $ELSE                                                          ICOM.342
1544   077000 /002442                      R = RLDAL                 .Ask owner to release it           ICOM.343
1546    <macro>                          $ENDIF                                                         ICOM.344
                                                                                      .<<<<< 1            $ENDIF   1
1546    <macro>                          GET R!%W1,DL@UNS,R!DAL      .Number physical units             ICOM.345
1554    <macro>                          $IF (A # 1)                 .If striped disk response          ICOM.346
                                                                                      .>>>>> 1            $IF      1
1557    <macro>                            STORE AM$BDRSP,DA@IFC,R!DAL .Code for AMSG                   ICOM.347
1564   002000  000000                      I = 0                                                        ICOM.348
```

```
1566   014000 /000000  024000          R!DZ = MIMDALQ            .AMSG queue                          ICOM.349
1571   020432  024000                  R!DY = R!DAL              .DAL address                         ICOM.350
1573    <macro>                        RKERN HPUT                .Queue it                            ICOM.351
1575   014000 /000000                  A = D$AMSG                .AMSG descriptor                     ICOM.352
1577    <macro>                        RKERN DACT                .Activate it                         ICOM.353
1601   003000                          I = 1                                                          ICOM.354
1602   075000 /000017                  P = NEXT                                                       ICOM.355
1604    <macro>                      $ENDIF                                                           ICOM.356
                                                                                .<<<<< 1                $ENDIF   1
1604    <macro>                      $IF (R!IOP = R!%MYID)     .If DAL originated in this IOP         ICOM.357
                                                                                .>>>>> 1                $IF      1
1610    <macro>                        FLDSUB 1,CPI@DP,R!%LSPI   .Adjust disk request population        ICOM.358
1614    <macro>                      $ENDIF                                                           ICOM.359
                                                                                .<<<<< 1                $ENDIF   1
1614    <macro>                      GET R!%W1,DA@SID,R!DAL    .Swap DID and SID                    ICOM.360
1621    <macro>                      GET R!%W2,DA@DID,R!DAL    .                                    ICOM.361
1626    <macro>                      PUT R!%W1,DA@DID,R!DAL    .                                    ICOM.362
1633    <macro>                      PUT R!%W2,DA@SID,R!DAL    .                                    ICOM.363
1640    <macro>                      STORE RQ$DISK,DA@SIL,R!DAL .Set source 'A' packet              ICOM.364
1650   002000  000000                I = 0                                                          ICOM.365
1652   020432  024000                R!DY = R!DAL              .DAL address                         ICOM.366
1654    <macro>                      RKERN IDALSND             .Send                                ICOM.367
1656   003000                        I = 1                                                          ICOM.368
                                                                                                      ICOM.369
                                 ***********************************************************************ICOM.370
                                 *        IM$ERRR:  Send error packet to CPU                         *ICOM.371
                                 ***********************************************************************ICOM.372
                                                                                                      ICOM.373
1657    <macro>                    $ELSEIF   (R!IFC = IM$ERRR)                                       ICOM.374
1666    <macro>                      $PUNTIF (R!%MYID # MIOP),CODE=PT$DD49                           ICOM.375
1671   077000 /002442                R = RLDAL                 .Release MOS DAL                     ICOM.376
                                                                                                      ICOM.377
                                 *        Build DAL's containing error info.                         ICOM.378
                                                                                                      ICOM.379
1673    <macro>                      GET R!MOSU,DL@BM0,R!DAL   .MOS address of error info           ICOM.380
1700    <macro>                      GET R!MOSL,DL@BM1,R!DAL                                         ICOM.381
1705   010010  012004  024453        R!OFF = DA@@LH + 4        .Offset to start of info             ICOM.382
1710   010030  004002  013001        R!LEN0 = DA@@LE > 2 - 1   .Words of info per DAL               ICOM.383
       024445
1714   010001  024447                R!SC0 = 1                 .Initialize                          ICOM.384
1716   020432  024450                R!SC1 = R!DAL             .First link                          ICOM.385
1720   020457  024451                R!SC2 = R!MOSU                                                  ICOM.386
```

```
1722   020460  024452                    R!SC3 = R!MOSL                                                    ICOM.387
1724    <macro>                          GET R!MSG,DA@DIU,R!DAL        .Task ID                            ICOM.388
1732    <macro>                          GET R!SUB,DE@SEQ,R!DAL        .Request number or error subtype    ICOM.389
1737   020466  004010  024465            R!DIA = R!SUB > D'8           .Upper byte                         ICOM.390
1742   020466  011377  024466            R!SUB = R!SUB & O'377         .Lower byte                         ICOM.391
1745    <macro>                          GET R!TOTAL,DE@TOT,R!DAL      .Number of DAL's needed             ICOM.392
1752    <macro>                          $UNTIL (R!SCO = R!TOTAL + 1) .Build error DAL's                   ICOM.393
1761    <macro>                            GETDAL R!DTR                .Get a DAL                          ICOM.394
1771    <macro>                            $IF (A # 0)                 .If error                           ICOM.395
                                                                                        .>>>>> 1              $IF      1
1774   030432  024447                      R!SCO = (R!DAL)             .First DAL                          ICOM.396
1776    <macro>                            $UNTIL (R!SCO = 0)          .Until all released                 ICOM.397
2001   020447  024454                        R!DTR = R!SCO             .Save address                       ICOM.398
2003   030447  024447                        R!SCO = (R!SCO)           .Pointer to next                    ICOM.399
2005    <macro>                              RELDAL R!DTR              .Release                            ICOM.400
2015    <macro>                            $ENDTIL                                                         ICOM.401
2016   070172                              P = RELBMBUF                .Release MOS disk buffer            ICOM.402
2017    <macro>                            $ENDIF                                                          ICOM.403
                                                                                        .<<<<< 1              $ENDIF   1
2017    <macro>                            CLEAR START=R!DTR,COUNT=R!OFF .Clear header                     ICOM.404
2032    <macro>                            MOSR R!SC2,R!SC3,R!DTR+R!OFF,R!LENO .Read in error info         ICOM.405
2053    <macro>                            STORE RQ$ERRR,DA@SIL,R!DTR .Source ID                           ICOM.406
2063    <macro>                            PUT R!MSG,DA@SIU,R!DTR      .Task ID                            ICOM.407
2076    <macro>                            PUT R!DIA,DE@RQN,R!DTR      .Request number upper byte          ICOM.408
2111    <macro>                            PUT R!SUB,DE@TYP,R!DTR      .Lower byte, or error subtype       ICOM.409
2124    <macro>                            STORE RQ$CPU,DA@DID,R!DTR .Destination                          ICOM.410
2132    <macro>                            PUT R!SCO,DE@SEQ,R!DTR      .Sequence number of this DAL        ICOM.411
2145    <macro>                            PUT R!TOTAL,DE@TOT,R!DTR    .Total DAL's in message             ICOM.412
2160    <macro>                            DADD R!SC2,R!SC3,R!LENO     .Next MOS info                      ICOM.413
2165   020454  034450                      (R!SC1) = R!DTR             .Link                               ICOM.414
2167   020454  024450                      R!SC1 = R!DTR                                                   ICOM.415
2171   026447                              R!SCO = R!SCO + 1                                               ICOM.416
2172    <macro>                          $ENDTIL                                                           ICOM.417
                                                                                                            ICOM.418
                                 *       Send DAL's to the CPU.                                             ICOM.419
                                                                                                            ICOM.420
2173   002000  000000                    I = 0                                                             ICOM.421
2175   030432  024447                    R!SCO = (R!DAL)               .First DAL                          ICOM.422
2177    <macro>                          $UNTIL (R!SCO = 0)            .Until all sent                     ICOM.423
2201   020447  024000                      R!DY = R!SCO                .DAL address                        ICOM.424
2203   030447  024447                      R!SCO = (R!SCO)             .Next DAL                           ICOM.425
2205    <macro>                            RKERN IDALSND               .Send                               ICOM.426
```

```
2207    <macro>                         $ENDTIL                                                         ICOM.427
2210                            RELBMBUF *                                                              ICOM.428
2210    <macro>                         STORE IM$RLBF,DA@IFC,R!DAL  .Release MOS disk buffer            ICOM.429
2215   002000  000000                   I = 0                                                           ICOM.430
2217   020432  024000                   R!DS = R!DAL                .DAL address                        ICOM.431
2221    <macro>                         RKERN DDALGETA              .Get an MOS DAL                     ICOM.432
2223    <macro>                         RKERN DSUM                  .Write to MOS                       ICOM.433
2225   020434  024000                   R!EI = R!IOP                .Target IOP                         ICOM.434
2227    <macro>                         RKERN EMSGIOP               .Send                               ICOM.435
2231    <macro>                         RELDAL  R!DAL               .Release local DAL                  ICOM.436
                                                                                                        ICOM.437
                                ***********************************************************************ICOM.438
                                *       IM$RLDL:  Release MOS DAL                                     *ICOM.439
                                ***********************************************************************ICOM.440
                                                                                                        ICOM.441
2241    <macro>                         $ELSEIF   (R!IFC = IM$RLDL)  .If request to release MOS         ICOM.442
2246   072240                           R = RELBMDAL                .Release the MOS DAL                ICOM.443
2247    <macro>                         RELDAL  R!DAL               .Release local DAL                  ICOM.444
                                                                                                        ICOM.445
                                ***********************************************************************ICOM.446
                                *       IM$RLBF:  Release MOS disk buffer                             *ICOM.447
                                ***********************************************************************ICOM.448
                                                                                                        ICOM.449
2257    <macro>                         $ELSEIF (R!IFC = IM$RLBF)                                       ICOM.450
2264    <macro>                         $PUNTIF (R!%MYID = MIOP),CODE=PT$DD49                           ICOM.451
2267   072153                           R = RLDAL                   .Release MOS DAL                    ICOM.452
2270    <macro>                         GET R!MOSU,DL@BM0,R!DAL     .MOS buffer address                 ICOM.453
2275    <macro>                         GET R!MOSL,DL@BM1,R!DAL                                         ICOM.454
2302    <macro>                         MPUT R!MOSU,R!MOSL          .Release                            ICOM.455
2316    <macro>                         RELDAL R!DAL                .Release local DAL                  ICOM.456
                                                                                                        ICOM.457
                                ***********************************************************************ICOM.458
                                *       All other function codes are illegal                          *ICOM.459
                                ***********************************************************************ICOM.460
                                                                                                        ICOM.461
2326    <macro>                         $ELSE                                                           ICOM.462
2327    <macro>                         $PUNTIF CODE=PT$DD49                                            ICOM.463
2331    <macro>                         $ENDIF                                                          ICOM.464
                                                                                .<<<<< 0                $ENDIF   1
2331   075000 /000017                   P = NEXT                    .Next message                       ICOM.465
```

```
                                 ********************************************************************    ICOM.467
                                 *        SUBROUTINES                                               *    ICOM.468
                                 ********************************************************************    ICOM.469
                                                                                                          ICOM.470
                                 ********************************************************************    ICOM.471
                                 *                                                                  *    ICOM.472
                                 *        STATUS:    Send status to MIOP.                            *    ICOM.473
                                 *                                                                  *    ICOM.474
                                 *        Entry:     R!DAL = DAL address                             *    ICOM.475
                                 *                                                                  *    ICOM.476
                                 ********************************************************************    ICOM.477
                                                                                                          ICOM.478
2333                             STATUS   *                                                               ICOM.479
2333    <macro>                          STORE IM$STAT,DA@IFC,R!DAL     .Status to mainframe              ICOM.480
2340   020432  024000                    R!DS = R!DAL                   .DAL address                      ICOM.481
2342    <macro>                          RKERN DMGA                     .Send to MIOP                     ICOM.482
2344    <macro>                          STORE 0,DL@ERC,R!DAL           .Clear errcount for next reply    ICOM.483
2351   001000                            EXIT                                                             ICOM.484
                                                                                                          ICOM.485
                                 ********************************************************************    ICOM.486
                                 *                                                                  *    ICOM.487
                                 *        LIMITS:    Get device configuration limits.                *    ICOM.488
                                 *                                                                  *    ICOM.489
                                 *        Entry:     R!DCB = DCB address                             *    ICOM.490
                                 *                   R!SUB = DAL subfunction code                    *    ICOM.491
                                 *                                                                  *    ICOM.492
                                 *        Exit:      R!CPD = Cylinders per disk                      *    ICOM.493
                                 *                   R!TPC = Tracks per cylinder                     *    ICOM.494
                                 *                   R!SPT = Sectors per track                       *    ICOM.495
                                 *                                                                  *    ICOM.496
                                 ********************************************************************    ICOM.497
                                                                                                          ICOM.498
2352                             LIMITS   *                                                               ICOM.499
2352    <macro>                          GET R!%W1,DK@DPT,R!DCB         .Device parameter table           ICOM.500
2357    <macro>                          $IF (R!SUB = DAS$ID),OR,(R!SUB = DAS$ABS) .If absolute mode      ICOM.501
                                                                                        .>>>>> 0              $IF      1
2366    <macro>                            GET R!SPT,DP@ASE,R!%W1       .Absolute sectors/track           ICOM.502
2374    <macro>                          $ELSE                                                            ICOM.503
2375    <macro>                            GET R!SPT,DP@DSE,R!%W1       .Default sectors/track            ICOM.504
2403    <macro>                          $ENDIF                                                           ICOM.505
                                                                                        .<<<<< 0              $ENDIF   1
2403    <macro>                          GET R!TPC,DK@TPC,R!DCB         .Tracks/cylinder                  ICOM.506
```

```
2412    <macro>                         $IF (R!DIA = 1)                 .If diagnostic request                  ICOM.507
                                                                                          .>>>>> 0                $IF      1
2416    <macro>                           GET R!CPD,DP@ACY,R!%W1        .Absolute cylinders/disk                ICOM.508
2423    <macro>                         $ELSE                                                                   ICOM.509
2424    <macro>                           GET R!CPD,DP@DCY,R!%W1        .Default cylinders/disk                 ICOM.510
2431    <macro>                         $ENDIF                                                                  ICOM.511
                                                                                          .<<<<< 0              $ENDIF   1
2431   001000                           EXIT                                                                    ICOM.512
                                                                                                                ICOM.513
                                 ***********************************************************************        ICOM.514
                                 *                                                                     *        ICOM.515
                                 *        ACTDEM:    Puts the DCB on queue for D4DEM.                   *        ICOM.516
                                 *                                                                     *        ICOM.517
                                 *        Entry:     R!DCB = DCB address                                *        ICOM.518
                                 *                                                                     *        ICOM.519
                                 ***********************************************************************        ICOM.520
                                                                                                                ICOM.521
2432                             ACTDEM   *                                                                     ICOM.522
2432   002000  000000                     I = 0                                                                 ICOM.523
2434   020441  024000                     R!DT = R!DCB                  .DCB address                            ICOM.524
2436    <macro>                           RKERN     D4ACT               .Activate D4DEM                         ICOM.525
2440   003000                             I = 1                                                                 ICOM.526
2441   001000                             EXIT                                                                  ICOM.527
                                                                                                                ICOM.528
                                 ***********************************************************************        ICOM.529
                                 *                                                                     *        ICOM.530
                                 *        RLDAL:     Send message to another IOP to release an          *        ICOM.531
                                 *                   MOS DAL.                                           *        ICOM.532
                                 *                                                                     *        ICOM.533
                                 *          Entry:     R!DAL = DAL address                              *        ICOM.534
                                 *                                                                     *        ICOM.535
                                 ***********************************************************************        ICOM.536
                                                                                                                ICOM.537
2442                             RLDAL    *                                                                     ICOM.538
2442   014000 /000000  024423             R!%W1 = IMDAL                 .Use immediate DAL                      ICOM.539
2445    <macro>                           COPY R!DAL,R!%W1,DA@@LH       .Copy header                            ICOM.540
2463    <macro>                           STORE IMSRLDL,DA@IFC,R!%W1    .Set request code                       ICOM.541
2470   002000  000000                     I = 0                                                                 ICOM.542
2472   020423  024000                     R!DS = R!%W1                  .DAL address                            ICOM.543
2474   010001  024000                     R!CE = 1                      .Words to write                         ICOM.544
2476    <macro>                           RKERN DALWRI                  .Write it                               ICOM.545
2500   020434  024000                     R!EI = R!IOP                  .Who to send it to                      ICOM.546
```

```
2502    <macro>                         RKERN EMSGIOP                   .Send it                                ICOM.547
2504   003000                           I = 1                                                                   ICOM.548
2505   001000                           EXIT                                                                    ICOM.549
                                                                                                                ICOM.550
                                 ***********************************************************************        ICOM.551
                                 *                                                                     *        ICOM.552
                                 *          RELBMDAL: Release MOS DAL for request.                     *        ICOM.553
                                 *                                                                     *        ICOM.554
                                 *          Entry:    R!DAL = DAL address                              *        ICOM.555
                                 *                                                                     *        ICOM.556
                                 ***********************************************************************        ICOM.557
                                                                                                                ICOM.558
2506                             RELBMDAL *                                                                     ICOM.559
2506   002000  000000                    I = 0                           .Release the MOS DAL                    ICOM.560
2510   010001  024000                    R!CA = 1                                                                ICOM.561
2512    <macro>                          GET R!CB,DA@DL0,R!DAL                                                   ICOM.562
2517    <macro>                          GET R!CC,DA@DL1,R!DAL                                                   ICOM.563
2524    <macro>                          RKERN PUTDAL                                                            ICOM.564
2526   003000                            I = 1                                                                   ICOM.565
2527   001000                            EXIT                                                                    ICOM.566
                                         END                                                                     ICOM.567
```

0X	%B		1: 4 D	7: 7 B	7: 8 B	7:14 B	7:18 B	7:25 B	7:27 B	7:35 B
			7:38 B	7:41 B	8: 7 B	8:11 B	8:16 B	8:31 B	9: 2 B	9: 5 B
			9:11 B	9:13 B	9:16 B	9:19 B	9:41 B	10: 2 B	10:12 B	10:13 B
			10:21 B	10:28 B	10:31 B	11: 2 B	11: 5 B	11:33 B	11:37 B	11:41 B
			12:34 B	12:39 B	13:22 B	13:28 B	13:33 B	13:35 B	14: 7 B	14:29 B
			14:31 B	15: 7 B	15:10 B	15:12 B	16:40 B			
0X	%DCBQ		3:19 D	10: 9	12:16					
0X	%EX		5:25 D	5:25	5:29 D	5:29	6:17 D	6:17	10:22 D	10:22
			12: 2 D	12: 2	13:21 D	13:21	15: 8 D	15: 8	15:15 D	15:15
			15:21 D	15:21	16:10 D	16:10	16:18 D	16:18	16:29 D	16:29
			16:30 D	16:30						
10	%GBLNUM	$APTEXT	2:10 F	2:10						
400	%GBLREG	$APTEXT	2:10							
0	%KERNEL	$APTEXT	5:17 D	5:17 B	5:34 D	5:34 B	5:38 D	5:38 B	10:11 D	10:11 B
			10:35 D	10:35 B	10:37 D	10:37 B	11:22 D	11:22 B	11:30 D	11:30 B
			12:15 D	12:15 B	12:17 D	12:17 B	14: 3 D	14: 3 B	14: 5 D	14: 5 B
			14:22 D	14:22 B	15:42 D	15:42 B	16: 6 D	16: 6 B	16: 7 D	16: 7 B
			16: 9 D	16: 9 B	17:16 D	17:16 B	18:21 D	18:21 B	18:41 D	18:41 B
			19: 1 D	19: 1 B	19:18 D	19:18 B				
0X	%LSPI		3:21 D	14:12						
0X	%MYID		3:20 D	7: 8	10:30	11: 4	13:29	13:32	14:11	14:30
			16:25							
0X	%PUNT		5: 4 D	5: 4	5:35 D	5:35	10:30 D	10:30	11: 4 D	11: 4
			11:23 D	11:23	13:29 D	13:29	14:30 D	14:30	16:25 D	16:25
			16:37 D	16:37						
0X	%TR1		6:33 D	6:33						
0X	%TR2		6:33 D	6:33						
0X	%TR3		6:33 D	6:33						
0X	%TR4		6:33 D	6:33						
0X	%TR5		6:33 D	6:33						
0X	%TRAC		6:33 D	6:33						
0X	%TRANQ		3:22 D	10:33						
0X	%TVNT		6:33 D	6:33						
42	AM$BDRSP	$APTEXT	13:41							
2	BP$DONE	$APTEXT	12:13							
0X	CA		3:41 D	19:15						
0X	CB		3:42 D	19:16						
0X	CC		4: 1 D	19:17						
0X	CE		4: 2 D	18:40						
20	CPI@DP@N	$APTEXT	14:12 F							
12	CPI@DP@P	$APTEXT	14:12 F	14:12						
0X	D$AMSG		3:23 D	14: 4						

0X	D$ICOM		3:24 D	5:16						
0X	D$TRANSF		3:25 D	10:36						
1000	D4$WPS	$APTEXT	9:34							
0X	D4ACI		3:34 D	18:21 D	18:21					
30	DA@@LI	$APTEXT	14:38							
10	DA@@LH	$APTEXT	14:37	18:36						
20	DA@DID@N	$APTEXT	14:16	14:16 F	14:17 F	15:26 F				
10	DA@DID@P	$APTEXT	14:16 F	14:16	14:17 F	14:17	15:26 F	15:26		
0	DA@DID@S	$APTEXT	14:16	14:16 F	14:17 F	15:26 F				
10	DA@DIU@N	$APTEXT	15: 2	15: 2 F						
10	DA@DIU@P	$APTEXT	15: 2 F	15: 2						
0	DA@DIU@S	$APTEXT	15: 2	15: 2 F						
20	DA@DLO@N	$APTEXT	19:16	19:16 F						
2	DA@DLO@P	$APTEXT	19:16 F	19:16						
0	DA@DLO@S	$APTEXT	19:16	19:16 F						
20	DA@DL1@N	$APTEXT	19:17	19:17 F						
3	DA@DL1@P	$APTEXT	19:17 F	19:17						
0	DA@DL1@S	$APTEXT	19:17	19:17 F						
20	DA@IFC@N	$APTEXT	6:18	6:18 F	11:24	11:24 F	12:26 F	13:41 F	16: 3 F	17:14 F
			18:37 F							
1	DA@IFC@P	$APTEXT	6:18 F	6:18	11:24 F	11:24	12:26 F	12:26	13:41 F	13:41
			16: 3 F	16: 3	17:14 F	17:14	18:37 F	18:37		
0	DA@IFC@S	$APTEXT	6:18	6:18 F	11:24	11:24 F	12:26 F	13:41 F	16: 3 F	17:14 F
			18:37 F							
10	DA@LEN@W	$APTEXT	6:17							
20	DA@SID@N	$APTEXT	14:15	14:15 F	14:18 F					
11	DA@SID@P	$APTEXT	14:15 F	14:15	14:18 F	14:18				
0	DA@SID@S	$APTEXT	14:15	14:15 F	14:18 F					
377	DA@SIL@M	$APTEXT	14:19 F	15:22 F						
10	DA@SIL@N	$APTEXT	14:19 F	15:22 F						
11	DA@SIL@P	$APTEXT	14:19 F	14:19	15:22 F	15:22				
10	DA@SIL@S	$APTEXT	14:19 F	15:22 F						
177400	DA@SIL@X	$APTEXT	14:19	15:22						
377	DA@SIU@M	$APTEXT	15:23							
10	DA@SIU@N	$APTEXT	15:23 F	15:23						
11	DA@SIU@P	$APTEXT	15:23 F	15:23						
0	DA@SIU@S	$APTEXT	15:23 F	15:23						
0X	DACI		3:26 D	10:37 D	10:37	14: 5 D	14: 5			
1	DAI$READ	$APTEXT	12: 9							
6	DAI$WBFR	$APTEXT	11:26	12:32						
14	DAI$WRBH	$APTEXT	10: 3	11:34	12:32					
2	DAI$WRIT	$APTEXT	9:41							

0X	DALWRT		3:27	D	18:41	D	18:41		
21	DAR$BKD	$APTEXT	7:24						
20	DAR$PAR	$APTEXT	10:18						
2	DAS$ABS	$APTEXT	17:36						
1	DAS$ID	$APTEXT	17:36						
2	DCB@DA@P	$APTEXT	10: 9						
4	DCB@DN@P	$APTEXT	12:16						
0X	DCCB		3:28	D	7:32		11: 8		
0X	DDALGETA		11:30	D	11:30		16: 6	D	16: 6
377	DE@RQN@M	$APTEXT	15:24						
10	DE@RQN@N	$APTEXT	15:24	F	15:24				
12	DE@RQN@P	$APTEXT	15:24	F	15:24				
0	DE@RQN@S	$APTEXT	15:24	F	15:24				
377	DE@SEQ@M	$APTEXT	15:27						
10	DE@SEQ@N	$APTEXT	15:27	F					
12	DE@SEQ@P	$APTEXT	15:27	F	15:27				
10	DE@SEQ@S	$APTEXT	15:27	F					
177400	DE@SEQ@X	$APTEXT	15:27						
377	DE@TOT@M	$APTEXT	15:28						
10	DE@TOT@N	$APTEXT	15:28	F	15:28				
13	DE@TOT@P	$APTEXT	15:28	F	15:28				
0	DE@TOT@S	$APTEXT	15:28	F	15:28				
377	DE@TYP@M	$APTEXT	15:25						
10	DE@TYP@N	$APTEXT	15:25	F					
13	DE@TYP@P	$APTEXT	15:25	F	15:25				
10	DE@TYP@S	$APTEXT	15:25	F					
177400	DE@TYP@X	$APTEXT	15:25						
0X	DGIVEUP		3:29	D	5:17	D	5:17		
14	DK@DLO@P	$APTEXT	11:21		12:14				
20	DK@DPT@N	$APTEXT	17:34		17:34	F			
61	DK@DPT@P	$APTEXT	17:34	F	17:34				
0	DK@DPT@S	$APTEXT	17:34		17:34	F			
20	DK@RAC@N	$APTEXT	13:12	F					
5	DK@RAC@P	$APTEXT	13:12	F	13:12				
0	DK@RAC@S	$APTEXT	13:12	F					
37	DK@RAH@M	$APTEXT	13:11						
5	DK@RAH@N	$APTEXT	13:11	F	13:11				
6	DK@RAH@P	$APTEXT	13:11	F	13:11				
2	DK@RAH@S	$APTEXT	13:11	F	13:11				
777	DK@RAS@M	$APTEXT	13:10						
11	DK@RAS@N	$APTEXT	13:10	F					
6	DK@RAS@P	$APTEXT	13:10	F	13:10				

7	DK@RAS@S	$APTEXT	13:10	F				
177000	DK@RAS@X	$APTEXT	13:10					
3	DK@RAU@M	$APTEXT	13:13					
2	DK@RAU@N	$APTEXT	13:13	F	13:13			
6	DK@RAU@P	$APTEXT	13:13	F	13:13			
0	DK@RAU@S	$APTEXT	13:13	F	13:13			
37	DK@TPC@M	$APTEXT	17:42					
5	DK@TPC@N	$APTEXT	17:42		17:42	F		
60	DK@TPC@P	$APTEXT	17:42	F	17:42			
2	DK@TPC@S	$APTEXT	17:42		17:42	F		
37	DK@TYP@M	$APTEXT	7:36					
5	DK@TYP@N	$APTEXT	7:36		7:36	F		
7	DK@TYP@P	$APTEXT	7:36	F	7:36			
2	DK@TYP@S	$APTEXT	7:36		7:36	F		
2	DK@UPC@N	$APTEXT	7:39		7:39	F		
60	DK@UPC@P	$APTEXT	7:39	F	7:39			
0	DK@UPC@S	$APTEXT	7:39		7:39	F		
1	DKF@ER@M	$APTEXT	12: 5	F				
1	DKF@ER@N	$APTEXT	12: 5	F				
0	DKF@ER@P	$APTEXT	12: 5	F				
10	DKF@ER@S	$APTEXT	12: 5	F				
177577	DKF@ER@X	$APTEXT	12: 5					
20	DL@BM0@N	$APTEXT	14:35		14:35	F	16:27	16:27 F
34	DL@BM0@P	$APTEXT	14:35	F	14:35		16:27 F	16:27
0	DL@BM0@S	$APTEXT	14:35		14:35	F	16:27	16:27 F
20	DL@BM1@N	$APTEXT	14:36		14:36	F	16:28	16:28 F
35	DL@BM1@P	$APTEXT	14:36	F	14:36		16:28 F	16:28
0	DL@BM1@S	$APTEXT	14:36		14:36	F	16:28	16:28 F
177	DL@CHN@M	$APTEXT	6:19					
7	DL@CHN@N	$APTEXT	6:19					
27	DL@CHN@P	$APTEXT	6:19	F	6:19			
11	DL@CHN@S	$APTEXT	6:19		6:19	F		
13	DL@CYL@N	$APTEXT	9:12		9:12	F	12:37	12:37 F
30	DL@CYL@P	$APTEXT	9:12	F	9:12		12:37 F	12:37
0	DL@CYL@S	$APTEXT	9:12		9:12	F	12:37	12:37 F
1	DL@DIA@N	$APTEXT	6:21		6:21	F		
15	DL@DIA@P	$APTEXT	6:21	F	6:21			
0	DL@DIA@S	$APTEXT	6:21		6:21	F		
20	DL@ERC@N	$APTEXT	17:17	F				
17	DL@ERC@P	$APTEXT	17:17	F	17:17			
0	DL@ERC@S	$APTEXT	17:17	F				
377	DL@LCOM	$APTEXT	10: 3	F				

10	DL@FC@N	$APTEXT	6:20	6:20 F	10: 3 F	10: 3
26	DL@FC@P	$APTEXT	6:20 F	6:20	10: 3 F	10: 3
0	DL@FC@S	$APTEXT	6:20	6:20 F	10: 3 F	10: 3
37	DL@HED@M	$APTEXT	9:15	12:36		
5	DL@HED@N	$APTEXT	9:15	12:36		
30	DL@HED@P	$APTEXT	9:15 F	9:15	12:36 F	12:36
13	DL@HED@S	$APTEXT	9:15	9:15 F	12:36	12:36 F
3	DL@IOP@M	$APTEXT	6:26			
2	DL@IOP@N	$APTEXT	6:26	6:26 F		
27	DL@IOP@P	$APTEXT	6:26 F	6:26		
5	DL@IOP@S	$APTEXT	6:26	6:26 F		
1777	DL@LLN@M	$APTEXT	9:38			
12	DL@LLN@N	$APTEXT	9:38 F			
15	DL@LLN@P	$APTEXT	9:38 F	9:38		
6	DL@LLN@S	$APTEXT	9:38 F			
176000	DL@LLN@X	$APTEXT	9:38			
20	DL@LN0@N	$APTEXT	8: 3	8: 3 F		
32	DL@LN0@P	$APTEXT	8: 3 F	8: 3		
0	DL@LN0@S	$APTEXT	8: 3	8: 3 F		
20	DL@LN1@N	$APTEXT	8: 4	8: 4 F		
33	DL@LN1@P	$APTEXT	8: 4 F	8: 4		
0	DL@LN1@S	$APTEXT	8: 4	8: 4 F		
777	DL@OFF@M	$APTEXT	9:24			
11	DL@OFF@N	$APTEXT	9:24			
31	DL@OFF@P	$APTEXT	9:24 F	9:24		
7	DL@OFF@S	$APTEXT	9:24	9:24 F		
377	DL@RC@M	$APTEXT	7:24 F	10:18 F		
10	DL@RC@N	$APTEXT	7:24 F	10:18 F		
26	DL@RC@P	$APTEXT	7:24 F	7:24	10:18 F	10:18
10	DL@RC@S	$APTEXT	7:24 F	10:18 F		
177400	DL@RC@X	$APTEXT	7:24	10:18		
20	DL@RCC@N	$APTEXT	9:42	9:42 F		
36	DL@RCC@P	$APTEXT	9:42 F	9:42		
0	DL@RCC@S	$APTEXT	9:42	9:42 F		
7	DL@SEC@N	$APTEXT	9:18	9:18 F	12:35	12:35 F
31	DL@SEC@P	$APTEXT	9:18 F	9:18	12:35 F	12:35
0	DL@SEC@S	$APTEXT	9:18	9:18 F	12:35	12:35 F
20	DL@SEQ@N	$APTEXT	15: 3	15: 3 F		
12	DL@SEQ@P	$APTEXT	15: 3 F	15: 3		
0	DL@SEQ@S	$APTEXT	15: 3	15: 3 F		
37	DL@SUB@M	$APTEXT	6:22			
5	DL@SUB@N	$APTEXT	6:22	6:22 F		

15	DL@SUB@P	$APIEXT	6:22 F	6:22				
1	DL@SUB@S	$APIEXT	6:22	6:22 F				
3	DL@IDL@M	$APIEXT	12:13 F					
2	DL@IDL@N	$APIEXT	12:13 F	12:13				
21	DL@IDL@P	$APIEXT	12:13 F	12:13				
7	DL@IDL@S	$APIEXT	12:13 F	12:13				
37777	DL@IMO@M	$APIEXT	8:20	8:35				
16	DL@IMO@N	$APIEXT	8:20	8:35 F				
24	DL@IMO@P	$APIEXT	8:20 F	8:20	8:35 F	8:35		
2	DL@IMO@S	$APIEXT	8:20	8:20 F	8:35 F			
140000	DL@IMO@X	$APIEXT	8:35					
20	DL@IM1@N	$APIEXT	8:21	8:21 F	8:36 F			
25	DL@IM1@P	$APIEXT	8:21 F	8:21	8:36 F	8:36		
0	DL@IM1@S	$APIEXT	8:21	8:21 F	8:36 F			
2	DL@IM@N	$APIEXT	6:23	6:23 F				
24	DL@IM@P	$APIEXT	6:23 F	6:23				
0	DL@IM@S	$APIEXT	6:23	6:23 F				
20	DL@IOI@N	$APIEXT	9:39 F	12:40	12:40 F	15: 6	15: 6 F	
14	DL@IOI@P	$APIEXT	9:39 F	9:39	12:40 F	12:40	15: 6 F	15: 6
0	DL@IOI@S	$APIEXT	9:39 F	12:40	12:40 F	15: 6	15: 6 F	
5	DL@IYP@N	$APIEXT	7:37	7:37 F				
27	DL@IYP@P	$APIEXT	7:37 F	7:37				
0	DL@IYP@S	$APIEXT	7:37	7:37 F				
4	DL@UNS@N	$APIEXT	13:38	13:38 F				
20	DL@UNS@P	$APIEXT	13:38 F	13:38				
0	DL@UNS@S	$APIEXT	13:38	13:38 F				
3	DL@UNI@M	$APIEXT	6:27	12:38				
2	DL@UNI@N	$APIEXT	6:27	6:27 F	12:38	12:38 F		
27	DL@UNI@P	$APIEXT	6:27 F	6:27	12:38 F	12:38		
7	DL@UNI@S	$APIEXT	6:27	6:27 F	12:38	12:38 F		
0X	DMGA		3:30 D	17:16 D	17:16			
20	DP@ACY@N	$APIEXT	18: 3	18: 3 F				
10	DP@ACY@P	$APIEXT	18: 3 F	18: 3				
0	DP@ACY@S	$APIEXT	18: 3	18: 3 F				
777	DP@ASE@M	$APIEXT	17:37					
11	DP@ASE@N	$APIEXT	17:37					
7	DP@ASE@P	$APIEXT	17:37 F	17:37				
7	DP@ASE@S	$APIEXT	17:37	17:37 F				
20	DP@DCY@N	$APIEXT	18: 5	18: 5 F				
6	DP@DCY@P	$APIEXT	18: 5 F	18: 5				
0	DP@DCY@S	$APIEXT	18: 5	18: 5 F				
777	DP@DSE@M	$APIEXT	17:39					

11	DP@DSE@N	$APTEXT	17:39														
5	DP@DSE@P	$APTEXT	17:39	F	17:39												
7	DP@DSE@S	$APTEXT	17:39		17:39	F											
0X	DS		4: 3	D	11:29		16: 5		17:15		18:39						
0X	DSUM		16: 7	D	16: 7												
0X	DT		4: 4	D	18:20												
0X	DY		4: 5	D	14: 2		14:21		15:40								
0X	DZ		4: 6	D	14: 1												
0X	ED		4: 7	D	5:33		10: 9		10:33		11:21		12:14		12:16		
0X	EDQU		11:22	D	11:22		12:15	D	12:15								
0X	EH		4: 8	D	5:35		5:36		10:10		10:34		11:23				
0X	EI		4: 9	D	16: 8		18:42										
0X	EMSGJOP		3:31	D	16: 9	D	16: 9		19: 1	D	19: 1						
0X	ENQU		3:32	D	10:11	D	10:11		10:35	D	10:35		12:17	D	12:17		
0X	ESWP		3:33	D	5:38	D	5:38										
0X	ETOQ		5:34	D	5:34												
26	F$GDAL	$APTEXT	5:25		15: 8												
46	F$MOSR	$APTEXT	6:17		15:21												
36	F$MPUT	$APTEXT	16:29														
7	F$PAUSE	$APTEXT	5:29														
27	F$RDAL	$APTEXT	10:22		12: 2		13:21		15:15		16:10		16:18		16:30		
0	F$SWAIT	$APTEXT	6:17		15:21												
0X	HPUT		3:35	D	14: 3	D	14: 3										
4	ICOM$LIM	$APTEXT	5:14														
0X	IDALSND		3:36	D	14:22	D	14:22		15:42	D	15:42						
6	IM$ABRT	$APTEXT	10:28														
11	IM$ARSP	$APTEXT	11: 2		11:17												
7	IM$ERRR	$APTEXT	14:29														
4	IM$RESP	$APTEXT	11: 2		12:26												
10	IM$RLBF	$APTEXT	16: 3		16:24												
2	IM$RLDL	$APTEXT	16:16		18:37												
5	IM$STAT	$APTEXT	11:28		11:36		13:28		17:14								
1	IM$STIO	$APTEXT	7: 7														
3	IM$TRAN	$APTEXT	10:28														
0X	IMDAL		3:37	D	18:35												
0X	MIAQ		3:38	D	5: 2		6: 8										
0X	MIMDALQ		3:39	D	14: 1												
0	MIOP	$APTEXT	7: 8		13:29		14:30		16:25								
77	MSK$SSD	$APTEXT	8:15		8:30												
0X	OSICOM		1: 4	D	1: 4												
17	OV@NUM@N	$APTEXT	1: 4														
1	OV@TYP@N	$APTEXT	1: 4														

50	PI$DD49	$APIEXT	5:35	10:30	11: 4	11:23	13:29	14:30	16:25	16:37
20	PI$MSGCH	$APIEXT	5: 4							
0X	PUIDAL		3:40 D	19:18 D	19:18					
4	QU$POP	$APIEXT	5:21							
41461	RQ$CPU	$APIEXT	15:26							
101	RQ$DISK	$APIEXT	14:19							
103	RQ$ERRR	$APIEXT	15:22							
11	SM@IST@N	$APIEXT	1: 4							
7	SM@NUM@N	$APIEXT	1: 4							
51	TF$ICOM	$APIEXT	6:33							
3	TM$MAX	$APIEXT	7:18							
2	TM$SSD	$APIEXT	7:23	8:14	8:29					
20	TM@FW0@N	$APIEXT	8:25	8:25 F						
1	TM@FW0@P	$APIEXT	8:25 F	8:25						
0	TM@FW0@S	$APIEXT	8:25	8:25 F						
20	TM@FW1@N	$APIEXT	8:26	8:26 F						
2	TM@FW1@P	$APIEXT	8:26 F	8:26						
0	TM@FW1@S	$APIEXT	8:26	8:26 F						
20	TM@IOP@N	$APIEXT	10:29	10:29 F	11: 3	11: 3 F				
5	TM@IOP@P	$APIEXT	10:29 F	10:29	11: 3 F	11: 3				
0	TM@IOP@S	$APIEXT	10:29	10:29 F	11: 3	11: 3 F				
20	TM@LMO@N	$APIEXT	8:41	8:41 F						
3	TM@LMO@P	$APIEXT	8:41 F	8:41						
0	TM@LMO@S	$APIEXT	8:41	8:41 F						
20	TM@LM1@N	$APIEXT	8:42	8:42 F						
4	TM@LM1@P	$APIEXT	8:42 F	8:42						
0	TM@LM1@S	$APIEXT	8:42	8:42 F						
0X	TMCB		4:10 D	6:24						
20	TR@DON@N	$APIEXT	11:11	11:11 F						
13	TR@DON@P	$APIEXT	11:11 F	11:11						
0	TR@DON@S	$APIEXT	11:11	11:11 F						
20	TR@MDL@N	$APIEXT	11: 7	11: 7 F						
11	TR@MDL@P	$APIEXT	11: 7 F	11: 7						
0	TR@MDL@S	$APIEXT	11: 7	11: 7 F						
20	TR@TOI@N	$APIEXT	11:12	11:12 F						
14	TR@TOI@P	$APIEXT	11:12 F	11:12						
0	TR@TOI@S	$APIEXT	11:12	11:12 F						
0	TRA@QU@P	$APIEXT	10:33							

SYMBOL QUALIFIER IS ICOM

2	$CONST	7:24 D	7:24 F	7:24	10: 3 D	10: 3 F	10: 3	10:18 D	10:18 F	

		10:18	12: 5 D	12: 5 F	12:13 D	12:13 F	12:13	12:26 D	12:26 F
		12:26	13:41 D	13:41 F	13:41	14:19 D	14:19 F	14:19	15:22 D
		15:22 F	15:22	15:26 D	15:26 F	15:26	16: 3 D	16: 3 F	16: 3
		17:14 D	17:14 F	17:14	17:17 D	17:17 F	18:37 D	18:37 F	18:37
471	$REGORG	2:10 D	2:10 F	2:11	2:11 D	2:12	2:12 D	2:13	2:13 D
		2:14	2:14 D	2:15	2:15 D	2:15 F	2:16	2:16 D	2:17
		2:17 D	2:18	2:18 D	2:19	2:19 D	2:20	2:20 D	2:21
		2:21 D	2:21 F	2:22	2:22 D	2:23	2:23 D	2:24	2:24 D
		2:25	2:25 D	2:26	2:26 D	2:26 F	2:27	2:27 D	2:28
		2:28 D	2:29	2:29 D	2:30	2:30 D	2:31	2:31 D	2:32
		2:32 D	2:33	2:33 D	2:34	2:34 D	2:35	2:35 D	2:36
		2:36 D	2:37	2:37 D	2:38	2:38 D	2:39	2:39 D	2:40
		2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2
		3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D	3: 6
		3: 6 D	3: 7	3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3:10
		3:10 D	3:11	3:11 D	3:12	3:12 D	3:13	3:13 D	3:14
		3:14 D	3:15	3:15 D	3:16	3:16 D	3:17	3:17 D	3:17 F
		5:25	5:29	6:17	10:22	12: 2	13:21	15: 8	15:15
		15:21	16:10	16:18	16:29	16:30			
0	%NP	1: 4	2:26 D						
430	%P	1: 4	2:26 D	2:26					
410	%S1	2:10	2:11 D	5: 2	5:21	6:18	6:19	6:20	6:21
		6:22	6:23	6:24	6:25	6:26	6:27	7:24	7:36
		7:37	7:39	8: 3	8: 4	8:10	8:11	8:15	8:16
		8:20	8:21	8:25	8:26	8:30	8:31	8:35	8:36
		8:41	8:42	9:12	9:15	9:18	9:24	9:38	9:39
		9:42	10: 3	10:18	10:29	11: 3	11: 7	11:11	11:12
		11:19	11:24	12:13	12:26	12:35	12:36	12:37	12:38
		12:40	13:10	13:11	13:12	13:13	13:38	13:41	14:12
		14:15	14:16	14:17	14:18	14:19	14:35	14:36	15: 2
		15: 3	15: 6	15: 7	15:22	15:23	15:24	15:25	15:26
		15:27	15:28	16: 3	16:27	16:28	17:14	17:17	17:34
		17:37	17:39	17:42	18: 3	18: 5	18:37	19:16	19:17
411	%S2	2:10	2:12 D						
412	%S3	2:10	2:13 D						
413	%S4	2:10	2:14 D						
414	%S5	2:10	2:15 D						
415	%I1	2:16 D	9:31	15:20	18:36				
416	%I2	2:17 D	15:20	18:36					
417	%I3	2:18 D	18:36						
420	%I4	2:19 D							
421	%I5	2:20 D							

422	%I6	2:21 D							
423	%W1	2:22 D	6:11	6:16	6:17	7:36	7:38	8:20	8:27
		8:35	8:40	9: 1	9: 4	9:25	9:27	11:24	11:28
		11:36	12:40	12:41	13:38	14:15	14:17	17:34	17:37
		17:39	18: 3	18: 5	18:35	18:36	18:37	18:39	
424	%W2	2:23 D	6:13	6:16	6:17	8:21	8:27	8:30	8:36
		8:40	9: 4	14:16	14:18				
425	%W3	2:24 D	6:15	6:16	8:25	8:27	8:41	9: 1	9: 4
426	%W4	2:25 D	8:26	8:27	8:42	9: 4			
427	%W5	2:26 D	6: 9	6:10	6:11	6:12	6:13	6:26	6:28
		6:30	6:33						
2432	ACTDEM	10:12	13:22	18:18 D					
444	CHAN	2:39 D	6:19	6:30	7:13	7:32	11: 8		
462	CPD	3:11 D	9:13	18: 3	18: 5				
436	CYL	2:33 D	9:12	12:37	13: 6	13:12			
432	DAL	2:29 D	5:25	6:17	6:18	6:19	6:20	6:21	6:22
		6:23	6:26	6:27	6:33	7:24	7:37	8: 3	8: 4
		8:20	8:21	8:35	8:36	9:12	9:15	9:18	9:24
		9:38	9:39	9:42	10: 3	10:10	10:18	10:22	10:34
		11: 6	11: 7	11:23	11:24	11:29	12: 2	12:13	12:26
		12:35	12:36	12:37	12:38	12:40	13:38	13:41	14: 2
		14:15	14:16	14:17	14:18	14:19	14:21	14:35	14:36
		14:41	15: 2	15: 3	15: 6	15:11	15:38	16: 3	16: 5
		16:10	16:18	16:27	16:28	16:30	17:14	17:15	17:17
		18:36	19:16	19:17					
441	DCB	2:36 D	7:33	7:34	7:36	7:39	11: 9	11:10	11:21
		12: 5	12:14	13:10	13:11	13:12	13:13	17:34	17:42
		18:20							
465	DIA	3:14 D	6:21	9:41	15: 4	15:24	18: 2		
442	DONE	2:37 D	11:11	12:11	12:25	12:30			
454	DTR	3: 5 D	11: 6	11: 7	11:11	11:12	13:21	15: 8	15:13
		15:15	15:20	15:21	15:22	15:23	15:24	15:25	15:26
		15:27	15:28	15:30	15:31				
435	ENC	2:32 D	6:20	6:33	9:41	11:19	11:26	11:34	12: 9
		12:32							
437	HED	2:34 D	9:15	12:36	13: 2	13: 5	13:11		
456	IFC	3: 7 D	6:18	6:33	7: 7	10:28	11: 2	11:17	13:28
		14:29	16:16	16:24					
434	IOP	2:31 D	6: 6	6: 8	13:32	14:11	16: 8	18:42	
445	LEN0	2:40 D	8: 3	8: 6	8:10	8:40	9:27	9:30	9:31
		14:38	15:21	15:29					
446	LEN1	2:41 D	8: 4	8: 6	8:15	8:40	9:27	9:28	9:29

		9:31	9:32						
2352	LIMITS	9:11	12:39	17:33 D					
443	LLNG	2:38 D	9:28	9:34	9:35	9:36	9:37	9:38	
460	MOSL	3: 9 D	14:36	15: 1	16:28	16:29			
457	MOSU	3: 8 D	14:35	14:42	16:27	16:29			
433	MSG	2:30 D	5:36	6: 6	6:14	6:33	15: 2	15:23	
17	NEXT	5:11 D	5:18	10:13	14: 7	16:40			
431	NUMDONE	2:28 D	5: 5	5:14	5:15	5:42			
453	OFF	3: 4 D	9:24	9:25	9:37	14:37	15:20	15:21	
1037	PARAM1	7:25	10:19 D						
1027	PARAMERR	7:14	7:18	7:27	7:35	7:38	7:41	8: 7	8:11
		8:16	8:31	9: 2	9: 5	9:13	9:16	9:19	10:17 D
430	QUE	2:27 D	5: 3	5:21	5:33				
2210	RELBMBUF	15:17	16: 2 D						
2506	RELBMDAL	13:33	16:17	19:13 D					
2442	RLDAL	10:31	11: 5	13:35	14:31	16:26	18:34 D		
447	SC0	2:42 D	14:40	15: 7	15:11	15:12	15:13	15:14	15:27
		15:32	15:38	15:39	15:40	15:41			
450	SC1	3: 1 D	14:41	15:30	15:31				
451	SC2	3: 2 D	14:42	15:21	15:29				
452	SC3	3: 3 D	15: 1	15:21	15:29				
440	SEC	2:35 D	9:18	9:25	9:32	12:35	12:41	12:42	13: 1
		13:10							
464	SP1	3:13 D	9:19	12:42	13: 1	17:37	17:39		
6	START	5: 1 D	5:30						
1535	STAT	7: 8	13:30 D						
2333	STATUS	10:21	11:33	11:37	11:41	12:34	17:13 D		
466	SUB	3:15 D	6:22	15: 3	15: 4	15: 5	15:25	17:36	
467	TM	3:16 D	6:23	6:24	7:18	7:23	8:14	8:29	
470	TMP	3:17 D	6:25	7:21	8:25	8:26	8:41	8:42	10:29
		11: 3							
455	TOTAL	3: 6 D	9:32	9:36	9:39	11:12	12:11	12:30	15: 6
		15: 7	15:28						
463	TPC	3:12 D	9:16	13: 4	17:42				
461	UNIT	3:10 D	6:27	6:28	7:40	12:38	13:13		

```
0     <macro>                    OVERLAY   BCOM                                                    BCOM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
*****************************************************************    BCOM.5
*                                                               *    BCOM.6
*  THE BCOM ROUTINE HANDLES ALL MESSAGE TRAFFIC FOR THE BLOCK MUX *    BCOM.7
*  TAPE SUBSYSTEM. MESSAGES ARE TAKEN OFF THE INPUT MESSAGE QUEUES. *    BCOM.8
*  THE MESSAGE POINTS TO THE DAL IN MOS WHICH IS THEN BROUGHT IN *    BCOM.9
*  TO LOCAL MEMORY. THE DAL CONTAINS A FUNCTION CODE WHICH DETERMINES *    BCOM.10
*  WHAT ACTION BCOM TAKES.                                       *    BCOM.11
*                                                               *    BCOM.12
*  FUNCTION CODES:                                               *    BCOM.13
*                                                               *    BCOM.14
*        MC$MSTIO - INITIATE EXECUTION OF REQUEST RECEIVED FROM *    BCOM.15
*                   THE CPU. BCOM FIRST LOOKS AT THE DEVICE CODE *    BCOM.16
*                   TO DETERMINE WHETHER THE DEVICE IS ACTIVE.  *    BCOM.17
*                   IF IT IS, BCOM LOOKS AT THE FUNCTION TO BE  *    BCOM.18
*                   EXECUTED. IF THE FUNCTION IS A READ OR WRITE, *    BCOM.19
*                   THE REQUEST IS CHECKED FOR  WHETHER OR NOT  *    BCOM.20
*                   DATA IS TO BE TRANSFERRED BETWEEN THE IOP AND *    BCOM.21
*                   THE CPU. IF THERE IS, THE REQUEST IS SENT TO *    BCOM.22
*                   'BYPASS', IF NOT, IT IS SENT TO 'TEX'.      *    BCOM.23
*                   IF THE DEVICE IS NOT ACTIVE, BCOM CREATES AN *    BCOM.24
*                   ACTIVITY AND SENDS THE REQUEST TO 'BMXOPE'. *    BCOM.25
*                                                               *    BCOM.26
*        MC$CPURD - INITIATE DATA TRANSFER FROM THE CPU TO MOS. *    BCOM.27
*                   REQUEST IS DIRECTED TO 'TDEM1'.             *    BCOM.28
*                                                               *    BCOM.29
*        MC$CPUWR - INITIATE DATA TRANSFER FROM MOS TO THE CPU. *    BCOM.30
*                   REQUEST IS DIRECTED TO 'TDEM1'.             *    BCOM.31
*                                                               *    BCOM.32
*        MC$CPRDN - RESPONSE FROM BIOP (TDEM1) THAT DATA HAS ALL *    BCOM.33
*                   BEEN TRANSFERRED FROM CPU TO MOS. A RESPONSE *    BCOM.34
*                   IS SENT TO THE CPU.                         *    BCOM.35
*                                                               *    BCOM.36
*        MC$CPWDN - RESPONSE FROM BIOP (TDEM1) THAT DATA HAS ALL *    BCOM.37
*                   BEEN TRANSFERRED FROM MOS TO CPU. A RESPONSE *    BCOM.38
*                   IS SENT TO THE CPU FOR DATA TRANSFERRED THAT *    BCOM.39
*                   HAS BEEN VERIFIED AS ERROR FREE.            *    BCOM.40
*                                                               *    BCOM.41
*        MC$CSTAT - REQUEST FROM XIOP TO MIOP TO SEND A RESPONSE TO *    BCOM.42
*                   THE CPU.                                    *    BCOM.43
*                                                               *    BCOM.44
*        MC$CSTRL - REQUEST FROM XIOP TO MIOP TO SEND AN ERROR  *    BCOM.45
*                   PACKET TO THE CPU.                          *    BCOM.46
```

```
                    *                                                          *    BCOM.47
                    *        MC$RLDAL - REQUEST FROM ONE IOP TO ANOTHER TO RELEASE THE    *    BCOM.48
                    *                   MOS DAL.                                          *    BCOM.49
                    *                                                                     *    BCOM.50
                    *        MC$CPMLT - REQUEST FROM XIOP TO SEND MULTIPLE PACKETS TO     *    BCOM.51
                    *                   THE CPU.                                          *    BCOM.52
                    *                                                                     *    BCOM.53
                    *        MC$RLMOS - REQUEST FROM MIOP TO RELEASE A MOS BUFFER THAT    *    BCOM.54
                    *                   IS OWNED BY XIOP.                                 *    BCOM.55
                    *                                                                     *    BCOM.56
                    ***********************************************************************    BCOM.57

6   <macro>                     REGDEFS   ,,(X0,X1,X2,X3,X4,X5,RSC,RBC,TSC,TBC,TCB,STAT,VMS,      BCOM.59
                    ,                     DAL,MSG,IOP,FCT,FUN,QUTBLE,QUPT,QUE,NUMDONE,DVC,FLG,   BCOM.60
                    ,                     MSU,MSL,LEN,DPT,TOT,PRI,OFF,DIA,TID,REQ,SLG)           BCOM.61
            410     %S1       EQUALS    $REGORG                                                $REG    3
            411     %S2       EQUALS    $REGORG                                                $REG    3
            412     %S3       EQUALS    $REGORG                                                $REG    3
            413     %S4       EQUALS    $REGORG                                                $REG    3
            414     %S5       EQUALS    $REGORG                                                $REG    3
            415     %T1       EQUALS    $REGORG                                                $REG    3
            416     %T2       EQUALS    $REGORG                                                $REG    3
            417     %T3       EQUALS    $REGORG                                                $REG    3
            420     %T4       EQUALS    $REGORG                                                $REG    3
            421     %T5       EQUALS    $REGORG                                                $REG    3
            422     %T6       EQUALS    $REGORG                                                $REG    3
            423     %W1       EQUALS    $REGORG                                                $REG    3
            424     %W2       EQUALS    $REGORG                                                $REG    3
            425     %W3       EQUALS    $REGORG                                                $REG    3
            426     %W4       EQUALS    $REGORG                                                $REG    3
            427     %W5       EQUALS    $REGORG                                                $REG    3
            430     X0        EQUALS    $REGORG                                                $REG    3
            431     X1        EQUALS    $REGORG                                                $REG    3
            432     X2        EQUALS    $REGORG                                                $REG    3
            433     X3        EQUALS    $REGORG                                                $REG    3
            434     X4        EQUALS    $REGORG                                                $REG    3
            435     X5        EQUALS    $REGORG                                                $REG    3
            436     RSC       EQUALS    $REGORG                                                $REG    3
            437     RBC       EQUALS    $REGORG                                                $REG    3
            440     TSC       EQUALS    $REGORG                                                $REG    3
            441     TBC       EQUALS    $REGORG                                                $REG    3
```

```
442     TCB       EQUALS    $REGORG                                          $REG     3
443     STAT      EQUALS    $REGORG                                          $REG     3
444     VMS       EQUALS    $REGORG                                          $REG     3
445     DAL       EQUALS    $REGORG                                          $REG     3
446     MSG       EQUALS    $REGORG                                          $REG     3
447     IOP       EQUALS    $REGORG                                          $REG     3
450     FCT       EQUALS    $REGORG                                          $REG     3
451     FUN       EQUALS    $REGORG                                          $REG     3
452     QUTBLE    EQUALS    $REGORG                                          $REG     3
453     QUPT      EQUALS    $REGORG                                          $REG     3
454     QUE       EQUALS    $REGORG                                          $REG     3
455     NUMDONE   EQUALS    $REGORG                                          $REG     3
456     DVC       EQUALS    $REGORG                                          $REG     3
457     FLG       EQUALS    $REGORG                                          $REG     3
460     MSU       EQUALS    $REGORG                                          $REG     3
461     MSL       EQUALS    $REGORG                                          $REG     3
462     LEN       EQUALS    $REGORG                                          $REG     3
463     DPT       EQUALS    $REGORG                                          $REG     3
464     TOT       EQUALS    $REGORG                                          $REG     3
465     PRI       EQUALS    $REGORG                                          $REG     3
466     OFF       EQUALS    $REGORG                                          $REG     3
467     DIA       EQUALS    $REGORG                                          $REG     3
470     TID       EQUALS    $REGORG                                          $REG     3
471     REQ       EQUALS    $REGORG                                          $REG     3
472     SLG       EQUALS    $REGORG                                          $REG     3
                                                                         BCOM.62

                  EXT       %ACTIVE                                        BCOM.64
                  EXT       %MYID                                          BCOM.65
                  EXT       BBDISK                                         BCOM.66
                  EXT       BXQQ                                           BCOM.67
                  EXT       CA                                             BCOM.68
                  EXT       CB                                             BCOM.69
                  EXT       CC                                             BCOM.70
                  EXT       CE                                             BCOM.71
                  EXT       D$BCOM                                         BCOM.72
                  EXT       DATQU                                          BCOM.73
                  EXT       DS                                             BCOM.74
                  EXT       DY                                             BCOM.75
                  EXT       DZ                                             BCOM.76
                  EXT       ED                                             BCOM.77
```

```
EXT     EH                                                      BCOM.78
EXT     EI                                                      BCOM.79
EXT     MBAQ                                                    BCOM.80
EXT     MBBQ                                                    BCOM.81
EXT     MBCQ                                                    BCOM.82
EXT     MBDQ                                                    BCOM.83
EXT     O$BMXOPE                                                BCOM.84
EXT     O$BMXSIO                                                BCOM.85
EXT     O$BUFMAN                                                BCOM.86
EXT     O$CONMAN                                                BCOM.87
EXT     TDMQ                                                    BCOM.88
EXT     TIMQU                                                   BCOM.89
EXT     XDEV                                                    BCOM.90
EXT     XDEVMAX                                                 BCOM.91
```

```
                                 BLOCK TABLE                                                    BCOM.93
                        ************************************************************************ BCOM.94
                                                                                                 BCOM.95
                                 *                                                               BCOM.96
                                 * MESSAGE QUEUE TABLE ADDRESSES                                 BCOM.97
                                 *                                                               BCOM.98
                                                                                                 BCOM.99
                                 * MIOP                                                          BCOM.100
                                                                                                 BCOM.101
2420                    QTBLE0   *                                                               BCOM.102
2420   000000X                   MBDQ                          .XIOP                            BCOM.103
2421   000000                    0                                                              BCOM.104
                                                                                                 BCOM.105
                                 * BIOP/DIOP                                                    BCOM.106
                                                                                                 BCOM.107
2422                    QTBLE1   *                                                               BCOM.108
2422   000000X                   MBDQ                          .XIOP                            BCOM.109
2423   000000                    0                                                              BCOM.110
                                                                                                 BCOM.111
                                 * XIOP                                                         BCOM.112
                                                                                                 BCOM.113
2424                    QTBLE2   *                                                               BCOM.114
2424   000000X                   MBBQ                          .BIOP                            BCOM.115
2425   000000X                   MBCQ                          .DIOP                            BCOM.116
2426   000000X                   MBAQ                          .MIOP                            BCOM.117
2427   000000                    0                                                              BCOM.118
                                                                                                 BCOM.119
                        10       CHKPT    EQUALS    10         .# OF CONSECUTIVE REQUESTS TO DO  BCOM.120
                                                                                                 BCOM.121
                                 BLOCK *                                                        BCOM.122
```

```
                                        *                                                                   BCOM.124
                                        * INITIALIZE                                                       BCOM.125
                                        *                                                                   BCOM.126
                                                                                                            BCOM.127
  6                                BEGIN *                                                                 BCOM.128
  6    <macro>                          $IF       (R!%MYID = MIOP)     .Select queue table                BCOM.129
                                                                                          .>>>>> 0          $IF      1
 11   014000 /002420  024452              R!QUTBLE = QTBLE0            .MIOP                              BCOM.130
 14    <macro>                          $ELSEIF   (R!%MYID = BIOP)                                        BCOM.131
 21   014000 /002422  024452              R!QUTBLE = QTBLE1            .BIOP                              BCOM.132
 24    <macro>                          $ELSEIF   (R!%MYID = DIOP)                                        BCOM.133
 31   014000 /002422  024452              R!QUTBLE = QTBLE1            .DIOP                              BCOM.134
 34    <macro>                          $ELSEIF   (R!%MYID = XIOP)                                        BCOM.135
 41   014000 /002424  024452              R!QUTBLE = QTBLE2            .XIOP                              BCOM.136
 44    <macro>                          $ELSE                                                             BCOM.137
 45    <macro>                            $PUNTIF                                                         BCOM.138
 47    <macro>                          $ENDIF                                                            BCOM.139
                                                                                          .<<<<< 0          $ENDIF   1
 47   020452  022000  024453           R!QUPT = R!QUTBLE + R!%B       .ADD BASE                          BCOM.140
 52   010000  024455                   R!NUMDONE = 0                  .NUMBER PROCESSED                  BCOM.141
                                                                                                            BCOM.142
                                        *                                                                   BCOM.143
                                        * GET NEXT MESSAGE TO PROCESS                                     BCOM.144
                                        *                                                                   BCOM.145
                                                                                                            BCOM.146
 54                                NEXT *                                                                  BCOM.147
 54   020455  013010  126000           P = YIELD, R!NUMDONE = CHKPT   .IF TIME FOR PAUSE                 BCOM.148
     /002253
 60   002000  000000                   I = 0                          .LOCK                              BCOM.149
 62   030453  024454                   R!QUE = (R!QUPT)               .QUEUE TO PROCESS                  BCOM.150
 64    <macro>                         $IF (A = 0)                    .IF ALL DONE                       BCOM.151
                                                                                          .>>>>> 0          $IF      1
 66   020452  022000  024453              R!QUPT = R!QUTBLE + R!%B    .SCAN QUEUES FOR ANY NEW ONES      BCOM.152
 71   010001  024454                      R!QUE = 1                   .SET NON-ZERO                      BCOM.153
 73    <macro>                            $UNTIL (R!QUE = 0)          .UNTIL ALL SCANNED                 BCOM.154
 75   030453  024454                        R!QUE = (R!QUPT)          .QUEUE TO CHECK                    BCOM.155
 77    <macro>                              $IF (A # 0)               .IF QUEUE THERE                    BCOM.156
                                                                                          .>>>>> 1          $IF      1
101   030454  024430                          R!X0 = (R!QUE)          .QUEUE ADDRESS                     BCOM.157
103    <macro>                                $IF (A # 0)             .IF QUEUE ESTABLISHED              BCOM.158
                                                                                          .>>>>> 2          $IF      1
105   020430  012004  024410                    A = (R!X0 + QU$POP)   .COUNT OF ENTRIES                  BCOM.159
```

```
        030410
111     107035                          P = NEXT, A # 0        .IF ANY TO PROCESS                    BCOM.160
112      <macro>                          $ENDIF                                                       BCOM.161
                                                                                .<<<<< 2                 $ENDIF    1
112     026453                            R!QUPT = R!QUPT + 1    .NEXT                                 BCOM.162
113      <macro>                        $ENDIF                                                         BCOM.163
                                                                                .<<<<< 1                 $ENDIF    1
113      <macro>                      $ENDTIL                                                          BCOM.164
114      <macro>                      PKERN ESWP                .END BCOM                              BCOM.165
116      <macro>                    $ELSE                                                              BCOM.166
117     030454  024454               R!QUE = (R!QUE)           .QUEUE ADDRESS                         BCOM.167
121      <macro>                     $IF (A = 0)               .IF NO QUEUE                           BCOM.168
                                                                                .>>>>> 1                 $IF       1
123     026453                         R!QUPT = R!QUPT + 1     .CHECK NEXT QUEUE                      BCOM.169
124     071050                         P = NEXT                                                       BCOM.170
125      <macro>                     $ENDIF                                                           BCOM.171
                                                                                .<<<<< 1                 $ENDIF    1
125     020454  012004  024410       A = (R!QUE + QU$POP)      .COUNT OF ENTRIES LEFT                 BCOM.172
        030410
                                                                                                        BCOM.173
                                     * ENTRY ON QUEUE                                                   BCOM.174
                                                                                                        BCOM.175
131      <macro>                     $IF (A # 0)               .IF ANY THERE                          BCOM.176
                                                                                .>>>>> 1                 $IF       1
133      <macro>                       RKERN HOFF              .GET A LOCAL DAL TO READ IT IN TO      BCOM.177
135     020000  024445                 R!DAL = DY              .SAVE ADDRESS                          BCOM.178
137      <macro>                       $IF (A = 0)             .IF NONE                               BCOM.179
                                                                                .>>>>> 2                 $IF       1
141      <macro>                     PAUSE 1,A1=0,A2=-1     .DELAY                                   BCOM.180
151     071143                       P = BEGIN                                                        BCOM.181
152      <macro>                       $ENDIF                                                         BCOM.182
                                                                                .<<<<< 2                 $ENDIF    1
152     020454  024000                 ED = R!QUE              .TAKE NEXT MESSAGE OFF QUEUE           BCOM.183
154      <macro>                       RKERN ETOQ                                                     BCOM.184
156     020000  024446                 R!MSG = EH              .SAVE IT                               BCOM.185
                                                                                                        BCOM.186
                                       * CHECK FOR OVERFLOW OF QUEUE                                    BCOM.187
                                                                                                        BCOM.188
160     020454  012006  024410         EH = (R!QUE + 6)        .OVERFLOW MESSAGE CELL                 BCOM.189
        030410  024000
165      <macro>                       $IF (EH # 0)            .IF QUEUE HAD OVERFLOW                 BCOM.190
                                                                                .>>>>> 2                 $IF       1
```

```
170   020454  024000                    ED = R!QUE                  .QUEUE THE OVERFLOWED MESSAGE        BCOM.191
172    <macro>                          RKERN EPOQ                                                       BCOM.192
174   030453  024430                    R!X0 = (R!QUPT)             .RE-ENABLE INPUT CHANNEL             BCOM.193
176   020430  012003  024410            B = (R!X0 + 3)                                                   BCOM.194
      030410  054000
203   167000                            IOB:7                                                            BCOM.195
204   020454  012006  024410            (R!QUE + 6) = 0             .CLEAR OVERFLOW CELL                 BCOM.196
      010000  034410
211    <macro>                        $ENDIF                                                             BCOM.197
                                                                                   .<<<<< 2                $ENDIF   1
211    <macro>                      $ELSE                                                                BCOM.198
212   026453                          R!QUPT = R!QUPT + 1         .CHECK NEXT QUEUE                    BCOM.199
213   071137                          P = NEXT                                                         BCOM.200
214    <macro>                      $ENDIF                                                               BCOM.201
                                                                                   .<<<<< 1                $ENDIF   1
214    <macro>                    $ENDIF                                                                 BCOM.202
                                                                                   .<<<<< 0                $ENDIF   1
214   026455                      R!NUMDONE = R!NUMDONE + 1       .INCREMENT NUMBER PROCESSED          BCOM.203
                                                                                                         BCOM.204
                                                                                                         BCOM.205
                                  *                                                                      BCOM.206
                                  * PROCESS MESSAGE                                                     BCOM.207
                                  *                                                                      BCOM.208
                                                                                                         BCOM.209
215   003000                      I = 1                           .ALLOW INTERRUPTS                    BCOM.210
216   020446  004014  013004      R!IOP = R!MSG > 14 - MQ$BMX0 & 17  .IOP FROM WHICH DAL ORIGINAT      BCOM.211
      011017  024447
                                                                                                         BCOM.212
                                  * COMPUTE MOS-DAL ADDRESS                                              BCOM.213
                                                                                                         BCOM.214
223   020447  005002  016000      R!X0 = R!IOP < 2 + MBAQ + 1     .POINTER TO MOS BASE ADDRESS         BCOM.215
     /000000  012001  024430
231   020446  015000 /007777      R!X1 = R!MSG & 7777 < 3         .OFFSET FROM BASE                    BCOM.216
      005003  024431
236   030430  024432              R!X2 = (R!X0)                   .BASE ADDRESS                        BCOM.217
240   020430  012001  024410      R!X3 = (R!X0 + 1)                                                    BCOM.218
      030410  024433
245    <macro>                    DADD R!X2,R!X3,0,R!X1,R!X4,R!X5 .ADD OFFSET                          BCOM.219
255    <macro>                    MOSR R!X4,R!X5,R!DAL,10         .READ DAL FROM MOS                   BCOM.220
                                                                                                         BCOM.221
                                  * GET DAL FUNCTION CODE                                                BCOM.222
                                                                                                         BCOM.223
```

```
275    <macro>                    GET R!FCT,DA@IFC,R!DAL                                  R15472BG.1
302    <macro>                    $IF (R!FCT = MC$MSTIO)        .I/O REQUEST FROM CPU         BCOM.225
                                                                                .>>>>> 0        $IF      1
                                                                                                BCOM.226
                                  * PROCESS INCOMING REQUEST FROM THE CPU                       BCOM.227
                                                                                                BCOM.228
306    <macro>                    $PUNTIF (R!%MYID # XIOP)      .If wrong IOP                   BCOM.229
312   077000 /002106              R = MOSREL                    .RETURN THE MOS DAL             BCOM.230
                                                                                                BCOM.231
                                  * GET PARAMETERS FROM PACKET                                  BCOM.232
                                                                                                BCOM.233
314   077000 /002153              R = DALSTRIP                  .GET DAL PARAMETERS             BCOM.234
                                                                                                BCOM.236
                                  *********************                                         BCOM.237
                                  *       TRACE       *                                         BCOM.238
                                  *                   *                                         BCOM.239
                                  * 1. DEVICE         *                                         BCOM.240
                                  * 2. FUNCTION       *                                         BCOM.241
                                  * 3. REQUEST BLOCKS *                                         BCOM.242
                                  * 4. REQUEST SECTS  *                                         BCOM.243
                                  * 5. FLAGS          *                                         BCOM.244
                                  *********************                                         BCOM.245
                                                                                                BCOM.246
316    <macro>                    KRECORD TF$TEXO,(R!DVC,R!FUN,R!RBC,R!RSC,R!FLG),INT=ON        BCOM.247
                                                                                                BCOM.248
                                  * CHECK FOR CONFIGURATION REQUEST                              BCOM.249
                                  * PACKET FOLLOWS DIFFERENT ROUTE                               BCOM.250
                                                                                                BCOM.251
336    <macro>                    $IF (R!FUN = FC$CHNGE)                                        BCOM.252
                                                                                .>>>>> 1        $IF      1
342    <macro>                      STORE 0,TQ@STS,R!DAL        .CLEAR STATUS                 R16767RD.1
347   014000 /000000  024431        R!X1 = O$CONMAN                                           BCOM.253
352   010001  024432                R!X2 = 1                    .PRIORITY                     BCOM.254
354   020445  024433                R!X3 = R!DAL                .PASS THE DAL ADDRESS         BCOM.255
356   077000 /002353                R = CREATE                                                BCOM.256
360   071304                        P = NEXT                    .CHECK FOR MORE               BCOM.257
361    <macro>                    $ENDIF                                                        BCOM.258
                                                                                .<<<<< 1        $ENDIF   1
                                                                                                BCOM.259
                                  * DO RANGE CHECK ON REQUESTED DEVICE NUMBER                   BCOM.260
                                                                                                BCOM.261
361   014000 /000000  024410      A = (XDEVMAX)                 .MAX DEVICE ORDINAL           BCOM.262
```

```
        030410
365      <macro>                    $IF (A <= R!DVC)            .IF OUT OF RANGE                    BCOM.263
                                                                                   .>>>>> 1          $IF      1
373     014000 /177776  024430        R!X0 = ST@DAL@X           .RETURN PROTOCOL ERROR              BCOM.264
376     077000 /002305                R = BADDO                                                     BCOM.265
400     077000 /002143                R = DALRET                .RETURN LOCAL DAL                   BCOM.266
402     071326                        P = NEXT                  .CHECK FOR MORE REQUESTS            BCOM.267
403      <macro>                    $ENDIF                                                          BCOM.268
                                                                                   .<<<<< 1          $ENDIF   1
                                                                                                     BCOM.269
                                    * ALL OTHER REQUESTS                                             BCOM.270
                                    * IF DEVICE ACTIVE QUEUE FOR TEX                                 BCOM.271
                                    * IF NOT, SEND TO BMXOPE (MUST BE MOUNT REQUEST)                 BCOM.272
                                                                                                     BCOM.273
403     014000 /000000  024410      R!X0 = R!DVC + (XDEV)        .POINTER INTO DEVICE TABLE          BCOM.274
        020456  032410  024430
411     030430  024430              R!X0 = (R!X0)               .DEVICE TABLE ADDRESS                BCOM.275
413      <macro>                    GET A,BDV@OP,R!X0            .OPEN FLAG                          BCOM.276
421      <macro>                    $IF (A = 0)                  .IF DEVICE NOT OPEN                 BCOM.277
                                                                                   .>>>>> 1          $IF      1
423      <macro>                      GET R!%W1,BDV@DT,R!X0      .GET DEVICE TYPE                R16767RD.2
427      <macro>                      $IF (R!%W1 # DT$BMT03)     .IF NOT 3480                    R16767RD.3
                                                                                   .>>>>> 2          $IF      1
433      <macro>                        STORE 0,TQ@STS,R!DAL     .CLEAR STATUS                   R16767RD.4
440      <macro>                      $ENDIF                                                      R16767RD.5
                                                                                   .<<<<< 2          $ENDIF   1
                                                                                                     BCOM.278
                                      * SEND TO OPEN ROUTINE                                         BCOM.279
                                                                                                     BCOM.280
440     014000 /000000  024431        R!X1 = O$BMXOPE            .OPEN ROUTINE                     BCOM.281
443     010001  024432                R!X2 = 1                   .PRIORITY                         BCOM.282
445     020445  024433                R!X3 = R!DAL               .PARAMETER                        BCOM.283
447     077000 /002353                R = CREATE                 .ACTIVATE                         BCOM.284
451      <macro>                    $ELSE                                                           BCOM.285
452      <macro>                      GET R!TCB,BDV@CR,R!X0      .CRW ADDRESS                      BCOM.286
457      <macro>                      $PUNTIF (R!TCB = 0)        .MUST BE ONE IF OPEN              BCOM.287
462      <macro>                      $IF (R!FUN # FC$DSP)       .IF NOT DISPLAY ON 3480'S       R16767RD.6
                                                                                   .>>>>> 2          $IF      1
466      <macro>                        STORE 0,TQ@STS,R!DAL     .CLEAR STATUS                   R16767RD.7
473      <macro>                      $ENDIF                                                      R16767RD.8
                                                                                   .<<<<< 2          $ENDIF   1
                                                                                                     BCOM.288
```

```
                                  * IF REQUEST IS FOR DATA I/O                                    BCOM.289
                                  * QUEUE FOR BYPASS                                              BCOM.290
                                                                                                  BCOM.291
473     <macro>                   $IF (R!FUN = FC$READ) OR (R!FUN = FC$WRITE)                     BCOM.292
                                                                          .>>>>> 2                    $IF       1
502    077000 /002401               R = BYPASS                .QUEUE FOR BYPASS                    BCOM.293
                                                                                                  BCOM.294
                                     * QUEUE CONTINUE-READS FOR BYPASS ALSO                       BCOM.295
                                                                                                  BCOM.296
504     <macro>                   $ELSEIF (R!FUN = FC$EORR) OR (R!FUN = FC$EOFR)                  BCOM.297
514    077000 /002401               R = BYPASS                                                   BCOM.298
516     <macro>                   $ELSEIF (R!FUN = FC$EODR) OR (R!FUN = FC$NOOP)                  BCOM.299
526    077000 /002401               R = BYPASS                                                   BCOM.300
530     <macro>                   $ELSE                                                           BCOM.301
                                                                                                  BCOM.302
                                     * QUEUE ALL OTHERS FOR TEX                                   BCOM.303
                                                                                                  BCOM.304
531    020451  013004  136000       R = HALTIO, R!FUN = FC$FREE  .HALT I/O IF FREE DEVICE        BCOM.305
      /002325
535    002000  000000               I = 0                     .LOCK                                BCOM.306
537    020445  024000               DY = R!DAL                .DAL ADDRESS                         BCOM.307
541     <macro>                     ADDRESS DZ,DC@MSG,R!TCB  .TCB DAL QUEUE                        BCOM.308
544     <macro>                     RKERN HPUT                                                   BCOM.309
546    003000                       I = 1                     .UNLOCK                              BCOM.310
547     <macro>                     ADDRESS R!QUE,DC@QUA,R!TCB  .TEX WAIT QUEUE                    BCOM.311
552    077000 /002265               R = POPPER                .LET TEX RIP                         BCOM.312
554     <macro>                   $ENDIF                                                          BCOM.313
                                                                          .<<<<< 2                    $ENDIF    1
554     <macro>                 $ENDIF                                                            BCOM.314
                                                                          .<<<<< 1                    $ENDIF    1
554    071500                     P = NEXT                  .CHECK FOR MORE                      BCOM.315
555     <macro>                 $ELSEIF (R!FCT = MC$CPURD) OR (R!FCT = MC$CPUWR)  .I/O TO CPU    BCOM.316
                                                                                                  BCOM.317
                                * REQUEST FROM XIOP TO READ/WRITE TO CPU                          BCOM.318
                                                                                                  BCOM.319
566     <macro>                 $PUNTIF (R!%MYID # BIOP) AND (R!%MYID # DIOP)  .If wrong IOP      BCOM.320
                                                                          .>>>>> 1                    $IF       2
                                                                          .<<<<< 1                    $ENDIF    2
600    002000  000000               I = 0                     .LOCK                                BCOM.321
602     <macro>                     STORE 0,AD@PRI,R!%ACTIVE  .SET PRIORITY 0 IN BIOP/DIOP         BCOM.322
611    020446  024000               EH = R!MSG                .LEAVE DAL IN MOS / QUEUE MESSAGE    BCOM.323
613    014000 /000000  024000       ED = BXQQ                 .QUEUE ADDRESS                       BCOM.324
```

```
616    <macro>                    RKERN EPOQ                                                    BCOM.325
620    <macro>                    $PUNTIF (EH = 0)            .IF QUEUE FULL                    BCOM.326
623   014000 /000000  024454      R!QUE = TDMQ                .WAIT QUEUE FOR TDEM1             BCOM.327
626   077000 /002265              R = POPPER                  .ACTIVATE IF WAITING              BCOM.328
630   077000 /002143              R = DALRET                  .RETURN LOCAL DAL                 BCOM.329
632   071556                      P = NEXT                    .CHECK FOR MORE                   BCOM.330
633    <macro>                   $ELSEIF (R!FCT = MC$CPRDN) OR (R!FCT = MC$CPWDN)  .CPU I/O DONE  BCOM.331
                                                                                                BCOM.332
                                 * REPLY TO XIOP REQUEST FOR CPU READ/WRITE                     BCOM.333
                                                                                                BCOM.334
645    <macro>                    $PUNTIF (R!%MYID # XIOP)    .If wrong IOP                     BCOM.335
651   077000 /002106              R = MOSREL                  .RETURN MOS DAL                   BCOM.336
                                                                                                BCOM.337
                                 * DECREMENT OUTSTANDING REQUEST COUNT                          BCOM.338
                                                                                                BCOM.339
653    <macro>                    GET R!TCB,DA@TCB,R!DAL      .DEVICE TABLE ADDRESS             BCOM.340
660    <macro>                    FLDSUB 1,DC@BRQ,R!TCB       .DECREMENT COUNT                  BCOM.341
664    <macro>                    $PUNTIF (A = 177777)        .IF NEGATIVE                      BCOM.342
                                                                                                BCOM.343
                                 * SEE IF ANYONE WAITING                                        BCOM.344
                                                                                                BCOM.345
670    <macro>                    ADDRESS R!QUE,DC@QUB,R!TCB  .BIOP WAIT QUEUE                  BCOM.346
673   077000 /002265              R = POPPER                                                    BCOM.347
                                                                                                BCOM.348
                                 * GET PARAMETERS FROM DAL                                      BCOM.349
                                                                                                BCOM.350
675   077000 /002153              R = DALSTRIP                                                  BCOM.351
677    <macro>                    GET R!VMS,DC@VMS,R!TCB      .VALID SECTOR/BLOCK COUNT         BCOM.352
                                                                                                BCOM.353
                                 ********************                                           BCOM.354
                                 *       TRACE      *                                           BCOM.355
                                 *                  *                                           BCOM.356
                                 * 1. DEVICE        *                                           BCOM.357
                                 * 2. TRAN. BLOCKS   *                                          BCOM.358
                                 * 3. TRAN. SECTS    *                                          BCOM.359
                                 * 4. VALID MOS      *                                          BCOM.360
                                 * 5. STATUS         *                                          BCOM.361
                                 ********************                                           BCOM.362
                                                                                                BCOM.363
704    <macro>                    KRECORD TF$TEX1,(R!DVC,R!TBC,R!TSC,R!VMS,R!STAT),INT=ON       BCOM.364
                                                                                                BCOM.365
                                 * CHECK STATUS FOR ERROR                                       BCOM.366
```

```
                                                                                    BCOM.367
 724    <macro>                  $IF (R!STAT # 0)            .IF ERROR DETECTED     BCOM.368
                                                                        .>>>>> 1        $IF      1
 727    <macro>                    STORE MC$CSTAT,DA@IFC,R!DAL .FUNCTION CODE     R15472BG.2
 734   077000 /002241              R = RESPOND                 .RESPOND TO CPU        BCOM.370
 736    <macro>                    STORE 1,DC@ERR,R!TCB        .SET ERROR FLAG        BCOM.371
 747   077000 /002143              R = DALRET                  .RETURN LOCAL DAL      BCOM.372
 751   071675                      P = NEXT                    .CHECK FOR MORE        BCOM.373
 752    <macro>                  $ENDIF                                              BCOM.374
                                                                        .<<<<< 1        $ENDIF   1
                                                                                    BCOM.375
                                 * SENSE READ FROM CPU COMPLETE                     BCOM.376
                                                                                    BCOM.377
 752    <macro>                  $IF (R!FCT = MC$CPRDN)                              BCOM.378
                                                                        .>>>>> 1        $IF      1
 756   020444  022441  024444      R!VMS = R!VMS + R!TBC       .UPDATE MOS BLOCK COUNT BCOM.379
                                                                                    BCOM.380
                                   * WRITE FROM MOS TO CPU COMPLETE                 BCOM.381
                                                                                    BCOM.382
 761    <macro>                  $ELSE                                               BCOM.383
 763    <macro>                    FLDSUB R!TSC,DC@TMS,R!TCB  .DECREMENT MOS SECTOR COUNT BCOM.384
                                                                                    BCOM.385
                                   * REPORT ONLY VALID SECTORS TRANSFERRED          BCOM.386
                                   * SAVE OTHEPS FOR END OF BLOCK.                  BCOM.387
                                                                                    BCOM.388
 771    <macro>                    $IF (R!VMS < R!TSC)         .IF SOME NON-VALID TRANSFERS BCOM.389
                                                                        .>>>>> 2        $IF      1
 776   020440  023444  024430        R!X0 = R!TSC - R!VMS      .COUNT                 BCOM.390
1001    <macro>                      FLDADD R!X0,DC@TSC,R!TCB  .SAVE FOR END OF BLOCK BCOM.391
1007   020444  024440                R!TSC = R!VMS             .NUMBER TO REPORT      BCOM.392
1011    <macro>                    $ENDIF                                              BCOM.393
                                                                        .<<<<< 2        $ENDIF   1
1011   020444  023440  024444      R!VMS = R!VMS - R!TSC       .UPDATE VALID SECTORS REMAINING BCOM.394
                                                                                    BCOM.395
                                   * IF DATA-BAD SET, CLEAR REQUESTED SECTOR COUNT  BCOM.396
                                                                                    BCOM.397
1014    <macro>                    GET A,TQ@DBF,R!DAL          .DATA BAD FLAG         BCOM.398
1021    <macro>                    $IF (A # 0)                 .IF SET                BCOM.399
                                                                        .>>>>> 2        $IF      1
1023    <macro>                      STORE 0,DC@RSC,R!TCB      .CLEAR REMAINING REQUEST BCOM.400
1030    <macro>                    $ENDIF                                              BCOM.401
                                                                        .<<<<< 2        $ENDIF   1
```

```
                                                                                                  BCOM.402
                                                                                                  BCOM.403
                                  * KEEP TRACK OF NUMBER OF SECTORS SENT TO CPU                   BCOM.404
                                  * THAT HAVE NOT BEEN RELEASED. IF ENOUGH ACCUMULATE             BCOM.405
                                  * CREATE BUFMAN TO RELEASE THEM.                                BCOM.406
                                                                                                  BCOM.407
1030    <macro>                   FLDADD R!TSC,DC@FSC,R!TCB  .ADD CURRENT SECTORS                BCOM.408
1036    <macro>                   GET R!X0,DC@FSC,R!TCB     .CHECK TOTAL                         BCOM.409
1043    <macro>                   GET R!X1,DC@BDN,R!TCB     .BUFMAN ACTIVE FLAG                  BCOM.410
1052    <macro>                   $IF (R!X0 > 20) AND (R!X1 = 0)                                  BCOM.411
                                                                          .>>>>> 2                $IF      1
1063   014000 /000000  024431       R!X1 = O$BUFMAN                                              BCOM.412
1066   010001  024432               R!X2 = 1                  .Task priority                     BCOM.413
1070   010001  024433               R!X3 = FN$DALL            .CODE TO DE-ALLOCATE               BCOM.414
1072   020442  024434               R!X4 = R!TCB              .DEVICE TABLE ADDRESS              BCOM.415
1074   020445  024435               R!X5 = R!DAL              .DAL ADDRESS                       BCOM.416
1076   077000 /002353               R = CREATE                .CALL                              BCOM.417
1100    <macro>                   $ENDIF                                                          BCOM.418
                                                                          .<<<<< 2                $ENDIF   1
1100    <macro>                  $ENDIF                                                           BCOM.419
                                                                          .<<<<< 1                $ENDIF   1
                                                                                                  BCOM.420
                                 * UPDATE UNUSED BIT AND PARTIAL WORD                             BCOM.421
                                 * COUNT IN CURRENT DAL.                                          BCOM.422
                                                                                                  BCOM.423
1100    <macro>                  GET R!X0,DC@DAL,R!TCB       .DAL ADDRESS                        BCOM.424
1105    <macro>                  $IF (R!TSC # 0) OR (R!TBC # 0)  .IF REPORTING TO CPU NOW        BCOM.425
                                                                          .>>>>> 1                $IF      1
                                                                                                  BCOM.426
                                  * SEND RESPONSE TO CPU                                          BCOM.427
                                                                                                  BCOM.428
1112    <macro>                   PUT R!TSC,TQ@TSC,R!DAL    .TRANSFERRED SECTORS                BCOM.429
1117    <macro>                   PUT R!TBC,TQ@TBC,R!DAL    .TRANSFERRED BLOCKS                 BCOM.430
1124    <macro>                   PUT R!VMS,TQ@VMS,R!DAL    .VALID MOS COUNT                    BCOM.431
1131    <macro>                   STORE 1,ST@DTR,R!DAL      .SET DATA TRANSFERRED               BCOM.432
1142   014000 /000000  024431     R!X1 = BBDISK             .PASS AVAILABLE MOS SECTOR COUNT    BCOM.433
1145    <macro>                   GET R!X2,BB@TOT,R!X1      .TOTAL MOS                          BCOM.434
1152    <macro>                   GET R!X3,BB@USE,R!X1      .NUMBER IN USE                      BCOM.435
1157   020432  023433  024432     R!X2 = R!X2 - R!X3        .NUMBER STILL AVAILABLE             BCOM.436
1162    <macro>                   PUT R!X2,TQ@MOS,R!DAL     .ENTER IN DAL                       BCOM.437
1167    <macro>                   STORE MC$CSTAT,DA@IFC,R!DAL .FUNCTION CODE                R15472BG.3
1174   077000 /002241             R = RESPOND               .SEND IT                            BCOM.439
```

```
1176    <macro>                 $ELSE                                                   BCOM.440
1177    <macro>                   GET R!X1,TQ@UBC,R!DAL     .UNUSED BIT COUNT           BCOM.441
1206    <macro>                   GET R!X2,TQ@PWC,R!DAL     .PARTIAL WORD COUNT         BCOM.442
1214    <macro>                   PUT R!X1,TQ@UBC,R!X0      .SAVE COUNTS FOR NEXT RESPONSE BCOM.443
1230    <macro>                   PUT R!X2,TQ@PWC,R!X0                                  BCOM.444
1243    <macro>                   GET R!X1,TQ@DBF,R!DAL     .MOVE BAD DATA FLAG         BCOM.445
1251    <macro>                   PUT R!X1,TQ@DBF,R!X0      .                           BCOM.446
1264    <macro>                 $ENDIF                                                  BCOM.447
                                                                        .<<<<< 1            $ENDIF    1
1264    <macro>                 PUT R!VMS,DC@VMS,R!TCB    .UPDATE MOS VALID COUNT     BCOM.448
1271   077000 /002143           R = DALREI                .RETURN LOCAL DAL           BCOM.449
1273   075000 /000054           P = NEXT                  .CHECK FOR MORE             BCOM.450
1275    <macro>               $ELSEIF (R!FCT = MC$CSTAT) OR (R!FCT = MC$CSTRL)  .STATUS TO CP BCOM.451
                                                                                        BCOM.452
                                * SEND RESPONSE TO CPU                                  BCOM.453
                                                                                        BCOM.454
1306    <macro>                 $PUNTIF (R!%MYID # MIOP)  .If wrong IOP               BCOM.455
1311   077000 /002153           R = DALSTRIP              .GET PARAMETERS INTO REGS   BCOM.456
                                                                                        BCOM.457
                                * SET SOURCE - EITHER BMX OR ERROR RESPONSE             BCOM.458
                                                                                        BCOM.459
1313    <macro>                 $IF (R!FCT = MC$CSTAT)    .BMX RESPONSE               BCOM.460
                                                                        .>>>>> 1            $IF       1
1317   010104  024430             R!X0 = RQ$BMX0                                        BCOM.461
1321    <macro>                 $ELSE                                                   BCOM.462
1322   010103  024430             R!X0 = RQ$ERRR                                        BCOM.463
1324    <macro>                 $ENDIF                                                  BCOM.464
                                                                        .<<<<< 1            $ENDIF    1
1324    <macro>                 GET R!%W1,TQ@SID,R!DAL    .SWAP DID AND SID           BCOM.465
1331    <macro>                 GET R!%W2,TQ@DID,R!DAL    .                           BCOM.466
1336    <macro>                 PUT R!%W1,TQ@DID,R!DAL    .                           BCOM.467
1343    <macro>                 PUT R!%W2,TQ@SID,R!DAL    .                           BCOM.468
1350    <macro>                 PUT R!X0,TQ@SIL,R!DAL     .SET SOURCE ID              BCOM.469
1363   002000  000000           I = 0                     .LOCK                       BCOM.470
1365   020445  024000           R!DY = R!DAL              .DAL TO SEND                BCOM.471
1367    <macro>                 RKERN IDALSND             .                           BCOM.472
1371   077000 /002106           R = MOSREL                .RETURN MOS DAL             BCOM.473
1373   075000 /000054           P = NEXT                  .CHECK FOR MORE             BCOM.474
1375    <macro>               $ELSEIF (R!FCT = MC$CPMLT)  .MULTIPLE PKTS IN MOS       BCOM.475
                                                                                        BCOM.476
1404    <macro>                 $PUNTIF (R!%MYID # MIOP)  .WRONG IOP                  BCOM.477
                                                                                        BCOM.478
```

```
                                        * GET MOS ADDRESS OF BUFFER AND REQUESTED NUMBER OF PACKETS      BCOM.479
                                                                                                         BCOM.480
1407    <macro>                         GET R!MSU,TQ@MSU,R!DAL       .MOS UPPER ADDRESS                  BCOM.481
1414    <macro>                         GET R!MSL,TQ@MSL,R!DAL       .MOS LOWER ADDRESS                  BCOM.482
1421    <macro>                         GET R!TOT,TQ@REQ,R!DAL       .NUMBER OF PACKETS REQUESTED        BCOM.483
1426    <macro>                         GET R!DIA,TQ@DIA,R!DAL       .GET DIAGNOSTIC REQUEST             BCOM.484
1434    <macro>                         GET R!TID,TQ@DIU,R!DAL       .TASK ID                            BCOM.485
1442    <macro>                         GET R!SLG,DI$SLG,R!DAL       .SUPPRESS SYSTEM LOG FLAG           BCOM.486
1450    <macro>                         GET R!REQ,TQ@RQN,R!DAL       .REQUEST NUMBER                     BCOM.487
1455    010030  004002  013001          R!LEN = DA@@LE > 2 - 1       .NUMBER OF WORDS - 1                BCOM.488
        024462
1461    010010  012004  024466          R!OFF = DA@@LH + 4           .HEADER + 1 WORD                    BCOM.489
                                                                                                         BCOM.490
                                        * ALLOCATE NEW DALS FOR LOG MESSAGE                              BCOM.491
                                                                                                         BCOM.492
1464    020460  024426                  R!%W4 = R!MSU                .MOS UPPER ADDRESS                  BCOM.493
1466    020461  024427                  R!%W5 = R!MSL                .MOS LOWER                          BCOM.494
1470    010001  024425                  R!%W3 = 1                    .SEQUENCE COUNTER                   BCOM.495
1472    020445  024424                  R!%W2 = R!DAL                                                    BCOM.496
1474    <macro>                         $UNTIL (R!%W3 = R!TOT + 1)   .GET NUMBER OF DALS NEEDED          BCOM.497
1502    <macro>                           GETDAL R!DPT                                                   BCOM.498
1512    <macro>                           $IF (A # 0)                .IF ERROR                           BCOM.499
                                                                                        .>>>>> 1            $IF      1
1514    030445  024423                      R!%W1 = (R!DAL)                                              BCOM.500
1516    <macro>                             $UNTIL (R!%W1 = 0)       .RELEASE ALL DALS                   BCOM.501
1520    020423  024463                        R!DPT = R!%W1                                              BCOM.502
1522    030423  024423                        R!%W1 = (R!%W1)        .NEXT LINK                          BCOM.503
1524    <macro>                               RELDAL R!DPT                                               BCOM.504
1534    <macro>                             $ENDTIL                                                      BCOM.505
1535    070245                              P = RELMOS                                                   BCOM.506
1536    <macro>                           $ENDIF                                                         BCOM.507
                                                                                        .<<<<< 1            $ENDIF   1
1536    <macro>                           CLEAR START=R!DPT, COUNT=R!OFF                                 BCOM.508
1551    <macro>                           MOSR R!%W4,R!%W5,R!DPT+R!OFF,R!LEN    .READ IN LOG          R16767MA.1
1572    <macro>                           STORE RQ$ERRR,TQ@SIL,R!DPT .SOURCE ID LOWER BYTE               BCOM.509
1602    <macro>                           STORE RQ$CPU,TQ@DID,R!DPT  .DESTINATION                        BCOM.510
1610    <macro>                           PUT R!%W3,EP@SEQ,R!DPT     .SEQUENCE NUMBER                    BCOM.511
1623    <macro>                           PUT R!TOT,EP@TOT,R!DPT     .TOTAL NUMBER                       BCOM.512
                                                                                                         BCOM.513
                                          * CHECK DIAGNOSTIC REQUEST                                     BCOM.514
                                                                                                         BCOM.515
1636    <macro>                           $IF (R!DIA # 0)                                                BCOM.516
```

```
                                                                  .>>>>> 1          $IF      1
1641   <macro>                    PUT R!TID,TQ@SIU,R!DPT     .TASK ID UPPER BYTE          BCOM.517
1654   020471  004010  024427     R!%W5 = R!REQ > D'8        .REQUEST UPPER               BCOM.518
1657   <macro>                    PUT R!%W5,EP@RQU,R!DPT     .STORE UPPER                 BCOM.519
1672   020471  011377  024427     R!%W5 = R!REQ & O'377      .REQUEST LOWER               BCOM.520
1675   <macro>                    PUT R!%W5,EP@RQL,R!DPT     .STORE LOWER                 BCOM.521
1710   <macro>                  $ELSE                                                      BCOM.522
1711   <macro>                    STORE 0,TQ@SIU,R!DPT       .TASK ID UPPER BYTE          BCOM.523
1722   <macro>                    STORE ER@TER,EP@PTY,R!DPT .ERROR CODE SUBTYPE           BCOM.524
1732   <macro>                  $ENDIF                                                     BCOM.525
                                                                  .<<<<< 1          $ENDIF   1
1732   <macro>                  DADD R!%W4,R!%W5,R!LEN     .REMAINING PACKET IN MOS     BCOM.526
1737   020463  034424           (R!%W2) = R!DPT            .LINK DAL ADDRESS            BCOM.527
1741   020463  024424           R!%W2 = R!DPT                                           BCOM.528
1743   026425                   R!%W3 = R!%W3 + 1          .INCREMENT COUNTER           BCOM.529
1744   <macro>                $ENDTIL                                                   BCOM.530
                                                                                         BCOM.531
                              * SEND CONTIGUOUS PACKETS                                  BCOM.532
                                                                                         BCOM.533
1745   030445  024423         R!%W1 = (R!DAL)            .FIRST LINK                  BCOM.534
1747   002000  000000         I = 0                      .LOCK INTERRUPTS             BCOM.535
1751   <macro>                $UNTIL (R!%W1 = 0)         .UNTIL ALL DALS SENT         BCOM.536
1753   020423  024000           R!DY = R!%W1                                            BCOM.537
1755   030423  024423           R!%W1 = (R!%W1)            .NEXT LINKED DAL             BCOM.538
1757   <macro>                  RKERN  IDALSND             .SEND IT                     BCOM.539
1761   <macro>                $ENDTIL                                                   BCOM.540
1762   003000                 I = 1                      .UNLOCK INTERRUPTS           BCOM.541
                                                                                         BCOM.542
                              *                                                          BCOM.543
                              * RELEASE ONLY DAL AND NOT MOS BUFFER IF THIS IS A         BCOM.544
                              * DIAGNOSTIC TASK AND SUPPRESS SYSTEM LOG NOT SET          BCOM.545
                              *                                                          BCOM.546
                                                                                         BCOM.547
1763   <macro>                $IF (R!DIA # 0)            .CHECK SUPPRESS SYSTEM LOG   BCOM.548
                                                                  .>>>>> 1          $IF      1
1766   <macro>                  $IF (R!SLG = 0)            .DO NOT RELEASE MOS BUFFER   BCOM.549
                                                                  .>>>>> 2          $IF      1
1771   <macro>                    STORE MC$RLDAL,DA@IFC,R!DAL                           BCOM.550
1776   077000 /002106             R = MOSREL                 .RELEASE JUST THE DAL        BCOM.551
2000   075000 /000054             P = NEXT                                                BCOM.552
2002   <macro>                  $ENDIF                                                    BCOM.553
                                                                  .<<<<< 2          $ENDIF   1
```

```
2002    <macro>                          $ENDIF                                                          BCOM.554
                                                                                  .<<<<< 1                $ENDIF    1
                                                                                                          BCOM.555
                                         * RELEASE MOS BUFFER                                             BCOM.556
                                                                                                          BCOM.557
2002                             RELMOS  *                                                                BCOM.558
2002    <macro>                          STORE MC$RLMOS,DA@IFC,R!DAL  .RELEASE MOS CODE            R15472BG.4
2007    072104                           R = MOSREL2                  .SEND TO CALLING IOP             BCOM.560
2010    072133                           R = DALRET                   .RELEASE LOCAL DAL               BCOM.561
2011    075000 /000054                   P = NEXT                                                      BCOM.562
2013    <macro>                        $ELSEIF (R!FCT = MC$RLMOS) OR (R!FCT = MC$RLDAL)                BCOM.563
                                                                                                          BCOM.564
                                         * RELEASE MOS BUFFER                                             BCOM.565
                                                                                                          BCOM.566
2023    <macro>                          $IF (R!FCT = MC$RLMOS)                                          BCOM.567
                                                                                  .>>>>> 1                $IF       1
2027    <macro>                            $PUNTIF (R!%MYID = MIOP)   .WRONG IOP                         BCOM.568
2032    <macro>                            GET R!MSU,IQ@MSU,R!DAL     .GET THE MOS BUFFER                BCOM.569
2037    <macro>                            GET R!MSL,IQ@MSL,R!DAL                                        BCOM.570
2044    <macro>                            MPUT R!MSU,R!MSL           .GIVE IT BACK                      BCOM.571
2060    <macro>                          $ENDIF                                                          BCOM.572
                                                                                  .<<<<< 1                $ENDIF    1
                                                                                                          BCOM.573
                                         * RETURN MOS DAL TO FREE POOL                                    BCOM.574
                                                                                                          BCOM.575
2060    072063                           R = DALRET                   .RETURN LOCAL DAL                BCOM.576
2061    002000  000000                   I = 0                        .LOCK                            BCOM.577
2063    <macro>                          GET R!CB,DA@DL0,R!DAL        .GET ADDRESS                 R15472BG.5
2070    <macro>                          GET R!CC,DA@DL1,R!DAL        .                            R15472BG.6
2075    010001  024000                   R!CA = 1                     .RELEASE 1 DAL                   BCOM.580
2077    <macro>                          RKERN   PUTDAL                                                BCOM.581
2101    075000 /000054                   P = NEXT                                                      BCOM.582
2103    <macro>                        $ELSE                                                           BCOM.583
                                                                                                          BCOM.584
                                         * ILLEGAL FUNCTION CODE                                          BCOM.585
                                                                                                          BCOM.586
2104    <macro>                          $PUNTIF                                                         BCOM.587
2106    <macro>                        $ENDIF                                                            BCOM.588
                                                                                  .<<<<< 0                $ENDIF    1
                                                                                                          BCOM.589
                                 ************************** SUBROUTINES ****************************      BCOM.590
                                                                                                          BCOM.591
```

```
                                        *                                                                   BCOM.592
                                        * RETURN MOS DAL TO APPROPRIATE IOP                                BCOM.593
                                        *                                                                   BCOM.594
                                                                                                            BCOM.595
2106                            MOSREL  *                                                                   BCOM.596
2106    <macro>                         STORE MC$RLDAL,DA@IFC,R!DAL    .FUNCTION CODE                   R15472BG.7
2113                            MOSREL2 *                                                                   BCOM.598
2113    002000  000000                  I = 0                          .LOCK                               BCOM.599
2115    020445  024000                  DS = R!DAL                     .DAL ADDRESS                        BCOM.600
2117    <macro>                         GET A,DA@IFC,R!DAL             .FUNCTION CODE                   R15472BG.8
2123    <macro>                         $IF (A = MC$RLMOS)             .IF RELEASE MOS                  R15472BG.9
                                                                                                .>>>>> 0        $IF      1
2126    <macro>                           RKERN   DSUM                 .SEND COMPLETE DAL                  BCOM.602
2130    <macro>                         $ELSE                                                               BCOM.603
2131    010001  024000                    R!CE = 1                                                          BCOM.604
2133    <macro>                           RKERN   DALWRT               .                                   BCOM.605
2135    <macro>                         $ENDIF                                                              BCOM.606
                                                                                                .<<<<< 0        $ENDIF   1
2135    020447  024000                  EI = R!IOP                     .ORIGINATING IOP                    BCOM.607
2137    <macro>                         RKERN EMSGIOP                  .SEND BACK                          BCOM.608
2141    003000                          I = 1                          .UNLOCK                             BCOM.609
2142    001000                          EXIT                                                               BCOM.610
                                                                                                            BCOM.611
                                        *                                                                   BCOM.612
                                        * RETURN LOCAL DAL                                                 BCOM.613
                                        *                                                                   BCOM.614
                                                                                                            BCOM.615
2143                            DALRET  *                                                                   BCOM.616
2143    002000  000000                  I = 0                          .LOCK                               BCOM.617
2145    020445  024000                  DY = R!DAL                     .ADDRESS                            BCOM.618
2147    <macro>                         RKERN HONN                     .RETURN TO POOL                     BCOM.619
2151    003000                          I = 1                          .UNLOCK                             BCOM.620
2152    001000                          EXIT                                                               BCOM.621
                                                                                                            BCOM.622
                                        *                                                                   BCOM.623
                                        * MOVE PARAMETERS FROM DAL TO REGISTERS                            BCOM.624
                                        *                                                                   BCOM.625
                                                                                                            BCOM.626
2153                            DALSTRIP *                                                                  BCOM.627
2153    <macro>                         GET R!FUN,TQ@FCN,R!DAL         .FUNCTION TYPE                      BCOM.628
2162    <macro>                         GET R!DVC,TQ@DVN,R!DAL         .DEVICE                             BCOM.629
2170    <macro>                         GET R!RSC,TQ@RSC,R!DAL         .REQUESTED SECTORS                  BCOM.630
```

```
2175    <macro>                    GET R!RBC,TQ@RBC,R!DAL        .REQUESTED BLOCKS                    BCOM.631
2202    <macro>                    GET R!TSC,TQ@TSC,R!DAL        .TRANSFERRED SECTORS                 BCOM.632
2207    <macro>                    GET R!TBC,TQ@TBC,R!DAL        .TRANSFERRED BLOCKS                  BCOM.633
2214    <macro>                    GET R!FLG,TQ@DDF,R!DAL        .DATASET FLAGS                       BCOM.634
2221    <macro>                    GET R!STAT,TQ@STS,R!DAL       .STATUS                              BCOM.635
2226    <macro>                    $IF (R!FUN = FC$MOUNT)                                           R16767RD.9
                                                                                   .>>>>> 0               $IF      1
2232   010000  024443                R!STAT = 0                    .CLEAR                           R16767RD.10
2234   010000  024437                R!RBC = 0                                                      R16767RD.11
2236   010000  024436                R!RSC = 0                                                      R16767RD.12
2240    <macro>                    $ENDIF                                                           R16767RD.13
                                                                                   .<<<<< 0               $ENDIF   1
2240   001000                       EXIT                                                                BCOM.636
                                                                                                       BCOM.637
                                    *                                                                  BCOM.638
                                    * SEND RESPONSE TO MIOP                                           BCOM.639
                                    *                                                                  BCOM.640
                                                                                                       BCOM.641
2241                        RESPOND *                                                                  BCOM.642
2241   002000  000000               I = 0                         .LOCK                                BCOM.643
2243   020445  024000               DS = R!DAL                    .ADDRESS                             BCOM.644
2245    <macro>                     RKERN DDALGETA                .ALLOCATE MOS DAL                    BCOM.645
2247    <macro>                     RKERN DMGA                    .SEND DAL                            BCOM.646
2251   003000                       I = 1                         .UNLOCK                              BCOM.647
2252   001000                       EXIT                                                               BCOM.648
                                                                                                       BCOM.649
                                    *                                                                  BCOM.650
                                    * GIVE UP CONTROL IF AN ACTIVITY                                   BCOM.651
                                    * OF EQUAL OR HIGHER PRIORITY IS WAITING.                          BCOM.652
                                    *                                                                  BCOM.653
                                                                                                       BCOM.654
2253                        YIELD   *                                                                  BCOM.655
2253   010000  024455               R!NUMDONE = 0                 .CLEAR CHECK POINT COUNT             BCOM.656
2255   002000  000000               I = 0                         .LOCK                                BCOM.657
2257   014000 /000000               A = D$BCOM                    .DEMON DESCRIPTOR                    BCOM.658
2261    <macro>                     RKERN     DGIVEUP             .Relinquish control if necessary     BCOM.659
2263   075000 /000054               P = NEXT                                                           BCOM.660
                                    *                                                                  BCOM.661
                                    * POP ACTIVITIES OFF WAIT QUEUE                                    BCOM.662
                                    *                                                                  BCOM.663
                                                                                                       BCOM.664
2265                        POPPER  *                                                                  BCOM.665
```

```
2265    <macro>                          $UNTIL ((R!QUE) = 0)          .UNTIL QUEUE FLUSHED                BCOM.666
2267    <macro>                            $IF ((R!QUE) # 0)           .IF ENTRY                           BCOM.667
                                                                                          .>>>>> 0            $IF      1
2272   002000  000000                        I = 0                     .LOCK                               BCOM.668
2274   020454  024000                        ED = R!QUE                .QUEUE ADDRESS                      BCOM.669
2276    <macro>                              RKERN EDQU                .POP OFF                            BCOM.670
2300    <macro>                              RKERN EQCP                .PUT ON CP QUEUE                    BCOM.671
2302   003000                                I = 1                     .UNLOCK                             BCOM.672
2303    <macro>                            $ENDIF                                                          BCOM.673
                                                                                          .<<<<< 0            $ENDIF   1
2303    <macro>                          $ENDTIL                                                           BCOM.674
2304   001000                            EXIT                                                              BCOM.675
                                                                                                           BCOM.676
                                         *                                                                 BCOM.677
                                         * RETURN ERROR RESPONSE TO MIOP                                  BCOM.678
                                         *                                                                 BCOM.679
                                                                                                           BCOM.680
2305                            BADDO    *                                                                 BCOM.681
2305   014000 /177777  023430            R!X0 = 177777 - R!X0          .COMPLEMENT                         BCOM.682
       024430
2311    <macro>                          PUT R!X0,TQ@STS,R!DAL         .ENTER IN DAL                       BCOM.683
2316    <macro>                          STORE MC$CSTAT,DA@IFC,R!DAL   .FUNCTION CODE                    R15472BG.10
2323   073062                            R = RESPOND                   .SEND DAL                           BCOM.685
2324   001000                            EXIT                                                              BCOM.686
                                                                                                           BCOM.687
                                         *                                                                 BCOM.688
                                         * STOP I/O ON CURRENT DEVICE                                     BCOM.689
                                         *                                                                 BCOM.690
                                                                                                           BCOM.691
2325                            HALTIO   *                                                                 BCOM.692
2325    <macro>                          STORE 1,FL$FRP,R!TCB          .SET FREE PENDING                   BCOM.693
2336   014000 /000000  024431            R!X1 = O$BMXSIO               .OVERLAY BMXSIO                     BCOM.694
2341   010001  024432                    R!X2 = 1                      .PRIORITY                           BCOM.695
2343   010021  024433                    R!X3 = RQ$HIO                 .HALT I/O                           BCOM.696
2345   020442  024434                    R!X4 = R!TCB                  .CRW ADDRESS                        BCOM.697
2347   020456  024435                    R!X5 = R!DVC                  .DEVICE CODE                        BCOM.698
2351   072002                            R = CREATE                    .CREATE ACTIVITY                    BCOM.699
2352   001000                            EXIT                                                              BCOM.700
                                                                                                           BCOM.701
                                         *                                                                 BCOM.702
                                         * CREATE AN ACTIVITY                                             BCOM.703
                                         *                                                                 BCOM.704
```

```
                                                                                                  BCOM.705
2353                          CREATE   *                                                          BCOM.706
2353    <macro>                        CREATE    R!X1,(R!X3,R!X4,R!X5),PRI=R!X2,TYPE=NUMBER      BCOM.707
2376    <macro>                        $PUNTIF (A # 0)                .IF CANT CREATE IT         BCOM.708
2400   001000                          EXIT                                                      BCOM.709
                                                                                                  BCOM.710
                                       *                                                          BCOM.711
                                       * QUEUE REQUEST FOR BYPASS ACTIVITY                       BCOM.712
                                       *                                                          BCOM.713
                                                                                                  BCOM.714
2401                          BYPASS   *                                                          BCOM.715
2401   002000  000000                  I = 0                          .LOCK                      BCOM.716
2403   020445  024000                  DY = R!DAL                     .QUEUE REQUEST             BCOM.717
2405   014000 /000000  024000          DZ = DATQU                     .QUEUE ADDRESS             BCOM.718
2410    <macro>                        RKERN HPUT                     .ROUTINE CALL              BCOM.719
2412   003000                          I = 1                          .UNLOCK                    BCOM.720
2413   014000 /000000  024454          R!QUE = TIMQU                  .BYPASS WAIT QUEUE         BCOM.721
2416   073131                          R = POPPER                     .ACTIVATE IF WAITING       BCOM.722
2417   001000                          EXIT                                                      BCOM.723
                                                                                                  BCOM.724
                                       END                                                        BCOM.725
```

0X	%ACTIVE		4:29 D	12:40							
0X	%B		1: 4 D	7:19	7:27 B	7:33	10: 8 B	10:12 B	10:35 B	11: 5 B	
			11: 6 B	11:33 B	12: 6 B	12:11 B	12:13 B	12:18 B	12:26 B	12:32 B	
			13: 4 B	13: 5 B	13: 7 B	13:12 B	13:23 B	13:27 B	14: 5 B	14: 7 B	
			14:20 B	14:27 B	15:17 B	15:42 B	16:11 B	16:12 B	16:13 B	16:18 B	
			16:37 B	16:38 B	16:39 B	18:39 B	18:40 B	19:10 B	19:32 B	21:37 B	
0X	%EX		8:29 D	8:29	9:39 D	9:39	17:21 D	17:21	17:28 D	17:28	
			17:34 D	17:34	19:20 D	19:20	23: 3 D	23: 3			
10	%GBLNUM	$APTEXT	3:16 F	3:16							
400	%GBLREG	$APTEXT	3:16								
0	%KERNEL	$APTEXT	8: 9 D	8: 9 B	8:25 D	8:25 B	8:34 D	8:34 B	9: 2 D	9: 2 B	
			12:23 D	12:23 B	13: 1 D	13: 1 B	16:36 D	16:36 B	18:25 D	18:25 B	
			19:31 D	19:31 B	20:13 D	20:13 B	20:16 D	20:16 B	20:20 D	20:20 B	
			20:31 D	20:31 B	21:22 D	21:22 B	21:23 D	21:23 B	21:36 D	21:36 B	
			22: 6 D	22: 6 B	22: 7 D	22: 7 B	23:15 D	23:15 B			
0X	%MYID		4:30 D	7: 7	7: 9	7:11	7:13	10: 7	12:37	13:11	
			16:17	16:41	19:17						
0X	%PUNT		7:16 D	7:16	10: 7 D	10: 7	11:36 D	11:36	12:36 D	12:37	
			13: 2 D	13: 2	13:11 D	13:11	13:18 D	13:18	16:17 D	16:17	
			16:41 D	16:41	19:17 D	19:17	19:37 D	19:37	23: 4 D	23: 4	
0X	%TR1		10:24 D	10:24	13:40 D	13:40					
0X	%TR2		10:24 D	10:24	13:40 D	13:40					
0X	%TR3		10:24 D	10:24	13:40 D	13:40					
0X	%TR4		10:24 D	10:24	13:40 D	13:40					
0X	%TR5		10:24 D	10:24	13:40 D	13:40					
0X	%TRAC		10:24 D	10:24	13:40 D	13:40					
0X	%TVNT		10:24 D	10:24	13:40 D	13:40					
177777	AD$NODEM	$APTEXT	23: 3								
17	AD@PRI@M	$APTEXT	12:40 F								
4	AD@PRI@N	$APTEXT	12:40 F								
2	AD@PRI@P	$APTEXT	12:40 F	12:40							
14	AD@PRI@S	$APTEXT	12:40 F								
177760	AD@PRI@X	$APTEXT	12:40								
20	BB@IOT@N	$APTEXT	15:37	15:37 F							
2	BB@IOT@P	$APTEXT	15:37 F	15:37							
0	BB@IOT@S	$APTEXT	15:37	15:37 F							
20	BB@USE@N	$APTEXT	15:38	15:38 F							
1	BB@USE@P	$APTEXT	15:38 F	15:38							
0	BB@USE@S	$APTEXT	15:38	15:38 F							
0X	BBDISK		4:31 D	15:36							
20	BDV@CR@N	$APTEXT	11:35	11:35 F							
7	BDV@CR@P	$APTEXT	11:35 F	11:35							

0	BDV@CR@S	$APTEXT	11:35		11:35	F											
77	BDV@DI@M	$APTEXT	11:21														
6	BDV@DI@N	$APTEXT	11:21		11:21	F											
0	BDV@DI@P	$APTEXT	11:21	F													
2	BDV@DI@S	$APTEXT	11:21		11:21	F											
1	BDV@OP@M	$APTEXT	11:18														
1	BDV@OP@N	$APTEXT	11:18		11:18	F											
5	BDV@OP@P	$APTEXT	11:18	F	11:18												
10	BDV@OP@S	$APTEXT	11:18		11:18	F											
1	BIOP	$APTEXT	7: 9		12:37												
0X	BXQQ		4:32	D	12:42												
0X	CA		4:33	D	19:30												
0X	CB		4:34	D	19:28												
0X	CC		4:35	D	19:29												
0X	CE		4:36	D	20:15												
0X	D$BCOM		4:37	D	21:35												
30	DA@@LE	$APTEXT	17:10														
10	DA@@LH	$APTEXT	17:12														
20	DA@DLO@N	$APTEXT	19:28		19:28	F											
2	DA@DLO@P	$APTEXT	19:28	F	19:28												
0	DA@DLO@S	$APTEXT	19:28		19:28	F											
20	DA@DL1@N	$APTEXT	19:29		19:29	F											
3	DA@DL1@P	$APTEXT	19:29	F	19:29												
0	DA@DL1@S	$APTEXT	19:29		19:29	F											
20	DA@IFC@N	$APTEXT	10: 1		10: 1	F	14: 4	F	15:41	F	18:38	F	19: 7	F	20: 6	F	20:10
			20:10	F	22:22	F											
1	DA@IFC@P	$APTEXT	10: 1	F	10: 1		14: 4	F	14: 4		15:41	F	15:41		18:38	F	18:38
			19: 7	F	19: 7		20: 6	F	20: 6		20:10	F	20:10		22:22	F	22:22
0	DA@IFC@S	$APTEXT	10: 1		10: 1	F	14: 4	F	15:41	F	18:38	F	19: 7	F	20: 6	F	20:10
			20:10	F	22:22	F											
20	DA@ICB@N	$APTEXT	13:16		13:16	F											
14	DA@ICB@P	$APTEXT	13:16	F	13:16												
0	DA@ICB@S	$APTEXT	13:16		13:16	F											
0X	DALWRT		20:16	D	20:16												
0X	DATQU		4:38	D	23:14												
1	DC@BDN@M	$APTEXT	15: 9														
1	DC@BDN@N	$APTEXT	15: 9		15: 9	F											
21	DC@BDN@P	$APTEXT	15: 9	F	15: 9												
10	DC@BDN@S	$APTEXT	15: 9		15: 9	F											
20	DC@BRQ@N	$APTEXT	13:17	F													
36	DC@BRQ@P	$APTEXT	13:17	F	13:17												
20	DC@DAL@N	$APTEXT	15:26		15:26	F											

22	DC@DAL@P	$APTEXT	15:26 F	15:26						
0	DC@DAL@S	$APTEXT	15:26	15:26 F						
1	DC@ERR@M	$APTEXT	14: 6 F							
1	DC@ERR@N	$APTEXT	14: 6 F	14: 6						
21	DC@ERR@P	$APTEXT	14: 6 F	14: 6						
6	DC@ERR@S	$APTEXT	14: 6 F	14: 6						
20	DC@FSC@N	$APTEXT	15: 7 F	15: 8	15: 8 F					
43	DC@FSC@P	$APTEXT	15: 7 F	15: 7	15: 8 F	15: 8				
0	DC@FSC@S	$APTEXT	15: 8	15: 8 F						
33	DC@MSG@P	$APTEXT	12:22							
23	DC@QUA@P	$APTEXT	12:25							
25	DC@QUB@P	$APTEXT	13:22							
20	DC@RSC@N	$APTEXT	14:40 F							
46	DC@RSC@P	$APTEXT	14:40 F	14:40						
0	DC@RSC@S	$APTEXT	14:40 F							
20	DC@TMS@N	$APTEXT	14:21 F							
45	DC@TMS@P	$APTEXT	14:21 F	14:21						
20	DC@TSC@N	$APTEXT	14:29 F							
41	DC@TSC@P	$APTEXT	14:29 F	14:29						
20	DC@VMS@N	$APTEXT	13:28	13:28 F	16:10 F					
44	DC@VMS@P	$APTEXT	13:28 F	13:28	16:10 F	16:10				
0	DC@VMS@S	$APTEXT	13:28	13:28 F	16:10 F					
0X	DDALGETA		21:22 D	21:22						
0X	DGIVEUP		21:36 D	21:36						
1	DI$SLG@M	$APTEXT	17: 8							
1	DI$SLG@N	$APTEXT	17: 8							
12	DI$SLG@P	$APTEXT	17: 8 F	17: 8						
17	DI$SLG@S	$APTEXT	17: 8	17: 8 F						
2	DIOP	$APTEXT	7:11	12:37						
0X	DMGA		21:23 D	21:23						
0X	DS		4:39 D	20: 9	21:21					
0X	DSUM		20:13 D	20:13						
6	DT$BMIO3	$APTEXT	11:23							
0X	DY		4:40 D	8:26	12:21	16:35	18:23	20:30	23:13	
0X	DZ		4:41 D	12:22	23:14					
0X	ED		4:42 D	8:33	9: 1	12:42	22: 5			
0X	EDQU		22: 6 D	22: 6						
0X	EH		5: 1 D	8:35	8:40	8:42	12:41	13: 2		
0X	EI		5: 2 D	20:19						
0X	EMSGIOP		20:20 D	20:20						
377	EP@PIY@M	$APTEXT	18: 9 F							
10	EP@PIY@N	$APTEXT	18: 9 F							

13	EP@PTY@P	$APTEXT	18: 9	F	18: 9				
10	EP@PTY@S	$APTEXT	18: 9	F					
177400	EP@PTY@X	$APTEXT	18: 9						
377	EP@RQL@M	$APTEXT	18: 6						
10	EP@RQL@N	$APTEXT	18: 6	F					
13	EP@RQL@P	$APTEXT	18: 6	F	18: 6				
10	EP@RQL@S	$APTEXT	18: 6	F					
177400	EP@RQL@X	$APTEXT	18: 6						
377	EP@RQU@M	$APTEXT	18: 4						
10	EP@RQU@N	$APTEXT	18: 4	F	18: 4				
12	EP@RQU@P	$APTEXT	18: 4	F	18: 4				
0	EP@RQU@S	$APTEXT	18: 4	F	18: 4				
377	EP@SEQ@M	$APTEXT	17:37						
10	EP@SEQ@N	$APTEXT	17:37	F					
12	EP@SEQ@P	$APTEXT	17:37	F	17:37				
10	EP@SEQ@S	$APTEXT	17:37	F					
177400	EP@SEQ@X	$APTEXT	17:37						
377	EP@TOT@M	$APTEXT	17:38						
10	EP@TOT@N	$APTEXT	17:38	F	17:38				
13	EP@TOT@P	$APTEXT	17:38	F	17:38				
0	EP@TOT@S	$APTEXT	17:38	F	17:38				
0X	EPOQ		9: 2	D	9: 2	13: 1	D	13: 1	
0X	EQCP		22: 7	D	22: 7				
5	ER@TER	$APTEXT	18: 9						
0X	ESWP		8: 9	D	8: 9				
0X	ETOQ		8:34	D	8:34				
55	F$CREATE	$APTEXT	23: 3						
26	F$GDAL	$APTEXT	17:21						
46	F$MOSR	$APTEXT	9:39		17:34				
36	F$MPUT	$APTEXT	19:20						
7	F$PAUSE	$APTEXT	8:29						
27	F$RDAL	$APTEXT	17:28						
17	FC$CHNGE	$APTEXT	10:30						
33	FC$DSP	$APTEXT	11:38						
7	FC$EODR	$APTEXT	12:12						
31	FC$EOFR	$APTEXT	12:10						
30	FC$EORR	$APTEXT	12:10						
4	FC$FRLE	$APTEXT	12:18						
3	FC$MOUNT	$APTEXT	21: 7						
32	FC$NOOP	$APTEXT	12:12						
1	FC$READ	$APTEXT	12: 5						
2	FC$WRITE	$APTEXT	12: 5						

1	FL$FRP@M	$APTEXT	22:31	F									
1	FL$FRP@N	$APTEXT	22:31	F	22:31								
4	FL$FRP@P	$APTEXT	22:31	F	22:31								
4	FL$FRP@S	$APTEXT	22:31	F	22:31								
1	FN$DALL	$APTEXT	15:14										
0	FS$WAIT	$APTEXT	9:39		17:34								
0X	HOFF		8:25	D	8:25								
0X	HONN		20:31	D	20:31								
0X	HPUT		12:23	D	12:23		23:15	D	23:15				
0X	IDALSND		16:36	D	16:36		18:25	D	18:25				
0X	MBAQ		5: 3	D	6:25	D	9:31						
0X	MBBQ		5: 4	D	6:23	D							
0X	MBCQ		5: 5	D	6:24	D							
0X	MBDQ		5: 6	D	6:11	D	6:17	D					
13	MC$CPMLT	$APTEXT	16:39										
6	MC$CPRDN	$APTEXT	13: 7		14:15								
3	MC$CPURD	$APTEXT	12:32										
4	MC$CPUWR	$APTEXT	12:32										
7	MC$CPWDN	$APTEXT	13: 7										
5	MC$CSTAT	$APTEXT	14: 4		15:41		16:13		16:23		22:22		
10	MC$CSTRL	$APTEXT	16:13										
1	MC$MSTIO	$APTEXT	10: 3										
2	MC$RLDAL	$APTEXT	18:38		19:11		20: 6						
14	MC$RLMOS	$APTEXT	19: 7		19:11		19:16		20:12				
0	MIOP	$APTEXT	7: 7		16:17		16:41		19:17				
4	MQ$BMXO	$APTEXT	9:26										
0X	O$BCOM		1: 4	D	1: 4								
0X	O$BMXOPE		5: 7	D	11:30								
0X	O$BMXSIO		5: 8	D	22:32								
0X	O$BUFMAN		5: 9	D	15:12								
0X	O$CONMAN		5:10	D	10:32								
17	OV@NUM@N	$APTEXT	1: 4										
1	OV@TYP@N	$APTEXT	1: 4										
0	PT$HUH	$APTEXT	7:16		10: 7		11:36		12:37		13: 2	13:11	13:18	16:17
			16:41		19:17		19:37		23: 4				
0X	PUTDAL		19:31	D	19:31								
4	QU$POP	$APTEXT	7:42		8:18								
104	RQ$BMXO	$APTEXT	16:24										
41461	RQ$CPU	$APTEXT	17:36										
103	RQ$ERRR	$APTEXT	16:26		17:35								
21	RQ$HIO	$APTEXT	22:34										
11	SM@FSI@N	$APTEXT	1: 4										

7	SM@NUM@N	$APTEXT	1: 4						
177776	ST@DAL@X	$APTEXT	11: 4						
1	ST@DTR@M	$APTEXT	15:35 F						
1	ST@DTR@N	$APTEXT	15:35 F	15:35					
32	ST@DTR@P	$APTEXT	15:35 F	15:35					
1	ST@DTR@S	$APTEXT	15:35 F	15:35					
0X	TDMQ		5:11 D	13: 3					
30	TF$TEX0	$APTEXT	10:24						
31	TF$TEX1	$APTEXT	13:40						
0X	TIMQU		5:12 D	23:17					
1	TQ@DBF@M	$APTEXT	16: 7						
1	TQ@DBF@N	$APTEXT	14:37	14:37 F	16: 6	16: 6 F	16: 7 F	16: 7	
33	TQ@DBF@P	$APTEXT	14:37 F	14:37	16: 6 F	16: 6	16: 7 F	16: 7	
0	TQ@DBF@S	$APTEXT	14:37	14:37 F	16: 6	16: 6 F	16: 7 F	16: 7	
20	TQ@DDF@N	$APTEXT	21: 4	21: 4 F					
17	TQ@DDF@P	$APTEXT	21: 4 F	21: 4					
0	TQ@DDF@S	$APTEXT	21: 4	21: 4 F					
1	TQ@DIA@N	$APTEXT	17: 6	17: 6 F					
16	TQ@DIA@P	$APTEXT	17: 6 F	17: 6					
0	TQ@DIA@S	$APTEXT	17: 6	17: 6 F					
20	TQ@DID@N	$APTEXT	16:30	16:30 F	16:31 F	17:36 F			
10	TQ@DID@P	$APTEXT	16:30 F	16:30	16:31 F	16:31	17:36 F	17:36	
0	TQ@DID@S	$APTEXT	16:30	16:30 F	16:31 F	17:36 F			
10	TQ@DIU@N	$APTEXT	17: 7	17: 7 F					
10	TQ@DIU@P	$APTEXT	17: 7 F	17: 7					
0	TQ@DIU@S	$APTEXT	17: 7	17: 7 F					
177	TQ@DVN@M	$APTEXT	20:41						
7	TQ@DVN@N	$APTEXT	20:41						
15	TQ@DVN@P	$APTEXT	20:41 F	20:41					
11	TQ@DVN@S	$APTEXT	20:41	20:41 F					
77777	TQ@FCN@M	$APTEXT	20:40						
17	TQ@FCN@N	$APTEXT	20:40						
16	TQ@FCN@P	$APTEXT	20:40 F	20:40					
1	TQ@FCN@S	$APTEXT	20:40	20:40 F					
20	TQ@MOS@N	$APTEXT	15:40 F						
37	TQ@MOS@P	$APTEXT	15:40 F	15:40					
0	TQ@MOS@S	$APTEXT	15:40 F						
20	TQ@MSI@N	$APTEXT	17: 4	17: 4 F	19:19	19:19 F			
25	TQ@MSI@P	$APTEXT	17: 4 F	17: 4	19:19 F	19:19			
0	TQ@MSI@S	$APTEXT	17: 4	17: 4 F	19:19	19:19 F			
20	TQ@MSU@N	$APTEXT	17: 3	17: 3 F	19:18	19:18 F			
24	TQ@MSU@P	$APTEXT	17: 3 F	17: 3	19:18 F	19:18			

0	TQ@MSU@S	$APTEXT	17: 3	17: 3 F	19:18	19:18 F				
777	TQ@PWC@M	$APTEXT	16: 3	16: 5						
11	TQ@PWC@N	$APTEXT	16: 3	16: 5 F						
33	TQ@PWC@P	$APTEXT	16: 3 F	16: 3	16: 5 F	16: 5				
7	TQ@PWC@S	$APTEXT	16: 3	16: 3 F	16: 5 F					
177000	TQ@PWC@X	$APTEXT	16: 5							
20	TQ@RBC@N	$APTEXT	21: 1	21: 1 F						
26	TQ@RBC@P	$APTEXT	21: 1 F	21: 1						
0	TQ@RBC@S	$APTEXT	21: 1	21: 1 F						
20	TQ@REQ@N	$APTEXT	17: 5	17: 5 F						
26	TQ@REQ@P	$APTEXT	17: 5 F	17: 5						
0	TQ@REQ@S	$APTEXT	17: 5	17: 5 F						
20	TQ@RQN@N	$APTEXT	17: 9	17: 9 F						
13	TQ@RQN@P	$APTEXT	17: 9 F	17: 9						
0	TQ@RQN@S	$APTEXT	17: 9	17: 9 F						
20	TQ@RSC@N	$APTEXT	20:42	20:42 F						
27	TQ@RSC@P	$APTEXT	20:42 F	20:42						
0	TQ@RSC@S	$APTEXT	20:42	20:42 F						
20	TQ@SID@N	$APTEXT	16:29	16:29 F	16:32 F					
11	TQ@SID@P	$APTEXT	16:29 F	16:29	16:32 F	16:32				
0	TQ@SID@S	$APTEXT	16:29	16:29 F	16:32 F					
377	TQ@SIL@M	$APTEXT	16:33	17:35 F						
10	TQ@SIL@N	$APTEXT	16:33 F	17:35 F						
11	TQ@SIL@P	$APTEXT	16:33 F	16:33	17:35 F	17:35				
10	TQ@SIL@S	$APTEXT	16:33 F	17:35 F						
177400	TQ@SIL@X	$APTEXT	16:33	17:35						
377	TQ@SIU@M	$APTEXT	18: 2	18: 8 F						
10	TQ@SIU@N	$APTEXT	18: 2 F	18: 2	18: 8 F	18: 8				
11	TQ@SIU@P	$APTEXT	18: 2 F	18: 2	18: 8 F	18: 8				
0	TQ@SIU@S	$APTEXT	18: 2 F	18: 2	18: 8 F	18: 8				
20	TQ@STS@N	$APTEXT	10:31 F	11:24 F	11:39 F	21: 5	21: 5 F	22:21 F		
32	TQ@STS@P	$APTEXT	10:31 F	10:31	11:24 F	11:24	11:39 F	11:39	21: 5 F	21: 5
			22:21 F	22:21						
0	TQ@STS@S	$APTEXT	10:31 F	11:24 F	11:39 F	21: 5	21: 5 F	22:21 F		
20	TQ@TBC@N	$APTEXT	15:33 F	21: 3	21: 3 F					
30	TQ@TBC@P	$APTEXT	15:33 F	15:33	21: 3 F	21: 3				
0	TQ@TBC@S	$APTEXT	15:33 F	21: 3	21: 3 F					
20	TQ@TSC@N	$APTEXT	15:32 F	21: 2	21: 2 F					
31	TQ@TSC@P	$APTEXT	15:32 F	15:32	21: 2 F	21: 2				
0	TQ@TSC@S	$APTEXT	15:32 F	21: 2	21: 2 F					
77	TQ@UBC@M	$APTEXT	16: 2	16: 4						
6	TQ@UBC@N	$APTEXT	16: 2	16: 2 F	16: 4 F	16: 4				

33	TQ@UBC@P	$APTEXT	16: 2 F	16: 2	16: 4 F	16: 4
1	TQ@UBC@S	$APTEXT	16: 2	16: 2 F	16: 4 F	16: 4
20	TQ@VMS@N	$APTEXT	15:34 F			
35	TQ@VMS@P	$APTEXT	15:34 F	15:34		
0	TQ@VMS@S	$APTEXT	15:34 F			
0X	XDEV		5:13 D	11:15		
0X	XDEVMAX		5:14 D	10:42		
3	XIOP	$APTEXT	7:13	10: 7	13:11	

SYMBOL QUALIFIER IS BCOM

1	$CONST	10:31 D	10:31 F	11:24 D	11:24 F	11:39 D	11:39 F	12:40 D	12:40 F
		14: 4 D	14: 4 F	14: 4	14: 6 D	14: 6 F	14: 6	14:40 D	14:40 F
		15:35 D	15:35 F	15:35	15:41 D	15:41 F	15:41	17:35 D	17:35 F
		17:35	17:36 D	17:36 F	17:36	18: 8 D	18: 8 F	18: 8	18: 9 D
		18: 9 F	18: 9	18:38 D	18:38 F	18:38	19: 7 D	19: 7 F	19: 7
		20: 6 D	20: 6 F	20: 6	22:22 D	22:22 F	22:22	22:31 D	22:31 F
		22:31							
473	$REGORG	3:16 D	3:16 F	3:17	3:17 D	3:18	3:18 D	3:19	3:19 D
		3:20	3:20 D	3:21	3:21 D	3:21 F	3:22	3:22 D	3:23
		3:23 D	3:24	3:24 D	3:25	3:25 D	3:26	3:26 D	3:27
		3:27 D	3:27 F	3:28	3:28 D	3:29	3:29 D	3:30	3:30 D
		3:31	3:31 D	3:32	3:32 D	3:32 F	3:33	3:33 D	3:34
		3:34 D	3:35	3:35 D	3:36	3:36 D	3:37	3:37 D	3:38
		3:38 D	3:39	3:39 D	3:40	3:40 D	3:41	3:41 D	3:42
		3:42 D	4: 1	4: 1 D	4: 2	4: 2 D	4: 3	4: 3 D	4: 4
		4: 4 D	4: 5	4: 5 D	4: 6	4: 6 D	4: 7	4: 7 D	4: 8
		4: 8 D	4: 9	4: 9 D	4:10	4:10 D	4:11	4:11 D	4:12
		4:12 D	4:13	4:13 D	4:14	4:14 D	4:15	4:15 D	4:16
		4:16 D	4:17	4:17 D	4:18	4:18 D	4:19	4:19 D	4:20
		4:20 D	4:21	4:21 D	4:22	4:22 D	4:23	4:23 D	4:24
		4:24 D	4:25	4:25 D	4:25 F	8:29	9:39	17:21	17:28
		17:34	19:20	23: 3					
0	%NP	1: 4	3:32 D						
430	%P	1: 4	3:32 D	3:32					
410	%S1	3:16	3:17 D	7:42	8:18	8:39	9: 4	9: 7	9: 8
		9:36	10: 1	10:31	10:42	11:15	11:16	11:18	11:24
		11:35	11:39	12:40	13:16	13:17	13:28	14: 4	14: 6
		14:21	14:29	14:37	14:40	15: 7	15: 8	15: 9	15:26
		15:32	15:33	15:34	15:35	15:37	15:38	15:40	15:41
		16: 2	16: 3	16: 4	16: 5	16: 6	16: 7	16:10	16:29
		16:30	16:31	16:32	16:33	17: 3	17: 4	17: 5	17: 6

		17: 7	17: 8	17: 9	17:20	17:35	17:36	17:37	17:38
		18: 2	18: 4	18: 6	18: 8	18: 9	18:38	19: 7	19:18
		19:19	19:28	19:29	20: 6	20:10	20:40	20:41	20:42
		21: 1	21: 2	21: 3	21: 4	21: 5	22:21	22:22	22:31
411	%S2	3:16	3:18 D						
412	%S3	3:16	3:19 D						
413	%S4	3:16	3:20 D						
414	%S5	3:16	3:21 D						
415	%T1	3:22 D	17:33						
416	%T2	3:23 D	17:33						
417	%T3	3:24 D							
420	%T4	3:25 D							
421	%T5	3:26 D							
422	%T6	3:27 D							
423	%W1	3:28 D	11:21	11:23	16:29	16:31	17:24	17:25	17:26
		17:27	18:20	18:22	18:23	18:24			
424	%W2	3:29 D	16:30	16:32	17:19	18:13	18:14		
425	%W3	3:30 D	17:18	17:20	17:37	18:15			
426	%W4	3:31 D	17:16	17:34	18:12				
427	%W5	3:32 D	17:17	17:34	18: 3	18: 4	18: 5	18: 6	18:12
2305	BADDO	11: 5	22:18 D						
6	BEGIN	7: 5 D	8:30						
2401	BYPASS	12: 6	12:11	12:13	23:11 D				
10	CHKPT	6:28 D	7:27						
2353	CREATE	10:35	11:33	15:17	22:37	23: 2 D			
445	DAL	4: 4 D	8:26	9:39	10: 1	10:31	10:34	11:24	11:32
		11:39	12:21	13:16	14: 4	14:37	15:16	15:32	15:33
		15:34	15:35	15:40	15:41	16: 2	16: 3	16: 6	16:29
		16:30	16:31	16:32	16:33	16:35	17: 3	17: 4	17: 5
		17: 6	17: 7	17: 8	17: 9	17:19	17:24	18:20	18:38
		19: 7	19:18	19:19	19:28	19:29	20: 6	20: 9	20:10
		20:30	20:40	20:41	20:42	21: 1	21: 2	21: 3	21: 4
		21: 5	21:21	22:21	22:22	23:13			
2143	DALRET	11: 6	13: 5	14: 7	16:11	19: 9	19:26	20:28 D	
2153	DALSTRIP	10:12	13:27	16:18	20:39 D				
467	DIA	4:22 D	17: 6	18: 1	18:35				
463	DPT	4:18 D	17:21	17:26	17:28	17:33	17:34	17:35	17:36
		17:37	17:38	18: 2	18: 4	18: 6	18: 8	18: 9	18:13
		18:14							
456	DVC	4:13 D	10:24	11: 3	11:15	13:40	20:41	22:36	
450	FCT	4: 7 D	10: 1	10: 3	12:32	13: 7	14:15	16:13	16:23
		16:39	19:11	19:16					

457	FLG	4:14 D	10:24	21: 4					
451	FUN	4: 8 D	10:24	10:30	11:38	12: 5	12:10	12:12	12:18
		20:40	21: 7						
2325	HALTTO	12:18	22:30 D						
447	TOP	4: 6 D	9:27	9:31	20:19				
462	LEN	4:17 D	17:10	17:34	18:12				
2106	MOSREL	10: 8	13:12	16:37	18:39	20: 5 D			
2113	MOSREL2	19: 8	20: 7 D						
446	MSG	4: 5 D	8:35	9:26	9:33	12:41			
461	MSL	4:16 D	17: 4	17:17	19:19	19:20			
460	MSU	4:15 D	17: 3	17:16	19:18	19:20			
54	NEXT	7:26 D	8: 2	8:15	9:13	10:36	11: 7	12:31	13: 6
		14: 8	16:12	16:38	18:40	19:10	19:32	21:37	
455	NUMDONE	4:12 D	7:20	7:27	9:18	21:33			
466	OFF	4:21 D	17:12	17:33	17:34				
2265	POPPER	12:26	13: 4	13:23	21:42 D	23:18			
465	PRT	4:20 D							
2420	QTBLE0	6:10 D	7: 8						
2422	QTBLE1	6:16 D	7:10	7:12					
2424	QTBLE2	6:22 D	7:14						
454	QUE	4:11 D	7:30	7:34	7:35	7:36	7:39	8:11	8:18
		8:33	8:39	9: 1	9: 7	12:25	13: 3	13:22	22: 1
		22: 3	22: 5	23:17					
453	QUPT	4:10 D	7:19	7:30	7:33	7:36	8: 5	8:14	9: 3
		9:12							
452	QUTBLE	4: 9 D	7: 8	7:10	7:12	7:14	7:19	7:33	
437	RBC	3:40 D	10:24	21: 1	21: 9				
2002	RELMOS	17:30	19: 6 D						
471	REQ	4:24 D	17: 9	18: 3	18: 5				
2241	RESPOND	14: 5	15:42	21:19 D	22:23				
436	RSC	3:39 D	10:24	20:42	21:10				
472	SLG	4:25 D	17: 8	18:37					
443	STAT	4: 2 D	13:40	14: 3	21: 5	21: 8			
441	TBC	3:42 D	13:40	14:16	15:28	15:33	21: 3		
442	TCB	4: 1 D	11:35	11:36	12:22	12:25	13:16	13:17	13:22
		13:28	14: 6	14:21	14:29	14:40	15: 7	15: 8	15: 9
		15:15	15:26	16:10	22:31	22:35			
470	TID	4:23 D	17: 7	18: 2					
464	TOT	4:19 D	17: 5	17:20	17:38				
440	TSC	3:41 D	13:40	14:21	14:27	14:28	14:30	14:33	15: 7
		15:28	15:32	21: 2					
444	VMS	4: 3 D	13:28	13:40	14:16	14:27	14:28	14:30	14:33

		15:34	16:10						
430	X0	3:33 D	7:39	7:42	9: 3	9: 4	9:32	9:35	9:36
		11: 4	11:16	11:17	11:18	11:21	11:35	14:28	14:29
		15: 8	15:11	15:26	16: 4	16: 5	16: 7	16:24	16:26
		16:33	22:19	22:21					
431	X1	3:34 D	9:34	9:38	10:32	11:30	15: 9	15:11	15:12
		15:36	15:37	15:38	16: 2	16: 4	16: 6	16: 7	22:32
		23: 3							
432	X2	3:35 D	9:35	9:38	10:33	11:31	15:13	15:37	15:39
		15:40	16: 3	16: 5	22:33	23: 3			
433	X3	3:36 D	9:37	9:38	10:34	11:32	15:14	15:38	15:39
		22:34	23: 3						
434	X4	3:37 D	9:38	9:39	15:15	22:35	23: 3		
435	X5	3:38 D	9:38	9:39	15:16	22:36	23: 3		
2253	YIELD	7:27	21:32 D						

```
0     <macro>                    OVERLAY   IDEM                                                    IDEM.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'                OVERLAY  1
```

```
*****************************************************************      TDEM.5
* TDEM IS ACTIVATED TO MOVE DATA BETWEEN LOCAL MEMORY AND MOS IN     *   TDEM.6
* TO INTERRUPTS RECEIVED FROM TAPE CHANNELS ON THE BLOCK MUX. INPUT  *   TDEM.7
* IS THE ADDRESS OF A TAPE DEVICE TABLE WHICH CONTAINS ALL PERTINANT *   TDEM.8
* INFORMATION TO PROCESS THE DATA. IF THE DEVICE IS DOING A WRITE TO *   TDEM.9
* TAPE, DATA IS MOVED FROM MOS TO A LOCAL BUFFER. IF DOING A READ,   *   TDEM.10
* DATA IS MOVED FROM A LOCAL BUFFER TO AN MOS BUFFER. IF THE DATA IS *   TDEM.11
* IN INTERCHANGE FORMAT, THE DATA IS BLOCKED ON THE WAY TO MOS.      *   TDEM.12
*                                                                    *   TDEM.13
* POINTERS RELATING TO THE LOCAL BUFFER ARE OBTAINED FROM THE CRW    *   TDEM.14
* CONTAINED IN TAPE DEVICE TABLE. POINTERS RELATING TO THE MOS BUFFER *   TDEM.15
* ARE FOUND IN A DATASET CONTROL TABLE LOCATED IN MOS. ITS ADDRESS IS *   TDEM.16
* IN THE TAPE DEVICE TABLE.                                          *   TDEM.17
*                                                                    *   TDEM.18
*                        TAPE DEVICE TABLE                           *   TDEM.19
*                                                                    *   TDEM.20
*          ***************************************************       *   TDEM.21
*          *                                                 *       *   TDEM.22
*    0-11  *                CHANNEL STATUS WORD              *       *   TDEM.23
*          *                                                 *       *   TDEM.24
*          ***************************************************       *   TDEM.25
*      12  *    DC@DSU: POINTER TO MOS TABLE                 *       *   TDEM.26
*          ***************************************************       *   TDEM.27
*      13  *    DC@DSL:                                      *       *   TDEM.28
*          ***************************************************       *   TDEM.29
*      14  *    DC@BFA: DATA BUFFER 'A'                      *       *   TDEM.30
*          ***************************************************       *   TDEM.31
*      15  *    DC@BFB: DATA BUFFER 'B'                      *       *   TDEM.32
*          ***************************************************       *   TDEM.33
*      16  *    DC@BFZ: DATA BUFFER SIZE                     *       *   TDEM.34
*          ***************************************************       *   TDEM.35
*      17  *                     FLAGS                       *       *   TDEM.36
*          ***************************************************       *   TDEM.37
*      18  *    DC@DAL: CURRENT DAL ADDRESS                  *       *   TDEM.38
*          ***************************************************       *   TDEM.39
*   19-20  *    DC@QUA: REQUEST WAIT QUEUE                   *       *   TDEM.40
*          ***************************************************       *   TDEM.41
*   21-22  *    DC@QUB: BIOP WAIT QUEUE                      *       *   TDEM.42
*          ***************************************************       *   TDEM.43
*      23  *    DC@DHU: HOLD-DATA MOS TABLE ADDRESS          *       *   TDEM.44
*          ***************************************************       *   TDEM.45
*      24  *    DC@DHL:                                      *       *   TDEM.46
```

```
*            ************************************************          *    TDEM.47
*    25-27   *   DC@MSG: REQUEST QUEUE                        *          *    TDEM.48
*            ************************************************          *    TDEM.49
*       28   *   DC@BRQ: OUTSTANDING BIOP REQUEST COUNT       *          *    TDEM.50
*            ************************************************          *    TDEM.51
*       29   *   DC@HSU: CPU DATA ADDRESS                     *          *    TDEM.52
*            ************************************************          *    TDEM.53
*       30   *   DC@HSL:                                      *          *    TDEM.54
*            ************************************************          *    TDEM.55
*       31   *   DC@TSC: TRANSFERRED SECTOR COUNT             *          *    TDEM.56
*            ************************************************          *    TDEM.57
*       32   *   DC@TBC: TRANSFERRED BLOCK COUNT              *          *    TDEM.58
*            ************************************************          *    TDEM.59
*       33   *   DC@VMS: VALID MOS SECTOR COUNT               *          *    TDEM.60
*            ************************************************          *    TDEM.61
*       34   *   DC@TMS: TOTAL MOS SECTOR COUNT               *          *    TDEM.62
*            ************************************************          *    TDEM.63
*       35   *   DC@RSC: REQ. SCTS   *   DC@RBC: REQ. BLKS    *          *    TDEM.64
*            ************************************************          *    TDEM.65
*       36   *   DC@BLK: CURRENT BLOCK COUNT                  *          *    TDEM.66
*            ************************************************          *    TDEM.67
*       37   *   DC@BCM: REVERSE READ TABLE POINTER           *          *    TDEM.68
*            ************************************************          *    TDEM.69
*    40-51   *                CPW LIST (DC@CPM)               *          *    TDEM.70
*            ************************************************          *    TDEM.71
*    52-71   *               CONTROL LIST 0 (DC@C0)           *          *    TDEM.72
*            ************************************************          *    TDEM.73
*    72-91   *               CONTROL LIST 1 (DC@C1)           *          *    TDEM.74
*            ************************************************          *    TDEM.75
*    92-95   *        ACTIVITY POINTER/QUEUE (DC@PRW)         *          *    TDEM.76
*            ************************************************          *    TDEM.77
*                                                                      *    TDEM.78
*    SEE COMTEX FOR DEFINITIONS                                        *    TDEM.79
************************************************************************    TDEM.80

6   <MACRO>          REGDEFS   ,,(CWRD,ENTRY,CUR,TCB,DST,DSU,DSL,DAL,RSC,HSU,HSL,      TDEM.82
                     ,         TMS,BSZ,FUNC,FMT,OST,COT,COB,C1T,C1B,DSCLIM,ADDU,       TDEM.83
                     ,         ADDL,SIZ,LOC,FCT,BPT,PTR,NCW,FWI,DATA,CPW,BUFF,CNT,     TDEM.84
                     ,         UBC,STAT,MODE,PBC,OFFSET,DONE,MSU,MSL,ENTU,ENTL,SENT,   TDEM.85
                     ,         NUL,P0,P1,P2,P3,P4,P5,DBI)                              TDEM.86
410         %S1      EQUALS    $REGORG                                                     $REG    3
```

```
411     %S2       EQUALS    $REGORG                                                    $REG     3
412     %S3       EQUALS    $REGORG                                                    $REG     3
413     %S4       EQUALS    $REGORG                                                    $REG     3
414     %S5       EQUALS    $REGORG                                                    $REG     3
415     %T1       EQUALS    $REGORG                                                    $REG     3
416     %T2       EQUALS    $REGORG                                                    $REG     3
417     %T3       EQUALS    $REGORG                                                    $REG     3
420     %T4       EQUALS    $REGORG                                                    $REG     3
421     %T5       EQUALS    $REGORG                                                    $REG     3
422     %T6       EQUALS    $REGORG                                                    $REG     3
423     %W1       EQUALS    $REGORG                                                    $REG     3
424     %W2       EQUALS    $REGORG                                                    $REG     3
425     %W3       EQUALS    $REGORG                                                    $REG     3
426     %W4       EQUALS    $REGORG                                                    $REG     3
427     %W5       EQUALS    $REGORG                                                    $REG     3
430     CWRD      EQUALS    $REGORG                                                    $REG     3
431     ENTRY     EQUALS    $REGORG                                                    $REG     3
432     CUR       EQUALS    $REGORG                                                    $REG     3
433     TCB       EQUALS    $REGORG                                                    $REG     3
434     DST       EQUALS    $REGORG                                                    $REG     3
435     DSU       EQUALS    $REGORG                                                    $REG     3
436     DSL       EQUALS    $REGORG                                                    $REG     3
437     DAL       EQUALS    $REGORG                                                    $REG     3
440     RSC       EQUALS    $REGORG                                                    $REG     3
441     HSU       EQUALS    $REGORG                                                    $REG     3
442     HSL       EQUALS    $REGORG                                                    $REG     3
443     TMS       EQUALS    $REGORG                                                    $REG     3
444     BSZ       EQUALS    $REGORG                                                    $REG     3
445     FUNC      EQUALS    $REGORG                                                    $REG     3
446     FMT       EQUALS    $REGORG                                                    $REG     3
447     OST       EQUALS    $REGORG                                                    $REG     3
450     C0T       EQUALS    $REGORG                                                    $REG     3
451     C0B       EQUALS    $REGORG                                                    $REG     3
452     C1T       EQUALS    $REGORG                                                    $REG     3
453     C1B       EQUALS    $REGORG                                                    $REG     3
454     DSCLIM    EQUALS    $REGORG                                                    $REG     3
455     ADDU      EQUALS    $REGORG                                                    $REG     3
456     ADDL      EQUALS    $REGORG                                                    $REG     3
457     SIZ       EQUALS    $REGORG                                                    $REG     3
460     LOC       EQUALS    $REGORG                                                    $REG     3
461     FCT       EQUALS    $REGORG                                                    $REG     3
462     BPT       EQUALS    $REGORG                                                    $REG     3
```

```
463     PTR     EQUALS  $REGORG                                         $REG    3
464     NCW     EQUALS  $REGORG                                         $REG    3
465     FWI     EQUALS  $REGORG                                         $REG    3
466     DATA    EQUALS  $REGORG                                         $REG    3
467     CPW     EQUALS  $REGORG                                         $REG    3
470     BUFF    EQUALS  $REGORG                                         $REG    3
471     CNT     EQUALS  $REGORG                                         $REG    3
472     UBC     EQUALS  $REGORG                                         $REG    3
473     STAT    EQUALS  $REGORG                                         $REG    3
474     MODE    EQUALS  $REGORG                                         $REG    3
475     PBC     EQUALS  $REGORG                                         $REG    3
476     OFFSET  EQUALS  $REGORG                                         $REG    3
477     DONE    EQUALS  $REGORG                                         $REG    3
500     MSU     EQUALS  $REGORG                                         $REG    3
501     MSL     EQUALS  $REGORG                                         $REG    3
502     ENTU    EQUALS  $REGORG                                         $REG    3
503     ENTL    EQUALS  $REGORG                                         $REG    3
504     SENT    EQUALS  $REGORG                                         $REG    3
505     NUL     EQUALS  $REGORG                                         $REG    3
506     P0      EQUALS  $REGORG                                         $REG    3
507     P1      EQUALS  $REGORG                                         $REG    3
510     P2      EQUALS  $REGORG                                         $REG    3
511     P3      EQUALS  $REGORG                                         $REG    3
512     P4      EQUALS  $REGORG                                         $REG    3
513     P5      EQUALS  $REGORG                                         $REG    3
514     DBI     EQUALS  $REGORG                                         $REG    3

                EXT     BXQQ                                          TDEM.88
                EXT     CA                                            TDEM.89
                EXT     CB                                            TDEM.90
                EXT     CC                                            TDEM.91
                EXT     CD                                            TDEM.92
                EXT     CE                                            TDEM.93
                EXT     DS                                            TDEM.94
                EXT     ED                                            TDEM.95
                EXT     EH                                            TDEM.96
                EXT     HB                                            TDEM.97
                EXT     HC                                            TDEM.98
                EXT     TDEMMEM                                       TDEM.99
                EXT     XCHT                                      R16767CH.1
                                                                  R16767CH.2
```

```
                    40      MEMSIZ   EQUALS    CU@@LE+BF@@LE+CW@@LE                                   TDEM.101
                100000      NOSTAT   EQUALS    100000              .STATUS IF PROCESSING DBLE INT.     TDEM.102
                                                                                                      TDEM.103
                                     *                                                                TDEM.104
                                     * ALLOCATE RESOURCES                                             TDEM.105
                                     *                                                                TDEM.106
                                                                                                      TDEM.107
                                     * LOCAL MEMORY SPACE                                             TDEM.108
                                                                                                      TDEM.109
  6   014000 /000000  024410         R!CWRD = (TDEMMEM)          .TDEM MEMORY ADDRESS                TDEM.110
      030410  024430
 13    <macro>                       $IF (A = 0)                 .IF NOT ALLOCATED                   TDEM.111
                                                                                  .>>>>> 0               $IF      1
 15    <macro>                         GETMEM MEMSIZ,R!CWRD      .ALLOCATE                           TDEM.112
 35    <macro>                         $PUNTIF (A # 0)           .IF NOT AVAILABLE                   TDEM.113
 37   014000 /000000  024410           (TDEMMEM) = R!CWRD        .SAVE ADDRESS                       TDEM.114
      020430  034410
 44    <macro>                       $ENDIF                                                          TDEM.115
                                                                                  .<<<<< 0               $ENDIF   1
 44   020430  012004  024431         R!ENTRY = R!CWRD + CW@@LE   .DESCRIPTOR WORD                    TDEM.116
 47   020431  012004  024432         R!CUR = R!ENTRY + BF@@LE    .XIOP HEADER                        TDEM.117
                                                                                                      TDEM.118
                                     *                                                                TDEM.119
                                     * TAKE NEXT ENTRY OFF QUEUE                                      TDEM.120
                                     *                                                                TDEM.121
                                                                                                      TDEM.122
 52                          DEQUEUE *                                                                TDEM.123
 52   002000  000000                 I = 0                       .LOCK                               TDEM.124
 54   014000 /000000  024000         ED = BXQQ                   .QUEUE ADDRESS                      TDEM.125
 57   020000  012004  024410         A = (ED + QU$POP)           .COUNT OF ENTRIES                   TDEM.126
      030410
 63    <macro>                       $IF (A # 0)                 .IF ANY THERE                       TDEM.127
                                                                                  .>>>>> 0               $IF      1
 65    <macro>                         RKERN ETOQ                .GET THE NEXT ONE                   TDEM.128
 67   020000  024433                   R!TCB = EH                .ADDRESS OF TAPE CONTROL TABLE      TDEM.129
 71    <macro>                         $PUNTIF (A = 0)           .IF 0 ENTRY                         TDEM.130
 73    <macro>                       $ELSE                                                           TDEM.131
                                       * TERMINATE ACTIVITY                                           TDEM.132
                                                                                                      TDEM.133
 74    <macro>                         PKERN ESWP                .GO TO IDLE LOOP                    TDEM.134
 76    <macro>                       $ENDIF                                                          TDEM.135
                                                                                  .<<<<< 0               $ENDIF   1
```

```
 76   010000  024472          R!UBC = 0                    .UNUSED BYTE COUNT          TDEM.136
100   010000  024505          R!NUL = 0                    .FLAG FOR NULL RCW          TDEM.137
102   010000  024514          R!DBI = 0                    .DOUBLE INTERRUPT           TDEM.138
                                                                                       TDEM.139
                              *                                                        TDEM.140
                              * MOVE POINTERS FROM TCB TO REGISTERS                   TDEM.141
                              *                                                        TDEM.142
                                                                                       TDEM.143
104   003000                  I = 1                        .ENABLE INTERRUPTS          TDEM.144
105    <macro>                GET R!DST,CRW@DS,R!TCB       .DEVICE STATUS              TDEM.145
113    <macro>                GET R!OST,CRW@OS,R!TCB       .OPERATION STATUS           TDEM.146
121    <macro>                GET R!DSU,DC@DSU,R!TCB       .MOS TABLE ADDRESS          TDEM.147
126    <macro>                GET R!DSL,DC@DSL,R!TCB                                   TDEM.148
133    <macro>                GET R!DAL,DC@DAL,R!TCB       .LOCAL DAL ADDRESS          TDEM.149
140    <macro>                GET R!RSC,DC@RSC,R!TCB       .REQUESTED SECTOR COUNT     TDEM.150
145    <macro>                GET R!HSU,DC@HSU,R!TCB       .CPU ADDRESS                TDEM.151
152    <macro>                GET R!HSL,DC@HSL,R!TCB                                   TDEM.152
157    <macro>                GET R!TMS,DC@TMS,R!TCB       .TOTAL MOS SECTOR COUNT     TDEM.153
164    <macro>                GET R!BSZ,DC@BFZ,R!TCB       .BUFFER SIZE                TDEM.154
                                                                                       TDEM.155
                              * NEED SOME INFO FROM DAL                                TDEM.156
                                                                                       TDEM.157
171    <macro>                GET R!FUNC,TQ@FCN,R!DAL      .FUNCTION TYPE              TDEM.158
200    <macro>                GET R!FMT,DD@FRM,R!DAL       .DATA FORMAT                TDEM.159
                                                                                       TDEM.160
                              * POINTERS TO CPW LISTS                                  TDEM.161
                                                                                       TDEM.162
207    <macro>                ADDRESS R!COT,DC@COT,R!TCB   .TOP OF LIST 0              TDEM.163
212    <macro>                ADDRESS R!COB,DC@COB,R!TCB   .BOTTOM                     TDEM.164
215    <macro>                ADDRESS R!C1T,DC@C1T,R!TCB   .TOP OF LIST 1              TDEM.165
220    <macro>                ADDRESS R!C1B,DC@C1B,R!TCB   .BOTTOM                     TDEM.166
                                                                                       TDEM.167
                              *                                                        TDEM.168
                              * READ IN XIOP AREA OF DSC                               TDEM.169
                              *                                                        TDEM.170
                                                                                       TDEM.171
223   020435  024455          R!ADDU = R!DSU               .XIOP HEADER ADDRESS        TDEM.172
225   020436  024456          R!ADDL = R!DSL               .                           TDEM.173
227   010006  024457          R!SIZ = CU@WDS               .LENGTH                     TDEM.174
231   020432  024460          R!LOC = R!CUR                .LOCAL ADDRESS              TDEM.175
233   010004  024461          R!FCT = FN$MOSRD             .READ                       TDEM.176
235   077000 /003057          R = MEMIO                                                TDEM.177
```

```
                                                                                TDEM.178
                                 *                                              TDEM.179
                                 * MOVE POINTERS TO REGISTERS                   TDEM.180
                                 *                                              TDEM.181
                                                                                TDEM.182
237     <macro>                  GET R!BPT,CU@BPT,R!CUR        .BUFFER POSITION TDEM.183
244     <macro>                  GET R!PTR,CU@PTR,R!CUR        .BUFFER POINTER  TDEM.184
251     <macro>                  GET R!NCW,CU@NCW,R!CUR        .POINTER TO NEXT CONTROL WORD TDEM.185
256     <macro>                  GET R!FWI,CU@FWI,R!CUR        .FORWARD WORD INDEX TDEM.186
264     <macro>                  GET R!DSCLIM,CU@LIM,R!CUR     .LENGTH OF LIST  TDEM.187
                                                                                TDEM.188
                                 ******************                             TDEM.189
                                 *      TRACE     *                             TDEM.190
                                 *                *                             TDEM.191
                                 * 1. DEVICE      *                             TDEM.192
                                 * 2. STATUS      *                             TDEM.193
                                 * 3. POINTER     *                             TDEM.194
                                 * 4. OFFSET      *                             TDEM.195
                                 * 5. MOS SECTS   *                             TDEM.196
                                 ******************                             TDEM.197
                                                                                TDEM.198
271     <macro>                  GET R!%W1,CRW@DC,R!TCB        .DEVICE ORDINAL  TDEM.199
277     <macro>                  KRECORD TF$TEX5,(R!%W1,R!DST,R!PTR,R!BPT,R!TMS),INT=ON TDEM.200
                                                                                TDEM.201
                                 *                                              TDEM.202
                                 * READ: MOVE DATA FROM LOCAL TO MOS             TDEM.203
                                 *                                              TDEM.204
                                                                                TDEM.205
317     <macro>                  $IF (R!FUNC = FC$READ)                         TDEM.206
                                                                  .>>>>> 0          $IF      1
                                                                                TDEM.207
                                   *                                            TDEM.208
                                   * GET CPW - CONTAINS BUFFER ADDRESS AND LENGTH OF DATA TDEM.209
                                   *                                            TDEM.210
                                                                                TDEM.211
324     <macro>                    GET R!CPW,CRW@LC,R!TCB       .LAST CPW ADDRESS TDEM.212
                                                                                TDEM.213
                                   * CHECK FOR DOUBLE INTERRUPT                 TDEM.214
                                   * CHECK FOR COMMAND RETRY                    TDEM.215
                                   * - IF SET, BACK UP 1 CPW                    TDEM.216
                                                                                TDEM.217
331     <macro>                    GET A,FL$DBI,R!TCB          .DOUBLE INTERRUPT FLAG TDEM.218
```

```
337    <macro>                          $IF (A # 0)                  .IF SET                                  TDEM.219
                                                                                          .>>>>> 1                $IF      1
341    <macro>                            STORE 0,FL$DBI,R!TCB       .CLEAR FLAG                              TDEM.220
350    014000 /100000  024434             R!DST = NOSTAT             .DELAY DEVICE END                        TDEM.221
353    010001  024514                     R!DBI = 1                  .SET DOUBLE INTERRUPT                    TDEM.222
                                                                                                               TDEM.223
                                          * FIND WHICH CONTROL LIST CURRENT CPW                               TDEM.224
                                          * BELONGS TO                                                        TDEM.225
                                                                                                               TDEM.226
355    <macro>                            $IF (R!CPW > R!COT) AND (R!CPW <= R!COB)  .LIST 0?                  TDEM.227
                                                                                          .>>>>> 2                $IF      1
371    020450  024506                       R!P0 = R!COT             .SET BOUNDARIES                          TDEM.228
373    020451  024507                       R!P1 = R!COB                                                        TDEM.229
375    <macro>                            $ELSE                                                               TDEM.230
376    020452  024506                       R!P0 = R!C1T             .LIST 1                                  TDEM.231
400    020453  024507                       R!P1 = R!C1B                                                        TDEM.232
402    <macro>                            $ENDIF                                                              TDEM.233
                                                                                          .<<<<< 2                $ENDIF   1
402    010004  025506                     R!P0 = R!P0 + 4            .SPACE PAST HEADER WORD                  TDEM.234
                                                                                                               TDEM.235
                                          * BACK UP 1 CPW - CHECK FOR WRAP                                    TDEM.236
                                                                                                               TDEM.237
404    020467  013004  024467             R!CPW = R!CPW - 4                                                   TDEM.238
407    020467  023506  101003             R!CPW = R!P1, R!CPW < R!P0  .IF WRAPPED                             TDEM.239
       020507  024467
414    <macro>                          $ELSEIF (0 # R!DST & ST$MD) AND (0 # R!DST & ST$UC)                   TDEM.240
433    <macro>                            $IF (0 # R!DST & ST$CE)                                             TDEM.241
                                                                                          .>>>>> 2                $IF      1
442    <macro>                              GET R!%W1,FL$TIP,R!TCB                                            TDEM.242
451    <macro>                              $IF (R!%W1 # 0)                                                   TDEM.243
                                                                                          .>>>>> 3                $IF      1
454    075000 /001675                         P = UPDATE             .RETRY IN MID-RECORD                     TDEM.244
456    <macro>                              $ELSE                                                             TDEM.245
457    010004  024434                         R!DST = ST$DE          .ASSUME PREVIOUS STATUS DONE             TDEM.246
461    <macro>                              $ENDIF                                                            TDEM.247
                                                                                          .<<<<< 3                $ENDIF   1
461    <macro>                            $ENDIF                                                              TDEM.248
                                                                                          .<<<<< 2                $ENDIF   1
461    <macro>                          $ENDIF                                                                TDEM.249
                                                                                          .<<<<< 1                $ENDIF   1
                                                                                                               TDEM.250
                                          * GET BUFFER INFO                                                   TDEM.251
```

```
                                                                                                TDEM.252
461    <macro>                          GET R!BUFF,CPW@DA,R!CPW      .BUFFER ADDRESS                TDEM.253
466    020470  015000 /177774           R!BUFF = R!BUFF & 177774     .MASK LOWER BITS               TDEM.254
       024470
472    <macro>                          GET R!CNT,CPW@BL,R!CPW       .BYTES READ                    TDEM.255
477    <macro>                          PUT R!BSZ,CPW@BL,R!CPW       .REPLACE BYTE LENGTH IN CPW    TDEM.256
                                                                                                TDEM.257
                                        * COMPUTE UNUSED BYTE COUNT                             TDEM.258
                                                                                                TDEM.259
504    020471  011007  024472           R!UBC = R!CNT & 7                                       TDEM.260
507    010010  023472  011007           R!UBC = 10 - R!UBC & 7                                  TDEM.261
       024472
513    010000  024504                   R!SENT = 0                   .CLEAR SECTORS SENT FLAG       TDEM.262
                                                                                                TDEM.263
                                        * IF DELAY FLAG SET, CLEAR IT                           TDEM.264
                                        * FLUSH DELAYED SECTORS.                                TDEM.265
                                                                                                TDEM.266
515    <macro>                          GET R!P0,DC@ERP,R!TCB        .DELAY FLAG                    TDEM.267
524    <macro>                          $IF (R!P0 # 0)                                          TDEM.268
                                                                                .>>>>> 1            $IF      1
527    <macro>                            STORE 0,DC@ERP,R!TCB       .CLEAR FLAG                    TDEM.269
536    <macro>                            GET R!P4,DC@DMS,R!TCB      .GET DELAY COUNT               TDEM.270
543    020512  024513                     R!P5 = R!P4                .SAVE COUNT                    TDEM.271
545    <macro>                            $UNTIL (R!P4 = 0)          .IF ANY                        TDEM.272
547    077000 /002667                       R = BUFSEND              .FLUSH                         TDEM.273
551    027512                               R!P4 = R!P4 - 1          .DECREMENT                     TDEM.274
552    <macro>                            $ENDTIL                                                  TDEM.275
553    <macro>                            STORE 0,DC@DMS,R!TCB       .CLEAR COUNT                   TDEM.276
560    020443  022513  024443             R!TMS = R!TMS + R!P5       .ADD DELAYED COUNT TO TOTAL    TDEM.277
563    <macro>                          $ENDIF                                                  TDEM.278
                                                                                .<<<<< 1            $ENDIF   1
                                                                                                TDEM.279
                                        * GO TO COMMON READ/WRITE PROCESSING ROUTINE            TDEM.280
                                                                                                TDEM.281
563    077000 /002040                   R = COMSUB                                              TDEM.282
                                                                                                TDEM.283
                                        *                                                       TDEM.284
                                        * DETERMINE DEVICE STATUS                               TDEM.285
                                        *                                                       TDEM.286
                                                                                                TDEM.287
565    <macro>                          $IF (0 = R!DST & ST$DE)      .DEVICE END NOT SET            TDEM.288
                                                                                .>>>>> 1            $IF      1
```

```
574   010000  024473                R!STAT = MD$BCW                                          TDEM.289
576    <macro>                     $ELSE                                                    TDEM.290
577   010010  024473                R!STAT = MD$EOR          .END OF RECORD                  TDEM.291
601    <macro>                      STORE 1,DC@CHK,R!TCB     .SET CHECKSUM FLAG FOR NEXT BLK  TDEM.292
612    <macro>                     $ENDIF                                                   TDEM.293
                                                                          .<<<<< 1              $ENDIF    1
                                                                                               TDEM.294
                                   * DONE IF END OF FILE                                       TDEM.295
                                                                                               TDEM.296
612   020434  011001  024410       P = UPDATE, 0 # R!DST & ST$UE .UNIT EXCEPTION               TDEM.297
      010000  023410  127000
     /001675
                                                                                                  .
                                                                                               TDEM.298
                                   * CHECK FOR DATA DISCARD FLAG                               TDEM.299
                                   * CHECK FOR LARGE BLOCK ENCOUNTERED                         TDEM.300
                                   * BYPASS DATA TRANSFER IF ALREADY SET                       TDEM.301
                                                                                               TDEM.302
621    <macro>                     GET A,DC@BIG,R!TCB       .FLAG                              TDEM.303
627   127000 /001675               P = UPDATE, A # 0        .IF SET                            TDEM.304
                                                                                               TDEM.305
                                   *                                                           TDEM.306
                                   * TRANSPARENT DATA                                          TDEM.307
                                   *                                                           TDEM.308
                                                                                               TDEM.309
631    <macro>                     $IF (R!FMT = FM$TRNS)                                       TDEM.310
                                                                                  .>>>>> 1          $IF       1
634   020471  126000 /001675         P = UPDATE, R!CNT = 0    .IF NOTHING TO TRANSFER            TDEM.311
637   020472  024475                 R!PBC = R!UBC            .SAVE AS PARTIAL BYTE COUNT        TDEM.312
641   077000 /002511                 R = BUFGET               .GET MOS BUFFER ADDRESS            TDEM.313
643    <macro>                       DADD R!MSU,R!MSL,0,R!BPT,R!ADDU,R!ADDL  .START ADDRESS     TDEM.314
653   020470  024460                 R!LOC = R!BUFF           .LOCAL DATA BUFFER                 TDEM.315
655   020471  012007  004003         R!SIZ = R!CNT + 7 > 3    .WORD COUNT                        TDEM.316
      024457
661   010005  024461                 R!FCT = FN$MOSWR         .WRITE TO MOS                      TDEM.317
663   077000 /003057                 R = MEMIO                                                   TDEM.318
665   020462  022457  024462         R!BPT = R!BPT + R!SIZ    .UPDATE BUFFER POINTER             TDEM.319
                                                                                               TDEM.320
                                     * CHECK FOR BUFFER FULL OR                                TDEM.321
                                     * END OF RECORD.                                          TDEM.322
                                                                                               TDEM.323
670    <macro>                       $IF (R!STAT = MD$EOR) OR (R!BPT = 1000)                   TDEM.324
                                                                                  .>>>>> 2          $IF       1
```

```
 700   077000 /002614                R = BUFOUT              .SEND BUFFER TO BIOP                TDEM.325
 702    <macro>                    $ENDIF                                                        TDEM.326
                                                                          .<<<<< 2                $ENDIF    1
                                                                                                  TDEM.327
                                   * INTERCHANGE MODE: BLOCK DATA                                 TDEM.328
                                                                                                  TDEM.329
 702    <macro>                  $ELSEIF (R!FMT = FM$BLKD)                                        TDEM.330
 711    <macro>                    STORE 1,CU@ROP,R!CUR      .MARK RECORD OPEN                   TDEM.331
 721   020462  103003  010001      R!BPT = 1, R!BPT = 0      .FIRST TIME IN - SET POSITION       TDEM.332
       024462
 725   010000  024475              R!PBC = 0                 .DATA ALWAYS ON WORD BOUND          TDEM.333
 727   010000  024476              R!OFFSET = 0              .INITIALIZE LOCAL OFFSET            TDEM.334
 731   010000  024477              R!DONE = 0                .FORCE PASS THROUGH LOOP            TDEM.335
 733   020471  012007  004003      R!DATA = R!CNT + 7 > 3    .WORDS TO TRANSFER                  TDEM.336
       024466
 737    <macro>                    $UNTIL (R!DATA = 0) AND (R!DONE = 1)  .UNTIL DATA ALL MOVED   TDEM.337
 746   020477  136000 /002511        R = BUFGET, R!DONE = 0  .GET CURRENT BUFFER ADDRESS        TDEM.338
 751   010001  024477                R!DONE = 1              .SET PASS FLAG                     TDEM.339
 753    <macro>                      $IF (R!DATA # 0)        .IF DATA TO MOVE                   TDEM.340
                                                                          .>>>>> 2                $IF       1
 757    <macro>                        DADD R!MSU,R!MSL,0,R!BPT,R!ADDU,R!ADDL  .WHERE TO WRITE  TDEM.341
 767   020470  022476  024460          R!LOC = R!BUFF + R!OFFSET  .DATA ORIGIN                   TDEM.342
                                                                                                  TDEM.343
                                        * COMPUTE SIZE TO WRITE                                   TDEM.344
                                                                                                  TDEM.345
 772   014000 /001000  023462          R!PO = 1000 - R!BPT   .ROOM LEFT IN BUFFER                TDEM.346
       024506
 776    <macro>                        $IF (R!DATA > R!PO)   .IF NOT ENOUGH ROOM                 TDEM.347
                                                                          .>>>>> 3                $IF       1
1006   020506  024457                    R!SIZ = R!PO        .FIT WHAT WE CAN                   TDEM.348
1010   005002  024476                    R!OFFSET = A < 2    .SET OFFSET TO REST OF DATA        TDEM.349
1012    <macro>                        $ELSE                                                     TDEM.350
1013   020466  024457                    R!SIZ = R!DATA      .WRITE IT ALL                      TDEM.351
1015   010000  024476                    R!OFFSET = 0        .CLEAR OFFSET                      TDEM.352
1017    <macro>                        $ENDIF                                                    TDEM.353
                                                                          .<<<<< 3                $ENDIF    1
1017   020466  023457  024466          R!DATA = R!DATA - R!SIZ  .DATA LEFT TO TRANSFER           TDEM.354
                                                                                                  TDEM.355
                                        * INITIATE WRITE                                          TDEM.356
                                                                                                  TDEM.357
1022   010005  024461                  R!FCT = FN$MOSWR      .WRITE TO MOS                       TDEM.358
1024   077000 /003057                  R = MEMIO                                                 TDEM.359
```

```
                                                                                    TDEM.360
                                  * UPDATE POINTERS                                 TDEM.361
                                                                                    TDEM.362
1026   020465  022457  024465     R!FWI = R!FWI + R!SIZ  .ADD TO POINTER TO NEXT CW  TDEM.363
1031   020462  022457  024462     R!BPT = R!BPT + R!SIZ  .DATA POINTER              TDEM.364
1034    <macro>                  $ENDIF                                             TDEM.365
                                                                  .<<<<< 2              $ENDIF    1
                                                                                    TDEM.366
                                  * END OF RECORD                                   TDEM.367
                                                                                    TDEM.368
1034    <macro>                  GET R!MODE,CU@MOD,R!CUR  .TYPE OF CW TO WRITE NEXT TDEM.369
1042    <macro>                  $IF (R!STAT = MD$EOR) AND (R!OFFSET = 0)           TDEM.370
                                                                  .>>>>> 2              $IF       1
                                                                                    TDEM.371
                                  * IF NEXT CONTROL WORD IS RCW                     TDEM.372
                                                                                    TDEM.373
1051    <macro>                    $IF (R!MODE = MD$BCW)  .IF NOT WRITING EOR       TDEM.374
                                                                  .>>>>> 3              $IF       1
1054    <macro>                      PUT R!UBC,CU@UBC,R!CUR  .SAVE UNUSED BIT COUNT TDEM.375
1070    <macro>                    $ENDIF                                           TDEM.376
                                                                  .<<<<< 3              $ENDIF    1
1070   077000 /003141              R = CWWRT              .WRITE CONTROL WORD       TDEM.377
                                                                                    TDEM.378
                                  * END CASES                                       TDEM.379
                                                                                    TDEM.380
1072    <macro>                    $IF (R!NCW = 0) OR (R!NCW = 777)  .CHECK ENDS    TDEM.381
                                                                  .>>>>> 3              $IF       1
1100   077000 /003141                R = CWWRT            .WRITE CONTROL WORD       TDEM.382
1102    <macro>                    $ENDIF                                           TDEM.383
                                                                  .<<<<< 3              $ENDIF    1
                                                                                    TDEM.384
                                  * BUFFER FULL                                     TDEM.385
                                                                                    TDEM.386
1102    <macro>                  $ELSEIF (R!BPT = 1000)                             TDEM.387
1110   077000 /003141              R = CWWRT              .WRITE BCW                TDEM.388
1112    <macro>                  $ENDIF                                             TDEM.389
                                                                  .<<<<< 2              $ENDIF    1
1112    <macro>                $ENDTIL                                              TDEM.390
1113    <macro>                $ENDIF                                               TDEM.391
                                                                  .<<<<< 1              $ENDIF    1
                                                                                    TDEM.392
                                *                                                   TDEM.393
```

```
                                  * WRITE: TRANSFER DATA FROM MOS TO LOCAL BUFFER                TDEM.394
                                  *                                                              TDEM.395
                                                                                                 TDEM.396
1113    <macro>                  $ELSEIF (R!FUNC = FC$WRITE)                                     TDEM.397
                                                                                                 TDEM.398
                                  * FIND CPW THAT CONTAINS BUFFER TO FILL                        TDEM.399
                                                                                                 TDEM.400
1120    <macro>                   GET R!CPW,CRW@NC,R!TCB        .NEXT CPW ADDRESS                TDEM.401
1125    <macro>                   GET R!BUFF,CPW@DA,R!CPW      .BUFFER ADDRESS                   TDEM.402
1132    020470  015000 /177774    R!BUFF = R!BUFF & 177774     .MASK LOWER BITS                  TDEM.403
        024470
                                                                                                 TDEM.404
                                  * DETECT IF START OF BLOCK NOT YET SAVED                       TDEM.405
                                                                                                 TDEM.406
1136    <macro>                   GET R!PO,DC@ERP,R!TCB        .SAVE POSITION DELAYED FLAG       TDEM.407
1145    <macro>                   $IF (R!PO = 1)                                                 TDEM.408
                                                                                .>>>>> 1             $IF      1
1151    020463  024506              R!PO = R!PTR               .SAVE CURRENT POSITION            TDEM.409
1153    020462  024507              R!P1 = R!BPT                                                 TDEM.410
1155    <macro>                     $IF (R!BPT # 0)            .NOT FULL SECTOR SIZE BUFFERS     TDEM.411
                                                                                .>>>>> 2             $IF      1
1160    010000  024462                R!BPT = 0                .BACK UP TO START OF BUFFER       TDEM.412
1162    <macro>                     $ELSE                                                        TDEM.413
1163    027463                        R!PTR = R!PTR - 1        .PREVIOUS BUFFER                  TDEM.414
1164    020463  013010  101004        R!PTR = R!DSCLIM - 1, R!PTR < HD@WDS  .WRAP LIST           TDEM.415
        020454  013001  024463
1172    <macro>                     $ENDIF                                                       TDEM.416
                                                                                .<<<<< 2             $ENDIF   1
1172    <macro>                     PUT R!PTR,CU@PTR,R!CUR     .SAVE                             TDEM.417
1177    <macro>                     PUT R!BPT,CU@BPT,R!CUR                                       TDEM.418
1204    077000 /002461              R = SAVPOS                 .SAVE POSITION FOR ERROR RECOV    TDEM.419
1206    020506  024463              R!PTR = R!PO               .RESTORE                          TDEM.420
1210    020507  024462              R!BPT = R!P1                                                 TDEM.421
1212    <macro>                     STORE 0,DC@ERP,R!TCB       .CLEAR DELAY FLAG                 TDEM.422
1221    <macro>                   $ENDIF                                                         TDEM.423
                                                                                .<<<<< 1             $ENDIF   1
1221    <macro>                   $IF (0 # R!DST & ST$UC)      .CCR OR ERROR                R17633AH.1
                                                                                .>>>>> 1             $IF      1
1230    070445                      P = UPDATE                                                   TDEM.425
1231    <macro>                   $ENDIF                                                         TDEM.426
                                                                                .<<<<< 1             $ENDIF   1
1231    077000 /002511            R = BUFGET                   .READ IN DESCRIPTOR               TDEM.427
```

```
                                                                                                TDEM.428
                                                                                                TDEM.429
                                  * GO TO COMMON HOUSEKEEPING ROUTINE                           TDEM.430
                                                                                                TDEM.431
1233   077000 /002040             R = COMSUB                                                    TDEM.432
                                                                                                TDEM.433
                                  * MOVE DATA                                                   TDEM.434
                                                                                                TDEM.435
1235   <macro>                    DADD R!MSU,R!MSL,0,R!BPT,R!ADDU,R!ADDL   .START ADDRESS       TDEM.436
1245   020470  024460             R!LOC = R!BUFF              .LOCAL ADDRESS                    TDEM.437
1247   020471  012007  004003     R!SIZ = R!CNT + 7 > 3       .WORD LENGTH                      TDEM.438
       024457
1253   010004  024461             R!FCT = FN$MOSRD            .READ IN                          TDEM.439
1255   077000 /003057             R = MEMIO                                                     TDEM.440
                                                                                                TDEM.441
                                  * UPDATE POINTERS                                             TDEM.442
                                                                                                TDEM.443
1257   020462  022457  024462     R!BPT = R!BPT + R!SIZ       .MOS POSITION                     TDEM.444
                                                                                                TDEM.445
                                  * MOVE BUFFER POINTER?                                        TDEM.446
                                                                                                TDEM.447
1262   <macro>                    GET R!PO,DC@CHK,R!TCB       .CHECK FOR END OF RECORD          TDEM.448
1271   <macro>                    $IF (R!BPT = 1000) OR (R!PO # 0)  .BUFFER FULL OR EOR         TDEM.449
                                                                                .>>>>> 1            $IF      1
1300   <macro>                      STORE 1,FL$TDM,R!TCB      .SHOW TDEM MOVED POINTER        R17633AH.2
1311   077000 /002603               R = NXTBUF                .MOVE POINTER                     TDEM.450
1313   <macro>                    $ENDIF                                                        TDEM.451
                                                                                .<<<<< 1            $ENDIF   1
                                                                                                R16767CH.3
                                  * SET BUFFER READY FOR I/O                                    R16767CH.4
                                                                                                R16767CH.5
1313   <macro>                    STORE 1,FL$RDY,R!TCB                                          R16767CH.6
                                                                                                R16767CH.7
                                  * CHECK IF THIS IS FIRST BUFFER'S WORTH OF RECORD             R16767CH.8
                                                                                                R16767CH.9
1324   <macro>                    GET R!%W1,F2$SIO,R!TCB      .CHECK FOR INITIAL I/O            R16767CH.10
1332   <macro>                    $IF (R!%W1 = 0)             .DATA CHAINING HAS STARTED        R16767CH.11
                                                                                .>>>>> 1            $IF      1
1335   <macro>                      STORE 1,F2$NVB,R!TCB      .MARK NEXT BUFFER DATA VALID      R16767CH.12
1346   <macro>                      GET R!%W1,CRW@DS,R!TCB    .GET CURRENT DEVICE STATUS        R16767CH.13
1354   <macro>                      $IF (0 # R!%W1 & ST$UC)   .IF ERROR                         R16767CH.14
                                                                                .>>>>> 2            $IF      1
```

```
1363   070312                                P = UPDATE              .DONE                                R16767CH.15
1364    <macro>                             $ENDIF                                                          R16767CH.16
                                                                                        .<<<<< 2                $ENDIF    1
                                                                                                            R16767CH.17
                                           * LOAD BYTE COUNT REGISTER                                       R16767CH.18
                                                                                                            R16767CH.19
1364    <macro>                            GET R!%W1,CRW@CN,R!TCB    .LAST CHANNEL USED FOR I/O             R16767CH.20
1372   020423  054000                      B = R!%W1                 .SET CHANNEL NUMBER                    R16767CH.21
1374   014000 /000000  024410              R!%W2 = R!%W1 - 20 + (XCHT) .LAST CHANNEL TABLE POINTER          R16767CH.22
       020423  013020  032410
       024424
1403   030424  024424                      R!%W2 = (R!%W2)           .TABLE ADDRESS                         R16767CH.23
                                                                                                            R16767CH.24
1405    <macro>                            $IF (IOB # DN)            .IF I/O STILL ACTIVE                   R16767CH.25
                                                                                        .>>>>> 2                $IF       1
                                                                                                            R16767CH.26
                                             * LOAD BYTE COUNT REGISTER                                     R16767CH.27
                                                                                                            R16767CH.28
1410   020471                                A = R!CNT                                                      R16767CH.29
1411   175000                                IOB:15                    .ISSUE                               R16767CH.30
                                                                                                            R16767CH.31
                                             * VERIFY BYTE COUNT REGISTER                                   R16767CH.32
                                                                                                            R16767CH.33
1412   171000                                IOB:11                    .ISSUE                               R16767CH.34
1413    <macro>                              $IF (A # R!CNT)                                                R16767CH.35
                                                                                        .>>>>> 3                $IF       1
1416    <macro>                                STORE ES$BRE,CHT@EC,R!%W2 .BAD HARDWARE                      R16767CH.36
1426    <macro>                                STORE 0,FL$RDY,R!TCB  .MARK BUFFER NOT READY                 R16767CH.37
1435   075000 /000052                          P = DEQUEUE           .NEXT REQUEST                          R16767CH.38
1437    <macro>                              $ENDIF                                                         R16767CH.39
                                                                                        .<<<<< 3                $ENDIF    1
                                                                                                            R16767CH.40
                                             * VERIFY CHANNEL IS KEEPING UP                                 R16767CH.41
                                                                                                            R16767CH.42
1437    <macro>                              $IF (IOB = DN)            .CHECK AGAIN                         R16767CH.43
                                                                                        .>>>>> 3                $IF       1
1442    <macro>                                STORE ES$SOR,CHT@EC,R!%W2 .RETRY BLOCK                       R16767CH.44
1452    <macro>                                STORE 1,CHT@DN,R!%W2                                         R16767CH.45
1463    <macro>                                STORE 0,FL$RDY,R!TCB  .MARK BUFFER NOT READY                 R16767CH.46
1472   075000 /000052                          P = DEQUEUE           .NEXT REQUEST                          R16767CH.47
1474    <macro>                              $ENDIF                                                         R16767CH.48
                                                                                        .<<<<< 3                $ENDIF    1
```

```
1474    <macro>                    $ENDIF                                                    R16767CH.49
                                                                        .<<<<< 2                $ENDIF    1
1474    <macro>                  $ELSE                      .DATA CHAIN HAS NOT STARTED       R16767CH.50
1475    <macro>                    STORE 0,F2$SIO,R!TCB     .CLEAR FIRST BUFFER I/O FLAG      R16767CH.51
1506    <macro>                  $ENDIF                                                      R16767CH.52
                                                                        .<<<<< 1                $ENDIF    1
                                                                                                R16767CH.53
                                                                                                    TDEM.452
                                 *                                                                  TDEM.453
                                 * TERMINATE INTERCHANGE DATA READ                                 TDEM.454
                                 *                                                                  TDEM.455
                                                                                                    TDEM.456
1506    <macro>                $ELSE                                                               TDEM.457
1507    010000  024475           R!PBC = 0                .CLEAR PARTIAL BYTE COUNT                TDEM.458
1511    077000 /002511           R = BUFGET               .GET DESCRIPTOR                          TDEM.459
1513    <macro>                  GET R!MODE,CU@MOD,R!CUR  .NEXT CW TYPE                            TDEM.460
1521    020474  024473           R!STAT = R!MODE          .SET BUFFER STATUS                       TDEM.461
                                                                                                    TDEM.462
                                 * WRITE LAST RCW                                                   TDEM.463
                                                                                                    TDEM.464
1523    <macro>                  $IF (R!FUNC = FC$EORR)                                             TDEM.465
                                                                        .>>>>> 1                    $IF       1
                                                                                                    TDEM.466
                                   * WRITE EOR CW WITH NULL FLAG                                    TDEM.467
                                                                                                    TDEM.468
1527    <macro>                    $IF (R!NCW # 0)        .DON'T START NEW SECTOR                  TDEM.469
                                                                        .>>>>> 2                    $IF       1
1532    010010  024474               R!MODE = MD$EOR      .EOR CW NEXT                             TDEM.470
1534    024473                       R!STAT = A           .SET STATUS                              TDEM.471
1535    010001  024505               R!NUL = 1            .SET NULL EOR FLAG                       TDEM.472
1537    010777  023464  024465       R!FWI = O'777 - R!NCW .SET POINTER TO NEXT SECTOR             TDEM.473
1542    077000 /003141               R = CWWRT            .WRITE EOR CW                            TDEM.474
1544    <macro>                    $ENDIF                                                           TDEM.475
                                                                        .<<<<< 2                    $ENDIF    1
                                                                                                    TDEM.476
1544    <macro>                  $ELSE                                                              TDEM.477
                                                                                                    TDEM.478
                                   * WRITE NORMAL EOR CW, IF RECORD OPEN                            TDEM.479
                                                                                                    TDEM.480
1545    <macro>                    $IF (R!NCW # 0)        .IF RECORD OPEN                          TDEM.481
                                                                        .>>>>> 2                    $IF       1
1550    010010  024474               R!MODE = MD$EOR      .EOR CW NEXT                             TDEM.482
```

```
1552   024473                          R!STAT = A              .SET STATUS                          TDEM.483
1553   077000 /003141                  R = CWWRT               .WRITE EOR CW                        TDEM.484
1555    <macro>                       $ENDIF                                                        TDEM.485
                                                                                .<<<<< 2                $ENDIF    1
                                                                                                      TDEM.486
1555    <macro>                       $IF (R!NCW = 0)         .IF BCW NEEDED                        TDEM.487
                                                                                .>>>>> 2                $IF       1
1560   077000 /003141                  R = CWWRT               .WRITE IT                            TDEM.488
1562    <macro>                       $ENDIF                                                        TDEM.489
                                                                                .<<<<< 2                $ENDIF    1
                                                                                                      TDEM.490
1562    <macro>                       $IF (R!FUNC = FC$EOFR)                                        TDEM.491
                                                                                .>>>>> 2                $IF       1
                                                                                                      TDEM.492
                                       * WRITE EOF CW WITH NULL FLAG                                 TDEM.493
                                                                                                      TDEM.494
1566   010016  024474                  R!MODE = MD$EOF         .EOF CW NEXT                         TDEM.495
1570   024473                          R!STAT = A              .SET STATUS                          TDEM.496
1571   010001  024505                  R!NUL = 1               .SET NULL EOR FLAG                   TDEM.497
1573   010777  023464  024465          R!FWI = O'777 - R!NCW   .SET POINTER TO NEXT SECTOR          TDEM.498
1576   077000 /003141                  R = CWWRT               .WRITE EOF CW                        TDEM.499
                                                                                                      TDEM.500
1600    <macro>                       $ELSE                                                         TDEM.501
                                                                                                      TDEM.502
                                       * WRITE NORMAL EOF CW                                         TDEM.503
                                                                                                      TDEM.504
1601   010016  024474                  R!MODE = MD$EOF         .EOF CW NEXT                         TDEM.505
1603   024473                          R!STAT = A              .SET STATUS                          TDEM.506
1604   077000 /003141                  R = CWWRT               .WRITE EOF CW                        TDEM.507
                                                                                                      TDEM.508
1606    <macro>                        $IF (R!NCW = 0)         .IF BCW NEEDED                       TDEM.509
                                                                                .>>>>> 3                $IF       1
1611   077000 /003141                    R = CWWRT             .WRITE IT                            TDEM.510
1613    <macro>                        $ENDIF                                                        TDEM.511
                                                                                .<<<<< 3                $ENDIF    1
                                                                                                      TDEM.512
1613    <macro>                        $IF (R!FUNC = FC$EODR)                                        TDEM.513
                                                                                .>>>>> 3                $IF       1
                                                                                                      TDEM.514
                                         * WRITE EOD CW WITH NULL FLAG                               TDEM.515
                                                                                                      TDEM.516
1617   010017  024474                    R!MODE = MD$EOD       .EOD CW NEXT                         TDEM.517
```

```
1621   024473                                R!STAT = A            .SET STATUS                          TDEM.518
1622   010001  024505                        R!NUL = 1             .SET NULL EOR FLAG                   TDEM.519
1624   010777  023464  024465                R!FWI = O'777 - R!NCW .SET POINTER TO NEXT SECTOR          TDEM.520
1627   077000 /003141                        R = CWWRT             .WRITE EOD CW                        TDEM.521
                                                                                                          TDEM.522
1631    <macro>                              STORE 0,CU@BL0,R!CUR  .CLEAR BLOCK NUMBER                  TDEM.523
1640    <macro>                              STORE 0,CU@BL1,R!CUR  .                                    TDEM.524
1645    <macro>                              STORE 0,CU@BL2,R!CUR  .                                    TDEM.525
1656    <macro>                            $ELSE                                                         TDEM.526
1657    <macro>                              $PUNTIF                                                     TDEM.527
1661    <macro>                            $ENDIF                                                        TDEM.528
                                                                                  .<<<<< 3                $ENDIF   1
1661    <macro>                          $ENDIF                                                          TDEM.529
                                                                                  .<<<<< 2                $ENDIF   1
1661    <macro>                        $ENDIF                                                            TDEM.530
                                                                                  .<<<<< 1                $ENDIF   1
                                                                                                          TDEM.531
1661   010000  024464                  R!NCW = 0             .CLEAR POINTER TO NEXT CW                   TDEM.532
1663   024462                          R!BPT = A             .CLEAR POINTER TO NEXT DATA                 TDEM.533
1664    <macro>                        STORE MD$BCW,CU@MOD,R!CUR  .RESET NEXT CONTROL WORD TYPE          TDEM.534
                                                                                                          TDEM.535
1675    <macro>                      $ENDIF                                                              TDEM.536
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                          TDEM.537
                                     *                                                                    TDEM.538
                                     * UPDATE MOS COPY OF XIOP POINTERS                                  TDEM.539
                                     *                                                                    TDEM.540
                                                                                                          TDEM.541
1675                         UPDATE  *                                                                    TDEM.542
1675    <macro>                      PUT R!PTR,CU@PTR,R!CUR        .POINTER TO MOS BUFFER                TDEM.543
1702    <macro>                      $IF (R!FUNC = FC$READ)        .IF READING - UPDATE START OF REC     TDEM.544
                                                                                  .>>>>> 0                $IF      1
1706   010004  012002  024423          R!%W1 = ST$DE+ST$UC           .DEVICE END + UNIT CHECK              TDEM.545
1711    <macro>                        $IF (ST$DE = R!DST & R!%W1) .IF DEVICE END ONLY                   TDEM.546
                                                                                  .>>>>> 1                $IF      1
1720    <macro>                          PUT R!PTR,CU@BKP,R!CUR    .UPDATE START OF RECORD               TDEM.547
1725    <macro>                        $ENDIF                                                            TDEM.548
                                                                                  .<<<<< 1                $ENDIF   1
1725    <macro>                      $ENDIF                                                              TDEM.549
                                                                                  .<<<<< 0                $ENDIF   1
1725    <macro>                      PUT R!BPT,CU@BPT,R!CUR        .POSITION IN BUFFER                   TDEM.550
1732    <macro>                      PUT R!NCW,CU@NCW,R!CUR        .POINTER TO NEXT CONTROL WORD         TDEM.551
```

```
1737    <macro>                 PUT R!FWI,CU@FWI,R!CUR        .FORWARD WORD INDEX          TDEM.552
                                                                                             TDEM.553
                                * WRITE IT BACK TO MOS                                       TDEM.554
                                                                                             TDEM.555
1752   020435  024455           R!ADDU = R!DSU                .XIOP HEADER ADDRESS         TDEM.556
1754   020436  024456           R!ADDL = R!DSL                .                            TDEM.557
1756   020432  024460           R!LOC = R!CUR                 .LOCAL ADDRESS               TDEM.558
1760   010006  024457           R!SIZ = CU@WDS                .LENGTH                      TDEM.559
1762   010005  024461           R!FCT = FN$MOSWR              .WRITE IT                    TDEM.560
1764   077000 /003057           R = MEMIO                                                  TDEM.561
                                                                                             TDEM.562
                                *                                                            TDEM.563
                                * RESTORE FIELDS IN TCB THAT MAY HAVE CHANGED               TDEM.564
                                *                                                            TDEM.565
                                                                                             TDEM.566
1766    <macro>                 PUT R!RSC,DC@RSC,R!TCB        .REQUEST REMAINDER           TDEM.567
1773    <macro>                 PUT R!HSU,DC@HSU,R!TCB        .CURRENT CPU ADDRESS         TDEM.568
2000    <macro>                 PUT R!HSL,DC@HSL,R!TCB                                     TDEM.569
2005    <macro>                 STORE 1,FL$RDY,R!TCB          .SHOW BUFFER READY           TDEM.570
                                                                                             TDEM.571
                                *****************                                            TDEM.572
                                *     TRACE     *                                            TDEM.573
                                *               *                                            TDEM.574
                                * 1. CPW        *                                            TDEM.575
                                * 2. REQ SECTS  *                                            TDEM.576
                                * 3. CPU-UPPER  *                                            TDEM.577
                                * 4. CPU-LWR    *                                            TDEM.578
                                * 5. POINTER    *                                            TDEM.579
                                *****************                                            TDEM.580
                                                                                             TDEM.581
2016    <macro>                 KRECORD TF$TEX6,(R!CPW,R!RSC,R!HSU,R!HSL,R!PTR),INT=ON     TDEM.582
                                                                                             TDEM.583
                                                                                             TDEM.584
                                *                                                            TDEM.585
                                * CHECK FOR MORE REQUESTS                                    TDEM.586
                                *                                                            TDEM.587
                                                                                             TDEM.588
2036   075000 /000052           P = DEQUEUE                                                TDEM.589
                                                                                             TDEM.590
                            ************************** SUBROUTINES ******************************  TDEM.591
                                                                                             TDEM.592
                                *                                                            TDEM.593
```

```
                                        * IF AT BEGINNING OF RECORD                                          TDEM.594
                                        *                                                                    TDEM.595
                                                                                                             TDEM.596
2040                            COMSUB  *                                                                    TDEM.597
2040    <macro>                         GET A,DC@CHK,R!TCB            .START OF NEW RECORD?                  TDEM.598
2046    <macro>                         $IF (A # 0)                   .IF SO -                               TDEM.599
                                                                                          .>>>>> 0               $IF      1
                                                                                                             TDEM.600
                                          * WRITE CHECKSUM                                                   TDEM.601
                                                                                                             TDEM.602
2050    030470  004010  011377            R!%W1 = (R!BUFF) > 10 & 377  .USE FIRST BYTE FOR NOW               TDEM.603
        024423
2054    <macro>                           PUT R!%W1,TQ@BCS,R!DAL      .ENTER IN RESPONSE PACKET              TDEM.604
2061    <macro>                           STORE 0,DC@CHK,R!TCB        .CLEAR INDICATOR                       TDEM.605
                                                                                                             TDEM.606
                                          * IF WRITING -                                                     TDEM.607
                                                                                                             TDEM.608
2070    <macro>                           $IF (R!FUNC = FC$WRITE)                                            TDEM.609
                                                                                          .>>>>> 1               $IF      1
                                                                                                             TDEM.610
                                            * VALIDATE START OF RECORD                                       TDEM.611
                                            * SAVE TOTAL RECORD LENGTH                                       TDEM.612
                                            * MARK CRW SO TDEM DOES NOT LOAD BYTE                        R16767CH.54
                                            * REGISTER ON FIRST BUFFER                                   R16767CH.55
                                                                                                         R16767CH.56
                                                                                                             TDEM.613
2074    <macro>                             GET A,BF@STA,R!ENTRY      .CHECK FOR BOR STATUS                  TDEM.614
2101    <macro>                             $IF (A # MD$BOR)          .IF NOT - ABORT                        TDEM.615
                                                                                          .>>>>> 2               $IF      1
2104    <macro>                               $PUNTIF                                                        TDEM.616
2106    <macro>                             $ELSE                                                            TDEM.617
2107    <macro>                               GET R!%W1,BF@RLU,R!ENTRY  .GET RECORD LENGTH                   TDEM.618
2116    <macro>                               GET R!%W2,BF@RLL,R!ENTRY                                       TDEM.619
2123    <macro>                               PUT R!%W1,CU@RLU,R!CUR  .SAVE                                  TDEM.620
2130    <macro>                               PUT R!%W2,CU@RLL,R!CUR                                         TDEM.621
2135    <macro>                               STORE 1,F2$SIO,R!TCB    .MARK FIRST BUFFER OF RECORD       R16767CH.57
2145    <macro>                             $ENDIF                                                           TDEM.622
                                                                                          .<<<<< 2               $ENDIF   1
                                                                                                             TDEM.623
                                            * IF PREVIOUS RECORD STILL BEING WRITTEN                         TDEM.624
                                            * - DELAY SAVING CURRENT POSITION FOR RECOVERY                   TDEM.625
                                                                                                             TDEM.626
```

```
2145    <macro>          $IF (R!OSI # OS$DN)          .                                  TDEM.627
                                                                          .>>>>> 2          $IF      1
2151    <macro>            STORE 1,DC@ERP,R!TCB     .SET DELAY FLAG                     TDEM.628
2162    <macro>          $ELSE                                                            TDEM.629
2163   072276              R = SAVPOS               .SAVE CURRENT POSITION NOW          TDEM.630
2164    <macro>          $ENDIF                                                           TDEM.631
                                                                          .<<<<< 2          $ENDIF   1
                                                                                           TDEM.632
                          * IF READING -                                                   TDEM.633
                                                                                           TDEM.634
2164    <macro>          $ELSE                                                            TDEM.635
                                                                                           TDEM.636
                          * IF AHEAD OF TAPEIO                                             TDEM.637
                          * - DELAY ADDING CURRENT SECTOR TO MOS TOTAL                     TDEM.638
                          *   SO THAT ITS NOT REPORTED AS PART OF LAST RECORD              TDEM.639
                                                                                           TDEM.640
2165   072274             R = SAVPOS                .SAVE CURRENT POSITION              TDEM.641
2166    <macro>           GET A,CRW@CC,R!TCB        .RECORD DONE COUNT                  TDEM.642
2172    <macro>           $IF (A # 0) AND (0 = R!DST & ST$DE)                           TDEM.643
                                                                          .>>>>> 2          $IF      1
2203    <macro>             STORE 1,DC@ERP,R!TCB    .SET DELAY FLAG                     TDEM.644
2214    <macro>           $ENDIF                                                          TDEM.645
                                                                          .<<<<< 2          $ENDIF   1
2214    <macro>          $ENDIF                                                           TDEM.646
                                                                          .<<<<< 1          $ENDIF   1
2214    <macro>        $ENDIF                                                             TDEM.647
                                                                          .<<<<< 0          $ENDIF   1
                                                                                           TDEM.648
                       *                                                                   TDEM.649
                       * IF WRITING, DETERMINE BYTE COUNT TO TRANSFER                      TDEM.650
                       *                                                                   TDEM.651
                                                                                           TDEM.652
2214    <macro>        $IF (R!FUNC = FC$WRITE)                                             TDEM.653
                                                                          .>>>>> 0          $IF      1
2220    <macro>          GET R!%W1,CU@RLU,R!CUR    .RESIDUE RECORD LENGTH                TDEM.654
2225    <macro>          GET R!%W2,CU@RLL,R!CUR                                          TDEM.655
2232    <macro>          $IF (R!%W1 # 0) OR (R!%W2 > R!BSZ)                              TDEM.656
                                                                          .>>>>> 1          $IF      1
2242   020444  024471      R!CNT = R!BSZ            .MAX                                TDEM.657
2244    <macro>          $ELSE                                                            TDEM.658
2245   020424  024471      R!CNT = R!%W2            .SET TO REMAINDER                   TDEM.659
2247    <macro>          $ENDIF                                                           TDEM.660
```

```
                                                                                .<<<<< 1          $ENDIF   1
2247    <macro>                   DSUB R!%W1,R!%W2,R!CNT       .ADJUST REMAINDER                TDEM.661
2260    <macro>                   PUT R!%W1,CU@RLU,R!CUR       .SAVE                            TDEM.662
2265    <macro>                   PUT R!%W2,CU@RLL,R!CUR                                        TDEM.663
                                                                                                  TDEM.664
                                  * CHECK FOR END OF RECORD                                     TDEM.665
                                                                                                  TDEM.666
2272    <macro>                   $IF (R!%W1 = 0) AND (R!%W2 = 0)  .IF NO MORE DATA IN BLOCK    TDEM.667
                                                                                .>>>>> 1          $IF      1
2300    <macro>                     STORE 1,DC@CHK,R!TCB       .INDICATE START OF NEXT          TDEM.668
2311    <macro>                   $ENDIF                                                        TDEM.669
                                                                                .<<<<< 1          $ENDIF   1
2311    <macro>                 $ENDIF                                                          TDEM.670
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                  TDEM.671
                                                                                                  TDEM.672
                                * TOGGLE BUFFER ADDRESSES IN OTHER                              TDEM.673
                                * CONTROL LIST. THIS MUST BE DONE SO                            TDEM.674
                                * THAT IF COMMAND CHAINING IS IN EFFECT                         TDEM.675
                                * THE FIRST BUFFER USED IN THE NEXT BLOCK                       TDEM.676
                                * IS NOT THE SAME AS THE LAST BUFFER IN THE                     TDEM.677
                                * CURRENT BLOCK.                                                TDEM.678
                                                                                                  TDEM.679
2311    020514  102002  001000  EXIT, R!DBI # 0              .DO NOT TOGGLE ON DOUBLE INT.       TDEM.680
                                                                                                  TDEM.681
2314    <macro>                 GET R!P0,CRW@NC,R!TCB         .NEXT CPW                         TDEM.682
2321    <macro>                 $IF (R!P0 > R!C0T) AND (R!P0 <= R!C0B)  .IF IN LIST 0           TDEM.683
                                                                                .>>>>> 0          $IF      1
2335    020452  024506            R!P0 = R!C1T               .SET UP LIST 1                   TDEM.684
2337    020453  024507            R!P1 = R!C1B                                                  TDEM.685
2341    <macro>                 $ELSE                        .SET UP LIST 0                   TDEM.686
2342    020450  024506            R!P0 = R!C0T                                                  TDEM.687
2344    020451  024507            R!P1 = R!C0B                                                  TDEM.688
2346    <macro>                 $ENDIF                                                          TDEM.689
                                                                                .<<<<< 0          $ENDIF   1
2346    010004  025506          R!P0 = R!P0 + 4              .BYPASS HEADER WORD                TDEM.690
                                                                                                  TDEM.691
                                * GET BUFFER ADDRESSES                                          TDEM.692
                                                                                                  TDEM.693
2350    <macro>                 GET R!P2,DC@BFA,R!TCB        .BUFFER A                          TDEM.694
2355    <macro>                 GET R!P3,DC@BFB,R!TCB        .BUFFER B                          TDEM.695
2362    020470  024512          R!P4 = R!BUFF                .SET TO CURRENT                    TDEM.696
```

```
                                                                                TDEM.697
                                 * CHECK FOR LEGAL VALUES IN CPW                TDEM.698
                                                                                TDEM.699
2364    <macro>                  $PUNTIF (R!BUFF # R!P2) AND (R!BUFF # R!P3)  .BAD BUFFER ADDRES   TDEM.700
                                                                     .>>>>> 0        $IF      2
                                                                     .<<<<< 0        $ENDIF   2
2376    <macro>                  $PUNTIF (R!CNT > R!BSZ)       .BAD LENGTH       TDEM.701
2404    <macro>                  $PUNTIF (R!CNT = 0) AND (R!FUNC = FC$WRITE)  .0 SIZE WRITE   TDEM.702
                                                                     .>>>>> 0        $IF      2
                                                                     .<<<<< 0        $ENDIF   2
                                                                                TDEM.703
                                                                                TDEM.704
                                 * IF READ - START WITH CURRENT BUFFER          TDEM.705
                                                                                TDEM.706
2415    <macro>                  $IF (R!FUNC = FC$READ)                         TDEM.707
                                                                     .>>>>> 0        $IF      1
2421    <macro>                    $IF (R!P4 = R!P2)           .SET TO OPPOSITE TDEM.708
                                                                     .>>>>> 1        $IF      1
2425    020511  024512               R!P4 = R!P3                                  TDEM.709
2427    <macro>                    $ELSE                                          TDEM.710
2430    020510  024512               R!P4 = R!P2                                  TDEM.711
2432    <macro>                    $ENDIF                                         TDEM.712
                                                                     .<<<<< 1        $ENDIF   1
2432    <macro>                  $ENDIF                                         TDEM.713
                                                                     .<<<<< 0        $ENDIF   1
                                 * TOGGLE ADDRESSES                             TDEM.714
                                                                                TDEM.715
                                                                                TDEM.716
2432    <macro>                  $UNTIL (R!P0 > R!P1)          .FOR EACH CPW    TDEM.717
2437    <macro>                    $IF (R!P4 = R!P2)           .SWITCH          TDEM.718
                                                                     .>>>>> 0        $IF      1
2443    020511  024512               R!P4 = R!P3                                  TDEM.719
2445    <macro>                    $ELSE                                          TDEM.720
2446    020510  024512               R!P4 = R!P2                                  TDEM.721
2450    <macro>                    $ENDIF                                         TDEM.722
                                                                     .<<<<< 0        $ENDIF   1
2450    <macro>                    PUT R!P4,CPW@DA,R!P0        .ENTER BUFFER ADDRESS IN CPW TDEM.723
2455    010004  025506             R!P0 = R!P0 + 4             .NEXT CPW        TDEM.724
2457    <macro>                  $ENDTIL                                        TDEM.725
2460    001000                   EXIT                                           TDEM.726
                                                                                TDEM.727
                                 *                                              TDEM.728
```

```
                                        * SAVE CURRENT POINTERS TO START OF BLOCK                    TDEM.729
                                        *                                                            TDEM.730
                                                                                                     TDEM.731
2461                            SAVPOS  *                                                            TDEM.732
2461    <macro>                         PUT R!PTR,CU@BKP,R!CUR          .POINTER TO FIRST BUFFER     TDEM.733
2466    <macro>                         GET R!ADDU,CU@ERU,R!CUR         .ERROR BUFFER ADDRESS        TDEM.734
2473    <macro>                         GET R!ADDL,CU@ERL,R!CUR                                      TDEM.735
2500    020432  024460                  R!LOC = R!CUR                   .LOCAL ADDRESS               TDEM.736
2502    010006  024457                  R!SIZ = CU@WDS                  .LENGTH                      TDEM.737
2504    010005  024461                  R!FCT = FN$MOSWR                .WRITE TO MOS                TDEM.738
2506    077000 /003057                  R = MEMIO                                                    TDEM.739
2510    001000                          EXIT                                                         TDEM.740
                                                                                                     TDEM.741
                                        *                                                            TDEM.742
                                        * GET THE CURRENT BUFFER ADDRESS                             TDEM.743
                                        *                                                            TDEM.744
                                                                                                     TDEM.745
2511                            BUFGET  *                                                            TDEM.746
2511    <macro>                         DADD R!DSU,R!DSL,0,R!PTR,R!ENTU,R!ENTL  .ADDRESS             TDEM.747
2521    020502  024455                  R!ADDU = R!ENTU                                              TDEM.748
2523    020503  024456                  R!ADDL = R!ENTL                                              TDEM.749
2525    020431  024460                  R!LOC = R!ENTRY                 .LOCAL BUFFER                TDEM.750
2527    010001  024457                  R!SIZ = 1                       .1 WORD DESCRIPTORS          TDEM.751
2531    010004  024461                  R!FCT = FN$MOSRD                .READ IN                     TDEM.752
2533    077000 /003057                  R = MEMIO                                                    TDEM.753
                                                                                                     TDEM.754
                                        * GET ADDRESS OF BUFFER                                      TDEM.755
                                                                                                     TDEM.756
2535    <macro>                         GET R!MSU,BF@ADU,R!ENTRY                                     TDEM.757
2537    <macro>                         GET R!MSL,BF@ADL,R!ENTRY                                     TDEM.758
                                                                                                     TDEM.759
                                        * SENSE RECORD OVERRUN                                       TDEM.760
                                                                                                     TDEM.761
2544    <macro>                         $IF (R!MSU = 0) AND (R!MSL = 0)  .IF NO BUFFER               TDEM.762
                                                                                .>>>>> 0                $IF      1
2552    <macro>                           $PUNTIF (R!FUNC = FC$WRITE)  .IF NO BUFFER ON WRITE        TDEM.763
2556    <macro>                           STORE 1,DC@BIG,R!TCB         .SET ERROR FLAG               TDEM.764
2567    002000  000000                    I = 0                        .LOCK                         TDEM.765
2571    <macro>                           EDECR                        .SHRINK STACK                 TDEM.766
2577    003000                            I = 1                        .UNLOCK                       TDEM.767
2600    071703                            P = UPDATE                   .END DEVICE                   TDEM.768
2601    <macro>                         $ELSE                                                        TDEM.769
```

```
2602   001000                          EXIT                                                          TDEM.770
2603    <macro>                       $ENDIF                                                         TDEM.771
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                      TDEM.772
                                      *                                                               TDEM.773
                                      * POSITION TO NEXT                                             TDEM.774
                                      *                                                               TDEM.775
                                                                                                      TDEM.776
2603                          NXTBUF  *                                                               TDEM.777
2603   026463                         R!PTR = R!PTR + 1              .BUMP POINTER                   TDEM.778
2604   020463  023454  103003         R!PTR = HD@WDS, R!PTR = R!DSCLIM .WRAP LIST                    TDEM.779
       010010  024463
2611   010000  024462                 R!BPT = 0                      .INITIALIZE BUFFER OFFSET       TDEM.780
2613   001000                         EXIT                                                            TDEM.781
                                                                                                      TDEM.782
                                      *                                                               TDEM.783
                                      * BUFFER FULL PROCESSING                                       TDEM.784
                                      *                                                               TDEM.785
                                                                                                      TDEM.786
2614                          BUFOUT  *                                                               TDEM.787
                                                                                                      TDEM.788
                                      * UPDATE DESCRIPTOR                                             TDEM.789
                                                                                                      TDEM.790
2614    <macro>                       GET R!PO,BF@STA,R!ENTRY        .CHECK FOR ERROR STATUS         TDEM.791
2622    <macro>                       $IF (R!PO # MD$ERR)            .IF NOT ERROR                   TDEM.792
                                                                                .>>>>> 0                $IF      1
2626    <macro>                         PUT R!STAT,BF@STA,R!ENTRY    .SAVE CURRENT STATUS            TDEM.793
2641    <macro>                       $ENDIF                                                          TDEM.794
                                                                                .<<<<< 0                $ENDIF   1
2641   020462  005003  023475         R!PO = R!BPT < 3 - R!PBC       .BYTES IN THIS BUFFER           TDEM.795
       024506
2645    <macro>                       PUT R!PO,BF@LNG,R!ENTRY        .ENTER                          TDEM.796
2652   020502  024455                 R!ADDU = R!ENTU                .ADDRESS OF DESCRIPTOR          TDEM.797
2654   020503  024456                 R!ADDL = R!ENTL                                                TDEM.798
2656   020431  024460                 R!LOC = R!ENTRY                .LOCAL ADDRESS                  TDEM.799
2660   010001  024457                 R!SIZ = 1                      .WORD LENGTH                    TDEM.800
2662   010005  024461                 R!FCT = FN$MOSWR               .WRITE BACK TO MOS              TDEM.801
2664   077000 /003057                 R = MEMIO                                                      TDEM.802
2666   073063                         R = NXTBUF                     .ADVANCE BUFFER POINTER         TDEM.803
                                                                                                      TDEM.804
                                      * IF REQUESTS OUTSTANDING - SEND TO BIOP                        TDEM.805
                                      * FOR TRANSFER TO CPU.                                          TDEM.806
```

```
                                                                                                TDEM.807
                                   * CHECK FOR SECTOR DELAY FLAG                                TDEM.808
                                                                                                TDEM.809
2667                       BUFSEND *                                                            TDEM.810
2667    <macro>                    GET R!P3,DC@ERP,R!TCB          .DELAY FLAG                  TDEM.811
2676    020443  012001  024506     R!P0 = R!TMS + 1               .ADD CURRENT TO TOTAL        TDEM.812
2701    <macro>                    GET R!P2,DC@BRQ,R!TCB          .BIOP REQUEST COUNT          TDEM.813
2706    <macro>                    $IF (R!P3 # 0)                 .IF CURRENT IS DELAYED       TDEM.814
                                                                                  .>>>>> 0        $IF      1
2711    <macro>                      $IF (R!P2 = 0)               .AND BIOP QUIET              TDEM.815
                                                                                  .>>>>> 1        $IF      1
2714    027506                         R!P0 = R!P0 - 1            .SUBTRACT CURRENT ONLY       TDEM.816
2715    <macro>                      $ELSE                                                      TDEM.817
2716    010000  024506                 R!P0 = 0                   .DONT SEND ANY               TDEM.818
2720    <macro>                      $ENDIF                                                     TDEM.819
                                                                                  .<<<<< 1        $ENDIF   1
2720    <macro>                    $ENDIF                                                       TDEM.820
                                                                                  .<<<<< 0        $ENDIF   1
                                                                                                TDEM.821
2720    <macro>                    $IF (R!RSC # 0) AND (R!P0 # 0)  .IF ANY TO TRANSFER          TDEM.822
                                                                                  .>>>>> 0        $IF      1
                                                                                                TDEM.823
                                     * DETERMINE SIZE - TRANSFER AS MANY AS POSSIBLE            TDEM.824
                                                                                                TDEM.825
2730    <macro>                      $IF (R!RSC > R!P0)           .DEMAND GREATER THAN SUPPLY  TDEM.826
                                                                                  .>>>>> 1        $IF      1
2736    020506  024507                 R!P1 = R!P0                .SET TO SUPPLY               TDEM.827
2740    <macro>                      $ELSE                                                      TDEM.828
2741    020440  024507                 R!P1 = R!RSC               .SET TO DEMAND               TDEM.829
2743    <macro>                      $ENDIF                                                     TDEM.830
                                                                                  .<<<<< 1        $ENDIF   1
                                                                                                TDEM.831
                                     * IF BIOP IS BUSY TRANSFERRING                             TDEM.832
                                     * LIMIT REQUEST TO CURRENT SECTOR.                         TDEM.833
                                                                                                TDEM.834
2743    020510  102003  010001       R!P1 = 1, R!P2 # 0           .IF BIOP BUSY                TDEM.835
        024507
                                                                                                TDEM.836
                                                                                                TDEM.837
                                     * SEND REQUEST                                             TDEM.838
                                                                                                TDEM.839
2747    <macro>                      PUT R!HSU,DA@HSU,R!DAL       .CPU ADDRESS                 TDEM.840
```

```
2754    <macro>                    PUT R!HSL,DA@HSL,R!DAL                                    TDEM.841
2761    <macro>                    PUT R!P1,DA@RSC,R!DAL          .REQUEST SIZE              TDEM.842
2766    <macro>                    STORE MC$CPUWR,DA@IFC,R!DAL .FUNCTION CODE            R15472BL.1
2773   002000  000000              I = 0                          .LOCK                      TDEM.844
2775   020437  024000              DS = R!DAL                     .DAL ADDRESS               TDEM.845
2777    <macro>                    RKERN DDALGETA                 .ALLOCATE AN MOS DAL       TDEM.846
3001    <macro>                    RKERN DMGB                     .SEND TO BIOP              TDEM.847
3003   003000                      I = 1                          .UNLOCK                    TDEM.848
                                                                                             TDEM.849
                                   * UPDATE                                                  TDEM.850
                                                                                             TDEM.851
3004    <macro>                    GET A,BF@S1A,R!ENTRY           .CHECK FOR BAD DATA        TDEM.852
3011    <macro>                    $IF (A = MD$ERR)               .IF IT IS                  TDEM.853
                                                                                .>>>>> 1         $IF      1
3014   010000  024440                R!RSC = 0                    .CLEAR REMAINING REQUEST   TDEM.854
3016    <macro>                    $ELSE                                                     TDEM.855
3017   020440  023507  024440        R!RSC = R!RSC - R!P1         .DECREMENT BY COUNT SENT   TDEM.856
3022    <macro>                    $ENDIF                                                    TDEM.857
                                                                                .<<<<< 1         $ENDIF   1
3022   020443  012001  023507      R!TMS = R!TMS + 1 - R!P1       .ADJUST TOTAL MOS COUNT    TDEM.858
       024443
3026   020507  005011  024506      R!P0 = R!P1 < 11               .UPDATE THE CPU ADDRESS    TDEM.859
3031    <macro>                    DADD R!HSU,R!HSL,R!P0                                     TDEM.860
3036    <macro>                    FLDADD 1,DC@BRQ,R!TCB          .ADD COUNT OF BIOP REQUESTS TDEM.861
3042    <macro>                  $ENDIF                                                      TDEM.862
                                                                                .<<<<< 0         $ENDIF   1
                                                                                             TDEM.863
                                 * KEEP TRACK OF SECTOR DISPOSITION                          TDEM.864
                                                                                             TDEM.865
3042    <macro>                  $IF (R!P3 = 0)                   .IF SECTOR NOT BEING DELAYED TDEM.866
                                                                                .>>>>> 0         $IF      1
3045    <macro>                    FLDADD 1,DC@TMS,R!TCB          .ADD TO MOS TOTAL          TDEM.867
3051    <macro>                  $ELSE                                                       TDEM.868
3052    <macro>                    FLDADD 1,DC@DMS,R!TCB          .ADD TO DELAY COUNT        TDEM.869
3056    <macro>                  $ENDIF                                                      TDEM.870
                                                                                .<<<<< 0         $ENDIF   1
3056   001000                    EXIT                                                        TDEM.871
                                                                                             TDEM.872
                                 *                                                           TDEM.873
                                 * EXECUTE MEMORY I/O                                        TDEM.874
                                 *                                                           TDEM.875
                                                                                             TDEM.876
```

```
3057                                MEMIO   *                                                                   TDEM.877
3057   002000  000000                       I = 0                          .DISABLE INTERRUPTS                  TDEM.878
3061    <macro>                             $IF (R!FCT = FN$MOSRD)         .READ FROM MOS                       TDEM.879
                                                                                                .>>>>> 0            $IF      1
3065   010002  024000                         R!CA = MIO$BMRI                                                   TDEM.880
3067    <macro>                             $ELSEIF (R!FCT = FN$MOSWR)     .WRITE TO MOS                        TDEM.881
3074   010004  024000                         R!CA = MIO$BMWI                                                   TDEM.882
3076    <macro>                             $ELSE                                                               TDEM.883
3077    <macro>                               $PUNTIF                      .ILLEGAL                             TDEM.884
3101    <macro>                             $ENDIF                                                              TDEM.885
                                                                                                .<<<<< 0            $ENDIF   1
3101   020455  024000                       R!CB = R!ADDU                  .MOS ADDRESS                         TDEM.886
3103   020456  024000                       R!CC = R!ADDL                                                       TDEM.887
3105   020460  024000                       R!CD = R!LOC                   .LOCAL ADDRESS                       TDEM.888
3107   020457  024000                       R!CE = R!SIZ                   .LENGTH                              TDEM.889
3111    <macro>                             RKERN COMIO                    .ISSUE CALL                          TDEM.890
                                                                                                                TDEM.891
                                            * WAIT FOR I/O TO COMPLETE IF READING                               TDEM.892
                                                                                                                TDEM.893
3113    <macro>                             $IF (R!FCT = FN$MOSRD)                                              TDEM.894
                                                                                                .>>>>> 0            $IF      1
3117    <macro>                               $UNTIL (IOB # BZ)            .UNTIL READ DONE                     TDEM.895
3121   003000                                   I = 1                      .ENABLE INTERRUPTS                   TDEM.896
3122    <macro>                                 RKERN CHNWTDN              .WAIT                                TDEM.897
3124   002000  000000                           I = 0                      .DISABLE INTERRUPTS                  TDEM.898
3126    <macro>                                 $IF (IOB = DN) AND (IOB = BZ) .IF ERROR                         TDEM.899
                                                                                                .>>>>> 1            $IF      1
3134    <macro>                                   RKERN CEREC              .ATTEMPT RECOVERY                    TDEM.900
3136    <macro>                                 $ENDIF                                                          TDEM.901
                                                                                                .<<<<< 1            $ENDIF   1
3136    <macro>                               $ENDTIL                                                           TDEM.902
3137    <macro>                             $ENDIF                                                              TDEM.903
                                                                                                .<<<<< 0            $ENDIF   1
3137   003000                               I = 1                          .ENABLE INTERRUPTS                   TDEM.904
3140   001000                               EXIT                                                                TDEM.905
                                                                                                                TDEM.906
                                            *                                                                   TDEM.907
                                            * WRITE THE NEXT CONTROL WORD (RCW/BCW/EOF/EOD)                     TDEM.908
                                            *                                                                   TDEM.909
                                                                                                                TDEM.910
                                            * CLEAR THE CONTROL WORD                                            TDEM.911
                                                                                                                TDEM.912
```

```
3141                          CWWRT   *                                                                  TDEM.913
3141   020430  024506                 R!P0 = R!CWRD                  .SAVE ADDRESS                      TDEM.914
3143    <macro>                       $UNTIL (R!P0 = R!CWRD + 4)                                        TDEM.915
3151   010000  034506                   (R!P0) = 0                                                      TDEM.916
3153   026506                           R!P0 = R!P0 + 1                                                 TDEM.917
3154    <macro>                       $ENDTIL                                                           TDEM.918
                                                                                                         TDEM.919
                                      * COMMON ENTRIES                                                  TDEM.920
                                                                                                         TDEM.921
3155    <macro>                       PUT R!MODE,CW@MOD,R!CWRD       .CW TYPE                           TDEM.922
3165    <macro>                       PUT R!FWI,CW@FWI,R!CWRD        .DATA TO NEXT CW                   TDEM.923
                                                                                                         TDEM.924
                                      * BCW                                                             TDEM.925
                                                                                                         TDEM.926
3200    <macro>                       $IF (R!MODE = MD$BCW)                                             TDEM.927
                                                                                  .>>>>> 0                $IF      1
3203    <macro>                         GET R!P0,CU@BL0,R!CUR        .BLOCK NUMBER                      TDEM.928
3211    <macro>                         GET R!P1,CU@BL1,R!CUR                                           TDEM.929
3216    <macro>                         GET R!P2,CU@BL2,R!CUR                                           TDEM.930
3224    <macro>                         PUT R!P0,CW@BL0,R!CWRD       .ENTER                             TDEM.931
3237    <macro>                         PUT R!P1,CW@BL1,R!CWRD                                          TDEM.932
3244    <macro>                         PUT R!P2,CW@BL2,R!CWRD                                          TDEM.933
                                                                                                         TDEM.934
                                        * RCW                                                           TDEM.935
                                                                                                         TDEM.936
3257    <macro>                       $ELSE                                                             TDEM.937
3260    <macro>                         GET R!P0,CU@UBC,R!CUR        .GET SAVED UBC                     TDEM.938
3267    <macro>                         PUT R!UBC,CU@UBC,R!CUR       .FOR NEXT RCW                      TDEM.939
3303   020506  005003  024506           R!P0 = R!P0 < 3              .CONVERT TO BITS                   TDEM.940
3306    <macro>                         PUT R!P0,CW@UBC,R!CWRD       .ENTER                             TDEM.941
3317    <macro>                         GET R!P0,CU@PFU,R!CUR        .PREVIOUS FILE INDEX               TDEM.942
3326    <macro>                         GET R!P1,CU@PFL,R!CUR                                           TDEM.943
3334    <macro>                         PUT R!P0,CW@PFU,R!CWRD       .ENTER                             TDEM.944
3350    <macro>                         PUT R!P1,CW@PFL,R!CWRD                                          TDEM.945
3363    <macro>                         GET R!P2,CU@PRU,R!CUR        .PREVIOUS RECORD INDEX             TDEM.946
3371    <macro>                         GET R!P3,CU@PRL,R!CUR                                           TDEM.947
3377    <macro>                         PUT R!P2,CW@PRU,R!CWRD       .ENTER                             TDEM.948
3412    <macro>                         PUT R!P3,CW@PRL,R!CWRD                                          TDEM.949
3425    <macro>                         PUT R!NUL,CW@NUL,R!CWRD      .NULL RECORD FLAG                  TDEM.950
3436    <macro>                       $ENDIF                                                            TDEM.951
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                         TDEM.952
```

```
                                 * WRITE CONTROL WORD TO BUFFER                                    TDEM.953
                                                                                                   TDEM.954
3436    <macro>                  DADD R!MSU,R!MSL,0,R!NCW,R!ADDU,R!ADDL  .ADDRESS                 TDEM.955
3446    020430  024460           R!LOC = R!CWRD                  .LOCAL                           TDEM.956
3450    010001  024457           R!SIZ = 1                       .1 WORD                          TDEM.957
3452    010005  024461           R!FCT = FN$MOSWR                .WRITE IT                        TDEM.958
3454    073375                   R = MEMIO                                                        TDEM.959
                                                                                                   TDEM.960
                                 * UPDATE CONTROL WORD FIELDS                                      TDEM.961
                                                                                                   TDEM.962
3455    <macro>                  $IF (R!MODE = MD$BCW)           .IF BCW JUST WRITTEN             TDEM.963
                                                                                .>>>>> 0             $IF      1
3460    <macro>                    FLDADD 1,CU@BL2,R!CUR         .UPDATE BLOCK NUMBER             TDEM.964
3503    <macro>                    $IF (A = 0)                                                      TDEM.965
                                                                                .>>>>> 1             $IF      1
3505    <macro>                      FLDADD 1,CU@BL1,R!CUR                                          TDEM.966
3511    <macro>                      $IF (A = 0)                                                    TDEM.967
                                                                                .>>>>> 2             $IF      1
3513    <macro>                        FLDADD 1,CU@BL0,R!CUR                                        TDEM.968
3536    <macro>                      $ENDIF                                                         TDEM.969
                                                                                .<<<<< 2             $ENDIF   1
3536    <macro>                    $ENDIF                                                           TDEM.970
                                                                                .<<<<< 1             $ENDIF   1
                                                                                                   TDEM.971
                                   * PREVIOUS FILE INDEX                                           TDEM.972
                                                                                                   TDEM.973
3536    <macro>                    GET A,CU@FOP,R!CUR            .FILE OPEN?                      TDEM.974
3544    <macro>                    $IF (A # 0)                   .IF SO                           TDEM.975
                                                                                .>>>>> 1             $IF      1
3546    <macro>                      FLDADD 1,CU@PFL,R!CUR       .INCREMENT IT                    TDEM.976
3571    <macro>                      $IF (A = 0)                                                    TDEM.977
                                                                                .>>>>> 2             $IF      1
3573    <macro>                        FLDADD 1,CU@PFU,R!CUR                                        TDEM.978
3620    <macro>                      $ENDIF                                                         TDEM.979
                                                                                .<<<<< 2             $ENDIF   1
                                                                                                   TDEM.980
                                     * PREVIOUS RECORD INDEX                                       TDEM.981
                                                                                                   TDEM.982
3620    <macro>                      GET A,CU@ROP,R!CUR          .RECORD OPEN?                    TDEM.983
3625    <macro>                      $IF (A # 0)                 .IF SO                           TDEM.984
                                                                                .>>>>> 2             $IF      1
3627    <macro>                        FLDADD 1,CU@PRL,R!CUR     .INCREMENT IT                    TDEM.985
```

```
3652    <macro>                         $IF (A = 0)                                                     TDEM.986
                                                                                .>>>>> 3                $IF      1
3654    <macro>                           FLDADD 1,CU@PRU,R!CUR                                         TDEM.987
3677    <macro>                         $ENDIF                                                          TDEM.988
                                                                                .<<<<< 3                $ENDIF   1
3677    <macro>                       $ENDIF                                                            TDEM.989
                                                                                .<<<<< 2                $ENDIF   1
                                                                                                        TDEM.990
                                      * MARK FILE OPEN                                                  TDEM.991
                                                                                                        TDEM.992
3677    <macro>                     $ELSE                                                               TDEM.993
3700    <macro>                       STORE 1,CU@FOP,R!CUR                                              TDEM.994
3711    <macro>                     $ENDIF                                                              TDEM.995
                                                                                .<<<<< 1                $ENDIF   1
                                                                                                        TDEM.996
                                    * RCW                                                               TDEM.997
                                                                                                        TDEM.998
3711    <macro>                   $ELSEIF (R!MODE = MD$EOR)                                             TDEM.999
3717    <macro>                     STORE 0,CU@PRU,R!CUR        .CLEAR PREVIOUS RECORD INDEX            TDEM.1000
3726    <macro>                     STORE 0,CU@PRL,R!CUR                                                TDEM.1001
3737    <macro>                     STORE 0,CU@ROP,R!CUR        .RECORD OPEN FLAG                       TDEM.1002
                                                                                                        TDEM.1003
                                    * EOF                                                               TDEM.1004
                                                                                                        TDEM.1005
3750    <macro>                   $ELSEIF (R!MODE = MD$EOF)                                             TDEM.1006
3757    <macro>                     STORE 0,CU@PFU,R!CUR        .CLEAR PREVIOUS FILE INDEX              TDEM.1007
3766    <macro>                     STORE 0,CU@PFL,R!CUR                                                TDEM.1008
3777    <macro>                     STORE 0,CU@FOP,R!CUR        .CLEAR FILE OPEN FLAG                   TDEM.1009
4006    <macro>                   $ENDIF                                                                TDEM.1010
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                        TDEM.1011
                                  * CHECK FOR BUFFER FULL                                               TDEM.1012
                                                                                                        TDEM.1013
4006   020505  102004  014000     R!BPT = D'512, R!NUL # 0    .FORCE FULL SECTOR TRANSFER             TDEM.1014
      /001000  024462
4013   020464  022465  012001     R!NCW = R!NCW + R!FWI + 1 & 777  .OFFSET FOR NEXT CW                TDEM.1015
       011777  024464
4020    <macro>                   $IF (A = 0)                 .IF BUFFER COMPLETE                     TDEM.1016
                                                                                .>>>>> 0                $IF      1
4022   077000 /002614               R = BUFOUT                  .FLUSH                                  TDEM.1017
4024   077000 /002511               R = BUFGET                  .GET NEW BUFFER DESCRIPTOR              TDEM.1018
4026   010000  024474               R!MODE = MD$BCW             .NEXT CW IS BCW                         TDEM.1019
```

```
4030    <macro>                          $ELSE                                                          TDEM.1020
4031   010010  024474                      R!MODE = MD$EOR              .NEXT CW IS RCW                TDEM.1021
4033    <macro>                          $ENDIF                                                         TDEM.1022
                                                                                   .<<<<< 0                $ENDIF    1
4033    <macro>                          PUT R!MODE,CU@MOD,R!CUR      .PUT IN TABLE                  TDEM.1023
4046   010000  024465                    R!FWI = 0                    .CLEAR FORWARD WORD INDEX      TDEM.1024
4050   020464  012001  024462            R!BPT = R!NCW + 1            .POINTER FOR NEXT DATA         TDEM.1025
4053   001000                            EXIT                                                        TDEM.1026
                                                                                                      TDEM.1027
                                         END                                                          TDEM.1028
```

0X	%B		1: 4 D	7:42 B	8:30 B	9:32 B	10:25 B	10:35 B	11:11 B	11:19 B	
			11:27 B	11:29 B	11:35 B	12: 1 B	12: 7 B	12:16 B	12:17 B	12:20 B	
			12:29 B	12:42 B	13:22 B	13:28 B	13:35 B	14:31 B	14:42 B	15: 5 B	
			15:14 B	15:26 B	16:29 B	16:40 B	17:15 B	17:32 B	18: 2 B	18: 8 B	
			18:21 B	18:29 B	18:33 B	19: 4 B	20:10 B	20:38 B	25:11 B	25:25 B	
			26:38 B	27:21 B	32:18 B	32:25 B	32:40 B	32:41 B			
0X	%EX		6:14 D	6:14							
10	%GBLNUM	$APTEXT	3:41 F	3:41							
400	%GBLREG	$APTEXT	3:41								
0	%KERNEL	$APTEXT	6:34 D	6:34 B	6:40 D	6:40 B	28: 6 D	28: 6 B	28: 7 D	28: 7 B	
			29:16 D	29:16 B	29:24 D	29:24 B	29:28 D	29:28 B			
0X	%PUNT		6:15 D	6:15	6:36 D	6:36	19:10 D	19:10	21:30 D	21:30	
			24: 4 D	24: 5	24: 7 D	24: 7	24: 8 D	24: 9	25:36 D	25:36	
			29: 9 D	29: 9							
0X	%TR1		8:23 D	8:23	20:31 D	20:31					
0X	%TR2		8:23 D	8:23	20:31 D	20:31					
0X	%TR3		8:23 D	8:23	20:31 D	20:31					
0X	%TR4		8:23 D	8:23	20:31 D	20:31					
0X	%TR5		8:23 D	8:23	20:31 D	20:31					
0X	%TRAC		8:23 D	8:23	20:31 D	20:31					
0X	%TVNT		8:23 D	8:23	20:31 D	20:31					
4	BF@@LE	$APTEXT	6: 1	6:21							
20	BF@ADL@N	$APTEXT	25:30	25:30 F							
1	BF@ADL@P	$APTEXT	25:30 F	25:30							
0	BF@ADL@S	$APTEXT	25:30	25:30 F							
20	BF@ADU@N	$APTEXT	25:29	25:29 F							
0	BF@ADU@P	$APTEXT	25:29 F								
0	BF@ADU@S	$APTEXT	25:29	25:29 F							
20	BF@LNG@N	$APTEXT	26:32 F								
3	BF@LNG@P	$APTEXT	26:32 F	26:32							
0	BF@LNG@S	$APTEXT	26:32 F								
20	BF@RLL@N	$APTEXT	21:33	21:33 F							
3	BF@RLL@P	$APTEXT	21:33 F	21:33							
0	BF@RLL@S	$APTEXT	21:33	21:33 F							
7777	BF@RLU@M	$APTEXT	21:32								
14	BF@RLU@N	$APTEXT	21:32								
2	BF@RLU@P	$APTEXT	21:32 F	21:32							
4	BF@RLU@S	$APTEXT	21:32	21:32 F							
17	BF@STA@M	$APTEXT	26:27								
4	BF@STA@N	$APTEXT	21:27	21:27 F	26:24	26:24 F	26:27 F	26:27	28:12	28:12 F	
2	BF@STA@P	$APTEXT	21:27 F	21:27	26:24 F	26:24	26:27 F	26:27	28:12 F	28:12	
0	BF@STA@S	$APTEXT	21:27	21:27 F	26:24	26:24 F	26:27 F	26:27	28:12	28:12 F	

0X	BXQQ		5:29 D	6:29							
0X	CA		5:30 D	29: 5		29: 7					
0X	CB		5:31 D	29:12							
0X	CC		5:32 D	29:13							
0X	CD		5:33 D	29:14							
0X	CE		5:34 D	29:15							
0X	CEREC		29:28 D	29:28							
0X	CHNWTDN		29:24 D	29:24							
1	CHI@DN@M	$APTEXT	16:38 F								
1	CHI@DN@N	$APTEXT	16:38 F	16:38							
13	CHI@DN@P	$APTEXT	16:38 F	16:38							
2	CHI@DN@S	$APTEXT	16:38 F	16:38							
377	CHI@EC@M	$APTEXT	16:27 F	16:37 F							
10	CHI@EC@N	$APTEXT	16:27 F	16:27		16:37 F		16:37			
14	CHI@EC@P	$APTEXT	16:27 F	16:27		16:37 F		16:37			
0	CHI@EC@S	$APTEXT	16:27 F	16:27		16:37 F		16:37			
0X	COM10		29:16 D	29:16							
20	CPW@BL@N	$APTEXT	10: 5	10: 5 F		10: 6 F					
2	CPW@BL@P	$APTEXT	10: 5 F	10: 5		10: 6 F		10: 6			
0	CPW@BL@S	$APTEXT	10: 5	10: 5 F		10: 6 F					
20	CPW@DA@N	$APTEXT	10: 2	10: 2 F		14: 9		14: 9 F		24:37 F	
1	CPW@DA@P	$APTEXT	10: 2 F	10: 2		14: 9 F		14: 9		24:37 F	24:37
0	CPW@DA@S	$APTEXT	10: 2	10: 2 F		14: 9		14: 9 F		24:37 F	
20	CRW@CC@N	$APTEXT	22:18	22:18 F							
7	CRW@CC@P	$APTEXT	22:18 F	22:18							
0	CRW@CC@S	$APTEXT	22:18	22:18 F							
10	CRW@CN@N	$APTEXT	16: 7	16: 7 F							
10	CRW@CN@P	$APTEXT	16: 7 F	16: 7							
0	CRW@CN@S	$APTEXT	16: 7	16: 7 F							
377	CRW@DC@M	$APTEXT	8:22								
10	CRW@DC@N	$APTEXT	8:22								
11	CRW@DC@P	$APTEXT	8:22 F	8:22							
10	CRW@DC@S	$APTEXT	8:22	8:22 F							
377	CRW@DS@M	$APTEXT	7:10	15:40							
10	CRW@DS@N	$APTEXT	7:10	15:40							
5	CRW@DS@P	$APTEXT	7:10 F	7:10		15:40 F		15:40			
10	CRW@DS@S	$APTEXT	7:10	7:10 F		15:40		15:40 F			
20	CRW@LC@N	$APTEXT	8:36	8:36 F							
2	CRW@LC@P	$APTEXT	8:36 F	8:36							
0	CRW@LC@S	$APTEXT	8:36	8:36 F							
20	CRW@NC@N	$APTEXT	14: 8	14: 8 F		23:26		23:26 F			
3	CRW@NC@P	$APTEXT	14: 8 F	14: 8		23:26 F		23:26			

0	CRW@NC@S	$APTEXT	14: 8	14: 8 F	23:26	23:26 F				
377	CRW@OS@M	$APTEXT	7:11							
10	CRW@OS@N	$APTEXT	7:11							
4	CRW@OS@P	$APTEXT	7:11 F	7:11						
10	CRW@OS@S	$APTEXT	7:11	7:11 F						
30	CU@@LE	$APTEXT	6: 1							
20	CU@BKP@N	$APTEXT	19:36 F	25: 5 F						
17	CU@BKP@P	$APTEXT	19:36 F	19:36	25: 5 F	25: 5				
0	CU@BKP@S	$APTEXT	19:36 F	25: 5 F						
1	CU@BLO@M	$APTEXT	19: 6 F	30:17	31:19					
1	CU@BLO@N	$APTEXT	19: 6 F	30:17	31:19 F	31:19				
11	CU@BLO@P	$APTEXT	19: 6 F	19: 6	30:17 F	30:17	31:19 F	31:19		
17	CU@BLO@S	$APTEXT	19: 6 F	30:17	30:17 F	31:19	31:19 F			
177776	CU@BLO@X	$APTEXT	19: 6	31:19						
20	CU@BL1@N	$APTEXT	19: 7 F	30:18	30:18 F	31:16 F				
12	CU@BL1@P	$APTEXT	19: 7 F	19: 7	30:18 F	30:18	31:16 F	31:16		
0	CU@BL1@S	$APTEXT	19: 7 F	30:18	30:18 F					
177	CU@BL2@M	$APTEXT	19: 8 F	31:13						
7	CU@BL2@N	$APTEXT	19: 8 F	19: 8	30:19	30:19 F	31:13 F	31:13		
13	CU@BL2@P	$APTEXT	19: 8 F	19: 8	30:19 F	30:19	31:13 F	31:13		
0	CU@BL2@S	$APTEXT	19: 8 F	19: 8	30:19	30:19 F	31:13	31:13 F		
20	CU@BPI@N	$APTEXT	8: 6	8: 6 F	14:30 F	19:41 F				
3	CU@BPI@P	$APTEXT	8: 6 F	8: 6	14:30 F	14:30	19:41 F	19:41		
0	CU@BPI@S	$APTEXT	8: 6	8: 6 F	14:30 F	19:41 F				
20	CU@ERL@N	$APTEXT	25: 7	25: 7 F						
16	CU@ERL@P	$APTEXT	25: 7 F	25: 7						
0	CU@ERL@S	$APTEXT	25: 7	25: 7 F						
20	CU@ERU@N	$APTEXT	25: 6	25: 6 F						
15	CU@ERU@P	$APTEXT	25: 6 F	25: 6						
0	CU@ERU@S	$APTEXT	25: 6	25: 6 F						
1	CU@FOP@M	$APTEXT	31:27	32:12 F	32:28 F					
1	CU@FOP@N	$APTEXT	31:27	31:27 F	32:12 F	32:12	32:28 F			
14	CU@FOP@P	$APTEXT	31:27 F	31:27	32:12 F	32:12	32:28 F	32:28		
1	CU@FOP@S	$APTEXT	31:27	31:27 F	32:12 F	32:12	32:28 F			
137777	CU@FOP@X	$APTEXT	32:28							
777	CU@FWI@M	$APTEXT	8: 9	20: 1						
11	CU@FWI@N	$APTEXT	8: 9	20: 1 F						
10	CU@FWI@P	$APTEXT	8: 9 F	8: 9	20: 1 F	20: 1				
7	CU@FWI@S	$APTEXT	8: 9	8: 9 F	20: 1 F					
177000	CU@FWI@X	$APTEXT	20: 1							
20	CU@LIM@N	$APTEXT	8:10	8:10 F						
24	CU@LIM@P	$APTEXT	8:10 F	8:10						

0	CU@LIM@S	$APTEXT	8:10	8:10 F							
17	CU@MOD@M	$APTEXT	19:20 F	33: 5							
4	CU@MOD@N	$APTEXT	13:11	13:11 F	17:16	17:16 F	19:20 F	19:20	33: 5 F	33: 5	
5	CU@MOD@P	$APTEXT	13:11 F	13:11	17:16 F	17:16	19:20 F	19:20	33: 5 F	33: 5	
0	CU@MOD@S	$APTEXT	13:11	13:11 F	17:16	17:16 F	19:20 F	19:20	33: 5 F	33: 5	
20	CU@NCW@N	$APTEXT	8: 8	8: 8 F	19:42 F						
4	CU@NCW@P	$APTEXT	8: 8 F	8: 8	19:42 F	19:42					
0	CU@NCW@S	$APTEXT	8: 8	8: 8 F	19:42 F						
377	CU@PFI@M	$APTEXT	31:30	32:27 F							
10	CU@PFI@N	$APTEXT	30:32	30:32 F	31:30 F	31:30	32:27 F	32:27			
7	CU@PFI@P	$APTEXT	30:32 F	30:32	31:30 F	31:30	32:27 F	32:27			
0	CU@PFI@S	$APTEXT	30:32	30:32 F	31:30	31:30 F	32:27 F	32:27			
7777	CU@PFU@M	$APTEXT	30:31	31:33	32:26 F						
14	CU@PFU@N	$APTEXT	30:31	31:33 F	31:33	32:26 F					
6	CU@PFU@P	$APTEXT	30:31 F	30:31	31:33 F	31:33	32:26 F	32:26			
4	CU@PFU@S	$APTEXT	30:31	30:31 F	31:33	31:33 F	32:26 F				
170000	CU@PFU@X	$APTEXT	31:33	32:26							
177	CU@PRI@M	$APTEXT	31:42	32:20 F							
7	CU@PRI@N	$APTEXT	30:36	30:36 F	31:42 F	31:42	32:20 F	32:20			
10	CU@PRI@P	$APTEXT	30:36 F	30:36	31:42 F	31:42	32:20 F	32:20			
0	CU@PRL@S	$APTEXT	30:36	30:36 F	31:42	31:42 F	32:20 F	32:20			
377	CU@PRU@M	$APTEXT	30:35	32: 3	32:19 F						
10	CU@PRU@N	$APTEXT	30:35	32: 3 F	32: 3	32:19 F					
7	CU@PRU@P	$APTEXT	30:35 F	30:35	32: 3 F	32: 3	32:19 F	32:19			
10	CU@PRU@S	$APTEXT	30:35	30:35 F	32: 3	32: 3 F	32:19 F				
177400	CU@PRU@X	$APTEXT	32: 3	32:19							
20	CU@PTR@N	$APTEXT	8: 7	8: 7 F	14:29 F	19:30 F					
2	CU@PTR@P	$APTEXT	8: 7 F	8: 7	14:29 F	14:29	19:30 F	19:30			
0	CU@PTR@S	$APTEXT	8: 7	8: 7 F	14:29 F	19:30 F					
20	CU@RLI@N	$APTEXT	21:35 F	22:36	22:36 F	23: 4 F					
23	CU@RLI@P	$APTEXT	21:35 F	21:35	22:36 F	22:36	23: 4 F	23: 4			
0	CU@RLI@S	$APTEXT	21:35 F	22:36	22:36 F	23: 4 F					
20	CU@RLU@N	$APTEXT	21:34 F	22:35	22:35 F	23: 3 F					
22	CU@RLU@P	$APTEXT	21:34 F	21:34	22:35 F	22:35	23: 3 F	23: 3			
0	CU@RLU@S	$APTEXT	21:34 F	22:35	22:35 F	23: 3 F					
1	CU@ROP@M	$APTEXT	12: 8 F	32:21 F							
1	CU@ROP@N	$APTEXT	12: 8 F	12: 8	31:39	31:39 F	32:21 F	32:21			
14	CU@ROP@P	$APTEXT	12: 8 F	12: 8	31:39 F	31:39	32:21 F	32:21			
0	CU@ROP@S	$APTEXT	12: 8 F	12: 8	31:39	31:39 F	32:21 F	32:21			
77	CU@UBC@M	$APTEXT	13:19	30:27	30:28						
6	CU@UBC@N	$APTEXT	13:19 F	13:19	30:27	30:27 F	30:28 F	30:28			
5	CU@UBC@P	$APTEXT	13:19 F	13:19	30:27 F	30:27	30:28 F	30:28			

4	CU@UBC@S	$APTEXT	13:19	F	13:19	30:27	30:27	F	30:28	F	30:28
6	CU@WDS	$APTEXT	7:39		20: 8	25: 9					
4	CW@@LE	$APTEXT	6: 1		6:20						
1	CW@BLO@M	$APTEXT	30:20								
1	CW@BLO@N	$APTEXT	30:20	F							
1	CW@BLO@P	$APTEXT	30:20	F	30:20						
17	CW@BLO@S	$APTEXT	30:20	F							
177776	CW@BLO@X	$APTEXT	30:20								
20	CW@BL1@N	$APTEXT	30:21	F							
2	CW@BL1@P	$APTEXT	30:21	F	30:21						
0	CW@BL1@S	$APTEXT	30:21	F							
177	CW@BL2@M	$APTEXT	30:22								
7	CW@BL2@N	$APTEXT	30:22	F	30:22						
3	CW@BL2@P	$APTEXT	30:22	F	30:22						
0	CW@BL2@S	$APTEXT	30:22	F	30:22						
777	CW@FWI@M	$APTEXT	30:11								
11	CW@FWI@N	$APTEXT	30:11	F							
3	CW@FWI@P	$APTEXT	30:11	F	30:11						
7	CW@FWI@S	$APTEXT	30:11	F							
177000	CW@FWI@X	$APTEXT	30:11								
17	CW@MOD@M	$APTEXT	30:10								
4	CW@MOD@N	$APTEXT	30:10	F	30:10						
0	CW@MOD@P	$APTEXT	30:10	F							
0	CW@MOD@S	$APTEXT	30:10	F	30:10						
1	CW@NUL@M	$APTEXT	30:39								
1	CW@NUL@N	$APTEXT	30:39	F	30:39						
0	CW@NUL@P	$APTEXT	30:39	F							
14	CW@NUL@S	$APTEXT	30:39	F	30:39						
377	CW@PFL@M	$APTEXT	30:34								
10	CW@PFL@N	$APTEXT	30:34	F	30:34						
2	CW@PFL@P	$APTEXT	30:34	F	30:34						
0	CW@PFL@S	$APTEXT	30:34	F	30:34						
7777	CW@PFU@M	$APTEXT	30:33								
14	CW@PFU@N	$APTEXT	30:33	F							
1	CW@PFU@P	$APTEXT	30:33	F	30:33						
4	CW@PFU@S	$APTEXT	30:33	F							
170000	CW@PFU@X	$APTEXT	30:33								
177	CW@PRL@M	$APTEXT	30:38								
7	CW@PRL@N	$APTEXT	30:38	F	30:38						
3	CW@PRL@P	$APTEXT	30:38	F	30:38						
0	CW@PRL@S	$APTEXT	30:38	F	30:38						
377	CW@PRU@M	$APTEXT	30:37								

10	CW@PRU@N	$APTEXT	30:37 F									
2	CW@PRU@P	$APTEXT	30:37 F	30:37								
10	CW@PRU@S	$APTEXT	30:37 F									
177400	CW@PRU@X	$APTEXT	30:37									
77	CW@UBC@M	$APTEXT	30:30									
6	CW@UBC@N	$APTEXT	30:30 F	30:30								
0	CW@UBC@P	$APTEXT	30:30 F									
4	CW@UBC@S	$APTEXT	30:30 F	30:30								
20	DA@HSI@N	$APTEXT	28: 1 F									
25	DA@HSI@P	$APTEXT	28: 1 F	28: 1								
0	DA@HSI@S	$APTEXT	28: 1 F									
20	DA@HSU@N	$APTEXT	27:42 F									
24	DA@HSU@P	$APTEXT	27:42 F	27:42								
0	DA@HSU@S	$APTEXT	27:42 F									
20	DA@IFC@N	$APTEXT	28: 3 F									
1	DA@IFC@P	$APTEXT	28: 3 F	28: 3								
0	DA@IFC@S	$APTEXT	28: 3 F									
20	DA@RSC@N	$APTEXT	28: 2 F									
27	DA@RSC@P	$APTEXT	28: 2 F	28: 2								
0	DA@RSC@S	$APTEXT	28: 2 F									
20	DC@BFA@N	$APTEXT	23:40	23:40 F								
16	DC@BFA@P	$APTEXT	23:40 F	23:40								
0	DC@BFA@S	$APTEXT	23:40	23:40 F								
20	DC@BFB@N	$APTEXT	23:41	23:41 F								
17	DC@BFB@P	$APTEXT	23:41 F	23:41								
0	DC@BFB@S	$APTEXT	23:41	23:41 F								
20	DC@BFZ@N	$APTEXT	7:19	7:19 F								
20	DC@BFZ@P	$APTEXT	7:19 F	7:19								
0	DC@BFZ@S	$APTEXT	7:19	7:19 F								
1	DC@BIG@M	$APTEXT	11:18	25:37 F								
1	DC@BIG@N	$APTEXT	11:18	11:18 F	25:37 F	25:37						
21	DC@BIG@P	$APTEXT	11:18 F	11:18	25:37 F	25:37						
11	DC@BIG@S	$APTEXT	11:18	11:18 F	25:37 F	25:37						
20	DC@BRQ@N	$APTEXT	27: 7	27: 7 F	28:24 F							
36	DC@BRQ@P	$APTEXT	27: 7 F	27: 7	28:24 F	28:24						
0	DC@BRQ@S	$APTEXT	27: 7	27: 7 F								
114	DC@C0B@P	$APTEXT	7:29									
74	DC@C0I@P	$APTEXT	7:28									
140	DC@C1B@P	$APTEXT	7:31									
120	DC@C1I@P	$APTEXT	7:30									
1	DC@CHK@M	$APTEXT	11: 4 F	15:22	21: 5	21:14 F	23:10 F					
1	DC@CHK@N	$APTEXT	11: 4 F	11: 4	15:22	15:22 F	21: 5	21: 5 F	21:14 F	23:10 F		

			23:10							
21	DC@CHK@P	$APTEXT	11: 4 F	11: 4	15:22 F	15:22	21: 5 F	21: 5	21:14 F	21:14
			23:10 F	23:10						
5	DC@CHK@S	$APTEXT	11: 4 F	11: 4	15:22	15:22 F	21: 5	21: 5 F	21:14 F	23:10 F
			23:10							
175777	DC@CHK@X	$APTEXT	21:14							
20	DC@DAL@N	$APTEXT	7:14	7:14 F						
22	DC@DAL@P	$APTEXT	7:14 F	7:14						
0	DC@DAL@S	$APTEXT	7:14	7:14 F						
20	DC@DMS@N	$APTEXT	10:22	10:22 F	10:28 F	28:34 F				
54	DC@DMS@P	$APTEXT	10:22 F	10:22	10:28 F	10:28	28:34 F	28:34		
0	DC@DMS@S	$APTEXT	10:22	10:22 F	10:28 F					
20	DC@DSL@N	$APTEXT	7:13	7:13 F						
15	DC@DSL@P	$APTEXT	7:13 F	7:13						
0	DC@DSL@S	$APTEXT	7:13	7:13 F						
20	DC@DSU@N	$APTEXT	7:12	7:12 F						
14	DC@DSU@P	$APTEXT	7:12 F	7:12						
0	DC@DSU@S	$APTEXT	7:12	7:12 F						
1	DC@ERP@M	$APTEXT	10:18	10:21 F	14:15	14:34 F	22: 3 F	22:21 F	27: 5	
1	DC@ERP@N	$APTEXT	10:18	10:18 F	10:21 F	14:15	14:15 F	14:34 F	22: 3 F	22: 3
			22:21 F	22:21	27: 5	27: 5 F				
21	DC@ERP@P	$APTEXT	10:18 F	10:18	10:21 F	10:21	14:15 F	14:15	14:34 F	14:34
			22: 3 F	22: 3	22:21 F	22:21	27: 5 F	27: 5		
13	DC@ERP@S	$APTEXT	10:18	10:18 F	10:21 F	14:15	14:15 F	14:34 F	22: 3 F	22: 3
			22:21 F	22:21	27: 5	27: 5 F				
177757	DC@ERP@X	$APTEXT	10:21	14:34						
20	DC@HSL@N	$APTEXT	7:17	7:17 F	20:18 F					
40	DC@HSL@P	$APTEXT	7:17 F	7:17	20:18 F	20:18				
0	DC@HSL@S	$APTEXT	7:17	7:17 F	20:18 F					
20	DC@HSU@N	$APTEXT	7:16	7:16 F	20:17 F					
37	DC@HSU@P	$APTEXT	7:16 F	7:16	20:17 F	20:17				
0	DC@HSU@S	$APTEXT	7:16	7:16 F	20:17 F					
20	DC@RSC@N	$APTEXT	7:15	7:15 F	20:16 F					
46	DC@RSC@P	$APTEXT	7:15 F	7:15	20:16 F	20:16				
0	DC@RSC@S	$APTEXT	7:15	7:15 F	20:16 F					
20	DC@IMS@N	$APTEXT	7:18	7:18 F	28:32 F					
45	DC@IMS@P	$APTEXT	7:18 F	7:18	28:32 F	28:32				
0	DC@IMS@S	$APTEXT	7:18	7:18 F						
3	DD@FRM@M	$APTEXT	7:24							
2	DD@FRM@N	$APTEXT	7:24	7:24 F						
17	DD@FRM@P	$APTEXT	7:24 F	7:24						
11	DD@FRM@S	$APTEXT	7:24	7:24 F						

0X	DDALGETA		28: 6 D	28: 6									
0X	DMGB		28: 7 D	28: 7									
0X	DS		5:35 D	28: 5									
0X	ED		5:36 D	6:29	6:30								
0X	EH		5:37 D	6:35									
7	ES$BRE	$APTEXT	16:27										
1	ES$SOR	$APTEXT	16:37										
0X	ESWP		6:40 D	6:40									
0X	ETOQ		6:34 D	6:34									
30	F$GMEM	$APTEXT	6:14										
1	F2$NVB@M	$APTEXT	15:39 F										
1	F2$NVB@N	$APTEXT	15:39 F	15:39									
13	F2$NVB@P	$APTEXT	15:39 F	15:39									
1	F2$NVB@S	$APTEXT	15:39 F	15:39									
1	F2$SIO@M	$APTEXT	17: 4 F	21:36 F									
1	F2$SIO@N	$APTEXT	15:36	15:36 F	17: 4 F	17: 4	21:36 F	21:36					
13	F2$SIO@P	$APTEXT	15:36 F	15:36	17: 4 F	17: 4	21:36 F	21:36					
0	F2$SIO@S	$APTEXT	15:36	15:36 F	17: 4 F	17: 4	21:36 F	21:36					
7	FC$EODR	$APTEXT	18:38										
31	FC$EOFR	$APTEXT	18:13										
30	FC$EORR	$APTEXT	17:22										
1	FC$READ	$APTEXT	8:30	19:32	24:16								
2	FC$WRITE	$APTEXT	14: 4	21:19	22:34	24: 9	25:36						
1	FL$DBI@M	$APTEXT	8:42	9: 3 F									
1	FL$DBI@N	$APTEXT	8:42	8:42 F	9: 3 F								
4	FL$DBI@P	$APTEXT	8:42 F	8:42	9: 3 F	9: 3							
3	FL$DBI@S	$APTEXT	8:42	8:42 F	9: 3 F								
167777	FL$DBI@X	$APTEXT	9: 3										
1	FL$RDY@M	$APTEXT	15:32 F	16:28 F	16:39 F	20:19 F							
1	FL$RDY@N	$APTEXT	15:32 F	15:32	16:28 F	16:39 F	20:19 F	20:19					
4	FL$RDY@P	$APTEXT	15:32 F	15:32	16:28 F	16:28	16:39 F	16:39	20:19 F	20:19			
2	FL$RDY@S	$APTEXT	15:32 F	15:32	16:28 F	16:39 F	20:19 F	20:19					
157777	FL$RDY@X	$APTEXT	16:28	16:39									
1	FL$TDM@M	$APTEXT	15:25 F										
1	FL$TDM@N	$APTEXT	15:25 F	15:25									
4	FL$TDM@P	$APTEXT	15:25 F	15:25									
1	FL$TDM@S	$APTEXT	15:25 F	15:25									
1	FL$TIP@M	$APTEXT	9:29										
1	FL$TIP@N	$APTEXT	9:29	9:29 F									
4	FL$TIP@P	$APTEXT	9:29 F	9:29									
5	FL$TIP@S	$APTEXT	9:29	9:29 F									
1	FM$BLKD	$APTEXT	12: 7										

0	FM$TRNS	$APTEXT	11:26							
4	FN$MOSRD	$APTEXT	7:41	15:13	25:24	29: 4	29:21			
5	FN$MOSWR	$APTEXT	11:34	12:41	20: 9	25:10	26:37	29: 6	31: 6	
0X	HB		5:38 D							
0X	HC		5:39 D							
10	HD@WDS	$APTEXT	14:25	26:11						
4	MC$CPUWR	$APTEXT	28: 3							
0	MD$BCW	$APTEXT	11: 1	13:18	19:20	30:16	31:12	32:42		
7	MD$BOR	$APTEXT	21:29							
17	MD$EOD	$APTEXT	18:42							
16	MD$EOF	$APTEXT	18:17	18:27	32:25					
10	MD$EOR	$APTEXT	11: 3	11:42	13:13	17:28	17:42	32:18	33: 2	
14	MD$ERR	$APTEXT	26:26	28:14						
2	MIO$BMRI	$APTEXT	29: 5							
4	MIO$BMWI	$APTEXT	29: 7							
0X	O$TDEM		1: 4 D	1: 4						
3	OS$DN	$APTEXT	22: 2							
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
0	PT$HUH	$APTEXT	6:15	6:36	19:10	21:30	24: 5	24: 7	24: 9	25:36
			29: 9							
2	PXS	$APTEXT	25:39							
4	QU$POP	$APTEXT	6:30							
11	SM@FST@N	$APTEXT	1: 4							
7	SM@NUM@N	$APTEXT	1: 4							
10	ST$CE	$APTEXT	9:28							
4	ST$DE	$APTEXT	9:34	10:42	19:33	19:35	22:20			
100	ST$MD	$APTEXT	9:26							
2	ST$UC	$APTEXT	9:26	14:38	15:42	19:33				
1	ST$UE	$APTEXT	11:10							
0X	TDEMMEM		5:40 D	6:10	6:16					
35	TF$TEX5	$APTEXT	8:23							
36	TF$TEX6	$APTEXT	20:31							
20	TQ@BCS@N	$APTEXT	21:13 F							
34	TQ@BCS@P	$APTEXT	21:13 F	21:13						
0	TQ@BCS@S	$APTEXT	21:13 F							
77777	TQ@FCN@M	$APTEXT	7:23							
17	TQ@FCN@N	$APTEXT	7:23							
16	TQ@FCN@P	$APTEXT	7:23 F	7:23						
1	TQ@FCN@S	$APTEXT	7:23	7:23 F						
0X	XCHT		5:41 D	16: 9						

SYMBOL QUALIFIER IS TDEM

0	$CONST	9: 3 D	9: 3 F	10:21 D	10:21 F	10:28 D	10:28 F	11: 4 D	11: 4 F
		11: 4	12: 8 D	12: 8 F	12: 8	14:34 D	14:34 F	15:25 D	15:25 F
		15:25	15:32 D	15:32 F	15:32	15:39 D	15:39 F	15:39	16:27 D
		16:27 F	16:27	16:28 D	16:28 F	16:37 D	16:37 F	16:37	16:38 D
		16:38 F	16:38	16:39 D	16:39 F	17: 4 D	17: 4 F	17: 4	19: 6 D
		19: 6 F	19: 7 D	19: 7 F	19: 8 D	19: 8 F	19: 8	19:20 D	19:20 F
		19:20	20:19 D	20:19 F	20:19	21:14 D	21:14 F	21:36 D	21:36 F
		21:36	22: 3 D	22: 3 F	22: 3	22:21 D	22:21 F	22:21	23:10 D
		23:10 F	23:10	25:37 D	25:37 F	25:37	28: 3 D	28: 3 F	28: 3
		32:12 D	32:12 F	32:12	32:19 D	32:19 F	32:20 D	32:20 F	32:20
		32:21 D	32:21 F	32:21	32:26 D	32:26 F	32:27 D	32:27 F	32:27
		32:28 D	32:28 F						
515	$REGORG	3:41 D	3:41 F	3:42	3:42 D	4: 1	4: 1 D	4: 2	4: 2 D
		4: 3	4: 3 D	4: 4	4: 4 D	4: 4 F	4: 5	4: 5 D	4: 6
		4: 6 D	4: 7	4: 7 D	4: 8	4: 8 D	4: 9	4: 9 D	4:10
		4:10 D	4:10 F	4:11	4:11 D	4:12	4:12 D	4:13	4:13 D
		4:14	4:14 D	4:15	4:15 D	4:15 F	4:16	4:16 D	4:17
		4:17 D	4:18	4:18 D	4:19	4:19 D	4:20	4:20 D	4:21
		4:21 D	4:22	4:22 D	4:23	4:23 D	4:24	4:24 D	4:25
		4:25 D	4:26	4:26 D	4:27	4:27 D	4:28	4:28 D	4:29
		4:29 D	4:30	4:30 D	4:31	4:31 D	4:32	4:32 D	4:33
		4:33 D	4:34	4:34 D	4:35	4:35 D	4:36	4:36 D	4:37
		4:37 D	4:38	4:38 D	4:39	4:39 D	4:40	4:40 D	4:41
		4:41 D	4:42	4:42 D	5: 1	5: 1 D	5: 2	5: 2 D	5: 3
		5: 3 D	5: 4	5: 4 D	5: 5	5: 5 D	5: 6	5: 6 D	5: 7
		5: 7 D	5: 8	5: 8 D	5: 9	5: 9 D	5:10	5:10 D	5:11
		5:11 D	5:12	5:12 D	5:13	5:13 D	5:14	5:14 D	5:15
		5:15 D	5:16	5:16 D	5:17	5:17 D	5:18	5:18 D	5:19
		5:19 D	5:20	5:20 D	5:21	5:21 D	5:22	5:22 D	5:23
		5:23 D	5:24	5:24 D	5:25	5:25 D	5:26	5:26 D	5:26 F
		6:14							
0	%NP	1: 4	4:15 D						
430	%P	1: 4	4:15 D	4:15					
410	%S1	3:41	3:42 D	6:10	6:16	6:17	6:30	7:10	7:11
		7:12	7:13	7:14	7:15	7:16	7:17	7:18	7:19
		7:23	7:24	8: 6	8: 7	8: 8	8: 9	8:10	8:22
		8:36	8:42	9: 3	9:26	9:28	9:29	10: 2	10: 5
		10: 6	10:18	10:21	10:22	10:28	10:42	11: 4	11:10
		11:11	11:18	12: 8	13:11	13:19	14: 8	14: 9	14:15
		14:29	14:30	14:34	14:38	15:22	15:25	15:32	15:36

		15:39	15:40	15:42	16: 7	16: 9	16:10	16:27	16:28
		16:37	16:38	16:39	17: 4	17:16	19: 6	19: 7	19: 8
		19:20	19:30	19:35	19:36	19:41	19:42	20: 1	20:16
		20:17	20:18	20:19	21: 5	21:13	21:14	21:27	21:32
		21:33	21:34	21:35	21:36	22: 3	22:18	22:20	22:21
		22:35	22:36	23: 3	23: 4	23:10	23:26	23:40	23:41
		24:37	25: 5	25: 6	25: 7	25:30	25:37	26:24	26:27
		26:32	27: 5	27: 7	27:42	28: 1	28: 2	28: 3	28:12
		28:24	28:32	28:34	30: 3	30:11	30:17	30:18	30:19
		30:20	30:21	30:22	30:27	30:28	30:31	30:32	30:33
		30:34	30:35	30:36	30:37	30:38	31:16	31:27	31:39
		32:12	32:19	32:20	32:21	32:26	32:27	32:28	33: 5
411	%S2	3:41	4: 1 D						
412	%S3	3:41	4: 2 D						
413	%S4	3:41	4: 3 D						
414	%S5	3:41	4: 4 D						
415	%T1	4: 5 D	31:13	31:19	31:30	31:33	31:42	32: 3	
416	%T2	4: 6 D	31:13	31:19	31:30	31:33	31:42	32: 3	
417	%T3	4: 7 D	23: 2	31:13	31:19	31:30	31:33	31:42	32: 3
420	%T4	4: 8 D							
421	%T5	4: 9 D							
422	%T6	4:10 D							
423	%W1	4:11 D	8:22	8:23	9:29	9:31	15:36	15:38	15:40
		15:42	16: 7	16: 8	16: 9	19:33	19:35	21:11	21:13
		21:32	21:34	22:35	22:38	23: 2	23: 3	23: 9	
424	%W2	4:12 D	16:10	16:12	16:27	16:37	16:38	21:33	21:35
		22:36	22:38	22:41	23: 2	23: 4	23: 9		
425	%W3	4:13 D							
426	%W4	4:14 D							
427	%W5	4:15 D							
0	@EPUT	25:39 D							
456	ADDL	4:38 D	7:38	11:30	12:21	15: 9	20: 6	25: 7	25:21
		26:34	29:13	31: 3					
455	ADDU	4:37 D	7:37	11:30	12:21	15: 9	20: 5	25: 6	25:20
		26:33	29:12	31: 3					
462	BPT	4:42 D	8: 6	8:23	11:30	11:36	11:42	12: 9	12:21
		12:26	13: 5	13:34	14:19	14:21	14:22	14:30	14:33
		15: 9	15:18	15:24	19:19	19:41	26:13	26:30	32:35
		33: 7							
444	BSZ	4:28 D	7:19	10: 6	22:38	22:39	24: 7		
470	BUFF	5: 6 D	10: 2	10: 3	11:31	12:22	14: 9	14:10	15:10
		21:11	23:42	24: 5					

2511	BUFGET	11:29	12:17	14:42	17:15	25:18 D	32:41		
2614	BUFOUT	12: 1	26:20 D	32:40					
2667	BUFSEND	10:25	27: 4 D						
451	COB	4:33 D	7:29	9:11	9:13	23:28	23:33		
450	COT	4:32 D	7:28	9:11	9:12	23:28	23:32		
453	CIB	4:35 D	7:31	9:16	23:30				
452	CIT	4:34 D	7:30	9:15	23:29				
471	CNT	5: 7 D	10: 5	10:10	11:27	11:32	12:14	15:11	16:19
		16:26	22:39	22:41	23: 2	24: 7	24: 9		
2040	COMSUB	10:35	15: 5	21: 4 D					
467	CPW	5: 5 D	8:36	9:11	9:23	9:24	9:25	10: 2	10: 5
		10: 6	14: 8	14: 9	20:31				
432	CUR	4:18 D	6:21	7:40	8: 6	8: 7	8: 8	8: 9	8:10
		12: 8	13:11	13:19	14:29	14:30	17:16	19: 6	19: 7
		19: 8	19:20	19:30	19:36	19:41	19:42	20: 1	20: 7
		21:34	21:35	22:35	22:36	23: 3	23: 4	25: 5	25: 6
		25: 7	25: 8	30:17	30:18	30:19	30:27	30:28	30:31
		30:32	30:35	30:36	31:13	31:16	31:19	31:27	31:30
		31:33	31:39	31:42	32: 3	32:12	32:19	32:20	32:21
		32:26	32:27	32:28	33: 5				
430	CWRD	4:16 D	6:11	6:14	6:16	6:20	30: 2	30: 3	30:10
		30:11	30:20	30:21	30:22	30:30	30:33	30:34	30:37
		30:38	30:39	31: 4					
3141	CWWRT	13:22	13:28	13:35	17:32	18: 2	18: 8	18:21	18:29
		18:33	19: 4	30: 1 D					
437	DAI	4:23 D	7:14	7:23	7:24	21:13	27:42	28: 1	28: 2
		28: 3	28: 5						
466	DATA	5: 4 D	12:14	12:16	12:20	12:29	12:33	12:37	
514	DBI	5:26 D	7: 3	9: 5	23:24				
52	DEQUEUE	6:27 D	16:29	16:40	20:38				
477	DONE	5:13 D	12:13	12:16	12:17	12:18			
454	DSCLIM	4:36 D	8:10	14:25	26:11				
436	DSL	4:22 D	7:13	7:38	20: 6	25:19			
434	DST	4:20 D	7:10	8:23	9: 4	9:26	9:28	9:34	10:42
		11:10	14:38	19:35	22:20				
435	DSU	4:21 D	7:12	7:37	20: 5	25:19			
503	ENTL	5:17 D	25:19	25:21	26:34				
431	ENTRY	4:17 D	6:20	6:21	21:27	21:32	21:33	25:22	25:29
		25:30	26:24	26:27	26:32	26:35	28:12		
502	ENTU	5:16 D	25:19	25:20	26:33				
461	ICT	4:41 D	7:41	11:34	12:41	15:13	20: 9	25:10	25:24
		26:37	29: 4	29: 6	29:21	31: 6			

446	FMT	4:30 D	7:24	11:26	12: 7				
445	FUNC	4:29 D	7:23	8:30	14: 4	17:22	18:13	18:38	19:32
		21:19	22:34	24: 9	24:16	25:36			
465	FWI	5: 3 D	8: 9	13: 4	17:31	18:20	19: 3	20: 1	30:11
		32:36	33: 6						
442	HSL	4:26 D	7:17	20:18	20:31	28: 1	28:23		
441	HSU	4:25 D	7:16	20:17	20:31	27:42	28:23		
460	LOC	4:40 D	7:40	11:31	12:22	15:10	20: 7	25: 8	25:22
		26:35	29:14	31: 4					
3057	MEMIO	7:42	11:35	12:42	15:14	20:10	25:11	25:25	26:38
		29: 1 D	31: 7						
40	MEMSIZ	6: 1 D	6:14						
474	MODE	5:10 D	13:11	13:18	17:16	17:17	17:28	17:42	18:17
		18:27	18:42	30:10	30:16	31:12	32:18	32:25	32:42
		33: 2	33: 5						
501	MSL	5:15 D	11:30	12:21	15: 9	25:30	25:35	31: 3	
500	MSU	5:14 D	11:30	12:21	15: 9	25:29	25:35	31: 3	
464	NCW	5: 2 D	8: 8	13:27	17:27	17:31	17:41	18: 7	18:20
		18:32	19: 3	19:18	19:42	31: 3	32:36	32:37	33: 7
100000	NOSTAT	6: 2 D	9: 4						
505	NUL	5:19 D	7: 2	17:30	18:19	19: 2	30:39	32:34	
2603	NXTBUF	15:26	26: 9 D	26:39					
476	OFFSET	5:12 D	12:12	12:22	12:31	12:34	13:13		
447	OST	4:31 D	7:11	22: 2					
506	P0	5:20 D	9:12	9:15	9:19	9:24	10:18	10:20	12:26
		12:29	12:30	14:15	14:17	14:18	14:32	15:22	15:24
		23:26	23:28	23:29	23:32	23:36	24:29	24:37	24:38
		26:24	26:26	26:30	26:32	27: 6	27:12	27:14	27:21
		27:26	27:27	28:22	28:23	30: 2	30: 3	30: 4	30: 5
		30:17	30:20	30:27	30:29	30:30	30:31	30:33	
507	P1	5:21 D	9:13	9:16	9:24	14:19	14:33	23:30	23:33
		24:29	27:27	27:29	27:36	28: 2	28:17	28:20	28:22
		30:18	30:21	30:32	30:34				
510	P2	5:22 D	23:40	24: 5	24:18	24:21	24:31	24:34	27: 7
		27:11	27:36	30:19	30:22	30:35	30:37		
511	P3	5:23 D	23:41	24: 5	24:19	24:32	27: 5	27: 9	28:31
		30:36	30:38						
512	P4	5:24 D	10:22	10:23	10:24	10:26	23:42	24:18	24:19
		24:21	24:31	24:32	24:34	24:37			
513	P5	5:25 D	10:23	10:29					
475	PBC	5:11 D	11:28	12:11	17:14	26:30			
463	PTR	5: 1 D	8: 7	8:23	14:18	14:24	14:25	14:26	14:29

		14:32	19:30	19:36	20:31	25: 5	25:19	26:10	26:11
		26:12							
440	RSC	4:24 D	7:15	20:16	20:31	27:21	27:26	27:29	28:15
		28:17							
2461	SAVPOS	14:31	22: 5	22:17	25: 4 D				
504	SENT	5:18 D	10:13						
457	SIZ	4:39 D	7:39	11:32	11:36	12:30	12:33	12:37	13: 4
		13: 5	15:11	15:18	20: 8	25: 9	25:23	26:36	29:15
		31: 5							
473	STAT	5: 9 D	11: 1	11: 3	11:42	13:13	17:17	17:29	18: 1
		18:18	18:28	19: 1	26:27				
433	TCB	4:19 D	6:35	7:10	7:11	7:12	7:13	7:14	7:15
		7:16	7:17	7:18	7:19	7:28	7:29	7:30	7:31
		8:22	8:36	8:42	9: 3	9:29	10:18	10:21	10:22
		10:28	11: 4	11:18	14: 8	14:15	14:34	15:22	15:25
		15:32	15:36	15:39	15:40	16: 7	16:28	16:39	17: 4
		20:16	20:17	20:18	20:19	21: 5	21:14	21:36	22: 3
		22:18	22:21	23:10	23:26	23:40	23:41	25:37	27: 5
		27: 7	28:24	28:32	28:34				
443	TMS	4:27 D	7:18	8:23	10:29	27: 6	28:20		
472	UBC	5: 8 D	7: 1	10:10	10:11	11:28	13:19	30:28	
1675	UPDATE	9:32	11:11	11:19	11:27	14:39	16: 1	19:29 D	25:41

```
0     <macro>                    OVERLAY   COMM01                                          COMM01.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'              OVERLAY  1
```

```
*****************************************************************  COMM01.5
*                                                               *  COMM01.6
*                    Station command overlay.                   *  COMM01.7
*                                                               *  COMM01.8
*        BABEL       Bring up the station messages display.     *  COMM01.9
*                                                               *  COMM01.10
*        CLEAR       Terminate any active displays and clear the display *  COMM01.11
*                    area of the console.                       *  COMM01.12
*                                                               *  COMM01.13
*        CONC        Monitor the concentrator activities        *  COMM01.14
*                                                               *  COMM01.15
*        DEBUG       Initiate the debug display directory display.  *  COMM01.16
*                                                               *  COMM01.17
*        DISK        Initiate the disk statistics display.      *  COMM01.18
*                                                               *  COMM01.19
*        DKERR       Initiate the disk error display.           *  COMM01.20
*                                                               *  COMM01.21
*        DSPL        Initiate the CPU formatted displays        *  COMM01.22
*                                                               *  COMM01.23
*        ERROR       Initiate the error information display.     *  COMM01.24
                                                                   COMM01.25
*        HELP        Help facility                              *  COMM01.26
*                                                               *  COMM01.27
*        LINK        Initiate the CRAY-1 link status display.   *  COMM01.28
*                                                               *  COMM01.29
*        LPATH       NSC/FEI logical path display               *  COMM01.30
*                                                               *  COMM01.31
*        MONITOR     Monitor IOP system statistics.             *  COMM01.32
*                                                               *  COMM01.33
*        RSTAT       Generic resource status display.           *  COMM01.34
*                                                               *  COMM01.35
*        SCROLL      Terminate any active displays.  The display area of *  COMM01.36
*                    the screen is cleared and used to scroll command *  COMM01.37
*                    lines.                                     *  COMM01.38
*                                                               *  COMM01.39
*        STATCLASS   Job class status.                          *  COMM01.40
*                                                               *  COMM01.41
*        STATION     Initiate the station status display.       *  COMM01.42
*                                                               *  COMM01.43
*        STATUS      Initiate the CRAY-1 system status display. *  COMM01.44
*                                                               *  COMM01.45
*        STORAGE     Initiate the CRAY-1 mass storage status display.  *  COMM01.46
```

```
*                                                                      *  COMM01.47
*        SIRSTAT       INITIATE THE STREAM STATUS DISPLAY              *  COMM01.48
*                                                                      *  COMM01.49
*        STP       INITIATE STP DISPLAY                                *  COMM01.50
*                                                                      *  COMM01.51
*        SUMMARY   INITIATE IOS SUMMARY DISPLAY                        *  COMM01.52
*                                                                      *  COMM01.53
*        SWAP      Display statistics on swap space availability       *  COMM01.54
*                                                                      *  COMM01.55
*        TAPE      Initiate the tape configuration display.            *  COMM01.56
*                                                                      *  COMM01.57
*        TJOB      Initiate the tape job(s) status display.            *  COMM01.58
*                                                                      *  COMM01.59
*        XJSTAT    Extended job status display.                        *  COMM01.60
*                                                                         COMM01.61
*        mn.       Select debug displays for the left and right halves *  COMM01.62
*                  of the screen.                                      *  COMM01.63
*                                                                      *  COMM01.64
************************************************************************  COMM01.65
```

```
6     <macro>                    REGDEFS   ,(PAR,BASE,CMDOFF,CMDLEN,OFFSET,COUNT),                COMM01.67
                                           (LINE,NUMBER,SCROLL,OUTPUT,MASTER,OLAY,PARAM,PARAM2,   COMM01.68
                                           QUEUES,LIST,LEFT,RIGHT,FEI)                            COMM01.69
                                                                                                        $REG    3
410          %S1       EQUALS    $REGORG                                                                $REG    3
411          %S2       EQUALS    $REGORG                                                                $REG    3
412          %S3       EQUALS    $REGORG                                                                $REG    3
413          %S4       EQUALS    $REGORG                                                                $REG    3
414          %S5       EQUALS    $REGORG                                                                $REG    3
415          %T1       EQUALS    $REGORG                                                                $REG    3
416          %T2       EQUALS    $REGORG                                                                $REG    3
417          %T3       EQUALS    $REGORG                                                                $REG    3
420          %T4       EQUALS    $REGORG                                                                $REG    3
421          %T5       EQUALS    $REGORG                                                                $REG    3
422          %T6       EQUALS    $REGORG                                                                $REG    3
423          %W1       EQUALS    $REGORG                                                                $REG    3
424          %W2       EQUALS    $REGORG                                                                $REG    3
425          %W3       EQUALS    $REGORG                                                                $REG    3
426          %W4       EQUALS    $REGORG                                                                $REG    3
427          %W5       EQUALS    $REGORG                                                                $REG    3
430          PAR       EQUALS    $REGORG                                                                $REG    3
431          BASE      EQUALS    $REGORG                                                                $REG    3
432          CMDOFF    EQUALS    $REGORG                                                                $REG    3
433          CMDLEN    EQUALS    $REGORG                                                                $REG    3
434          OFFSET    EQUALS    $REGORG                                                                $REG    3
435          COUNT     EQUALS    $REGORG                                                                $REG    3
436          LINE      EQUALS    $REGORG                                                                $REG    3
437          NUMBER    EQUALS    $REGORG                                                                $REG    3
440          SCROLL    EQUALS    $REGORG                                                                $REG    3
441          OUTPUT    EQUALS    $REGORG                                                                $REG    3
442          MASTER    EQUALS    $REGORG                                                                $REG    3
443          OLAY      EQUALS    $REGORG                                                                $REG    3
444          PARAM     EQUALS    $REGORG                                                                $REG    3
445          PARAM2    EQUALS    $REGORG                                                                $REG    3
446          QUEUES    EQUALS    $REGORG                                                                $REG    3
447          LIST      EQUALS    $REGORG                                                                $REG    3
450          LEFT      EQUALS    $REGORG                                                                $REG    3
451          RIGHT     EQUALS    $REGORG                                                                $REG    3
452          FEI       EQUALS    $REGORG                                                                $REG    3
                                                                                                   COMM01.70
                                 EXT       %FEI                                                    COMM01.71
                                 EXT       CF$LINE                                                 COMM01.72
                                 EXT       CF$WRITE                                                COMM01.73
```

```
EXT       CP$CLEAR                                          COMM01.74
EXT       DESC$BAB                                          COMM01.75
EXT       DESC$CNC                                          COMM01.76
EXT       DESC$DKE                                          COMM01.77
EXT       DESC$DKS                                          COMM01.78
EXT       DESC$DSP                                          COMM01.79
EXT       DESC$LPA                                          COMM01.80
EXT       DESC$MON                                          COMM01.81
EXT       DESC$RST                                          COMM01.82
EXT       DESC$QUE                                          COMM01.83
EXT     DESC$STS                                            COMM01.84
EXT       DESC$SUM                                          COMM01.85
EXT       DESC$TJO                                          COMM01.86
EXT       DESC$HEL                                          COMM01.87
EXT       DESC$XJS                                          COMM01.88
EXT       DP$BABEL                                          COMM01.89
EXT       DP$DEBUG                                          COMM01.90
EXT       DP$CNC                                            COMM01.91
EXT       DP$DKERR                                          COMM01.92
EXT       DP$DKSTA                                          COMM01.93
EXT       DP$DSPL                                           COMM01.94
EXT       DP$NSCLP                                          COMM01.95
EXT       DP$ERROR                                          COMM01.96
EXT       DP$LINK                                           COMM01.97
EXT       DP$MONIT                                          COMM01.98
EXT       DP$RSTAT                                          COMM01.99
EXT       DP$SPLIT                                          COMM01.100
EXT       DP$STATC                                          COMM01.101
EXT       DP$STATI                                          COMM01.102
EXT       DP$STATU                                          COMM01.103
EXT       DP$STORA                                          COMM01.104
EXT       DP$STP                                            COMM01.105
EXT     DP$STRS                                             COMM01.106
EXT       DP$SUMMA                                          COMM01.107
EXT       DP$SWAP                                           COMM01.108
EXT       DP$TAPE                                           COMM01.109
EXT       DP$TJOB                                           COMM01.110
EXT       DP$HELP                                           COMM01.111
EXT       DP$XJST                                           COMM01.112
EXT       DV$BABEL                                          COMM01.113
EXT       DV$CNC                                            COMM01.114
EXT       DV$DEBUG                                          COMM01.115
```

```
                    EXT       DV$DKERR                                          COMM01.116
                    EXT       DV$DKSTA                                          COMM01.117
                    EXT       DV$DSPL                                           COMM01.118
                    EXT       DV$NSCLP                                          COMM01.119
                    EXT       DV$ERROR                                          COMM01.120
                    EXT       DV$LINK                                           COMM01.121
                    EXT       DV$MONIT                                          COMM01.122
                    EXT       DV$RSTAT                                          COMM01.123
                    EXT       DV$SPLIT                                          COMM01.124
                    EXT       DV$STATC                                          COMM01.125
                    EXT       DV$STATI                                          COMM01.126
                    EXT       DV$STATU                                          COMM01.127
                    EXT       DV$STORA                                          COMM01.128
                    EXT       DV$STP                                            COMM01.129
                    EXT     DV$STRS                                             COMM01.130
                    EXT       DV$SUMMA                                          COMM01.131
                    EXT       DV$SWAP                                           COMM01.132
                    EXT       DV$TAPE                                           COMM01.133
                    EXT       DV$TJOB                                           COMM01.134
                    EXT       DV$HELP                                           COMM01.135
                    EXT       DV$XJST                                           COMM01.136
                    EXT       EC$BDPAR                                          COMM01.137
                    EXT       EC$NOHLP                                          COMM01.138
3       VA$LGH      EQUALS    3               .Offset to parameter 2            COMM01.139
```

```
                                 *        Lock the display interface; process the command; release the     COMM01.141
                                 *        interface; return to the caller.                                 COMM01.142
                                                                                                           COMM01.143
     6   010000  024445                   R!PARAM2 = 0                                                     COMM01.144
    10    <macro>                         ADDRESS   R!MASTER,$L@DIS,R!%CLI                                 COMM01.145
    13    <macro>                         GET       R!LINE,$C@DIS,R!%CLI . First display line.             COMM01.146
    20    <macro>                         GET       R!NUMBER,$C@CNT,R!%CLI . Display area size.            COMM01.147
    25    <macro>                         $GOTO     (ERASE,DEBUG,ERR,LINK,ERASE,STATION,STATUS,STORAGE,SE  COMM01.148
                                 ,LECT,MONITOR,BABEL,STATC,TAPE,TJOB,DKSTAT,SUMMARY,STP,XJSTAT,STRS,HELP,  COMM01.149
                                 ,RSTAT,CONC,DSPL,LPATH,SWAP,DKERR),R!PAR                                  COMM01.150
                                                                                                           COMM01.151
    74                           RELEASE  *                                                               COMM01.152
    74    <macro>                         UNLOCK    R!MASTER            . Release the display interface.   COMM01.153
                                                                                                           COMM01.154
   113                           DONE     *                                                               COMM01.155
   113    <macro>                         RETURN                                                           COMM01.156
```

```
                                 ***************************************************************************  COMM01.158
                                 *                                                                         *  COMM01.159
                                 *        CLEAR command.                                                   *  COMM01.160
                                 *        SCROLL command.                                                  *  COMM01.161
                                 *                                                                         *  COMM01.162
                                 ***************************************************************************  COMM01.163


121                              ERASE    *                                                                  COMM01.165
                                                                                                             COMM01.166
                                 *        Establish the new scroll line number.                              COMM01.167
                                                                                                             COMM01.168
121     <macro>                          LOCK      R!MASTER              . Lock the interface.              COMM01.169
150     <macro>                          STORE     DIS$NONE,$D@TYP,R!%CLI . No displays active.             COMM01.170
155    026437                            R!NUMBER = R!NUMBER+1           . Number of lines to clear.        COMM01.171
156    020436  024440                    R!SCROLL = R!LINE               . Line for full screen scroll.     COMM01.172
160    020430  017000 /000000            R!SCROLL = R!SCROLL+R!NUMBER, R!PAR = CP$CLEAR                     COMM01.173
       103004  020440  022437
       024440
167    077000 /004412                    R = SCREEN                      . Set screen parameters.           COMM01.174
171    071075                            P = RELEASE                                                        COMM01.175
                                 ***************************************************************************  COMM01.176
                                 *                                                                         *  COMM01.177
                                 *          LPATH  chord  logical path                                     *  COMM01.178
                                 *                                                                         *  COMM01.179
                                 ***************************************************************************  COMM01.180
                                                                                                             COMM01.181
172                              LPATH    *                                                                  COMM01.182
                                                                                                             COMM01.183
172     <macro>                          GETSTACK  PAR$COM,R!LIST        .Get memory                        COMM01.184
214     <macro>                          CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$LPA,R!LIST,RO=    COMM01.185
                                 ,R!OFFSET,RO=R!COUNT)                                                       COMM01.186
                                                                                                             COMM01.187
245     <macro>                          $IF (A # 0)                     .If an error occurred              COMM01.188
                                                                                          .>>>>> 0                $IF       1
247     <macro>                            FRESTACK                                                         COMM01.189
                                                                                          .>>>>> 1                $IF       3
                                                                                          .<<<<< 1                $ENDIF    3
265    071152                              P = DONE                                                         COMM01.190
266     <macro>                          $ENDIF                                                             COMM01.191
                                                                                          .<<<<< 0                $ENDIF    1
                                                                                                             COMM01.192
```

```
266    <macro>                    GET R!%W1,VA@TYP,R!LIST                                          COMM01.193
271    <macro>                    $IF (R!%W1 # VAL$NONE)          .Check for parameter 1           COMM01.194
                                                                                .>>>>> 0                $IF      1
275    <macro>                      GET R!PARAM2,VA@PR2,R!LIST                                     COMM01.195
302    <macro>                      PUT R!PARAM2,$D@P2,R!%CLI     .Save the Channel Number-input   COMM01.196
307    010003  025447               R!LIST = R!LIST + VA$LGH      .Point to next list entry        COMM01.197
311    <macro>                      GET R!%W2,VA@PR2,R!LIST                                        COMM01.198
316    <macro>                      PUT R!%W2,$D@P3,R!%CLI        .Save the Logial Path number     COMM01.199
323    <macro>                    $ENDIF                                                           COMM01.200
                                                                                .<<<<< 0                $ENDIF   1
                                                                                                     COMM01.201
                                                                                                     COMM01.202
323    <macro>                    FRESTACK                                                         COMM01.203
                                                                                .>>>>> 0                $IF      3
                                                                                .<<<<< 0                $ENDIF   3
                                                                                                     COMM01.204
341    014000 /000000  024443     R!OLAY = DV$NSCLP                                                COMM01.205
344    014000 /000000  024444     R!PARAM = DP$NSCLP                                               COMM01.206
347    075000 /001566             P = NEW                                                          COMM01.207
                                                                                                     COMM01.208

                          ***********************************************************************  COMM01.210
                          *                                                                     *  COMM01.211
                          *        CONC command                                                 *  COMM01.212
                          *                                                                     *  COMM01.213
                          ***********************************************************************  COMM01.214
                                                                                                     COMM01.215
351                       CONC     *                                                               COMM01.216
                                                                                                     COMM01.217
351    <macro>                    GETSTACK  PAR$COM,R!LIST                                         COMM01.218
373    <macro>                    CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$CNC,R!LIST,RO=  COMM01.219
                          ,R!OFFSET,RO=R!COUNT)                                                      COMM01.220
                                                                                                     COMM01.221
424    <macro>                    $IF (A # 0)                .if an error encountered              COMM01.222
                                                                                .>>>>> 0                $IF      1
426    <macro>                      FRESTACK                                                       COMM01.223
                                                                                .>>>>> 1                $IF      3
                                                                                .<<<<< 1                $ENDIF   3
444    071331                       P = DONE                                                       COMM01.224
445    <macro>                    $ENDIF                                                           COMM01.225
                                                                                .<<<<< 0                $ENDIF   1
```

```
                                                                                                COMM01.226
445    <macro>                    GET  R!%W1,VA@TYP,R!LIST  .see if anything entered           COMM01.227
450    <macro>                    $IF (A # VAL$NONE)                                           COMM01.228
                                                                          .>>>>> 0                $IF      1
453    <macro>                      GET  R!%W2,VA@PR2,R!LIST  .get the entry                   COMM01.229
460    <macro>                    $ELSE                                                        COMM01.230
461   010003  024424                R!%W2 = I@COCO          .first concentrator ordinal          COMM01.231
463    <macro>                    $ENDIF                                                       COMM01.232
                                                                          .<<<<< 0                $ENDIF   1
                                                                                                COMM01.233
463    <macro>                    FRESTACK                                                     COMM01.234
                                                                          .>>>>> 0                $IF      3
                                                                          .<<<<< 0                $ENDIF   3
                                                                                                COMM01.235
501    <macro>                    $IF (R!%W2 <= I@NCOR)     .a basic concentrator              COMM01.236
                                                                          .>>>>> 0                $IF      1
507    <macro>                      FNDFEI   R!PARAM2,ORD=R!%W2  .locate the table entry       COMM01.237
                                                                          .>>>>> 1                $IF      2
                                                                          .<<<<< 1                $ENDIF   2
543    <macro>                      $IF (R!PARAM2 = 0)      .not found                         COMM01.238
                                                                          .>>>>> 1                $IF      1
546   075000 /002125                  P = ERROR                                                 COMM01.239
550    <macro>                      $ELSE                                                      COMM01.240
551    <macro>                        GET  R!%W1,FEI@CT,R!PARAM2  .check the channel           COMM01.241
554   020423  013036  127000          P = ERROR, R!%W1 # CH$CN                                  COMM01.242
     /002125
560    <macro>                      $ENDIF                                                     COMM01.243
                                                                          .<<<<< 1                $ENDIF   1
560    <macro>                    $ELSE                                                        COMM01.244
561   020424  013014  005001        R!%W1 = R!%W2 - I@SNID < 1  .base NSC ordinal * 2          COMM01.245
      024423
565    <macro>                      GET  R!PARAM2,FEI@ON,R!%FEI                                COMM01.246
572   020445  022423  024445        R!PARAM2 = R!PARAM2 + R!%W1  .COR entry                    COMM01.247
575    <macro>                    $ENDIF                                                       COMM01.248
                                                                          .<<<<< 0                $ENDIF   1
                                                                                                COMM01.249
575    <macro>                    PUT       R!%W2,$D@P3,R!%CLI  .save the ordinal              COMM01.250
602   014000 /000000  024443      R!OLAY = DV$CNC                                              COMM01.251
605   014000 /000000  024444      R!PARAM = DP$CNC                                             COMM01.252
610   075000 /001566              P = NEW                                                      COMM01.253
```

```
                                 ********************************************************************  COMM01.255
                                 *                                                                  *  COMM01.256
                                 *        STREAM STATUS COMMAND (STRSTAT)                           *  COMM01.257
                                 *                                                                  *  COMM01.258
                                 ********************************************************************  COMM01.259
                                                                                                       COMM01.260
612                              STRS     *                                                           COMM01.261
                                                                                                       COMM01.262
                                          * CHECK FOR ID PASSED WITH COMMAND                          COMM01.263
                                                                                                       COMM01.264
612     <macro>                           GETSTACK PAR$COM,R!LIST        .ALLOCATE BUFFER            COMM01.265
634     <macro>                           CALL DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$STS,R!LIST,RO=R!OFF  COMM01.266
                                 ,SET,RO=R!COUNT)                                                      COMM01.267
                                                                                                       COMM01.268
665     <macro>                           $IF (A # 0)                    .IF ERROR OCCURRED          COMM01.269
                                                                                          .>>>>> 0          $IF      1
667     <macro>                             FRESTACK                     .RELEASE BUFFER             COMM01.270
                                                                                          .>>>>> 1          $IF      3
                                                                                          .<<<<< 1          $ENDIF   3
705     071572                              P = DONE                     .TERMINATE                  COMM01.271
706     <macro>                           $ELSE                                                       COMM01.272
707     <macro>                             GET R!%W1,VA@TYP,R!LIST      .VALIDATE TYPE              COMM01.273
712     <macro>                             $IF (A = VAL$ID)             .IF ID                      COMM01.274
                                                                                          .>>>>> 1          $IF      1
714     <macro>                               GET R!PARAM2,VA@PR1,R!LIST .USE FOR DISPLAY            COMM01.275
721     <macro>                             $ELSE                                                     COMM01.276
722     010000  024445                        R!PARAM2 = 0               .SPECIFY ALL IDS            COMM01.277
724     <macro>                             $ENDIF                                                    COMM01.278
                                                                                          .<<<<< 1          $ENDIF   1
724     <macro>                           $ENDIF                                                      COMM01.279
                                                                                          .<<<<< 0          $ENDIF   1
                                                                                                       COMM01.280
724     <macro>                           FRESTACK                       .RELEASE MEMORY             COMM01.281
                                                                                          .>>>>> 0          $IF      3
                                                                                          .<<<<< 0          $ENDIF   3
742     014000 /000000  024443            R!OLAY = DV$STRS               .OVERLAY                    COMM01.282
745     014000 /000000  024444            R!PARAM = DP$STRS              .PARAMETER                  COMM01.283
750     075000 /001566                    P = NEW                        .ACTIVATE DISPLAY           COMM01.284
```

```
                                                                                  COMM01.287
                                                                                  COMM01.288
                                                                                  COMM01.289
                                                                                  COMM01.290
                                                                                  COMM01.291
                                                                                  COMM01.292
                                                                                  COMM01.293
                                                                                  COMM01.294
                                                                                  COMM01.295

                                                                                  COMM01.296
                                                                                  [illegible]
                                                                                  [illegible]
                                                                                  [illegible]
  2    014000 /000000  024443          R!OLAY = DV$DEBUG                        COMM01.301
755    014000 /000000  024444          R!PARAM = DP$DEBUG                       COMM01.302
760    075000 /001566                  P = NEW                                  COMM01.303

762                            ERR     *                                        COMM01.305
762    014000 /000000  024443          R!OLAY = DV$ERROR                        COMM01.306
765    014000 /000000  024444          R!PARAM = DP$ERROR                       COMM01.307
770    075000 /001566                  P = NEW                                  COMM01.308

772                            LINK    *                                        COMM01.310
772    014000 /000000  024443          R!OLAY = DV$LINK                         COMM01.311
775    014000 /000000  024444          R!PARAM = DP$LINK                        COMM01.312
1000   075000 /001566                  P = NEW                                  COMM01.313

1002                           STATION *                                        COMM01.315
1002   014000 /000000  024443          R!OLAY = DV$STATI                        COMM01.316
1005   014000 /000000  024444          R!PARAM = DP$STATI                       COMM01.317
1010   075000 /001566                  P = NEW                                  COMM01.318

1012                           STATC   *                                        COMM01.320
1012   [illegible]                     R!OLAY = DV$STATC                        COMM01.321
[illegible]                                                                     [illegible]
```

```
1020                                STORAGE  *                                                          COMM01.325
1020   014000 /000000  024443               R!OLAY = DV$STORA                                          COMM01.326
1023   014000 /000000  024444               R!PARAM = DP$STORA                                         COMM01.327
1026   070540                               P = NEW                                                    COMM01.328

1027                                STIP     *                                                          COMM01.330
1027   014000 /000000  024443               R!OLAY = DV$STP                                            COMM01.331
1032   014000 /000000  024444               R!PARAM = DP$STP                                           COMM01.332
1035   070531                               P = NEW                                                    COMM01.333
                                                                                                             .
1036                                BABEL    *                                                          COMM01.335
1036   014000 /000000  024443               R!OLAY = DV$BABEL                                          COMM01.336
1041   014000 /000000  024444               R!PARAM = DP$BABEL                                         COMM01.337
1044    <macro>                             GETSTACK  PAR$COM,R!LIST                                   COMM01.338
1066    <macro>                             CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$BAB,R!LIST,RO=  COMM01.339
                                    ,R!OFFSET,RO=R!COUNT)                                               COMM01.340
1117   024423                               R!%W1 = A                                                  COMM01.341
1120    <macro>                             GET       R!PARAM2,VA@PR1,R!LIST  . Get display parameter.  COMM01.342
1125    <macro>                             GET       R!%W2,VA@TYP,R!LIST                              COMM01.343
1130   020424  013014  103004               R!PARAM2 = 'RR'R, R!%W2 = VAL$NONE  . Use default.         COMM01.344
       014000 /051122  024445
1136    <macro>                             FRESTACK                                                   COMM01.345
                                                                                  .>>>>> 0                $IF      3
                                                                                  .<<<<< 0                $ENDIF   3
1154   020423  127000 /000113               P = DONE, R!%W1 # 0          . If error.                   COMM01.346
1157   070407                               P = NEW                                                    COMM01.347

1160                                MONITOR  *                                                          COMM01.349
1160   014000 /000000  024443               R!OLAY = DV$MONIT                                          COMM01.350
1163   014000 /000000  024444               R!PARAM = DP$MONIT                                         COMM01.351
1166    <macro>                             GETSTACK  PAR$COM,R!LIST                                   COMM01.352
1210    <macro>                             CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$MON,R!LIST,RO=  COMM01.353
                                    ,R!OFFSET,RO=R!COUNT)                                               COMM01.354
1241   024423                               R!%W1 = A                                                  COMM01.355
1242    <macro>                             GET       R!PARAM2,VA@PR1,R!LIST  . Get display parameter.  COMM01.356
1247    <macro>                             FRESTACK                                                   COMM01.357
                                                                                  .>>>>> 0                $IF      3
                                                                                  .<<<<< 0                $ENDIF   3
1265   020423  127000 /000113               P = DONE, R!%W1 # 0          . If error.                   COMM01.358
```

```
1270   <macro>                GET       A,$D@MON,R!%CLI                                   COMM01.359
                                                                                          COMM01.360
1274   <macro>                $IF       (A = 0)                                           COMM01.361
                                                                          .>>>>> 0              $IF      1
1276   <macro>                  GETMEM  0'140,R!LIST                                      COMM01.362
1316   <macro>                  $IF     (A # 0)              .If buffer not available     COMM01.363
                                                                          .>>>>> 1              $IF      1
1320   024423                     R!%W1 = A                                               COMM01.364
1321   <macro>                    CALL  ERROR,(R!%W1,0,0,0)                               COMM01.365
1343   075000 /000113             P = DONE                                                COMM01.366
1345   <macro>                  $ENDIF                                                    COMM01.367
                                                                          .<<<<< 1              $ENDIF   1
1345   <macro>                  PUT     R!LIST,$D@MON,R!%CLI                              COMM01.368
1352   <macro>                $ENDIF                                                      COMM01.369
                                                                          .<<<<< 0              $ENDIF   1
                                                                                          COMM01.370
1352   070214                 P = NEW                                                    COMM01.371
```

```
                                 ********************************************************************  COMM01.373
                                 *                                                                  *  COMM01.374
                                 *        STATUS command.                                           *  COMM01.375
                                 *                                                                  *  COMM01.376
                                 ********************************************************************  COMM01.377


                                 *        Scan the command line for queue designators.                COMM01.379
                                                                                                      COMM01.380
1353                             STATUS   *                                                           COMM01.381
1353   010000  024446                     R!QUEUES = 0                                                COMM01.382
1355    <macro>                           STORE     0,$D@SCI,R!%CLI      . Clear the scroll interval. COMM01.383
1362    <macro>                           GETSTACK  PAR$COM,R!LIST                                   COMM01.384
                                                                                                      COMM01.385
1404                             WHICH    *                                                           COMM01.386
1404    <macro>                           CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$QUE,R!LIST,RO=  COMM01.387
                                 ,R!OFFSET,RO=R!COUNT)                                                COMM01.388
                                                                                                      COMM01.389
1435    <macro>                           $IF       (A # 0)              . If error encountered.     COMM01.390
                                                                                          .>>>>> 0        $IF      1
1437    <macro>                             FRESTACK                     . Release buffer space.     COMM01.391
                                                                                          .>>>>> 1        $IF      3
                                                                                          .<<<<< 1        $ENDIF   3
1455   075000 /000113                       P = DONE                                                   COMM01.392
1457    <macro>                           $ENDIF                                                     COMM01.393
                                                                                          .<<<<< 0        $ENDIF   1
                                                                                                      COMM01.394
1457    <macro>                           GET       R!%W1,VA@TYP,R!LIST . Get the entry type.          COMM01.395
                                                                                                      COMM01.396
1462    <macro>                           $IF       (R!%W1 = VAL$LIT)   . If a queue specified.        COMM01.397
                                                                                          .>>>>> 0        $IF      1
1466    <macro>                             GET     R!%W1,VA@PR1,R!LIST . Get the queue flag and include  COMM01.398
1473    <macro>                             OR      R!QUEUES,R!%W1,R!QUEUES . it with the others.      COMM01.399
1501   071075                               P = WHICH                    . Check for more designators. COMM01.400
1502    <macro>                           $ELSEIF (R!%W1 = VAL$DEC)                                  COMM01.401
1507    <macro>                             GET       R!%W2,VA@PR2,R!LIST . Get the scroll interval.   COMM01.402
1514   020424  005001  024425               R!%W3 = R!%W2 < 1            . Convert to tenths of a second.  COMM01.403
1517   020424  005003  022425               R!%W2 = R!%W2 < 3 + R!%W3                                  COMM01.404
       024424
1523    <macro>                             PUT       R!%W2,$D@SCI,R!%CLI . Store the scroll interval. COMM01.405
1530    <macro>                             PUT       R!%W2,$D@SCT,R!%CLI . Store the scroll timer.    COMM01.406
1535   071131                               P = WHICH                    . Check for more designators. COMM01.407
```

```
15.     <macro>                         $ENDIF                                                          COMM01.408
                                                                                  .<<<<< 0                $ENDIF    1
                                                                                                          COMM01.409
1536    <macro>                         FRESTACK                                                        COMM01.410
                                                                                  .>>>>> 0                $IF       3
                                                                                  .<<<<< 0                $ENDIF    3
1554    020446  103003  010073          R!QUEUES = QF$ALL, R!QUEUES = 0 . Use default if none entered.  COMM01.411
        024446
1560    014000 /000000  024443          R!OLAY = DV$STATU                                               COMM01.412
1563    014000 /000000  024444          R!PARAM = DP$STATU                                              COMM01.413


                        *               Initiate the new display.                                       COMM01.415
                                                                                                        COMM01.416
1566                    NEW             *                                                               COMM01.417
1566    <macro>                         LOCK      R!MASTER                . Lock the interface.         COMM01.418
1615    <macro>                         PUT       R!QUEUES,$D@QUE,R!%CLI . Save status queue flags.     COMM01.419
1622    <macro>                         STORE     D'30000,$D@FRM,R!%CLI . Initialize frame number.      COMM01.420
1630    010001  024423                  R!%W1 = DIS$OPER                . Operator display indicator.   COMM01.421
                                                                                                        COMM01.422
1632                    SAVE            *                                                               COMM01.423
1632    <macro>                         PUT       R!OLAY,$D@OVL,R!%CLI . Set display overlay number.    COMM01.424
1637    <macro>                         PUT       R!PARAM,$D@PAR,R!%CLI . Set new display parameter.    COMM01.425
1644    <macro>                         PUT       R!PARAM2,$D@P2,R!%CLI                               COMM01.426
1651    <macro>                         PUT       R!%W1,$D@TYP,R!%CLI . Set display type.               COMM01.427
1656    <macro>                         ADDRESS   R!%W1,$R@DIS,R!%CLI                                 COMM01.428
1661    <macro>                         SIGNAL    R!%W1,1                 . Activate task for refresh.  COMM01.429
1702    020436  022437  024436          R!LINE = R!LINE+R!NUMBER        . Line to clear.              COMM01.430
1705    010001  024437                  R!NUMBER = 1                    . Clear line after display.   COMM01.431
1707    020436  012001  024440          R!SCROLL = R!LINE+1             . Set scroll line number.     COMM01.432
1712    077000 /004412                  R = SCREEN                      . Set screen parameters.      COMM01.433
1714    075000 /000074                  P = RELEASE                                                   COMM01.434
```

```
                                 ********************************************************************  COMM01.436
                                 *                                                                  *  COMM01.437
                                 *        Debug display selection.                                  *  COMM01.438
                                 *                                                                  *  COMM01.439
                                 ********************************************************************  COMM01.440
                                                                                                       COMM01.441
1716                             SELECT   *                                                            COMM01.442
                                                                                                       COMM01.443
                                 *        Validate the display characters and translate them to     COMM01.444
                                 *        descriptor addresses.                                     COMM01.445
                                                                                                       COMM01.446
1716   020433  013002  100004            P = ERROR, R!CMDLEN > 2        . If more than two selections.  COMM01.447
       102003  075000 /002125
1724    <macro>                          GETBYTE   R!%W1,R!CMDOFF,R!BASE . First display designator.  COMM01.448
                                                                                                       COMM01.449
1746    <macro>                          $IF       (R!%W1 < 'A'R),OR,(R!%W1 > 'Z'R)                   COMM01.450
                                                                                  .>>>>> 0             $IF      1
1757   075000 /002125                      P = ERROR                    . If illegal display character. COMM01.451
1761    <macro>                          $ENDIF                                                      COMM01.452
                                                                                  .<<<<< 0             $ENDIF   1
                                                                                                       COMM01.453
1761   020423  013101  005002            R!%W1 = R!%W1-'A'R<2           . Form descriptor offset.       COMM01.454
       024423
1765    <macro>                          ADDRESS   R!LEFT,$D@DEB,R!%CLI                              COMM01.455
1770   020450  022423  024450            R!LEFT = R!LEFT+R!%W1          . Get descriptor address.       COMM01.456
1773   010000  024451                    R!RIGHT = 0                                                 COMM01.457
                                                                                                       COMM01.458
1775    <macro>                          $IF       (R!CMDLEN = 2)       . If two displays selected.     COMM01.459
                                                                                  .>>>>> 0             $IF      1
2002   020432  012001  024424              R!%W2 = R!CMDOFF+1                                        COMM01.460
2005    <macro>                          GETBYTE R!%W1,R!%W2,R!BASE   . Get the second character.    COMM01.461
                                                                                                       COMM01.462
2027    <macro>                          $IF       (R!%W1 < 'A'R),OR,(R!%W1 > 'Z'R)                   COMM01.463
                                                                                  .>>>>> 1             $IF      1
2040   070065                              P = ERROR                    . If illegal display character. COMM01.464
2041    <macro>                          $ENDIF                                                      COMM01.465
                                                                                  .<<<<< 1             $ENDIF   1
                                                                                                       COMM01.466
2041   020423  013101  005002              R!%W1 = R!%W1-'A'R<2         . Form descriptor offset.       COMM01.467
       024423
2045    <macro>                          ADDRESS R!RIGHT,$D@DEB,R!%CLI                               COMM01.468
2050   020451  022423  024451            R!RIGHT = R!RIGHT+R!%W1        . Get descriptor address.       COMM01.469
```

```
20.     <macro>                         $ENDIF                                                          COMM01.470
                                                                                .<<<<< 0                $ENDIF   1


                                *       Save the display descriptor parameters.                         COMM01.472
                                                                                                        COMM01.473
2053    <macro>                         LOCK      R!MASTER              . Lock the interface.           COMM01.474
2102    <macro>                         PUT       R!LEFT,$D@LFT,R!%CLI . Save the descriptor addresses. COMM01.475
2107    <macro>                         PUT       R!RIGHT,$D@RGT,R!%CLI                               COMM01.476
2114    014000 /000000  024443          R!OLAY = DV$SPLIT               . Get debug display parameters. COMM01.477
2117    014000 /000000  024444          R!PARAM = DP$SPLIT                                              COMM01.478
2122    010002  024423                  R!%W1 = DIS$DEB                                                 COMM01.479
2124    071272                          P = SAVE                                                        COMM01.480


                                *       The display designator is invalid.  Output an error message.    COMM01.482
                                                                                                        COMM01.483
2125                            ERROR   *                                                               COMM01.484
2125    <macro>                         CALL      ERROR,(EC$BDPAR,R!BASE,R!CMDOFF,R!CMDLEN)             COMM01.485
2150    075000 /000113                  P = DONE                                                        COMM01.486
```

```
                                 ********************************************************************  COMM01.488
                                 *                                                                  *  COMM01.489
                                 *        TAPE command                                              *  COMM01.490
                                 *                                                                  *  COMM01.491
                                 ********************************************************************  COMM01.492
                                                                                                       COMM01.493
                                                                                                       COMM01.494
2152                             TAPE     *                                                            COMM01.495
                                                                                                       COMM01.496
2152   014000 /000000  024443            R!OLAY  = DV$TAPE                                            COMM01.497
2155   014000 /000000  024444            R!PARAM = DP$TAPE                                            COMM01.498
2160   071372                            P = NEW                                                      COMM01.499


                                 ********************************************************************  COMM01.501
                                 *                                                                  *  COMM01.502
                                 *        TJOB command                                              *  COMM01.503
                                 *                                                                  *  COMM01.504
                                 ********************************************************************  COMM01.505
                                                                                                       COMM01.506
2161                             TJOB     *                                                            COMM01.507
                                                                                                       COMM01.508
2161    <macro>                          STORE     0,$D@SCI,R!%CLI     . Clear the scroll interval.   COMM01.509
2166    <macro>                          GETSTACK  PAR$COM,R!LIST                                     COMM01.510
                                                                                                       COMM01.511
2210    <macro>                          CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$TJO,R!LIST,RO=  COMM01.512
                                 ,R!OFFSET,RO=R!COUNT)                                                 COMM01.513
                                                                                                       COMM01.514
2241    <macro>                          $IF       (A # 0)             . If error encountered.        COMM01.515
                                                                                        .>>>>> 0          $IF      1
2243    <macro>                            FRESTACK                    . Release buffer space.        COMM01.516
                                                                                        .>>>>> 1          $IF      3
                                                                                        .<<<<< 1          $ENDIF   3
2261   075000 /000113                      P = DONE                                                   COMM01.517
2263    <macro>                          $ENDIF                                                       COMM01.518
                                                                                        .<<<<< 0          $ENDIF   1
                                                                                                       COMM01.519
2263    <macro>                          GET       R!%W1,VA@TYP,R!LIST . Get the entry type.          COMM01.520
                                                                                                       COMM01.521
2266    <macro>                          $IF       (R!%W1 = VAL$DEC)                                  COMM01.522
                                                                                        .>>>>> 0          $IF      1
```

```
2272   <macro>                          GET       R!%W2,VA@PR2,R!LIST . Get the scroll interval.      COMM01.523
2277   020424  005001  024425           R!%W3 = R!%W2 < 1             . Convert to tenths of a second.  COMM01.524
2302   020424  005003  022425           R!%W2 = R!%W2 < 3 + R!%W3                                     COMM01.525
       024424
                                                                                                       COMM01.526
2306   <macro>                          PUT       R!%W2,$D@SCI,R!%CLI . Store the scroll interval.    COMM01.527
2313   <macro>                          PUT       R!%W2,$D@SCT,R!%CLI . Store the scroll timer.       COMM01.528
2320   <macro>                        $ENDIF                                                          COMM01.529
                                                                                   .<<<<< 0                $ENDIF   1
                                                                                                       COMM01.530
2320   <macro>                        FRESTACK                                                        COMM01.531
                                                                                   .>>>>> 0                $IF      3
                                                                                   .<<<<< 0                $ENDIF   3
2336   014000 /000000  024443         R!OLAY  = DV$TJOB                                               COMM01.532
2341   014000 /000000  024444         R!PARAM = DP$TJOB                                               COMM01.533
2344   071556                         P = NEW                                                         COMM01.534


                                ***********************************************************************COMM01.536
                                *                                                                     *COMM01.537
                                *       DKERR command.                                                *COMM01.538
                                *                                                                     *COMM01.539
                                ***********************************************************************COMM01.540
                                                                                                       COMM01.541
2345                            DKERR  *                                                               COMM01.542
2345   <macro>                         GETSTACK  PAR$COM,R!LIST      .Get buffer for parameter list   COMM01.543
2367   <macro>                         CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$DKE,R!LIST,RO=COMM01.544
                                ,R!OFFSET,RO=R!COUNT)                                                  COMM01.545
2420   <macro>                         $IF (A # 0)                   .If error encountered            COMM01.546
                                                                                   .>>>>> 0                $IF      1
2422   <macro>                           FRESTACK                    .Release buffer                  COMM01.547
                                                                                   .>>>>> 1                $IF      3
                                                                                   .<<<<< 1                $ENDIF   3
2440   075000 /000113                    P = DONE                                                     COMM01.548
2442   <macro>                         $ENDIF                                                          COMM01.549
                                                                                   .<<<<< 0                $ENDIF   1
2442   <macro>                         GET R!%W1,VA@TYP,R!LIST       .Get the entry type              COMM01.550
2445   <macro>                         GET R!PARAM2,VA@PR2,R!LIST    .Get the parameter:  IOP#        COMM01.551
2452   020423  013014  103003          R!PARAM2 = 1, R!%W1 = VAL$NONE .Default is IOP-1               COMM01.552
       010001  024445
2457   <macro>                         FRESTACK                                                        COMM01.553
```

```
                                                                                .>>>>> 0          $IF      3
                                                                                .<<<<< 0          $ENDIF   3
2475   014000 /000000  024443          R!OLAY = DV$DKERR                                     COMM01.554
2500   014000 /000000  024444          R!PARAM = DP$DKERR                                    COMM01.555
2503   071715                          P = NEW                                               COMM01.556


                                 ********************************************************** COMM01.558
                                 *                                                        * COMM01.559
                                 *        DISK command                                    * COMM01.560
                                 *                                                        * COMM01.561
                                 ********************************************************** COMM01.562
                                                                                            COMM01.563
2504                             DKSTAT   *                                                 COMM01.564
2504    <macro>                          GETSTACK  PAR$COM,R!LIST      .Get buffer for parameter list  COMM01.565
2526    <macro>                          CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$DKS,R!LIST,RO= COMM01.566
                                 ,R!OFFSET,RO=R!COUNT)                                      COMM01.567
2557    <macro>                          $IF       (A # 0)             .If error encountered - COMM01.568
                                                                                .>>>>> 0          $IF      1
2561    <macro>                            FRESTACK                    .Release buffer       COMM01.569
                                                                                .>>>>> 1          $IF      3
                                                                                .<<<<< 1          $ENDIF   3
2577   075000 /000113                     P = DONE                                          COMM01.570
2601    <macro>                          $ENDIF                                             COMM01.571
                                                                                .<<<<< 0          $ENDIF   1
2601    <macro>                          GET       R!%W1,VA@TYP,R!LIST  .Get the entry type COMM01.572
2604    <macro>                          GET       R!PARAM2,VA@PR2,R!LIST  .Get the parameter: IOP# COMM01.573
2611   020423  013014  103003           R!PARAM2 = 1, R!%W1 = VAL$NONE  .Default is IOP-1 COMM01.574
       010001  024445
2616    <macro>                          FRESTACK                                           COMM01.575
                                                                                .>>>>> 0          $IF      3
                                                                                .<<<<< 0          $ENDIF   3
2634   014000 /000000  024443           R!OLAY = DV$DKSTA                                     COMM01.576
2637   014000 /000000  024444           R!PARAM = DP$DKSTA                                    COMM01.577
2642   075000 /001566                   P = NEW                                               COMM01.578
                                                                                            COMM01.579


                                 ********************************************************** COMM01.581
```

```
                                *                                                       *   COMM01.582
                                *       DSPL COMMAND                                    *   COMM01.583
                                *       Format: DSPL,displayname                        *   COMM01.584
                                *                                                       *   COMM01.585
                                ***********************************************************   COMM01.586


2644                            DSPL    *                                                   COMM01.588
                                                                                            COMM01.589
                                *       Get the display name from the command line and build an   COMM01.590
                                *       task display request segment.                       COMM01.591
                                                                                            COMM01.592
2644    <macro>                         GETSTACK  PAR$COM,R!LIST                            COMM01.593
                                                                                            COMM01.594
2666    <macro>                         CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$DSP,R!LIST,RO=   COMM01.595
                                ,R!OFFSET,RO=R!COUNT)                                       COMM01.596
                                                                                            COMM01.597
2717    <macro>                         $IF       (A # 0)                 . Error encountered...   COMM01.598
                                                                                  .>>>>> 0        $IF      1
2721    <macro>                           FRESTACK                        . Release buffer    COMM01.599
                                                                                  .>>>>> 1        $IF      3
                                                                                  .<<<<< 1        $ENDIF   3
2737    075000 /000113                    P = DONE                        . And return        COMM01.600
2741    <macro>                         $ENDIF                                                COMM01.601
                                                                                  .<<<<< 0        $ENDIF   1
                                                                                                COMM01.602
2741    <macro>                         GET       R!PARAM,VA@PR1,R!LIST   . Display name string start   COMM01.603
2746    <macro>                         GET       R!PARAM2,VA@PR2,R!LIST  . Length of string  COMM01.604
                                                                                            COMM01.605
2753    <macro>                         $IF       (R!PARAM2 # 2)          . Only 2 characters allowed   COMM01.606
                                                                                  .>>>>> 0        $IF      1
2757    <macro>                           FRESTACK                                          COMM01.607
                                                                                  .>>>>> 1        $IF      3
                                                                                  .<<<<< 1        $ENDIF   3
2775    075000 /000113                    P = DONE                                          COMM01.608
2777    <macro>                         $ENDIF                                                COMM01.609
                                                                                  .<<<<< 0        $ENDIF   1
                                                                                                COMM01.610
                                *       Move the display name into the parameter area of the CLI table   COMM01.611
                                                                                            COMM01.612
2777    010030  005001  024423          R!%W1 = $D@P3@P < 1               . Destination byte offset   COMM01.613
3002    <macro>                         MOVECHAR  R!BASE,R!PARAM,R!%CLI,R!%W1,R!PARAM2      COMM01.614
```

```
                                                                                                COMM01.615
3070    <macro>                         FRESTACK                          . Release buffer      COMM01.616
                                                                                   .>>>>> 0          $IF      3
                                                                                   .<<<<< 0          $ENDIF   3
                                                                                                COMM01.617
3106    014000 /000000  024443          R!OLAY = DV$DSPL                                        COMM01.618
3111    014000 /000000  024444          R!PARAM = DP$DSPL                                       COMM01.619
3114    075000 /001566                  P = NEW                           . Initiate the display COMM01.620
                                                                                                COMM01.621


                        ***********************************************************************COMM01.623
                        *                                                                     *COMM01.624
                        *       SUMMARY COMMAND                                               *COMM01.625
                        *                                                                     *COMM01.626
                        ***********************************************************************COMM01.627
                                                                                                COMM01.628
3116                    SUMMARY  *                                                              COMM01.629
3116    <macro>                         GETSTACK  PAR$COM,R!LIST       .BUFFER FOR DESCRIBE     COMM01.630
3140    <macro>                         CALL DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$SUM,R!LIST,RO=R!OFF COMM01.631
                        ,SET,RO=R!COUNT)                                                        COMM01.632
3171    <macro>                         $IF (A # 0)                    .SENSE ERROR             COMM01.633
                                                                                   .>>>>> 0          $IF      1
3173    <macro>                           FRESTACK                     .RELEASE MEMORY          COMM01.634
                                                                                   .>>>>> 1          $IF      3
                                                                                   .<<<<< 1          $ENDIF   3
3211    075000 /000113                    P = DONE                     .BYE                     COMM01.635
3213    <macro>                         $ELSE                                                   COMM01.636
3214    <macro>                           GET R!%W1,VA@TYP,R!LIST      .PARAMETER TYPE          COMM01.637
3217    <macro>                           $IF (R!%W1 = VAL$NONE)       .IF NOT ENTERED          COMM01.638
                                                                                   .>>>>> 1          $IF      1
3223    010000  024445                      R!PARAM2 = 0               .DEFAULT TO 0            COMM01.639
3225    <macro>                           $ELSE                                                 COMM01.640
3226    <macro>                             GET R!PARAM2,VA@PR2,R!LIST .USE ENTERED VALUE        COMM01.641
3233    <macro>                           $ENDIF                                                COMM01.642
                                                                                   .<<<<< 1          $ENDIF   1
3233    <macro>                           FRESTACK                     .RELEASE MEMORY          COMM01.643
                                                                                   .>>>>> 1          $IF      3
                                                                                   .<<<<< 1          $ENDIF   3
3251    014000 /000000  024443            R!OLAY = DV$SUMMA            .DISPLAY HANDLER         COMM01.644
```

```
3254   014000 /000000  024444          R!PARAM = DP$SUMMA          .DISPLAY DRIVER                    COMM01.645
3257   075000 /001566                  P = NEW                                                        COMM01.646
3261    <macro>                      $ENDIF                                                           COMM01.647
                                                                                   .<<<<< 0              $ENDIF    1


                                 ************************************************************************ COMM01.649
                                 *                                                                      COMM01.650
                                 *        HELP command.                                                 COMM01.651
                                 *                                                                      COMM01.652
                                 ************************************************************************ COMM01.653
                                                                                                        COMM01.654
3261                             HELP     *                                                             COMM01.655
3261    <macro>                          GETSTACK  PAR$COM,R!LIST      .Buffer for DESCRIBE            COMM01.656
3303    <macro>                          CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$HEL,R!LIST,RO= COMM01.657
                                 ,R!OFFSET,RO=R!COUNT)                                                  COMM01.658
3334    <macro>                          $IF (A # 0)                   .If error                       COMM01.659
                                                                                   .>>>>> 0              $IF       1
3336    <macro>                            FRESTACK                    .Release stack space            COMM01.660
                                                                                   .>>>>> 1              $IF       3
                                                                                   .<<<<< 1              $ENDIF    3
3354   075000 /000113                    P = DONE                                                       COMM01.661
3356    <macro>                          $ENDIF                                                         COMM01.662
                                                                                   .<<<<< 0              $ENDIF    1
                                                                                                        COMM01.663
3356    <macro>                          GET       R!%W1,VA@TYP,R!LIST                                  COMM01.664
3361    <macro>                          $IF (R!%W1 = VAL$NONE)                                         COMM01.665
                                                                                   .>>>>> 0              $IF       1
3365   010000  024445                    R!PARAM2 = 0                                                   COMM01.666
3367    <macro>                          $ELSE                                                          COMM01.667
3370    <macro>                            GET       R!PARAM,VA@PR1,R!LIST  .Start of string            COMM01.668
3375    <macro>                            GET       R!PARAM2,VA@PR2,R!LIST  .Character count           COMM01.669
3402    <macro>                          $ENDIF                                                         COMM01.670
                                                                                   .<<<<< 0              $ENDIF    1
                                                                                                        COMM01.671
3402    <macro>                          GET       R!%W2,VA@NXT,R!LIST                                  COMM01.672
3405   020447  022424  024424            R!%W2 = R!LIST + R!%W2                                         COMM01.673
3410    <macro>                          GET       R!%W1,VA@TYP,R!%W2                                   COMM01.674
3413    <macro>                          $IF (R!%W1 = VAL$NONE)                                         COMM01.675
                                                                                   .>>>>> 0              $IF       1
```

```
3417   010001  024423                R!%W1 = HELP$STA                                          COMM01.676
3421    <macro>                     $ELSE                                                     COMM01.677
3422    <macro>                       GET       R!%W1,VA@PR1,R!%W2                              COMM01.678
3427    <macro>                     $ENDIF                                                    COMM01.679
                                                                                  .<<<<< 0          $ENDIF    1
                                                                                                COMM01.680
3427   020444  024424                R!%W2 = R!PARAM                                           COMM01.681
3431   010000  024425                R!%W3 = 0                                                 COMM01.682
3433    <macro>                     $UNTIL (R!%W3 = R!PARAM2)      .Move string               COMM01.683
3436    <macro>                       GETBYTE   R!%W5,R!%W2,R!BASE,INC=1                      COMM01.684
3461    <macro>                       PUTBYTE   R!%W5,R!%W3,R!LIST,INC=1                      COMM01.685
3512    <macro>                     $ENDTIL                                                   COMM01.686
3513   010000  024424                R!%W2 = 0                      .Terminate with zero byte  COMM01.687
3515    <macro>                     PUTBYTE   R!%W2,R!%W3,R!LIST                              COMM01.688
                                                                                              COMM01.689
3545    <macro>                     CALL      HELP,(O'177777,R!%W1,R!LIST,RO=PARAM2,RO=PARAM) COMM01.690
3572    <macro>                     FRESTACK                                                  COMM01.691
                                                                                  .>>>>> 0          $IF       3
                                                                                  .<<<<< 0          $ENDIF    3
3610    <macro>                     $IF (R!PARAM2 = 0)             .If no help found          COMM01.692
                                                                                  .>>>>> 0          $IF       1
3613    <macro>                       CALL      ERROR,(EC$NOHLP,0,0,0)                          COMM01.693
3636   075000 /000113                P = DONE                                                  COMM01.694
3640    <macro>                     $ENDIF                                                    COMM01.695
                                                                                  .<<<<< 0          $ENDIF    1
                                                                                                COMM01.696
3640    <macro>                     PUT       R!PARAM,$D@P3,R!%CLI                            COMM01.697
3645   014000 /000000  024443       R!OLAY = DV$HELP               .Display handler           COMM01.698
3650   014000 /000000  024444       R!PARAM = DP$HELP              .Display driver            COMM01.699
3653   075000 /001566               P = NEW                        .Update display            COMM01.700


                                    ************************************************************** COMM01.702
                                    *                                                            * COMM01.703
                                    *        XJSTAT command.                                     * COMM01.704
                                    *                                                            * COMM01.705
                                    ************************************************************** COMM01.706
                                                                                                COMM01.707
3655                                XJSTAT    *                                                 COMM01.708
3655    <macro>                             GETSTACK  PAR$COM,R!LIST       .Get buffer for parameter list COMM01.709
```

```
3730    <macro>                         CALL      DESCRIBE  .!BASE,R!OFFSET,R!COUNT,DESC$XJS,R!LIST,RO=   COMM01.710
                                  ,R!OFFSET,RO=R!COUNT)                                                    COMM01.711
3730    <macro>                         $IF (A # 0)                    .If error                          COMM01.712
                                                                                          .>>>>> 0          $IF      1
3732    <macro>                           FRESTACK                     .Release buffer                    COMM01.713
                                                                                          .>>>>> 1          $IF      3
                                                                                          .<<<<< 1          $ENDIF   3
3750    075000 /000113                    P = DONE                                                        COMM01.714
3752    <macro>                         $ENDIF                                                            COMM01.715
                                                                                          .<<<<< 0          $ENDIF   1
3752    <macro>                         GET       R!PARAM2,VA@PR2,R!LIST  .Get JSQ                        COMM01.716
3757    <macro>                         FRESTACK                                                          COMM01.717
                                                                                          .>>>>> 0          $IF      3
                                                                                          .<<<<< 0          $ENDIF   3
3775    014000 /000000  024443          R!OLAY = DV$XJST                                                  COMM01.718
4000    014000 /000000  024444          R!PARAM = DP$XJST                                                 COMM01.719
4003    075000 /001566                  P = NEW                                                           COMM01.720

                                  ***********************************************************************   COMM01.722
                                  *                                                                     *   COMM01.723
                                  *       RSTAT command                                                 *   COMM01.724
                                  *                                                                     *   COMM01.725
                                  ***********************************************************************   COMM01.726
                                                                                                            COMM01.727
4005                              RSTAT   *                                                                 COMM01.728
4005    <macro>                         STORE     0,$D@P3,R!%CLI  .clear JSQ parameter                    COMM01.729
4012    020402  012032  024423          R!%W1 = R!%CLI + $D@GRN@P                                         COMM01.730
4015    <macro>                         CLEAR     START=R!%W1,COUNT=4  .Clear the resource name           COMM01.731
4030    <macro>                         GETSTACK  PAR$COM,R!LIST  .Get buffer for parameter list          COMM01.732
4052    <macro>                         CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$RST,R!LIST,RO=   COMM01.733
                                  ,R!OFFSET,RO=R!COUNT)                                                    COMM01.734
4103    <macro>                         $IF (A # 0)                .if an error encountered               COMM01.735
                                                                                          .>>>>> 0          $IF      1
4105    <macro>                           FRESTACK                 .release buffer                        COMM01.736
                                                                                          .>>>>> 1          $IF      3
                                                                                          .<<<<< 1          $ENDIF   3
4123    075000 /000113                    P = DONE                                                        COMM01.737
4125    <macro>                         $ENDIF                                                            COMM01.738
                                                                                          .<<<<< 0          $ENDIF   1
4125    010000  024446                  R!QUEUES = 0                                                      COMM01.739
4127    <macro>                         GET       R!%W1,VA@TYP,R!LIST  .get the entry type                COMM01.740
```

```
4132    <macro>                          GET       R!PARAM2,VA@PR1,R!LIST  .get the display type          COMM01.741
4137    020423  013014  103003          R!PARAM2 = GRD$P, R!%W1 = VAL$NONE  .default profile display   COMM01.742
        010001  024445
4144    <macro>                          $IF (R!PARAM2 = GQT$I) OR (R!PARAM2 = GQT$E)                   COMM01.743
                                                                                   .>>>>> 0              $IF      1
4153    010003  024445                     R!PARAM2 = GRD$Q       .if queue was entered, set display   COMM01.744
4155    <macro>                          $ENDIF                                                         COMM01.745
                                                                                   .<<<<< 0              $ENDIF   1
4155    020423  013001  103003          R!PARAM2 = GRD$J, R!%W1 = VAL$DEC  .if a JSQ entered           COMM01.746
        010002  024445
                                                                                                         COMM01.747
4162    <macro>                          $UNTIL (R!%W1 = VAL$END),OR,(R!%W1 = VAL$NONE)                 COMM01.748
4170    <macro>                            $IF (R!%W1 = VAL$LIT)                                        COMM01.749
                                                                                   .>>>>> 0              $IF      1
4174    <macro>                              GET       R!%W2,VA@PR1,R!LIST  .get the parameter          COMM01.750
4201    <macro>                              $IF (R!%W2 = GRD$G)  .if a named request                   COMM01.751
                                                                                   .>>>>> 1              $IF      1
4205    072350                                 R = NEXT                                                 COMM01.752
4206    <macro>                                GET     R!%W4,VA@PR2,R!LIST  .string length              COMM01.753
4213    <macro>                                GET     R!%W3,VA@PR1,R!LIST  .string offset              COMM01.754
4220    010032  005001  024427               R!%W5 = $D@GRN@P < 1                                       COMM01.755
4223    <macro>                                MOVECHAR R!BASE,R!%W3,R!%CLI,R!%W5,R!%W4                 COMM01.756
4311    <macro>                              $ELSEIF (R!%W2 = GQT$I),OR,(R!%W2 = GQT$E)  .queue         COMM01.757
4321    020446  022424  024446                 R!QUEUES = R!QUEUES + R!%W2                              COMM01.758
4324    <macro>                              $ENDIF                                                     COMM01.759
                                                                                   .<<<<< 1              $ENDIF   1
4324    <macro>                            $ELSEIF (R!%W1 = VAL$DEC)   .if a JSQ was entered            COMM01.760
4331    <macro>                              GET     R!%W3,VA@PR2,R!LIST                                COMM01.761
4336    <macro>                              PUT     R!%W3,$D@P3,R!%CLI                                 COMM01.762
4343    <macro>                            $ENDIF                                                       COMM01.763
                                                                                   .<<<<< 0              $ENDIF   1
4343    072212                             R = NEXT                .look for the next parameter        COMM01.764
4344    <macro>                            GET     R!%W1,VA@TYP,R!LIST                                  COMM01.765
4347    <macro>                          $ENDTIL                                                        COMM01.766
                                                                                                         COMM01.767
4350    <macro>                          FRESTACK                                                       COMM01.768
                                                                                   .>>>>> 0              $IF      3
                                                                                   .<<<<< 0              $ENDIF   3
4366    020446  103003  010050          R!QUEUES = GQT$IE, R!QUEUES = 0                                COMM01.769
        024446
4372    014000 /000000  024443          R!OLAY = DV$RSTAT                                              COMM01.770
4375    014000 /000000  024444          R!PARAM = DP$RSTAT                                             COMM01.771
```

```
                                 ********************************************************************  COMM01.774
                                 *                                                                  *  COMM01.775
                                 *                    SWAP  command.                                *  COMM01.776
                                 *                                                                  *  COMM01.777
                                 ********************************************************************  COMM01.778
4402                             SWAP     *                                                           COMM01.779
4402   014000 /000000  024443            R!OLAY  = DV$SWAP                                            COMM01.780
4405   014000 /000000  024444            R!PARAM = DP$SWAP                                            COMM01.781
4410   075000 /001566                    P = NEW                                                      COMM01.782


                                 *        Establish the command scroll line and clear the display area.  COMM01.785
                                 *                                                                     COMM01.786
                                 *        SCROLL    Command scroll line.                               COMM01.787
                                 *        LINE      First display line to clear.                       COMM01.788
                                 *        NUMBER    Number of lines to clear.                          COMM01.789
                                                                                                      COMM01.790
4412                             SCREEN   *                                                           COMM01.791
4412    <macro>                          CALL      CONSL,(CF$LINE,R!SCROLL) . Set new scroll line.    COMM01.792
4431    <macro>                          GETSTACK  1,R!OUTPUT          . Allocate a line buffer.      COMM01.793
4452    <macro>                          PUTBYTE   SC$EEOL,0,R!OUTPUT                                 COMM01.794
                                                                                                      COMM01.795
4502    <macro>                          $UNTIL    (R!NUMBER = 0)      . Until display area is cleared. COMM01.796
4504    <macro>                            CALL    CONSL,(CF$WRITE,R!LINE,0,R!OUTPUT,0,1) . Clear line. COMM01.797
4533   026436                              R!LINE = R!LINE+1                                          COMM01.798
4534   027437                              R!NUMBER = R!NUMBER-1                                      COMM01.799
4535    <macro>                          $ENDTIL                                                      COMM01.800
                                                                                                      COMM01.801
4536    <macro>                          FRESTACK                      . Release the buffer.          COMM01.802
                                                                                   .>>>>> 0              $IF      3
                                                                                   .<<<<< 0              $ENDIF   3
4554   001000                           EXIT                                                         COMM01.803


                                 *                                                                    COMM01.805
                                 *        Advance to the next entry on the command information buffer COMM01.806
                                 *                                                                    COMM01.807
```

```
                                                                                COMM01.808
4555                          NEXT     *                                        COMM01.809
4555   <macro>                        GET       R!%W1,VA@NXT,R!LIST             COMM01.810
4560   020447  022423  024447          R!LIST = R!LIST + R!%W1                   COMM01.811
4563   001000                         EXIT                                      COMM01.812

                                      END                                       COMM01.814
```

20	$C@CNT@N	$APTEXT	7: 7	7: 7 F						
5	$C@CNT@P	$APTEXT	7: 7 F	7: 7						
0	$C@CNT@S	$APTEXT	7: 7	7: 7 F						
20	$C@DIS@N	$APTEXT	7: 6	7: 6 F						
4	$C@DIS@P	$APTEXT	7: 6 F	7: 6						
0	$C@DIS@S	$APTEXT	7: 6	7: 6 F						
60	$D@DEB@P	$APTEXT	17:24	17:41						
20	$D@FRM@N	$APTEXT	16:20 F							
11	$D@FRM@P	$APTEXT	16:20 F	16:20						
0	$D@FRM@S	$APTEXT	16:20 F							
32	$D@GRN@P	$APTEXT	26:28	27:21						
20	$D@LFT@N	$APTEXT	18:10 F							
13	$D@LFT@P	$APTEXT	18:10 F	18:10						
0	$D@LFT@S	$APTEXT	18:10 F							
20	$D@MON@N	$APTEXT	14: 1	14: 1 F	14:13 F					
23	$D@MON@P	$APTEXT	14: 1 F	14: 1	14:13 F	14:13				
0	$D@MON@S	$APTEXT	14: 1	14: 1 F	14:13 F					
20	$D@OVL@N	$APTEXT	16:24 F							
10	$D@OVL@P	$APTEXT	16:24 F	16:24						
0	$D@OVL@S	$APTEXT	16:24 F							
20	$D@P2@N	$APTEXT	9: 5 F	16:26 F						
27	$D@P2@P	$APTEXT	9: 5 F	9: 5	16:26 F	16:26				
0	$D@P2@S	$APTEXT	9: 5 F	16:26 F						
20	$D@P3@N	$APTEXT	9: 8 F	10:37 F	25:27 F	26:27 F	27:29 F			
30	$D@P3@P	$APTEXT	9: 8 F	9: 8	10:37 F	10:37	22:41	25:27 F	25:27	26:27 F
			26:27	27:29 F	27:29					
0	$D@P3@S	$APTEXT	9: 8 F	10:37 F	25:27 F	26:27 F	27:29 F			
20	$D@PAR@N	$APTEXT	16:25 F							
26	$D@PAR@P	$APTEXT	16:25 F	16:25						
0	$D@PAR@S	$APTEXT	16:25 F							
20	$D@QUE@N	$APTEXT	16:19 F							
12	$D@QUE@P	$APTEXT	16:19 F	16:19						
0	$D@QUE@S	$APTEXT	16:19 F							
20	$D@RGT@N	$APTEXT	18:11 F							
14	$D@RGT@P	$APTEXT	18:11 F	18:11						
0	$D@RGT@S	$APTEXT	18:11 F							
20	$D@SCI@N	$APTEXT	15:12 F	15:40 F	19:24 F	20: 6 F				
24	$D@SCI@P	$APTEXT	15:12 F	15:12	15:40 F	15:40	19:24 F	19:24	20: 6 F	20: 6
0	$D@SCI@S	$APTEXT	15:12 F	15:40 F	19:24 F	20: 6 F				
20	$D@SCI@N	$APTEXT	15:41 F	20: 7 F						
25	$D@SCI@P	$APTEXT	15:41 F	15:41	20: 7 F	20: 7				
0	$D@SCI@S	$APTEXT	15:41 F	20: 7 F						

20	$D@TYP@N	$APTEXT	8:14 F	16:27 F						
6	$D@TYP@P	$APTEXT	8:14 F	8:14	16:27 F	16:27				
0	$D@TYP@S	$APTEXT	8:14 F	16:27 F						
41	$I@DIS@P	$APTEXT	7: 5							
55	$R@DIS@P	$APTEXT	16:28							
0X	%ACTIVE		8:13 D	8:13	16:18 D	16:18	18: 9 D	18: 9		
0X	%B		1: 4 D	7:10	8:20 B	9:19 B	10:22 B	10:25 B	10:40 B	11:38 B
			12:18 B	12:24 B	12:30 B	13:27 B	13:42 B	14:10 B	15:24 B	16:33 B
			16:34 B	17:13 B	17:18 B	17:29 B	18:24 B	19:35 B	20:35 B	21:25 B
			21:37 B	22:23 B	22:35 B	23: 8 B	23:29 B	24: 2 B	24:24 B	25:23 B
			25:30 B	26: 8 B	26:17 B	26:38 B	28: 1 B	29: 9 B		
406	%BASE	$APTEXT	8:37	9:14	9:38	10:12	11:18	11:34	13:25	13:40
			15:22	16: 5	19:33	20:12	20:33	21: 1	21:23	21:33
			22:21	22:33	23: 3	23:27	23:40	24:22	25:18	26: 6
			26:13	26:36	27:37	29:34				
402	%CLI	$APTEXT	7: 5	7: 6	7: 7	8:14	9: 5	9: 8	10:37	14: 1
			14:13	15:12	15:40	15:41	16:19	16:20	16:24	16:25
			16:26	16:27	16:28	17:24	17:41	18:10	18:11	19:24
			20: 6	20: 7	22:42	25:27	26:27	26:28	27:22	27:29
0X	%EX		7:13 D	7:13	7:16 D	7:16	8:13 D	8:13	8:32 D	8:32
			9:33 D	9:33	11:13 D	11:13	13:18 D	13:18	13:36 D	13:36
			14: 5 D	14: 5	14: 9 D	14: 9	15:17 D	15:17	16:18 D	16:18
			16:29 D	16:29	18: 9 D	18: 9	18:23 D	18:23	19:28 D	19:28
			20:29 D	20:29	21:19 D	21:19	22:16 D	22:16	23:23 D	23:23
			24:18 D	24:18	25:16 D	25:16	25:22 D	25:22	26: 2 D	26: 2
			26:32 D	26:32	29:23 D	29:23	29:28 D	29:28		
0X	%FEI		4:40 D	10:18	10:32					
10	%GBLNUM	$APTEXT	4: 3 F	4: 3						
400	%GBLREG	$APTEXT	4: 3							
405	%LIMIT	$APTEXT	8:30	8:37	9:14	9:31	9:38	10:12	11:11	11:18
			11:34	13:16	13:25	13:34	13:40	15:13	15:22	16: 5
			19:25	19:33	20:12	20:27	20:33	21: 1	21:17	21:23
			21:33	22:13	22:21	22:33	23: 3	23:21	23:27	23:40
			24:16	24:22	25:18	25:42	26: 6	26:13	26:30	26:36
			27:37	29:24	29:34					
0X	%PUNT		8:30 D	8:30	8:36 D	8:37	9:13 D	9:14	9:31 D	9:31
			9:37 D	9:38	10:11 D	10:12	11:11 D	11:11	11:17 D	11:18
			11:33 D	11:34	13:16 D	13:16	13:24 D	13:25	13:34 D	13:34
			13:39 D	13:40	15:13 D	15:13	15:21 D	15:22	16: 4 D	16: 5
			19:25 D	19:25	19:32 D	19:33	20:11 D	20:12	20:27 D	20:27
			20:32 D	20:33	20:42 D	21: 1	21:17 D	21:17	21:22 D	21:23
			21:32 D	21:33	22:13 D	22:13	22:20 D	22:21	22:32 D	22:33

			23: 2	D	23: 3		23:21	D	23:21		23:26	D	23:27		23:39	D	23:40
			24:16	D	24:16		24:21	D	24:22		25:17	D	25:18		25:42	D	25:42
			26: 5	D	26: 6		26:12	D	26:13		26:30	D	26:30		26:35	D	26:36
			27:36	D	27:37		29:24	D	29:24		29:33	D	29:34				
404	%STACK	$APTEXT	8:30		8:36		8:37		9:13		9:14		9:31		9:37		9:38
			10:11		10:12		11:11		11:17		11:18		11:33		11:34		13:16
			13:24		13:25		13:34		13:39		13:40		15:13		15:21		15:22
			16: 4		16: 5		19:25		19:32		19:33		20:11		20:12		20:27
			20:32		20:33		20;42		21: 1		21:17		21:22		21:23		21:32
			21:33		22:13		22:20		22:21		22:32		22:33		23: 2		23: 3
			23:21		23:26		23:27		23:39		23:40		24:16		24:21		24:22
			25:17		25:18		25:42		26: 5		26: 6		26:12		26:13		26:30
			26:35		26:36		27:36		27:37		29:24		29:33		29:34		
20	AD@RC@N	$APTEXT	8:13		8:13	F	16:18		16:18	F	18: 9		18: 9	F			
13	AD@RC@P	$APTEXT	8:13	F	8:13		16:18	F	16:18		18: 9	F	18: 9				
0	AD@RC@S	$APTEXT	8:13		8:13	F	16:18		16:18	F	18: 9		18: 9	F			
0X	CF$LINE		4:41	D	29:23												
0X	CF$WRITE		4:42	D	29:28												
36	CH$CN	$APTEXT	10:25														
0X	CP$CLEAR		5: 1	D	8:17												
0X	DESC$BAB		5: 2	D	13:18												
0X	DESC$CNC		5: 3	D	9:33												
0X	DESC$DKE		5: 4	D	20:29												
0X	DESC$DKS		5: 5	D	21:19												
0X	DESC$DSP		5: 6	D	22:16												
0X	DESC$HEL		5:14	D	24:18												
0X	DESC$LPA		5: 7	D	8:32												
0X	DESC$MON		5: 8	D	13:36												
0X	DESC$QUE		5:10	D	15:17												
0X	DESC$RST		5: 9	D	26:32												
0X	DESC$STS		5:11	D	11:13												
0X	DESC$SUM		5:12	D	23:23												
0X	DESC$TJO		5:13	D	19:28												
0X	DESC$XJS		5:15	D	26: 2												
2	DIS$DEB	$APTEXT	18:14														
0	DIS$NONE	$APTEXT	8:14														
1	DIS$OPER	$APTEXT	16:21														
0X	DP$BABEL		5:16	D	13:15												
0X	DP$CNC		5:18	D	10:39												
0X	DP$DEBUG		5:17	D	12:17												
0X	DP$DKERR		5:19	D	21: 4												
0X	DP$DKSTA		5:20	D	21:36												

0X	DP$DSPL	5:21	D	23: 7
0X	DP$ERROR	5:23	D	12:23
0X	DP$HELP	5:38	D	25:29
0X	DP$LINK	5:24	D	12:29
0X	DP$MONIT	5:25	D	13:33
0X	DP$NSCLP	5:22	D	9:18
0X	DP$RSTAT	5:26	D	27:42
0X	DP$SPLIT	5:27	D	18:13
0X	DP$STATC	5:28	D	12:41
0X	DP$STATI	5:29	D	12:35
0X	DP$STATU	5:30	D	16:10
0X	DP$STORA	5:31	D	13: 3
0X	DP$STP	5:32	D	13: 9
0X	DP$STRS	5:33	D	11:37
0X	DP$SUMMA	5:34	D	24: 1
0X	DP$SWAP	5:35	D	29: 8
0X	DP$TAPE	5:36	D	19:11
0X	DP$TJOB	5:37	D	20:15
0X	DP$XJST	5:39	D	26:16
0X	DV$BABEL	5:40	D	13:14
0X	DV$CNC	5:41	D	10:38
0X	DV$DEBUG	5:42	D	12:16
0X	DV$DKERR	6: 1	D	21: 3
0X	DV$DKSTA	6: 2	D	21:35
0X	DV$DSPL	6: 3	D	23: 6
0X	DV$ERROR	6: 5	D	12:22
0X	DV$HELP	6:20	D	25:28
0X	DV$LINK	6: 6	D	12:28
0X	DV$MONIT	6: 7	D	13:32
0X	DV$NSCLP	6: 4	D	9:17
0X	DV$RSTAT	6: 8	D	27:41
0X	DV$SPLIT	6: 9	D	18:12
0X	DV$STATC	6:10	D	12:40
0X	DV$STATI	6:11	D	12:34
0X	DV$STATU	6:12	D	16: 9
0X	DV$STORA	6:13	D	13: 2
0X	DV$STP	6:14	D	13: 8
0X	DV$STRS	6:15	D	11:36
0X	DV$SUMMA	6:16	D	23:42
0X	DV$SWAP	6:17	D	29: 7
0X	DV$TAPE	6:18	D	19:10
0X	DV$TJOB	6:19	D	20:14

0X	DV$XJST		6:21 D	26:15						
0X	EC$BDPAR		6:22 D	18:23						
0X	EC$NOHLP		6:23 D	25:22						
50	F$CALL	$APTEXT	8:32	9:33	11:13	13:18	13:36	14: 9	15:17	18:23
			19:28	20:29	21:19	22:16	23:23	24:18	25:16	25:22
			26: 2	26:32	29:23	29:28				
30	F$GMEM	$APTEXT	14: 5							
2	F$POP	$APTEXT	7:13	16:29						
1	F$PUSH	$APTEXT	8:13	16:18	18: 9					
52	F$RETURN	$APTEXT	7:16							
377	FEI@CI@M	$APTEXT	10:24							
10	FEI@CI@N	$APTEXT	10:24							
0	FEI@CI@P	$APTEXT	10:24 F							
10	FEI@CI@S	$APTEXT	10:24	10:24 F						
7	FEI@LE	$APTEXT	10:18							
2	FEI@LH	$APTEXT	10:18							
20	FEI@ON@N	$APTEXT	10:32	10:32 F						
1	FEI@ON@P	$APTEXT	10:32 F	10:32						
0	FEI@ON@S	$APTEXT	10:32	10:32 F						
0	FS$PRI	$APTEXT	8:13	16:18	18: 9					
10	GQT$E	$APTEXT	27: 5	27:23						
40	GQT$I	$APTEXT	27: 5	27:23						
50	GQT$IE	$APTEXT	27:39							
4	GRD$G	$APTEXT	27:17							
2	GRD$J	$APTEXT	27: 9							
1	GRD$P	$APTEXT	27: 2							
3	GRD$Q	$APTEXT	27: 6							
1	HELP$STA	$APTEXT	25: 1							
3	I@COCO	$APTEXT	10: 7							
13	I@NCOR	$APTEXT	10:16	10:18						
14	I@SNID	$APTEXT	10:30							
0X	O$COMMO1		1: 4 D	1: 4						
0X	O$CONSL		29:23 D	29:23	29:28 D	29:28				
0X	O$DESCRI		8:32 D	8:32	9:33 D	9:33	11:13 D	11:13	13:18 D	13:18
			13:36 D	13:36	15:17 D	15:17	19:28 D	19:28	20:29 D	20:29
			21:19 D	21:19	22:16 D	22:16	23:23 D	23:23	24:18 D	24:18
			26: 2 D	26: 2	26:32 D	26:32				
0X	O$ERROR		14: 9 D	14: 9	18:23 D	18:23	25:22 D	25:22		
0X	O$HELP		25:16 D	25:16						
17	OV@NUM@N	$APTEXT	1: 4							
1	OV@TYP@N	$APTEXT	1: 4							
24	PAR$COM	$APTEXT	8:30	9:31	11:11	13:16	13:34	15:13	19:25	20:27

			21:17	22:13	23:21	24:16	25:42	26:30		
13	PI$STACK	$APIEXT	8:30	8:37	9:14	9:31	9:38	10:12	11:11	11:18
			11:34	13:16	13:25	13:34	13:40	15:13	15:22	16: 5
			19:25	19:33	20:12	20:27	20:33	21: 1	21:17	21:23
			21:33	22:13	22:21	22:33	23: 3	23:21	23:27	23:40
			24:16	24:22	25:18	25:42	26: 6	26:13	26:30	26:36
			27:37	29:24	29:34					
73	QI$ALI	$APIEXT	16: 7							
205	SC$EEOL	$APIEXT	29:25							
11	SM@ESI@N	$APIEXT	1: 4							
7	SM@NUM@N	$APIEXT	1: 4							
14	SM@REG@P	$APIEXT	8:32	9:33	11:13	13:18	13:36	15:17	19:28	20:29
			21:19	22:16	23:23	24:18	25:16	26: 2	26:32	
377	VA@NXI@M	$APIEXT	24:38	30: 3						
10	VA@NXI@N	$APIEXT	24:38	30: 3						
0	VA@NXI@P	$APIEXT	24:38 F	30: 3 F						
10	VA@NXI@S	$APIEXT	24:38	24:38 F	30: 3	30: 3 F				
20	VA@PRI@N	$APIEXT	11:25	11:25 F	13:20	13:20 F	13:38	13:38 F	15:32	15:32 F
			22:27	22:27 F	24:33	24:33 F	25: 3	25: 3 F	27: 1	27: 1 F
			27:15	27:15 F	27:20	27:20 F				
1	VA@PRI@P	$APIEXT	11:25 F	11:25	13:20 F	13:20	13:38 F	13:38	15:32 F	15:32
			22:27 F	22:27	24:33 F	24:33	25: 3 F	25: 3	27: 1 F	27: 1
			27:15 F	27:15	27:20 F	27:20				
0	VA@PRI@S	$APIEXT	11:25	11:25 F	13:20	13:20 F	13:38	13:38 F	15:32	15:32 F
			22:27	22:27 F	24:33	24:33 F	25: 3	25: 3 F	27: 1	27: 1 F
			27:15	27:15 F	27:20	27:20 F				
20	VA@PR2@N	$APIEXT	9: 4	9: 4 F	9: 7	9: 7 F	10: 5	10: 5 F	15:36	15:36 F
			20: 1	20: 1 F	20:39	20:39 F	21:29	21:29 F	22:28	22:28 F
			23:36	23:36 F	24:34	24:34 F	26:11	26:11 F	27:19	27:19 F
			27:28	27:28 F						
2	VA@PR2@P	$APIEXT	9: 4 F	9: 4	9: 7 F	9: 7	10: 5 F	10: 5	15:36 F	15:36
			20: 1 F	20: 1	20:39 F	20:39	21:29 F	21:29	22:28 F	22:28
			23:36 F	23:36	24:34 F	24:34	26:11 F	26:11	27:19 F	27:19
			27:28 F	27:28						
0	VA@PR2@S	$APIEXT	9: 4	9: 4 F	9: 7	9: 7 F	10: 5	10: 5 F	15:36	15:36 F
			20: 1	20: 1 F	20:39	20:39 F	21:29	21:29 F	22:28	22:28 F
			23:36	23:36 F	24:34	24:34 F	26:11	26:11 F	27:19	27:19 F
			27:28	27:28 F						
10	VA@TYP@N	$APIEXT	9: 1	9: 1 F	10: 2	10: 2 F	11:22	11:22 F	13:21	13:21 F
			15:28	15:28 F	19:39	19:39 F	20:38	20:38 F	21:28	21:28 F
			23:31	23:31 F	24:28	24:28 F	24:40	24:40 F	26:42	26:42 F
			27:33	27:33 F						

0	VA@TYP@P	$APTEXT	9: 1 F	10: 2 F	11:22 F	13:21 F	15:28 F	19:39 F	20:38 F	21:28 F
			23:31 F	24:28 F	24:40 F	26:42 F	27:33 F			
0	VA@TYP@S	$APTEXT	9: 1	9: 1 F	10: 2	10: 2 F	11:22	11:22 F	13:21	13:21 F
			15:28	15:28 F	19:39	19:39 F	20:38	20:38 F	21:28	21:28 F
			23:31	23:31 F	24:28	24:28 F	24:40	24:40 F	26:42	26:42 F
			27:33	27:33 F						
1	VAL$DEC	$APTEXT	15:35	19:42	27: 9	27:27				
15	VAL$END	$APTEXT	27:12							
0	VAL$ID	$APTEXT	11:24							
3	VAL$LIT	$APTEXT	15:31	27:14						
14	VAL$NONE	$APTEXT	9: 3	10: 4	13:22	20:40	21:30	23:33	24:30	24:42
			27: 2	27:12						

SYMBOL QUALIFIER IS COMMO1

21	$$RO	8:32 D	8:32	9:33 D	9:33	11:13 D	11:13	13:18 D	13:18	
		13:36 D	13:36	15:17 D	15:17	19:28 D	19:28	20:29 D	20:29	
		21:19 D	21:19	22:16 D	22:16	23:23 D	23:23	24:18 D	24:18	
		25:16 D	25:16	26: 2 D	26: 2	26:32 D	26:32			
0	$CONST	8:14 D	8:14 F	15:12 D	15:12 F	16:20 D	16:20 F	16:20	19:24 D	
		19:24 F	26:27 D	26:27 F						
453	$REGORG	4: 3 D	4: 3 F	4: 4	4: 4 D	4: 5	4: 5 D	4: 6	4: 6 D	
		4: 7	4: 7 D	4: 8	4: 8 D	4: 8 F	4: 9	4: 9 D	4:10	
		4:10 D	4:11	4:11 D	4:12	4:12 D	4:13	4:13 D	4:14	
		4:14 D	4:14 F	4:15	4:15 D	4:16	4:16 D	4:17	4:17 D	
		4:18	4:18 D	4:19	4:19 D	4:19 F	4:20	4:20 D	4:21	
		4:21 D	4:22	4:22 D	4:23	4:23 D	4:24	4:24 D	4:25	
		4:25 D	4:25 F	4:26	4:26 D	4:27	4:27 D	4:28	4:28 D	
		4:29	4:29 D	4:30	4:30 D	4:31	4:31 D	4:32	4:32 D	
		4:33	4:33 D	4:34	4:34 D	4:35	4:35 D	4:36	4:36 D	
		4:37	4:37 D	4:38	4:38 D	4:38 F	7:13	7:16	8:13	
		8:13 F	8:32	8:32 F	9:33	9:33 F	11:13	11:13 F	13:18	
		13:18 F	13:36	13:36 F	14: 5	14: 9	14: 9 F	15:17	15:17 F	
		16:18	16:18 F	16:29	18: 9	18: 9 F	18:23	18:23 F	19:28	
		19:28 F	20:29	20:29 F	21:19	21:19 F	22:16	22:16 F	23:23	
		23:23 F	24:18	24:18 F	25:16	25:16 F	25:22	25:22 F	26: 2	
		26: 2 F	26:32	26:32 F	29:23	29:23 F	29:28	29:28 F		
6	%NP	1: 4	4:25 D							
430	%P	1: 4	4:19 D	4:25	8:13 F	8:13	8:32 F	8:32	9:33 F	
		9:33	11:13 F	11:13	13:18 F	13:18	13:36 F	13:36	14: 9 F	
		14: 9	15:17 F	15:17	16:18 F	16:18	18: 9 F	18: 9	18:23 F	
		18:23	19:28 F	19:28	20:29 F	20:29	21:19 F	21:19	22:16 F	

		22:16	23:23 F	25:23	24:18 F	24:18	25:16 F	25:16	25:22 F
		25:22	26: 2 F	26: 2	26:32 F	26:32	29:23 F	29:23	29:28 F
		29:28							
410	%S1	4: 3	4: 4 D	7: 6	7: 7	8:13	8:14	9: 4	9: 5
		9: 7	9: 8	10: 5	10:32	10:37	11:25	13:20	13:38
		14: 1	14:13	15:12	15:32	15:36	15:40	15:41	16:18
		16:19	16:20	16:24	16:25	16:26	16:27	17:14	17:31
		18: 9	18:10	18:11	19:24	20: 1	20: 6	20: 7	20:39
		21:29	22:27	22:28	22:42	23:36	24:33	24:34	25: 3
		25:10	25:11	25:14	25:27	26:11	26:27	27: 1	27:15
		27:19	27:20	27:22	27:28	27:29	29:25		
411	%S2	4: 3	4: 5 D						
412	%S3	4: 3	4: 6 D						
413	%S4	4: 3	4: 7 D						
414	%S5	4: 3	4: 8 D						
415	%I1	4: 9 D	7:10	7:13	8:13	8:30	9:31	10:17	10:18
		11:11	13:16	13:34	15:13	16:18	16:29	17:14	17:31
		18: 9	19:25	20:27	21:17	22:13	22:42	23:21	24:16
		25:10	25:11	25:14	25:42	26:29	26:30	27:22	29:24
		29:25							
416	%I2	4:10 D	10:18	10:19	22:42	25:11	25:14	26:29	27:22
		29:25							
417	%I3	4:11 D	22:42	27:22					
420	%I4	4:12 D	22:42	27:22					
421	%I5	4:13 D	22:42	27:22					
422	%I6	4:14 D	22:42	27:22					
423	%W1	4:15 D	9: 1	9: 3	10: 2	10:24	10:25	10:30	10:33
		11:22	13:19	13:27	13:37	13:42	14: 8	14: 9	15:28
		15:31	15:32	15:33	15:35	16:21	16:27	16:28	16:29
		17:14	17:17	17:22	17:25	17:31	17:34	17:39	17:42
		18:14	19:39	19:42	20:38	20:40	21:28	21:30	22:41
		22:42	23:31	23:33	24:28	24:30	24:40	24:42	25: 1
		25: 3	25:16	26:28	26:29	26:42	27: 2	27: 9	27:12
		27:14	27:27	27:33	30: 3	30: 4			
424	%W2	4:16 D	9: 7	9: 8	10: 5	10: 7	10:16	10:17	10:30
		10:37	13:21	13:22	15:36	15:37	15:38	15:40	15:41
		17:30	17:31	20: 1	20: 2	20: 3	20: 6	20: 7	24:38
		24:39	24:40	25: 3	25: 7	25:10	25:13	25:14	27:15
		27:17	27:23	27:24					
425	%W3	4:17 D	15:37	15:38	20: 2	20: 3	25: 8	25: 9	25:11
		25:14	27:20	27:22	27:28	27:29			
426	%W4	4:18 D	27:19	27:22					

427	%W5	4:19 D	25:10	25:11	27:21	27:22			
1036	BABEL	7:10	13:13 D						
431	BASE	4:21 D	8:32	9:33	11:13	13:18	13:36	15:17	17:14
		17:31	18:23	19:28	20:29	21:19	22:16	22:42	23:23
		24:18	25:10	26: 2	26:32	27:22			
433	CMDLEN	4:23 D	17:12	17:29	18:23				
432	CMDOFF	4:22 D	17:14	17:30	18:23				
351	CONC	7:10	9:29 D						
435	COUNT	4:25 D	8:32	8:32 F	9:33	9:33 F	11:13	11:13 F	13:18
		13:18 F	13:36	13:36 F	15:17	15:17 F	19:28	19:28 F	20:29
		20:29 F	21:19	21:19 F	22:16	22:16 F	23:23	23:23 F	24:18
		24:18 F	26: 2	26: 2 F	26:32	26:32 F			
752	DEBUG	7:10	12:15 D						
2345	DKERR	7:10	20:26 D						
2504	DKSTAT	7:10	21:16 D						
113	DONE	7:15 D	8:39	9:40	11:20	13:27	13:42	14:10	15:24
		18:24	19:35	20:35	21:25	22:23	22:35	23:29	24:24
		25:23	26: 8	26:38					
2644	DSPL	7:10	22: 8 D						
121	ERASE	7:10	8: 9 D						
762	ERR	7:10	12:21 D						
2125	ERROR	10:22	10:25	17:13	17:18	17:35	18:22 D		
452	FET	4:38 D							
3261	HELP	7:10	24:15 D						
450	LEFT	4:36 D	17:24	17:25	18:10				
436	LINE	4:26 D	7: 6	8:16	16:30	16:32	29:28	29:29	
772	LINK	7:10	12:27 D						
447	LIST	4:35 D	8:30	8:32	9: 1	9: 4	9: 6	9: 7	9:31
		9:33	10: 2	10: 5	11:11	11:13	11:22	11:25	13:16
		13:18	13:20	13:21	13:34	13:36	13:38	14: 5	14:13
		15:13	15:17	15:28	15:32	15:36	19:25	19:28	19:39
		20: 1	20:27	20:29	20:38	20:39	21:17	21:19	21:28
		21:29	22:13	22:16	22:27	22:28	23:21	23:23	23:31
		23:36	24:16	24:18	24:28	24:33	24:34	24:38	24:39
		25:11	25:14	25:16	25:42	26: 2	26:11	26:30	26:32
		26:42	27: 1	27:15	27:19	27:20	27:28	27:33	30: 3
		30: 4							
172	LPATH	7:10	8:28 D						
442	MASTER	4:30 D	7: 5	7:13	8:13	16:18	18: 9		
1160	MONITOR	7:10	13:31 D						
1566	NEW	9:19	10:40	11:38	12:18	12:24	12:30	12:36	12:42
		13: 4	13:10	13:28	14:17	16:17 D	19:12	20:16	21: 5

		21:37	23: 8	24	25:30	26:17	28: 1	29: 9	
4555	NEXT	27:18	27:32	30: 2 D					
437	NUMBER	4:27 D	7: 7	8:15	8:18	16:30	16:31	29:27	29:30
434	OFFSET	4:24 D	8:32	8:32 F	9:33	9:33 F	11:13	11:13 F	13:18
		13:18 F	13:36	13:36 F	15:17	15:17 F	19:28	19:28 F	20:29
		20:29 F	21:19	21:19 F	22:16	22:16 F	23:23	23:23 F	24:18
		24:18 F	26: 2	26: 2 F	26:32	26:32 F			
443	OLAY	4:31 D	9:17	10:38	11:36	12:16	12:22	12:28	12:34
		12:40	13: 2	13: 8	13:14	13:32	16: 9	16:24	18:12
		19:10	20:14	21: 3	21:35	23: 6	23:42	25:28	26:15
		27:41	29: 7						
441	OUTPUT	4:29 D	29:24	29:25	29:28				
430	PAR	4:20 D	7:10	8:17					
444	PARAM	4:32 D	9:18	10:39	11:37	12:17	12:23	12:29	12:35
		12:41	13: 3	13: 9	13:15	13:33	16:10	16:25	18:13
		19:11	20:15	21: 4	21:36	22:27	22:42	23: 7	24: 1
		24:33	25: 7	25:16 F	25:16	25:27	25:29	26:16	27:42
		29: 8							
445	PARAM2	4:33 D	7: 4	9: 4	9: 5	10:17	10:19	10:21	10:24
		10:32	10:33	11:25	11:27	13:20	13:23	13:38	16:26
		20:39	20:41	21:29	21:31	22:28	22:31	22:42	23:34
		23:36	24:31	24:34	25: 9	25:16 F	25:16	25:21	26:11
		27: 1	27: 3	27: 5	27: 6	27:10			
446	QUEUES	4:34 D	15:11	15:33	16: 7	16:19	26:41	27:24	27:39
74	RELEASE	7:12 D	8:21	16:34					
451	RIGHT	4:37 D	17:26	17:41	17:42	18:11			
4005	RSTAT	7:10	26:26 D						
1632	SAVE	16:23 D	18:15						
4412	SCREEN	8:20	16:33	29:22 D					
440	SCROLL	4:28 D	8:16	8:18	16:32	29:23			
1716	SELECT	7:10	17: 7 D						
1011	STATC	7:10	12:39 D						
1002	STATION	7:10	12:33 D						
1353	STATUS	7:10	15:10 D						
1020	STORAGE	7:10	13: 1 D						
1027	STP	7:10	13: 7 D						
612	STRS	7:10	11: 7 D						
3116	SUMMARY	7:10	23:20 D						
4402	SWAP	7:10	29: 6 D						
2152	TAPE	7:10	19: 8 D						
2161	TJOB	7:10	19:22 D						
3	VASLGH	6:24 D	9: 6						

1404	WHICH	15:15 D	15:34	15:42
3655	XJSIAI	7:10	25:41 D	

```
G     <macro>                    OVERLAY   COMMO2                                          COMMO2.4

                                 COMMENT   'Copyright (C) Cray Research, Inc., 1984'            OVERLAY  1
```

```
                    ************************************************************************  COMM02.5
                    *                                                                      *  COMM02.6
                    *                    Station command overlay.                          *  COMM02.7
                    *                                                                      *  COMM02.8
                    *         END        Terminate the A processor station.  This requires *  COMM02.9
                    *                    freeing all devices, releasing all local and MOS  *  COMM02.10
                    *                    memory, and terminating all station tasks.        *  COMM02.11
                    *                                                                      *  COMM02.12
                    *         IACON      Bring up the interactive console.                 *  COMM02.13
                    *                                                                      *  COMM02.14
                    *         LOGOFF     Requests the protocol task to logoff the CRAY-1   *  COMM02.15
                    *                    and to terminate itself.                          *  COMM02.16
                    *                                                                      *  COMM02.17
                    *         LOGON      Starts the protocol task which initializes the    *  COMM02.18
                    *                    communication interface and logs the station onto *  COMM02.19
                    *                    the CRAY-1.                                       *  COMM02.20
                    *                                                                      *  COMM02.21
                    ************************************************************************  COMM02.22

6   <macro>                              REGDEFS    ,(PAR,BASE,CMDOFF,CMDLEN,OFFSET,COUNT),  COMM02.24
                                                    (LOCK,ERROR,ID,LIST,POINTER)              COMM02.25
                410     %S1       EQUALS    $REGORG                                                $REG    3
                411     %S2       EQUALS    $REGORG                                                $REG    3
                412     %S3       EQUALS    $REGORG                                                $REG    3
                413     %S4       EQUALS    $REGORG                                                $REG    3
                414     %S5       EQUALS    $REGORG                                                $REG    3
                415     %T1       EQUALS    $REGORG                                                $REG    3
                416     %T2       EQUALS    $REGORG                                                $REG    3
                417     %T3       EQUALS    $REGORG                                                $REG    3
                420     %T4       EQUALS    $REGORG                                                $REG    3
                421     %T5       EQUALS    $REGORG                                                $REG    3
                422     %T6       EQUALS    $REGORG                                                $REG    3
                423     %W1       EQUALS    $REGORG                                                $REG    3
                424     %W2       EQUALS    $REGORG                                                $REG    3
                425     %W3       EQUALS    $REGORG                                                $REG    3
                426     %W4       EQUALS    $REGORG                                                $REG    3
                427     %W5       EQUALS    $REGORG                                                $REG    3
                430     PAR       EQUALS    $REGORG                                                $REG    3
                431     BASE      EQUALS    $REGORG                                                $REG    3
                432     CMDOFF    EQUALS    $REGORG                                                $REG    3
                433     CMDLEN    EQUALS    $REGORG                                                $REG    3
```

```
434     OFFSET   EQUALS   $REGORG                                          $REG    3
435     COUNT    EQUALS   $REGORG                                          $REG    3
436     LOCK     EQUALS   $REGORG                                          $REG    3
437     ERROR    EQUALS   $REGORG                                          $REG    3
440     ID       EQUALS   $REGORG                                          $REG    3
441     LIST     EQUALS   $REGORG                                          $REG    3
442     POINTER  EQUALS   $REGORG                                          $REG    3

                 EXT      %INTRACT                                   COMM02.27
                 EXT      CF$SCROL                                   COMM02.28
                 EXT      CF$WRITE                                   COMM02.29
                 EXT      CLI$IAC                                    COMM02.30
                 EXT      CLI$TERM                                   COMM02.31
                 EXT      CP$END                                     COMM02.32
                 EXT      CP$IACON                                   COMM02.33
                 EXT      CP$LOGOF                                   COMM02.34
                 EXT      DESC$LOG                                   COMM02.35
                 EXT      EC$ASTR                                    COMM02.36
                 EXT      EC$IAIOP                                   COMM02.37
                 EXT      EC$LOG                                     COMM02.38
                 EXT      EC$LOGIN                                   COMM02.39
                 EXT      EC$OFF                                     COMM02.40
                 EXT      PRO$INIT                                   COMM02.41
```

```
                        *       Lock the operator interface.                                    COMM02.43
                                                                                                COMM02.44
     6    <macro>                ADDRESS   R!LOCK,SS@OP,R!%STAT                                COMM02.45
    11    <macro>                LOCK      R!LOCK              . Lock the operator interface.    COMM02.46


                        *       Process the appropriate command.                                COMM02.48
                                                                                                COMM02.49
    40    <macro>                $IF       (R!PAR = CP$END)    . If END command.               COMM02.50
                                                                                .>>>>> 0        $IF      1
                                                                                                COMM02.51
    45    <macro>                  GET     R!%W1,SS@NUM,R!%STAT . Get number of active consoles. COMM02.52
    47    020423  013001  112206   R = LOGOFF, R!%W1 = 1       . If no other consoles logged on. COMM02.53
    52    <macro>                  UNLOCK  R!LOCK                                              COMM02.54
    71    <macro>                  GOTO    CLINIT,(CLI$TERM)   . Shut it down.                 COMM02.55
                                                                                                COMM02.56
   105    <macro>                $ELSEIF (R!PAR = CP$IACON)                                    COMM02.57
   113    <macro>                  $IF (R!%INTRACT = 0)                                        COMM02.58
                                                                                .>>>>> 1        $IF      1
                        *            The interactive concentrator is not active.                COMM02.59
   116    <macro>                    CALL      ERROR,(EC$IAIOP,0,0,0)                          COMM02.60
   141    <macro>                  $ELSE                                                       COMM02.61
   142    <macro>                    UNLOCK    R!LOCK                                          COMM02.62
   161    <macro>                    GOTO      CLINIT,(CLI$IAC)                                COMM02.63
   175    <macro>                  $ENDIF                                                      COMM02.64
                                                                                .<<<<< 1        $ENDIF   1
                                                                                                COMM02.65
   175    <macro>                $ELSEIF   (R!PAR = CP$LOGOF)  . If LOGOFF command.            COMM02.66
                                                                                                COMM02.67
   203    072054                   R = LOGOFF                                                  COMM02.68
                                                                                                COMM02.69
   204    <macro>                $ELSE                         . If LOGON command.             COMM02.70
                                                                                                COMM02.71
   205    072215                   R = LOGON                                                   COMM02.72
                                                                                                COMM02.73
   206    <macro>                $ENDIF                                                        COMM02.74
                                                                                .<<<<< 0        $ENDIF   1
                                                                                                COMM02.75
   206    <macro>                UNLOCK    R!LOCK              . Release the interface.          COMM02.76
   225    <macro>                ADDRESS   R!%W1,$R@DIS,R!%CLI                                 COMM02.77
```

```
250     <macro>                 SIGNAL    R!%W1,1          . Request a display refresh.     COMM02.78
251     <macro>                 RETURN                                                      COMM02.79
```

```
                                 ***********************************************************************  COMM02.81
                                 *                                                                     *  COMM02.82
                                 *        LOGOFF command.                                              *  COMM02.83
                                 *                                                                     *  COMM02.84
                                 ***********************************************************************  COMM02.85
                                                                                                          COMM02.86
 257                             LOGOFF   *                                                               COMM02.87
                                                                                                          COMM02.88
                                 *        Output an error message if not logged on.                       COMM02.89
                                                                                                          COMM02.90
 257    <macro>                          ADDRESS   R!%W1,SS@FLG,R!%STAT                                  COMM02.91
 262    <macro>                          GETBIT    A,STA$LOG,R!%W1                                       COMM02.92
 275    <macro>                          $IF       (A = 0)              . If not logged on.              COMM02.93
                                                                                          .>>>>> 0          $IF      1
                                                                                                          COMM02.94
 277    <macro>                            $IF     (R!PAR # CP$END)     . If not end command.            COMM02.95
                                                                                          .>>>>> 1          $IF      1
 304    <macro>                              CALL  ERROR,(EC$OFF,0,0,0) .Output an error message         COMM02.96
 327    <macro>                            $ENDIF                                                        COMM02.97
                                                                                          .<<<<< 1          $ENDIF   1


                                 *        Send termination request to the protocol task and wait for a    COMM02.99
                                 *        response.                                                       COMM02.100
                                                                                                          COMM02.101
 327    <macro>                          $ELSE                        . If logged on.                  COMM02.102
                                                                                                          COMM02.103
                                 *          CHECK FOR LOGOFF IN PROGRESS                                  COMM02.104
                                                                                                          COMM02.105
 330    <macro>                            GETBIT A,STA$OFF,R!%W1      .LOGOFF IN PROGRESS FLAG          COMM02.106
 343   127000 /001252                      P = ERREI, A # 0            .IF SO                            COMM02.107
                                                                                                          COMM02.108
                                                                                                          COMM02.109
 345    <macro>                            STORE   1,SS@PRO,R!%STAT    . Set termination request flag.   COMM02.110
 352    <macro>                            ADDRESS R!%W1,SS@REQ,R!%STAT                                  COMM02.111
 355    <macro>                            SIGNAL  R!%W1,1             . Activate protocol task.         COMM02.112
 376    <macro>                            ADDRESS R!%W1,SS@PRO,R!%STAT                                  COMM02.113
 401    <macro>                            PUSH    R!%W1+1             . Wait for termination response.  COMM02.114
                                                                                                          COMM02.115
 421    <macro>                          $ENDIF                                                          COMM02.116
```

```
                                                    .<<<<< 0          $ENDI.
                                                                  COMM02.117
421   001000                EXIT                                  COMM02.118
```

```
                                 ********************************************************************  COMM02.120
                                 *                                                                  *  COMM02.121
                                 *        LOGON command.                                            *  COMM02.122
                                 *                                                                  *  COMM02.123
                                 ********************************************************************  COMM02.124
                                                                                                       COMM02.125
422                              LOGON    *                                                           COMM02.126
                                                                                                       COMM02.127
                                 *        Output an error message if already logged on.               COMM02.128
                                                                                                       COMM02.129
422   014000 /000000  024437             R!ERROR = EC$LOG                                             COMM02.130
425    <macro>                           ADDRESS   R!%W1,SS@FLG,R!%STAT                               COMM02.131
430    <macro>                           GETBIT    A,STA$LOG,R!%W1                                    COMM02.132
443    <macro>                           $IF       (A = 0)              . ******************** LOGGED ON  COMM02.133
                                                                                          .>>>>> 0          $IF      1


                                 *        Establish the logon ID and TID.                             COMM02.135
                                                                                                       COMM02.136
446    <macro>                           GETSTACK  PAR$COM,R!LIST       . Buffer for returned info.   COMM02.137
470    <macro>                           CALL      DESCRIBE,(R!BASE,R!OFFSET,R!COUNT,DESC$LOG,R!LIST,RO=  COMM02.138
                                 ,R!OFFSET,RO=R!COUNT)                                                 COMM02.139
521   024437                             R!ERROR = A                                                  COMM02.140
                                                                                                       COMM02.141
522    <macro>                           $IF       (A # 0)              . If parameter error.         COMM02.142
                                                                                          .>>>>> 1          $IF      1
524    <macro>                             FRESTACK                                                   COMM02.143
                                                                                          .>>>>> 2          $IF      3
                                                                                          .<<<<< 2          $ENDIF   3
542   075000 /001341                       P = DONE                                                   COMM02.144
544    <macro>                           $ELSE                          . If input is valid.          COMM02.145
545    <macro>                             GET       R!ID,SS@DID,R!%STAT . Get the default ID.         COMM02.146
552    <macro>                             GET       R!%W1,VA@TYP,R!LIST                               COMM02.147
                                                                                                       COMM02.148
555    <macro>                             $IF       (R!%W1 = VAL$ID)   . If ID entered with command.  COMM02.149
                                                                                          .>>>>> 2          $IF      1
560    <macro>                               GET     R!ID,VA@PR1,R!LIST                                COMM02.150
565    <macro>                             $ENDIF                                                     COMM02.151
                                                                                          .<<<<< 2          $ENDIF   1
                                                                                                       COMM02.152
```

```
 567   <macro>                    PUT     R!ID,SS@T...!%STAT  . Establish the logon ID.          COMM02.153
 572   <macro>                    GET     R!%W1,VA@NXT,R!LIST                                     COMM02.154
 575   020441  022423  024441     R!LIST = R!LIST+R!%W1       . Advance to the next parameter.    COMM02.155
 600   <macro>                    GET     R!%W1,VA@TYP,R!LIST                                     COMM02.156
 603   <macro>                    ADDRESS R!%W4,SS@TID,R!%STAT                                    COMM02.157
                                                                                                   COMM02.158
 606   <macro>                    $IF     (R!%W1 = VAL$STNG)  . If TID entered with command.      COMM02.159
                                                                                .>>>>> 2          $IF      1
 612   020431  024423               R!%W1 = R!BASE            . Get the TID location.             COMM02.160
 614   <macro>                      GET   R!%W2,VA@PR1,R!LIST                                     COMM02.161
 621   <macro>                      GET   R!%W3,VA@PR2,R!LIST                                     COMM02.162
 626   <macro>                      CLEAR START=R!%W4,COUNT=4 . Clear the field.                  COMM02.163
                                                                                                   COMM02.164
 641   <macro>                    $ELSE                       . Use the default TID.              COMM02.165
 642   <macro>                      ADDRESS R!%W1,SS@DTI,R!%STAT                                  COMM02.166
 645   010000  024424               R!%W2 = 0                                                     COMM02.167
 647   010010  024425               R!%W3 = D'8                                                   COMM02.168
 651   <macro>                    $ENDIF                                                          COMM02.169
                                                                                .<<<<< 2          $ENDIF   1
                                                                                                   COMM02.170
 651   <macro>                    MOVECHAR R!%W1,R!%W2,R!%W4,0,R!%W3 . Establish the TID.         COMM02.171
 737   <macro>                    FRESTACK                                                        COMM02.172
                                                                                .>>>>> 2          $IF      3
                                                                                .<<<<< 2          $ENDIF   3
 755   <macro>                  $ENDIF                                                            COMM02.173
                                                                                .<<<<< 1          $ENDIF   1


                          *       Create the PROTOCOL task.  Wait for a response to see if the     COMM02.175
                          *       protocol task initialization and logon processing was success-   COMM02.176
                          *       ful.                                                             COMM02.177
                                                                                                   COMM02.178
 755   <macro>                  GETSTACK  LM@LEN@P,R!POINTER  . Buffer for START segment.         COMM02.179
 777   <macro>                  CREATE    PROTINIT,(PRO$INIT,R!POINTER),PRI=PR$PROT               COMM02.180
1022   024437                   R!ERROR = A                                                       COMM02.181
1023   <macro>                  $IF       (A = 0)             . ************************ CREATE   COMM02.182
                                                                                .>>>>> 1          $IF      1
                                                                                                   COMM02.183
1025   <macro>                  ADDRESS   R!%W1,SS@PRO,R!%STAT                                    COMM02.184
1030   010000  034423           (R!%W1) = 0                   . Clear logoff request word.        COMM02.185
```

```
1032    <macro>                 PUSH      R!%W1+1            . Wait for the logon response.     COMM02.186
1052   024437                   R!ERROR = A                                                     COMM02.187
1053    <macro>                 $IF       (A = 0)            . ************************* LOGON  COMM02.188
                                                                                .>>>>> 2            $IF      1


                        *       The logon was successful.  Display the system name and genera-   COMM02.190
                        *       tion date which are returned in the start message segment.       COMM02.191
                                                                                                 COMM02.192
1055   010020  024423           R!%W1 = LM@SYS@B             . Get pointers to system name.     COMM02.193
1057   010010  024424           R!%W2 = LM@SYS@C                                                COMM02.194
1061   072261                   R = XLATE                                                       COMM02.195
1062    <macro>                 GET       R!%W1,$C@LST,R!%CLI                                   COMM02.196
1067    <macro>                 CALL      CONSL,(CF$WRITE,R!%W1,1,R!POINTER,LM@SYS@B,LM@SYS@C)  COMM02.197
1116   010030  024423           R!%W1 = LM@DAT@B             . Get pointers to date.            COMM02.198
1120   010010  024424           R!%W2 = LM@DAT@C                                                COMM02.199
1122   072220                   R = XLATE                                                       COMM02.200
1123    <macro>                 GET       R!%W1,$C@LST,R!%CLI                                   COMM02.201
1130    <macro>                 CALL      CONSL,(CF$WRITE,R!%W1,3+LM@SYS@C,R!POINTER,LM@DAT@B,L COMM02.202
                        ,M@DAT@C)                                                                COMM02.203
1160    <macro>                 CALL      CONSL,(CF$SCROL)                                      COMM02.204
1175    <macro>                 FRESTACK                     . Release segment buffer.          COMM02.205
                                                                                .>>>>> 3            $IF      3
                                                                                .<<<<< 3            $ENDIF   3
                                                                                                 COMM02.206
1213   070126                   P = DONE                                                        COMM02.207


1214    <macro>                 $ENDIF                       . ************************* LOGON  COMM02.209
                                                                                .<<<<< 2            $ENDIF   1
                                                                                                 COMM02.210
1214    <macro>                 $ENDIF                       . ************************* CREATE COMM02.211
                                                                                .<<<<< 1            $ENDIF   1
                                                                                                 COMM02.212
1214    <macro>                 FRESTACK                     . Release segment buffer.          COMM02.213
                                                                                .>>>>> 1            $IF      3
                                                                                .<<<<< 1            $ENDIF   3
                                                                                                 COMM02.214
```

```
1232    <macro>                         $ENDIF                          . ******************** LOGGED ON  COMM02.215
                                                                                              .<<<<< 0          $ENDIF   1
                                                                                                                COMM02.216
                                                                                                                COMM02.217
                                *       CHECK FOR LOGOFF IN PROGRESS DUE TO ACTIVE STREAM                       COMM02.218
                                                                                                                COMM02.219
1232    <macro>                         ADDRESS R!%W1,SS@FLG,R!%STAT   .STATION FLAG WORD                      COMM02.220
1235    <macro>                         GETBIT A,STA$OFF,R!%W1         .SEE IF LOGOFF IN PROGRESS              COMM02.221
1250    <macro>                         $IF (A # 0)                    .IF SO                                  COMM02.222
                                                                                              .>>>>> 0          $IF      1
1252                            ERRET     *                                                                     COMM02.223
1252    014000 /000000  024437            R!ERROR = EC$LOGIN             .Logoff in progress error code          COMM02.224
                                                                                                                COMM02.225
                                          * CHECK FOR ACTIVE STREAM(S)                                          COMM02.226
                                                                                                                COMM02.227
1255    <macro>                           GET R!%W2,SS@IST,R!%STAT       .COUNT OF STREAMS                       COMM02.228
1262    <macro>                           GET R!%W3,SS@OST,R!%STAT       .                                       COMM02.229
1267    020425  022424  024425            R!%W3 = R!%W3 + R!%W2          .                                       COMM02.230
1272    010000  024426                    R!%W4 = 0                      .CLEAR ACTIVE FLAG                      COMM02.231
1274    <macro>                           ADDRESS R!%W1,PT@TAB,R!%PROT .FIRST STREAM TABLE                       COMM02.232
1277    <macro>                           $UNTIL (R!%W3 = 0)             .CHECK EACH TABLE                       COMM02.233
1301    <macro>                             GET R!%W4,IO@FLG,R!%W1       .ACTIVE FLAG                            COMM02.234
1306    <macro>                             ADDRESS R!%W1,IO@LEN,R!%W1 .NEXT TABLE                               COMM02.235
1310    027425                              R!%W3 = R!%W3 - 1            .DECREMENT COUNT                        COMM02.236
1311    <macro>                           $ENDTIL                                                               COMM02.237
1312    020426  102004  014000            R!ERROR = EC$ASTR, R!%W4 # 0 .Set stream active error code           COMM02.238
       /000000  024437
1317    <macro>                         $ENDIF                                                                  COMM02.239
                                                                                              .<<<<< 0          $ENDIF   1
                                                                                                                COMM02.240
                                *       Output an error message explaining why the logon failed.                 COMM02.241
                                                                                                                COMM02.242
1317    <macro>                         CALL      ERROR,(R!ERROR,0,0,0) . Output an error message.              COMM02.243
                                                                                                                COMM02.244
1341                            DONE    *                                                                       COMM02.245
1341    001000                          EXIT                                                                    COMM02.246
```

```
                        *       Translate illegal characters to spaces.                 COMM02.248
                        *                                                               COMM02.249
                        *       R!%W1     Byte offset of the character string (R!POINTER is  COMM02.250
                        *                 assumed to be the base address).              COMM02.251
                        *       R!%W2     Character string length.                      COMM02.252
                        *                                                               COMM02.253
                        *       On exit, R!%W1 and R!%W2 are undefined.                 COMM02.254
                                                                                        COMM02.255
1342                    XLATE   *                                                       COMM02.256
                                                                                        COMM02.257
1342   <macro>                  $UNTIL    (R!%W2 = 0)          . Until all characters scanned.  COMM02.258
1344   <macro>                    GETBYTE R!%W3,R!%W1,R!POINTER . Grab the next character.  COMM02.259
1366   020425  013040  101003     R!%W3 = ' 'R, R!%W3 < CC$FIRST . Translate illegal characters  COMM02.260
       010040  024425
1373   020425  013177  100004     R!%W3 = ' 'R, R!%W3 > CC$LAST . to spaces.              COMM02.261
       102003  010040  024425
1401   <macro>                    PUTBYTE R!%W3,R!%W1,R!POINTER,INC=1                     COMM02.262
1432   027424                     R!%W2 = R!%W2-1              . Decrement the character count.  COMM02.263
1433   <macro>                  $ENDTIL                                                  COMM02.264
                                                                                        COMM02.265
1434   001000                   EXIT                                                    COMM02.266


                                END                                                     COMM02.268
```

20	$C@LST@N	$APTEXT	10:15	10:15 F	1[illegible]	10:20 F					
3	$C@LST@P	$APTEXT	10:15 F	10:15	10:20 F	10:20					
0	$C@LST@S	$APTEXT	10:15	10:15 F	10:20	10:20 F					
55	$R@DIS@P	$APTEXT	4:42								
0X	%ACTIVE		4: 4 D	4: 4	6:40 D	6:40	10: 1 D	10: 1			
0X	%B		1: 4 D	6:33 B	8:15 B	8:32 B					
406	%BASE	$APTEXT	8:30	9:23	10:25	10:40					
402	%CLI	$APTEXT	4:42	10:15	10:20						
0X	%EX		4: 4 D	4: 4	4:16 D	4:16	4:17 D	4:17	4:23 D	4:23	
			4:25 D	4:25	4:26 D	4:26	4:41 D	4:41	5: 1 D	5: 1	
			5: 2 D	5: 2	6:18 D	6:18	6:38 D	6:38	6:40 D	6:40	
			8:24 D	8:24	9:36 D	9:36	10: 1 D	10: 1	10:16 D	10:16	
			10:22 D	10:22	10:23 D	10:23	11:33 D	11:33			
10	%GBLNUM	$APTEXT	2:22 F	2:22							
400	%GBLREG	$APTEXT	2:22								
0X	%INTRACT		3:10 D	4:21							
405	%LIMIT	$APTEXT	8:22	8:30	9:23	9:35	10:25	10:40			
401	%PROT	$APTEXT	11:20								
0X	%PUNT		8:22 D	8:22	8.29 D	8:30	9:22 D	9:23	9:35 D	9:35	
			10:24 D	10:25	10:39 D	10:40					
404	%STACL	$APTEXT	8:22	8:29	8:30	9:22	9:23	9:35	10:24	10:25	
			10:39	10:40							
400	%STAT	$APTEXT	4: 3	4:14	6:11	6:36	6:37	6:39	8:12	8:34	
			9: 1	9: 5	9:15	9:41	11: 7	11:16	11:17		
177777	AD$NODLM	$APTEXT	9:36								
20	AD@RC@N	$APTEXT	4: 4	4: 4 F	6:40	6:40 F	10: 1	10: 1 F			
13	AD@RC@P	$APTEXT	4: 4 F	4: 4	6:40 F	6:40	10: 1 F	10: 1			
0	AD@RC@S	$APTEXT	4: 4	4: 4 F	6:40	6:40 F	10: 1	10: 1 F			
40	CC$FIRST	$APTEXT	12:13								
177	CC$LAST	$APTEXT	12:15								
0X	CF$SCROL		3:11 D	10:23							
0X	CF$WRITE		3:12 D	10:16	10:22						
0X	CLI$TAC		3:13 D	4:26							
0X	CLI$TERM		3:14 D	4:17							
0X	CP$END		3:15 D	4:12	6:17						
0X	CP$TACON		3:16 D	4:19							
0X	CP$LOGOT		3:17 D	4:30							
0X	DESC$LOG		3:18 D	8:24							
0X	EC$ASTR		3:19 D	11:26							
0X	EC$TATOP		3:20 D	4:23							
0X	EC$LOG		3:21 D	8:11							
0X	EC$LOGIN		3:22 D	11:12							

0X	ECSOFF		3:23 D	6:18					
50	F$CALL	$APTEXT	4:23	6:18	8:24	10:16	10:22	10:23	11:33
55	F$CREATE	$APTEXT	9:36						
51	F$GOTO	$APTEXT	4:17	4:26					
2	F$POP	$APTEXT	4:16	4:25	4:41	5: 1	6:38		
1	F$PUSH	$APTEXT	4: 4	6:40	10: 1				
52	F$RETURN	$APTEXT	5: 2						
0	F$SPRT	$APTEXT	4: 4	6:40	10: 1				
20	IO@FLG@N	$APTEXT	11:22	11:22 F					
1	IO@FLG@P	$APTEXT	11:22 F	11:22					
0	IO@FLG@S	$APTEXT	11:22	11:22 F					
64	IO@LEN@P	$APTEXT	11:23						
30	LM@DAT@B	$APTEXT	10:17	10:22					
10	LM@DAT@C	$APTEXT	10:18	10:22					
30	LM@LEN@P	$APTEXT	9:35						
20	LM@SYS@B	$APTEXT	10:12	10:16					
10	LM@SYS@C	$APTEXT	10:13	10:16	10:22				
0X	O$CLINIT		4:17 D	4:17	4:26 D	4:26			
0X	O$COMM02		1: 4 D	1: 4					
0X	O$CONSL		10:16 D	10:16	10:22 D	10:22	10:23 D	10:23	
0X	O$DESCRI		8:24 D	8:24					
0X	O$ERROR		4:23 D	4:23	6:18 D	6:18	11:33 D	11:33	
0X	O$PROTIN		9:36 D	9:36					
17	OV@NUM@N	$APTEXT	1: 4						
1	OV@TYP@N	$APTEXT	1: 4						
24	PARSCOM	$APTEXT	8:22						
5	PRSPROT	$APTEXT	9:36						
0X	PROSINIT		3:24 D	9:36					
13	PISSTACK	$APTEXT	8:22	8:30	9:23	9:35	10:25	10:40	
222	PI@TAB@P	$APTEXT	11:20						
11	SM@FST@N	$APTEXT	1: 4						
7	SM@NUM@N	$APTEXT	1: 4						
14	SM@REG@P	$APTEXT	8:24						
20	SS@DID@N	$APTEXT	8:34	8:34 F					
11	SS@DID@P	$APTEXT	8:34 F	8:34					
0	SS@DID@S	$APTEXT	8:34	8:34 F					
12	SS@DIT@P	$APTEXT	9:15						
35	SS@FLG@P	$APTEXT	6:11	8:12	11: 7				
20	SS@ID@N	$APTEXT	9: 1 F						
4	SS@ID@P	$APTEXT	9: 1 F	9: 1					
0	SS@ID@S	$APTEXT	9: 1 F						
20	SS@ISTO@N	$APTEXT	11:16	11:16 F					

16	SS@ISI@P	$APTEXT	11:16 F	11:16							
0	SS@ISI@S	$APTEXT	11:16	11:16 F							
20	SS@NUM@N	$APTEXT	4:14	4:14 F							
0	SS@NUM@P	$APTEXT	4:14 F								
0	SS@NUM@S	$APTEXT	4:14	4:14 F							
67	SS@OPOP	$APTEXT	4: 3								
20	SS@OSI@N	$APTEXT	11:17	11:17 F							
17	SS@OSI@P	$APTEXT	11:17 F	11:17							
0	SS@OSI@S	$APTEXT	11:17	11:17 F							
20	SS@PRO@N	$APTEXT	6:36 F								
75	SS@PRO@P	$APTEXT	6:36 F	6:36	6:39	9:41					
0	SS@PRO@S	$APTEXT	6:36 F								
100	SS@RJQ@P	$APTEXT	6:37								
5	SS@TID@P	$APTEXT	9: 5								
0	STASLOG	$APTEXT	6:12	8:13							
2	STASOFF	$APTEXT	6:32	11: 8							
377	VA@NXI@M	$APTEXT	9: 2								
10	VA@NXI@N	$APTEXT	9: 2								
0	VA@NXI@P	$APTEXT	9: 2 F								
10	VA@NXI@S	$APTEXT	9: 2	9: 2 F							
20	VA@PRI@N	$APTEXT	8:39	8:39 F	9:10	9:10 F					
1	VA@PRI@P	$APTEXT	8:39 F	8:39	9:10 F	9:10					
0	VA@PRI@S	$APTEXT	8:39	8:39 F	9:10	9:10 F					
20	VA@PR2@N	$APTEXT	9:11	9:11 F							
2	VA@PR2@P	$APTEXT	9:11 F	9:11							
0	VA@PR2@S	$APTEXT	9:11	9:11 F							
10	VA@TYP@N	$APTEXT	8:35	8:35 F	9: 4	9: 4 F					
0	VA@TYP@P	$APTEXT	8:35 F	9: 4 F							
0	VA@TYP@S	$APTEXT	8:35	8:35 F	9: 4	9: 4 F					
0	VALSID	$APTEXT	8:38								
4	VALSSING	$APTEXT	9: 8								

SYMBOL QUALIFIER IS COMM02

21	$SEO		8:24 D	8:24							
1	$CONST		6:36 D	6:36 F	6:36						
443	$REGDEG		2:22 D	2:22 F	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D	
			2:26	2:26 D	2:27	2:27 D	2:27 F	2:28	2:28 D	2:29	
			2:29 D	2:30	2:30 D	2:31	2:31 D	2:32	2:32 D	2:33	
			2:33 D	2:33 F	2:34	2:34 D	2:35	2:35 D	2:36	2:36 D	
			2:37	2:37 D	2:38	2:38 D	2:38 F	2:39	2:39 D	2:40	
			2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1	3: 1 D	3: 2	

		3: 2 D	3: 2 F	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D
		3: 6	3: 6 D	3: 7	3: 7 D	3: 7 F	4: 4	4: 4 F	4:16
		4:17	4:23	4:23 F	4:25	4:26	4:41	5: 1	5: 2
		6:18	6:18 F	6:38	6:40	6:40 F	8:24	8:24 F	9:36
		10: 1	10: 1 F	10:16	10:16 F	10:22	10:22 F	10:23	10:23 F
		11:33	11:33 F						
6	%NP	1: 4	3: 2 D						
430	%P	1: 4	2:38 D	3: 2	4: 4 F	4: 4	4:23 F	4:23	6:18 F
		6:18	6:40 F	6:40	8:24 F	8:24	10: 1 F	10: 1	10:16 F
		10:16	10:22 F	10:22	10:23 F	10:23	11:33 F	11:33	
410	%S1	2:22	2:23 D	4: 4	6:36	6:40	8:34	8:39	9: 1
		9:10	9:11	9:21	10: 1	10:15	10:20	11:16	11:17
		11:22	12:12	12:17					
411	%S2	2:22	2:24 D						
412	%S3	2:22	2:25 D						
413	%S4	2:22	2:26 D						
414	%S5	2:22	2:27 D						
415	%T1	2:28 D	4: 4	4:16	4:25	4:41	5: 1	6:12	6:32
		6:38	8:13	8:22	9:12	9:21	9:35	11: 8	12:12
		12:17							
416	%T2	2:29 D	9:12	9:21	12:17				
417	%T3	2:30 D	9:21						
420	%T4	2:31 D	9:21						
421	%T5	2:32 D	9:21						
422	%T6	2:33 D	9:21						
423	%W1	2:34 D	4:14	4:15	4:42	5: 1	6:11	6:12	6:32
		6:37	6:38	6:39	6:40	8:12	8:13	8:35	8:38
		9: 2	9: 3	9: 4	9: 8	9: 9	9:15	9:21	9:41
		9:42	10: 1	10:12	10:15	10:16	10:17	10:20	10:22
		11: 7	11: 8	11:20	11:22	11:23	12:12	12:17	
424	%W2	2:35 D	9:10	9:16	9:21	10:13	10:18	11:16	11:18
		12:11	12:18						
425	%W3	2:36 D	9:11	9:17	9:21	11:17	11:18	11:21	11:24
		12:12	12:13	12:14	12:15	12:16	12:17		
426	%W4	2:37 D	9: 5	9:12	9:21	11:19	11:22	11:26	
427	%W5	2:38 D							
431	BASE	2:40 D	8:24	9: 9					
433	CMDLEN	2:42 D							
432	CMDOFF	2:41 D							
435	COUNT	3: 2 D	8:24	8:24 F					
1341	DONE	8:32	10:28	11:35 D					
1252	ERRET	6:33	11:11 D						

437	ERROR	3: 4 D	8:11	8:25	9:37	10: 2	11:12	11:27	11:33
440	ID	3: 5 D	8:34	8:39	9: 1				
441	LIST	3: 6 D	8:22	8:24	8:35	8:39	9: 2	9: 3	9: 4
		9:10	9:11						
436	LOCK	3: 3 D	4: 3	4: 4	4:16	4:25	4:41		
257	LOGOFF	4:15	4:32	6: 7 D					
422	LOGON	4:36	8: 7 D						
434	OFFSET	3: 1 D	8:24	8:24 F					
430	PAR	2:39 D	4:12	4:19	4:30	6:17			
442	POINTER	3: 7 D	9:35	9:36	10:16	10:22	12:12	12:17	
1342	XLATE	10:14	10:19	12: 9 D					

```
0    <macro>                    OVERLAY   ISPDRV                                          ISPDRV.4

                                COMMENT   'Copyright (C) Cray Research, Inc., 1984'              OVERLAY  1
```

```
                    ***************************  ****************************************   ISPDRV.5
                    *                                                                     *   ISPDRV.6
                    *  THE ISPDRV OVERLAY COMPRISES THE DRIVER FOR ISP CHANNEL DEVICES    *   ISPDRV.7
                    *  UNDER THE USER CHANNEL SHELL STRUCTURE.                            *   ISPDRV.8
                    *  IT IS CALLED BY THE SHELL TO PROCESS REQUESTS FROM THE COS TASK    *   ISPDRV.9
                    *  TIM FOR I/O TO THE INTEGRATED SUPPORT PROCESSOR.                   *   ISPDRV.10
                    *                                                                     *   ISPDRV.11
                    ***********************************************************************   ISPDRV.12
                                                                                              ISPDRV.13
6   <macro>                     REGDEFS ,(IBL),                                               ISPDRV.14
                                        (CMD,UST,BUF,SIZ,END,CHN,L1U,L1L,L2U,L2L,TLB,TO2,TMO)  ISPDRV.15
                410     %S1     EQUALS  $REGORG                                                   $REG    3
                411     %S2     EQUALS  $REGORG                                                   $REG    3
                412     %S3     EQUALS  $REGORG                                                   $REG    3
                413     %S4     EQUALS  $REGORG                                                   $REG    3
                414     %S5     EQUALS  $REGORG                                                   $REG    3
                415     %T1     EQUALS  $REGORG                                                   $REG    3
                416     %T2     EQUALS  $REGORG                                                   $REG    3
                417     %T3     EQUALS  $REGORG                                                   $REG    3
                420     %T4     EQUALS  $REGORG                                                   $REG    3
                421     %T5     EQUALS  $REGORG                                                   $REG    3
                422     %T6     EQUALS  $REGORG                                                   $REG    3
                423     %W1     EQUALS  $REGORG                                                   $REG    3
                424     %W2     EQUALS  $REGORG                                                   $REG    3
                425     %W3     EQUALS  $REGORG                                                   $REG    3
                426     %W4     EQUALS  $REGORG                                                   $REG    3
                427     %W5     EQUALS  $REGORG                                                   $REG    3
                430     TBL     EQUALS  $REGORG                                                   $REG    3
                431     CMD     EQUALS  $REGORG                                                   $REG    3
                432     UST     EQUALS  $REGORG                                                   $REG    3
                433     BUF     EQUALS  $REGORG                                                   $REG    3
                434     SIZ     EQUALS  $REGORG                                                   $REG    3
                435     END     EQUALS  $REGORG                                                   $REG    3
                436     CHN     EQUALS  $REGORG                                                   $REG    3
                437     L1U     EQUALS  $REGORG                                                   $REG    3
                440     L1L     EQUALS  $REGORG                                                   $REG    3
                441     L2U     EQUALS  $REGORG                                                   $REG    3
                442     L2L     EQUALS  $REGORG                                                   $REG    3
                443     TLB     EQUALS  $REGORG                                                   $REG    3
                444     TO2     EQUALS  $REGORG                                                   $REG    3
                445     TMO     EQUALS  $REGORG                                                   $REG    3
                                                                                              ISPDRV.16
```

```
*       REGISTERS:                                                      ISPDRV.17
*                                                                       ISPDRV.18
*       CMD = Last command received from the shell.                     ISPDRV.19
*             UC$OPN = Driver open.                                     ISPDRV.20
*             UC$CLS = Driver close.                                    ISPDRV.21
*             UC$RD  = Read data.                                       ISPDRV.22
*             UC$RDL = Read last data.                                  ISPDRV.23
*             UC$WRT = Write data.                                      ISPDRV.24
*             UC$WRTL = Write last data.                                ISPDRV.25
*                                                                       ISPDRV.26
*       UST = Driver status returned to the shell.                      ISPDRV.27
*             UC$CMPT = Request completed.                              ISPDRV.28
*             UC$CONT = Request partially completed - send more.        ISPDRV.29
*             UC$NOOP = Acknowledgement (returned as first reply        ISPDRV.30
*                       to a series of RD or WT requests if in          ISPDRV.31
*                       double-buffer mode.                             ISPDRV.32
*             UC$ERR and greater = error codes.                         ISPDRV.33
*                                                                       ISPDRV.34
*       BUF = Address of buffer received from shell.                    ISPDRV.35
*                                                                       ISPDRV.36
*       SIZ = Length of data transfer requested by shell.               ISPDRV.37
*                                                                       ISPDRV.38
*       END = 1 to terminate activity.                                  ISPDRV.39
*                                                                       ISPDRV.40
*       CHN = Channel number.                                           ISPDRV.41
*                                                                       ISPDRV.42
*       L1U,L1L = Anticipated length of current read block (words).     ISPDRV.43
*                                                                       ISPDRV.44
*       L2U,L2L = Anticipated length of next read block (words).        ISPDRV.45
*                                                                       ISPDRV.46
*       TLB = Transmitted or received length (bytes).                   ISPDRV.47
*                                                                       ISPDRV.48
*       TO2 = 1 during 2nd timeout of I/O operation.                    ISPDRV.49
*                                                                       ISPDRV.50
*       TMO = Nonzero (set to the timeout interval) after a timeout     ISPDRV.51
*             on an output channel.  Causes subsequent outputs to       ISPDRV.52
*             wait until the other side puts up a read.                 ISPDRV.53
                                                                        ISPDRV.54
                                                                        ISPDRV.55
        *                                                               ISPDRV.56
        * ERROR CODES RETURNED TO CALLER:                               ISPDRV.57
        *                                                               ISPDRV.58
```

```
                                                                                ISPDRV.59
6    <macro>          IE$TMO   NEXT     UC$ERR          .TIMEOUT ERROR CODE     ISPDRV.60
6    <macro>          IE$HWE   NEXT                     .CHANNEL HARDWARE ERROR ISPDRV.61
6    <macro>          IE$CTY   NEXT                     .ILLEGAL CHANNEL TYPE   ISPDRV.62
6    <macro>          IE$FCN   NEXT                     .ILLEGAL DRIVER FUNCTION CODE ISPDRV.63
6    <macro>          IE$LEN   NEXT                     .DATA LENGTH ERROR      ISPDRV.64
                                                                                ISPDRV.65
                               *                                                ISPDRV.66
                               * ISP CHANNEL HEADER FORMAT:                     ISPDRV.67
                               *                                                ISPDRV.68
                                                                                ISPDRV.69
6    <macro>          IH       TABLE    LE=24            .Length = 24 parcels    ISPDRV.70
                      *        FIELD    0                .Destination I.D.       ISPDRV.71
                      *        FIELD    1                .Source I.D.            ISPDRV.72
6    <macro>          IH@HLI   FIELD    2,0,8            .Header length in bytes (48) ISPDRV.73
                      *        FIELD    2,8,8            .Message number         ISPDRV.74
                      *        FIELD    3,0,8            . (unused)              ISPDRV.75
6    <macro>          IH@FC    FIELD    3,8,8            .Function code          ISPDRV.76
                 0     IHFC$DAT EQUALS   0                . Data header           ISPDRV.77
6    <macro>          IH@HCS   FIELD    4                .Header checksum        ISPDRV.78
6    <macro>          IH@HB1   FIELD    5,0,8            .Header block size (1st byte) ISPDRV.79
6    <macro>          IH@HB2   FIELD    5,8,8            .Header block size (2nd byte) ISPDRV.80
6    <macro>          IH@HB3   FIELD    6,0,8            .Header block size (3rd byte) ISPDRV.81
6    <macro>          IH@DB1   FIELD    6,8,8            .Data block size (1st byte) ISPDRV.82
6    <macro>          IH@DB2   FIELD    7,0,8            .Data block size (2nd byte) ISPDRV.83
6    <macro>          IH@DB3   FIELD    7,8,8            .Data block size (3rd byte) ISPDRV.84
                      *        FIELD    8,*              .Full word of ones      ISPDRV.85
                      *        FIELD    12,*             .Full word of zeros     ISPDRV.86
                      *        FIELD    16,*             .Diag. bit pattern (full word) ISPDRV.87
                      *        FIELD    20,*             .Header block checksum (full word ISPDRV.88
                                                                                ISPDRV.89
               600    DDB@LE   EQUALS   D'384            .Dummy data block length (words) ISPDRV.90
                                                                                ISPDRV.91
                               *                                                ISPDRV.92
                               *        APF@DRV WORD USAGE IN DRIVER-OPEN REQUEST ISPDRV.93
                               *                                                ISPDRV.94
                                                                                ISPDRV.95
6    <macro>          APF@XB   FIELD    28,0,1           .Extra-buffer flag      ISPDRV.96
                                                                                ISPDRV.97
                               *                                                ISPDRV.98
                               *        LAYOUT OF APDRV WORD RETURNED IN F-PACKET: ISPDRV.99
                               *                                                ISPDRV.100
```

```
                                                                                                ISPDRV.101
     6    <macro>                APF@FC   FIELD     28                  .Function code from link header   ISPDRV.102
     6    <macro>                APF@HZ   FIELD     29                  .Header size in words             ISPDRV.103
     6    <macro>                APF@DZ   FIELD     30                  .Data size in words (upper)       ISPDRV.104
     6    <macro>                APF@DL   FIELD     31                  .Data size in words (lower)       ISPDRV.105
                                                                                                ISPDRV.106
                                         *                                                      ISPDRV.107
                                         * INITIALIZE                                           ISPDRV.108
                                         *                                                      ISPDRV.109
                                                                                                ISPDRV.110
     6   010000  024435                  R!END = 0                                              ISPDRV.111
    10    <macro>                        GET R!CHN,UC@CHN,R!TBL                                 ISPDRV.112
                                                                                                ISPDRV.113
                                         *                                                      ISPDRV.114
                                         * MAIN LOOP:                                           ISPDRV.115
                                         *                                                      ISPDRV.116
                                                                                                ISPDRV.117
    13    <macro>                        $UNTIL (R!END # 0)                                     ISPDRV.118
                                                                                                ISPDRV.119
    15    <macro>                          WATCH R!TBL+UC@SDR@P,R!CMD  .GET NEXT COMMAND FROM SHELL   ISPDRV.120
                                                                                                ISPDRV.121
    61    <macro>                          $IF (R!CMD = UC$OPN)        .OPEN COMMAND              ISPDRV.122
                                                                                     .>>>>> 0        $IF      1
    65   072101                              R = DOOPN                                           ISPDRV.123
                                                                                                ISPDRV.124
    66    <macro>                          $ELSEIF (R!CMD = UC$CLS)    .CLOSE COMMAND             ISPDRV.125
    73   072125                              R = DOCLS                                           ISPDRV.126
                                                                                                ISPDRV.127
    74    <macro>                          $ELSEIF (R!CMD = UC$RD)     .READ COMMAND              ISPDRV.128
   101   072124                              R = DOIO                                            ISPDRV.129
                                                                                                ISPDRV.130
   102    <macro>                          $ELSEIF (R!CMD = UC$RDL)    .READ-LAST COMMAND         ISPDRV.131
   107   072116                              R = DOIO                                            ISPDRV.132
                                                                                                ISPDRV.133
   110    <macro>                          $ELSEIF (R!CMD = UC$WRT)    .WRITE COMMAND             ISPDRV.134
   115   072110                              R = DOIO                                            ISPDRV.135
                                                                                                ISPDRV.136
   116    <macro>                          $ELSEIF (R!CMD = UC$WRTL)   .WRITE-LAST COMMAND        ISPDRV.137
   123   072102                              R = DOIO                                            ISPDRV.138
                                                                                                ISPDRV.139
   124    <macro>                          $ELSEIF (R!CMD >= UC$DRV)   .DRIVER COMMAND            ISPDRV.140
   131   072341                              R = DODRV                                           ISPDRV.141
```

```
                                                                                ISPDRV.142
132    <macro>                    $ELSE                    .ELSE ILLEGAL COMMAND  ISPDRV.143
133   010043  024432                R!UST = IE$FCN                               ISPDRV.144
                                                                                ISPDRV.145
135    <macro>                    $ENDIF                                         ISPDRV.146
                                                                  .<<<<< 0            $ENDIF   1
                                                                                ISPDRV.147
135    <macro>                    SIGNAL R!TBL+UC@DSR@P,R!UST .RETURN STATUS TO SHELL  ISPDRV.148
                                                                                ISPDRV.149
157    <macro>                  $ENDTIL                                        ISPDRV.150
160    <macro>                  TERM                                           ISPDRV.151
                                                                                ISPDRV.152
```

```
                                 ********************************************************************   ISPDRV.154
                                 *                                                                  *   ISPDRV.155
                                 * THE DOOPN ROUTINE EXECUTES THE DRIVER OPEN FUNCTION FOR ISP      *   ISPDRV.156
                                 * CHANNELS.                                                        *   ISPDRV.157
                                 *                                                                  *   ISPDRV.158
                                 * DOOPN SETS THE UC@XBF FLAG ACCORDING TO THE SPECIFICATION IN THE *   ISPDRV.159
                                 * REQUEST.  WHEN THIS BIT IS SET, THE SHELL WILL MAINTAIN DOUBLE   *   ISPDRV.160
                                 * BUFFERS DURING UC$RD AND UC$WRT REQUESTS, TO ALLOW OVERLAP OF I/O *   ISPDRV.161
                                 * WITH SHELL AND DRIVER EXECUTION.                                 *   ISPDRV.162
                                 *                                                                  *   ISPDRV.163
                                 ********************************************************************   ISPDRV.164
                                                                                                        ISPDRV.165
166                              DOOPN    *                                                             ISPDRV.166
                                                                                                        ISPDRV.167
166   010000  024432                     R!UST = UC$CMPT                .ASSUME NO ERRORS              ISPDRV.168
170    <macro>                           GET R!%W1,UC@REQ,R!TBL         .ADDRESS OF REQUEST PACKET     ISPDRV.169
175    <macro>                           GET R!%W2,APF@XB,R!%W1         .1 IF 2-BUFFER I/O DESIRED     ISPDRV.170
203    <macro>                           PUT R!%W2,UC@XBF,R!TBL         .SET EXTRA-BUFFER FLAG         ISPDRV.171
210   010000  024437                     R!L1U = 0                      .CLEAR 1ST DATA LENGTH         ISPDRV.172
212   024440                             R!L1L = A                                                     ISPDRV.173
213   024441                             R!L2U = A                      .CLEAR 2ND DATA LENGTH         ISPDRV.174
214   024442                             R!L2L = A                                                     ISPDRV.175
215   024445                             R!TMO = A                      .CLEAR TIMEOUT PENDING         ISPDRV.176
                                                                                                        ISPDRV.177
216   072430                             R = CLEAR                      .CLEAR THE CHANNEL             ISPDRV.178
                                                                                                        ISPDRV.179
217   001000                             EXIT                                                          ISPDRV.180
```

```
                                 *************************.  .*****************************************   ISPDRV.182
                                 *                                                                    *   ISPDRV.183
                                 * THE DOCLS ROUTINE EXECUTES THE DRIVER CLOSE FUNCTION FOR ISP       *   ISPDRV.184
                                 * CHANNELS.                                                          *   ISPDRV.185
                                 *                                                                    *   ISPDRV.186
                                 **********************************************************************   ISPDRV.187
                                                                                                        ISPDRV.188
220                              DOCLS    *                                                             ISPDRV.189
220    010000  024432                     R!USI = UC$CMPT                .SET STATUS RETURN            ISPDRV.190
222    010001  024435                     R!END = 1                      .TERMINATE ACTIVITY           ISPDRV.191
224    001000                             EXIT                                                         ISPDRV.192
```

```
                                 ********************************************************************   ISPDRV.194
                                 *                                                                  *   ISPDRV.195
                                 * THE DOIO ROUTINE EXECUTES READ ,READ-LAST, WRITE AND WRITE-LAST  *   ISPDRV.196
                                 * COMMANDS FOR ISP CHANNELS.                                       *   ISPDRV.197
                                 *                                                                  *   ISPDRV.198
                                 * ONE OF TWO MAIN LOOPS IS USED, DEPENDING ON WHETHER THE UC$XBF   *   ISPDRV.199
                                 * FLAG IS SET (XIO) OR NOT (IO).  XBF INDICATES THAT THE SHELL HAS  *   ISPDRV.200
                                 * A SECOND LOCAL I/O BUFFER, AND CAN GIVE THE DRIVER A SECOND REQUEST *   ISPDRV.201
                                 * IMMEDIATELY.  A SIGNAL WITH R!UST = UC$NOOP AFTER THE FIRST UC$RD *   ISPDRV.202
                                 * OR UC$WRT REQUEST ALLOWS THE SHELL TO DO THIS.                   *   ISPDRV.203
                                 *                                                                  *   ISPDRV.204
                                 * THE TWO LOOPS CONTAIN THE SAME BASIC STEPS, BUT IN DIFFERENT ORDER: *   ISPDRV.205
                                 *                                                                  *   ISPDRV.206
                                 *                XIO                                  IO            *   ISPDRV.207
                                 *    ----------------------------      ----------------------------  *   ISPDRV.208
                                 *    R = STARTIO   .START I/O          R = STARTIO   .START I/O      *   ISPDRV.209
                                 *                                                                  *   ISPDRV.210
                                 *    SIGNAL        .RETURN STATUS      R = WAITIO    .WAIT DONE      *   ISPDRV.211
                                 *                   OF LAST REQ.                                   *   ISPDRV.212
                                 *                                                                  *   ISPDRV.213
                                 *    WATCH         .PREFETCH NEXT      SIGNAL        .RETURN STATUS  *   ISPDRV.214
                                 *                   REQUEST                            OF THIS REQ.  *   ISPDRV.215
                                 *                                                                  *   ISPDRV.216
                                 *    R = WAITIO    .WAIT DONE          WATCH         .GET NEXT REQ.  *   ISPDRV.217
                                 *                                                                  *   ISPDRV.218
                                 ********************************************************************   ISPDRV.219
                                                                                                        ISPDRV.220
225                              DOIO     *                                                             ISPDRV.221
225                                       GET R!%W1,UC@XBF,R!TBL        .1 IF DOING 2-BUFFER I/O        ISPDRV.222
232   020423  102122                      P = SIO, R!%W1 = 0            .IF SINGLE-BUFFER I/O           ISPDRV.223
                                                                                                        ISPDRV.224
234                              XIO      *                                                             ISPDRV.225
234   010002  024432                      R!UST = UC$NOOP               .FIRST REPLY STATUS             ISPDRV.226
236    <macro>                            $UNTIL (R!UST = UC$CMPT) OR (R!UST >= UC$ERR)                 ISPDRV.227
243   .072232                               R = STARTIO                .START UP DEVICE                ISPDRV.228
244    <macro>                              $IF (R!UST < UC$ERR)       .IF NO ERROR YET                ISPDRV.229
                                                                                        .>>>>> 0          $IF      1
250    <macro>                                $IF (R!CMD = UC$RD) OR (R!CMD = UC$WRT)  .IF MORE TO COME  ISPDRV.230
                                                                                        .>>>>> 1          $IF      1
257    <macro>                                  SIGNAL R!TBL+UC@DSR@P,R!UST                            ISPDRV.231
301    <macro>                                  WATCH  R!TBL+UC@SDR@P,R!CMD                            ISPDRV.232
345   010001  024432                            R!UST = UC$CONT                                        ISPDRV.233
```

```
 349    <macro>                          $ELSE                    .ELSE THIS IS LAST I/O                ISPDRV.234
 350    010000  024432                     R!UST = UC$CMPT        .COMPLETE IF NO ERRORS                ISPDRV.235
 352    <macro>                          $ENDIF                                                         ISPDRV.236
                                                                                   .<<<<< 1                $ENDIF   1
 352    072350                           R = WAITIO               .WAIT FOR INTERRUPT ON CHANNEL        ISPDRV.237
 353    <macro>                        $ENDIF                                                           ISPDRV.238
                                                                                   .<<<<< 0                $ENDIF   1
 353    <macro>                        $ENDTIL                                                          ISPDRV.239
 354    001000                         EXIT                                                             ISPDRV.240
                                                                                                         ISPDRV.241
 355                            SIO     *                                                               ISPDRV.242
 355    <macro>                        $IF (R!CMD = UC$RD) OR (R!CMD = UC$WRT)  .IF MORE TO COME        ISPDRV.243
                                                                                   .>>>>> 0                $IF      1
 364    010001  024432                   R!UST = UC$CONT                                                 ISPDRV.244
 366    <macro>                        $ELSE                                                            ISPDRV.245
 367    010000  024432                   R!UST = UC$CMPT                                                 ISPDRV.246
 371    <macro>                        $ENDIF                                                           ISPDRV.247
                                                                                   .<<<<< 0                $ENDIF   1
 371    072104                         R = STARTIO                                                      ISPDRV.248
 372    020432  013040  100002         EXIT, R!UST >= UC$ERR                                            ISPDRV.249
        001000
 376    072324                         R = WAITIO                                                       ISPDRV.250
 377    020432  013001  102002         EXIT, R!UST # UC$CONT    .EXIT IF I/O ALL DONE                   ISPDRV.251
        001000
 403    <macro>                        SIGNAL R!TBL+UC@DSR@P,R!UST                                     ISPDRV.252
 425    <macro>                        WATCH  R!TBL+UC@SDR@P,R!CMD                                     ISPDRV.253
 471    071114                         P = SIO                                                          ISPDRV.254
                                                                                                         ISPDRV.255
```

```
                                 ************************************************************************   ISPDRV.257
                                 *                                                                      *   ISPDRV.258
                                 * THE DODRV ROUTINE EXECUTES DRIVER REQUEST FUNCTIONS.                 *   ISPDRV.259
                                 *                                                                      *   ISPDRV.260
                                 * NO DRIVER FUNCTIONS ARE IMPLEMENTED AT AT THIS TIME.                 *   ISPDRV.261
                                 *                                                                      *   ISPDRV.262
                                 ************************************************************************   ISPDRV.263
                                                                                                            ISPDRV.264
472                              DODRV    *                                                                 ISPDRV.265
472   010043  024432                      R!UST = IE$FCN                .NO DRIVER FUNCTIONS DEFINED       ISPDRV.266
474   001000                              EXIT                                                              ISPDRV.267
```

```
                                  ***************************************************************************   ISPDRV.269
                                  *                                                                         *   ISPDRV.270
                                  * STARTIO                                                                 *   ISPDRV.271
                                  *                                                                         *   ISPDRV.272
                                  * THIS SUBROUTINE IS RESPONSIBLE FOR STARTING THE CHANNEL ON THE          *   ISPDRV.273
                                  * NEXT BLOCK OF I/O.  IT ALSO HOLDS DISCONNECT ON ALL BUT THE LAST        *   ISPDRV.274
                                  * BLOCK OF A SERIES OF OUTPUTS.                                           *   ISPDRV.275
                                  *                                                                         *   ISPDRV.276
                                  * ENTRY CONDITIONS:                                                       *   ISPDRV.277
                                  *                                                                         *   ISPDRV.278
                                  * UC@SDB = I/O BUFFER ADDRESS FOR THIS I/O.                               *   ISPDRV.279
                                  * UC@SDZ = NUMBER OF BYTES TO TRANSFER.                                   *   ISPDRV.280
                                  * R!CHN  = CHANNEL NUMBER                                                 *   ISPDRV.281
                                  * R!CMD  = UC$RD, UC$WRT, UC$RDL, OR UC$WRTL                              *   ISPDRV.282
                                  *                                                                         *   ISPDRV.283
                                  * EXIT:                                                                   *   ISPDRV.284
                                  *                                                                         *   ISPDRV.285
                                  * UC@SDB = 0                                                              *   ISPDRV.286
                                  * UD@SDZ = 0                                                              *   ISPDRV.287
                                  * CHANNEL BUSY, OR                                                        *   ISPDRV.288
                                  * R!UST  = IE$LEN (THIS INDICATES A BUG IN THE SHELL, AND SHOULD          *   ISPDRV.289
                                  *          NEVER OCCUR.)                                                  *   ISPDRV.290
                                  *                                                                         *   ISPDRV.291
                                  ***************************************************************************   ISPDRV.292
                                                                                                                ISPDRV.293
                                                                                                                ISPDRV.294
475                               STARTIO  *                                                                    ISPDRV.295
                                                                                                                ISPDRV.296
475   020445  102041                      P = STARTIO2, R!TMO = 0        .IF NO TIMEOUT PENDING                 ISPDRV.297
                                                                                                                ISPDRV.298
477   020436  054000                      B = R!CHN                                                             ISPDRV.299
501   171000                              IOB:11                         .READ CHANNEL STATUS FLAGS             ISPDRV.300
502    <macro>                            $IF (0 # A & O'100000)         .IF SEQ ERR FLAG                       ISPDRV.301
                                                                                                  .>>>>> 0          $IF      1
511   072135                                R = CLEAR                    .DO I/O MASTER CLEAR                   ISPDRV.302
512   070025                                P = STARTIO2                                                        ISPDRV.303
513    <macro>                            $ENDIF                                                                ISPDRV.304
                                                                                                  .<<<<< 0          $ENDIF   1
513    <macro>                            PAUSE   1                      .DELAY ONE TENTH                       ISPDRV.305
524   027445                              R!TMO = R!TMO - 1                                                     ISPDRV.306
525   020445  107031                      P=STARTIO, R!TMO # 0           .TRY AGAIN UNTIL TIME IS UP            ISPDRV.307
                                                                                                                ISPDRV.308
```

```
527    <macro>                          GET R!TMO,UC@TMO,R!TBL        .KEEP TIMEOUT PENDING                ISPDRV.309
534    010040  024432                   R!UST = IE$TMO                .TIMEOUT ERROR CODE                  ISPDRV.310
536    001000                           EXIT                                                               ISPDRV.311
                                                                                                           ISPDRV.312
537                             STARTIO2 *                                                                  ISPDRV.313
537    010000  024445                   R!TMO = 0                     .CLEAR TIMEOUT PENDING               ISPDRV.314
                                                                                                           ISPDRV.315
541    <macro>                          GET R!BUF,UC@SDB,R!TBL        .BUFFER ADDRESS                      ISPDRV.316
546    <macro>                          $PUNTIF (R!BUF = 0)           .SHELL DIDN'T SUPPLY A BUFFER        ISPDRV.317
551    <macro>                          STORE   0,UC@SDB,R!TBL                                             ISPDRV.318
556    <macro>                          GET R!SIZ,UC@SDZ,R!TBL        .DATA LENGTH                         ISPDRV.319
563    <macro>                          STORE   0,UC@SDZ,R!TBL                                             ISPDRV.320
                                                                                                           ISPDRV.321
570    <macro>                          $IF (R!SIZ # 0)               .IF LENGTH NONZERO                   ISPDRV.322
                                                                                          .>>>>> 0               $IF      1
573    020436  054000                     B = R!CHN                                                        ISPDRV.323
575    160000                             IOB:0                       .CLEAR CHANNEL                       ISPDRV.324
576    163000                             IOB:3                                                            ISPDRV.325
577    166000                             IOB:6                       .CLEAR INTERRUPT ENABLE              ISPDRV.326
600    <macro>                            STORE 0,UC@IRT,R!TBL        .CLEAR INTERRUPT RETURNED FLAG       ISPDRV.327
607    <macro>                            $IF (R!CMD = UC$WRT)        .IF WRITE, NOT LAST                  ISPDRV.328
                                                                                          .>>>>> 1               $IF      1
613    014000 /001000                       A = O'1000                                                     ISPDRV.329
615    164000                               IOB:4                     .SET HOLD-DISCONNECT                 ISPDRV.330
616    <macro>                            $ELSEIF (R!CMD = UC$WRTL)   .ELSE IF WRITE LAST                  ISPDRV.331
623    010000                               A = 0                                                          ISPDRV.332
624    164000                               IOB:4                     .CLEAR HOLD-DISCONNECT               ISPDRV.333
625    <macro>                            $ENDIF                                                           ISPDRV.334
                                                                                          .<<<<< 1               $ENDIF   1
625    020434  004001                     A = R!SIZ > BYT$PAR         .I/O PARCEL COUNT                    ISPDRV.335
627    162000                             IOB:2                       .ENTER PARCEL COUNT                  ISPDRV.336
630    020433                             A = R!BUF                   .BUFFER ADDRESS                      ISPDRV.337
631    161000                             IOB:1                       .START UP THE CHANNEL                ISPDRV.338
632    <macro>                            STORE 1,UC@IPN,R!TBL        .SET INTERRUPT PENDING               ISPDRV.339
642    <macro>                          $ELSE                         .ELSE ZERO-LENGTH TRANSFER           ISPDRV.340
643    010044  024432                     R!UST = IE$LEN              .SET LENGTH ERROR STATUS             ISPDRV.341
645    <macro>                          $ENDIF                                                             ISPDRV.342
                                                                                          .<<<<< 0               $ENDIF   1
                                                                                                           ISPDRV.343
645    001000                           EXIT                                                               ISPDRV.344
```

```
                                 ***************************************************************** ISPDRV.346
                                 *                                                               * ISPDRV.347
                                 * CLEAR                                                         * ISPDRV.348
                                 *                                                               * ISPDRV.349
                                 * CLEAR SENDS AN I/O MASTER CLEAR ON AN OUTPUT CHANNEL           * ISPDRV.350
                                 *                                                               * ISPDRV.351
                                 * IF THIS CHANNEL IS AN INPUT CHANNEL, WE USE THE CO-CHANNEL NUMBER * ISPDRV.352
                                 * AS THE CHANNEL TO DO THE MASTER CLEAR ON.  THE CO-CHANNEL NUMBER * ISPDRV.353
                                 * OF AN INPUT CHANNEL IS THE OUTPUT CHANNEL TO THE SAME FEI DEVICE. * ISPDRV.354
                                 *                                                               * ISPDRV.355
                                 ***************************************************************** ISPDRV.356
                                                                                                   ISPDRV.357
                                                                                                   ISPDRV.358
646                              CLEAR    *                                                        ISPDRV.359
                                                                                                   ISPDRV.360
646   020436  054000                      B = R!CHN                    .CHANNEL NUMBER             ISPDRV.361
650    <macro>                            $IF (0 = B & 1)              .IF INPUT CHANNEL           ISPDRV.362
                                                                                   .>>>>> 0          $IF      1
657    <macro>                              GET R!CHN,UC@CCN,R!TBL     .GET CO-CHANNEL NUMBER      ISPDRV.363
662   020436  054000                        B = R!CHN                                              ISPDRV.364
664    <macro>                            $ENDIF                                                  ISPDRV.365
                                                                                   .<<<<< 0          $ENDIF   1
664    <macro>                            $IF (1 = B & 1)              .IF OUTPUT CHANNEL          ISPDRV.366
                                                                                   .>>>>> 0          $IF      1
673   160000                               IOB:0                      .CLEAR CHANNEL DONE & BUSY  ISPDRV.367
674   163000                               IOB:3                      .CLEAR PARITY ERROR FLAGS   ISPDRV.368
675   014000 /040000                       A = O'40000                .I/O MASTER CLEAR           ISPDRV.369
677   164000                               IOB:4                                                  ISPDRV.370
700   010012                               A = D'10                                               ISPDRV.371
701    <macro>                             $UNTIL A=0                 .HOLD MC UP A BIT           ISPDRV.372
702   013001                                 A = A-1                                              ISPDRV.373
703    <macro>                             $ENDTIL                                                ISPDRV.374
704   020436  054000                       B = R!CHN                  .RESTORE B AFTER KERNEL CALL ISPDRV.375
706   010000                               A = 0                      .CLEAR CHANNEL COMMAND      ISPDRV.376
707   164000                               IOB:4                                                  ISPDRV.377
710   160000                               IOB:0                      .CLEAR DONE AND BUSY BITS   ISPDRV.378
711   163000                               IOB:3                      .CLEAR ERROR FLAGS          ISPDRV.379
712   010024                               A = D'20                                               ISPDRV.380
713    <macro>                             $UNTIL A=0                 .WAIT FOR MC TO SETTLE      ISPDRV.381
714   013001                                 A = A-1                                              ISPDRV.382
715    <macro>                             $ENDTIL                                                ISPDRV.383
716    <macro>                           $ENDIF                                                   ISPDRV.384
```

```
                                                                    .<<<<< 0          $ENDIF   1
716    <macro>                GET R!CHN,UC@CHN,R!TBL      .RESTORE CHANNEL NUMBER      ISPDRV.385
                                                                                     ISPDRV.386
721    001000                 EXIT                                                  ISPDRV.387
```

```
*************************  ****************************************  ISPDRV.389
*                                                                  *  ISPDRV.390
* WAITIO                                                           *  ISPDRV.391
*                                                                  *  ISPDRV.392
* WAITIO DOES 2 THINGS:                                            *  ISPDRV.393
*                                                                  *  ISPDRV.394
*  1.  WAIT FOR THE CURRENT I/O TO COMPLETE OR TIME OUT.           *  ISPDRV.395
*                                                                  *  ISPDRV.396
*  2.  PROCESS THE COMPLETION OF I/O, COMPUTING THE AMOUNT OF DATA *  ISPDRV.397
*      SENT OR RECEIVED, CHECKING FOR ERRORS, AND CHECKING FOR THE *  ISPDRV.398
*      END OF A BLOCK.                                             *  ISPDRV.399
*                                                                  *  ISPDRV.400
* ENTRY CONDITIONS:                                                *  ISPDRV.401
*                                                                  *  ISPDRV.402
* UC@IRT = 1 IF THE KERNEL HAS RECEIVED AN INTERRUPT AFTER I/O WAS *  ISPDRV.403
*          STARTED.  THIS MAY HAPPEN DURING EXECUTION OF WAITIO.   *  ISPDRV.404
*                                                                  *  ISPDRV.405
* R!CHN  = CHANNEL NUMBER.                                         *  ISPDRV.406
*                                                                  *  ISPDRV.407
* R!BUF  = BEGINNING ADDRESS OF I/O BLOCK.                         *  ISPDRV.408
*                                                                  *  ISPDRV.409
* UC@IBF = CHANNEL ADDRESS AT TIME OF INTERRUPT (SET BY KERNEL).   *  ISPDRV.410
*                                                                  *  ISPDRV.411
* R!L1U,R!L1L = LENGTH OF BLOCK CURRENTLY BEING READ (WORDS),      *  ISPDRV.412
*               ZERO IF UNKNOWN.                                   *  ISPDRV.413
*                                                                  *  ISPDRV.414
* R!L2U,R!L2L = LENGTH OF NEXT BLOCK TO READ (WORDS),              *  ISPDRV.415
*               ZERO IF UNKNOWN.                                   *  ISPDRV.416
*                                                                  *  ISPDRV.417
* EXIT CONDITIONS:                                                 *  ISPDRV.418
*                                                                  *  ISPDRV.419
* R!UST  = IE$IMO IF NO INTERRUPT IS RECEIVED.                     *  ISPDRV.420
*        = IE$HWE IF HARDWARE ERROR ON INPUT CHANNEL.              *  ISPDRV.421
*        = IE$CMPT IF ENTIRE BLOCK READ ON INPUT CHANNEL.          *  ISPDRV.422
*                                                                  *  ISPDRV.423
* UC@DSZ = AMOUNT OF DATA TRANSFERRED (BYTES).                     *  ISPDRV.424
*                                                                  *  ISPDRV.425
* R!L1U,L1L,L2U,L2L UPDATED.                                       *  ISPDRV.426
*                                                                  *  ISPDRV.427
*                                                                  *  ISPDRV.428
* PROBLEMS:                                                        *  ISPDRV.429
*                                                                  *  ISPDRV.430
```

```
                        * 1.  IT IS POSSIBLE FOR THE KERNEL TO TIME OUT THE TPUSH AFTER       *   ISPDRV.431
                        * I/O HAS ACTUALLY BEGUN, BUT BEFORE AN INTERRUPT HAS BEEN RECEIVED.  *   ISPDRV.432
                        * IF WE CLEARED THE CHANNEL ON SUCH A TIMEOUT, IT WOULD BE DIFFICULT  *   ISPDRV.433
                        * TO RECOVER, SINCE THE OTHER SIDE WOULD SEE AN INTERRUPTED TRANS-    *   ISPDRV.434
                        * MISSION, NOT A TIMEOUT.  FOR THIS REASON, WAITIO CHECKS FOR I/O     *   ISPDRV.435
                        * IN PROGRESS AFTER A TIMEOUT, AND WAITS ANOTHER 100 MSEC IF THE      *   ISPDRV.436
                        * I/O HAS STARTED.  THIS IS AMPLE TIME FOR ANY I/O TO COMPLETE, SO    *   ISPDRV.437
                        * IF THIS SECOND INTERVAL EXPIRES, WE THEN RETURN A TIME-OUT          *   ISPDRV.438
                        * STATUS CODE.                                                        *   ISPDRV.439
                        *                                                                    *   ISPDRV.440
                        * 2.  ISP TRANSMISSION BLOCKS ARE VARIABLE-LENGTH.  THIS MEANS THAT   *   ISPDRV.441
                        * IQM ISSUES A READ FOR A MAXIMUM-LENGTH BLOCK, AND EXPECTS ISPDRV TO *   ISPDRV.442
                        * FIGURE OUT HOW LONG THE BLOCK ACTUALLY IS.  IF THE BLOCK HAPPENS TO *   ISPDRV.443
                        * BE AN EXACT MULTIPLE OF THE LOCAL BUFFER SIZE, THERE IS NO WAY TO   *   ISPDRV.444
                        * DETERMINE FROM HARDWARE WHEN THE DISCONNECT AT THE END OF THE BLOCK *   ISPDRV.445
                        * HAS ARRIVED, SINCE IT COINCIDES WITH THE INTERRUPT AT THE END OF    *   ISPDRV.446
                        * THE LOCAL I/O BUFFER.  THEREFORE THE DRIVER HAS TO BE SMART ENOUGH  *   ISPDRV.447
                        * TO DO THE FOLLOWING THINGS:                                         *   ISPDRV.448
                        *                                                                    *   ISPDRV.449
                        *  A.  KNOW WHETHER THE CURRENT INPUT BLOCK IS A HEADER BLOCK OR A   *   ISPDRV.450
                        *  DATA BLOCK (THESE GENERALLY COME IN ALTERNATING SEQUENCE).        *   ISPDRV.451
                        *                                                                    *   ISPDRV.452
                        *  B.  CAPTURE THE HEADER LENGTH AND DATA LENGTH FIELDS FROM HEADER  *   ISPDRV.453
                        *  BLOCKS AS THEY ARE RECEIVED (IF THE CHECKSUM IS CORRECT).         *   ISPDRV.454
                        *                                                                    *   ISPDRV.455
                        *  C.  RETURN UC$CMPT STATUS IN R!UST (END OF BLOCK) WHEN THE AMOUNT *   ISPDRV.456
                        *  OF DATA RECEIVED EQUALS WHAT IS EXPECTED ACCORDING TO THESE       *   ISPDRV.457
                        *  LENGTHS.                                                          *   ISPDRV.458
                        *                                                                    *   ISPDRV.459
                        * THERE IS AN IMPLICIT ASSUMPTION IN THIS PROCESSING THAT IQM WILL   *   ISPDRV.460
                        * ALWAYS ISSUE READS FOR MULTIPLES OF 8 BYTES.                        *   ISPDRV.461
                        *                                                                    *   ISPDRV.462
                        ************************************************************************   ISPDRV.463
                                                                                                   ISPDRV.464
                                                                                                   ISPDRV.465
722                     WAITIO   *                                                                 ISPDRV.466
722   010000  024444            R!TO2 = 0                        .CLEAR 2ND TIMEOUT FLAG           ISPDRV.467
                                                                                                   ISPDRV.468
                                 * WAIT FOR AN INTERRUPT ON THE CHANNEL:                           ISPDRV.469
                                                                                                   ISPDRV.470
724                     WAITIO2  *                                                                 ISPDRV.471
724   020436  054000            B = R!CHN                                                          ISPDRV.472
```

```
 72    002000  000000          I = 0                                                    ISPDRV.473
730     <macro>               GET R!%W1,UC@IRT,R!TBL      .1 IF INTERRUPT ALREADY HAPPENED  ISPDRV.474
737     <macro>               $IF (R!%W1 = 0)             .IF NO INTERRUPT YET              ISPDRV.475
                                                                          .>>>>> 0              $IF      1
                                                                                            ISPDRV.476
743     <macro>                 ADDRESS R!%W1,UC@IWQ,R!TBL  .I/O WAIT QUEUE ADDRESS         ISPDRV.477
746    020444  024424           R!%W2 = R!TO2               .= 1 IF 2ND TIMEOUT ON THIS I/O  ISPDRV.478
750     <macro>                 $IF (R!%W2 = 0)             .IF NOT 2ND TIMEOUT              ISPDRV.479
                                                                          .>>>>> 1              $IF      1
753     <macro>                   GET R!%W2,UC@TMO,R!TBL    .  USE DEFAULT TIMEOUT           ISPDRV.480
760     <macro>                 $ENDIF                                                     ISPDRV.481
                                                                          .<<<<< 1              $ENDIF   1
760    167000                   IOB:7                       .ENABLE INTERRUPTS ON CHANNEL    ISPDRV.482
                                                                                            ISPDRV.483
761     <macro>                 TPUSH R!%W1,R!%W2           .SUSPEND UNTIL INTERRUPT         ISPDRV.484
                                                                                            ISPDRV.485
1000   002000  000000           I = 0                                                      ISPDRV.486
1002   020436  054000           B = R!CHN                   .CHANNEL NUMBER                  ISPDRV.487
1004    <macro>                 GET R!%W1,UC@IRT,R!TBL      .CHECK INTERRUPT RETURNED FLAG   ISPDRV.488
1013    <macro>                 $IF (R!%W1 = 0)             .IF NO INTERRUPT YET             ISPDRV.489
                                                                          .>>>>> 1              $IF      1
1016    <macro>                   $IF (R!TO2 = 0)           .IF FIRST TIMEOUT                ISPDRV.490
                                                                          .>>>>> 2              $IF      1
1021   170000                       IOB:10                  .READ CHANNEL ADDRESS            ISPDRV.491
1022    <macro>                     $IF (A # R!BUF)         .IF I/O HAS STARTED              ISPDRV.492
                                                                          .>>>>> 3              $IF      1
1025   010001  024444                R!TO2 = 1               .SET 2ND TIMEOUT FLAG            ISPDRV.493
1027   071103                         P = WAITIO2             .WAIT 100 MSEC MORE              ISPDRV.494
1030    <macro>                     $ENDIF                                                   ISPDRV.495
                                                                          .<<<<< 3              $ENDIF   1
1030    <macro>                   $ENDIF                    .CHANNEL HAS TIMED OUT:          ISPDRV.496
                                                                          .<<<<< 2              $ENDIF   1
1030   160000                     IOB:0                     .CLEAR CHANNEL                   ISPDRV.497
1031   163000                     IOB:3                                                      ISPDRV.498
1032   166000                     IOB:6                     .DISABLE INTERRUPTS              ISPDRV.499
1033    <macro>                   $IF (1 = B & 1)           .IF OUTPUT CHANNEL               ISPDRV.500
                                                                          .>>>>> 2              $IF      1
1042    <macro>                     GET R!TMO,UC@TMO,R!TBL  .  SET TIMEOUT PENDING           ISPDRV.501
1047    <macro>                   $ENDIF                                                     ISPDRV.502
                                                                          .<<<<< 2              $ENDIF   1
1047   003000                     I = 1                                                      ISPDRV.503
1050   010040  024432             R!UST = IE$TMO            .SET TIMEOUT ERROR CODE          ISPDRV.504
```

```
1052   001000                          EXIT                        .RETURN TO CALLER                    ISPDRV.505
1053    <macro>                      $ENDIF                                                           ISPDRV.506
                                                                                   .<<<<< 1                $ENDIF   1
                                                                                                        ISPDRV.507
1053    <macro>                    $ENDIF                                                             ISPDRV.508
                                                                                   .<<<<< 0                $ENDIF   1
1053   003000                      I = 1                                                              ISPDRV.509
                                                                                                        ISPDRV.510
                                   * RETURN NUMBER OF BYTES SENT OR RECEIVED:                          ISPDRV.511
                                                                                                        ISPDRV.512
1054    <macro>                    PUT R!BUF,UC@DSB,R!TBL      .RETURN BUFFER                         ISPDRV.513
1061    <macro>                    GET R!TLB,UC@IBF,R!TBL      .CHANNEL BUFFER ADDRESS                ISPDRV.514
1066   020443  023433              A = R!TLB - R!BUF           .PARCELS OF DATA TRANSFERRED           ISPDRV.515
1070   005001  024443              R!TLB = A < BYT$PAR         .CONVERT PARCELS TO BYTES              ISPDRV.516
1072    <macro>                    PUT R!TLB,UC@DSZ,R!TBL      .RETURN DATA LENGTH                    ISPDRV.517
1077   020443  023434  101003      R!UST = UC$CMPT, R!TLB < R!SIZ   .COMPLETE IF SHORT I/O            ISPDRV.518
       010000  024432
                                                                                                        ISPDRV.519
                                   * CHECK FOR HARDWARE ERRORS ON INPUT CHANNEL:                       ISPDRV.520
                                                                                                        ISPDRV.521
1104    <macro>                    $IF (0 = R!CHN & 1)         .IF INPUT CHANNEL                      ISPDRV.522
                                                                                   .>>>>> 0                $IF      1
                                                                                                        ISPDRV.523
1113    <macro>                      GET R!%W1,UC@IST,R!TBL    .GET CHANNEL STATUS                    ISPDRV.524
1120    <macro>                      $IF (0 # R!%W1 & O'77777) .IF HARDWARE ERROR                     ISPDRV.525
                                                                                   .>>>>> 1                $IF      1
1130   010000  024437                R!L1U = 0                                                        ISPDRV.526
1132   010000  024440                R!L1L = 0                                                        ISPDRV.527
1134   010041  024432                R!UST = IE$HWE                                                   ISPDRV.528
1136   001000                        EXIT                                                             ISPDRV.529
1137    <macro>                      $ENDIF                                                           ISPDRV.530
                                                                                   .<<<<< 1                $ENDIF   1
                                                                                                        ISPDRV.531
                                     * IF LINK HEADER EXPECTED, EXAMINE IT FOR DATA LENGTHS:           ISPDRV.532
                                                                                                        ISPDRV.533
1137    <macro>                      $IF (R!L1U = 0) AND (R!L1L = 0)                                  ISPDRV.534
                                                                                   .>>>>> 1                $IF      1
1145   072044                        R = GETL                    .GET HEADER AND DATA LENGTHS           ISPDRV.535
1146    <macro>                      $ENDIF                                                           ISPDRV.536
                                                                                   .<<<<< 1                $ENDIF   1
                                                                                                        ISPDRV.537
                                     * SET COMPLETE STATUS IF ENTIRE BLOCK HAS BEEN READ:              ISPDRV.538
```

```
                                                                                                ISPDRV.539
1146    <macro>                          $IF (R!UST # UC$CMPT)        .IF NOT ALREADY COMPLETE   ISPDRV.540
                                                                                  .>>>>> 1          $IF      1
1151    020443  012007  004003            R!%W1 = R!TLB + 7 > BYT$WRD .CONVERT BYTES TO WORDS     ISPDRV.541
        024423
1155    <macro>                          DSUB R!L1U,R!L1L,R!%W1       .SUBTRACT FROM TOTAL DATA LENGTH  ISPDRV.542
1166    <macro>                          $IF (A # 0)                  .IF NO UNDERFLOW           ISPDRV.543
                                                                                  .>>>>> 2          $IF      1
1170    020440  102002  001000              EXIT, R!L1L # 0           .EXIT IF MORE DATA TO READ ISPDRV.544
1173    020437  102002  001000              EXIT, R!L1U # 0                                      ISPDRV.545
1176    <macro>                          $ENDIF                                                  ISPDRV.546
                                                                                  .<<<<< 2          $ENDIF   1
1176    <macro>                          $ENDIF                                                  ISPDRV.547
                                                                                  .<<<<< 1          $ENDIF   1
1176    010000  024432                   R!UST = UC$CMPT              .SET TRANSFER COMPLETE     ISPDRV.548
1200    020441  024437                   R!L1U = R!L2U                .SET NEXT DATA LENGTH      ISPDRV.549
1202    020442  024440                   R!L1L = R!L2L                                           ISPDRV.550
1204    010000  024441                   R!L2U = 0                    .CLEAR 2ND DATA LENGTH     ISPDRV.551
1206    010000  024442                   R!L2L = 0                                               ISPDRV.552
                                                                                                ISPDRV.553
1210    <macro>                         $ENDIF                                                   ISPDRV.554
                                                                                  .<<<<< 0          $ENDIF   1
1210    001000                          EXIT                                                     ISPDRV.555
```

```
*********************************************************************   ISPDRV.557
*                                                                   *   ISPDRV.558
* GETL                                                              *   ISPDRV.559
*                                                                   *   ISPDRV.560
* GETL EXAMINES A BLOCK JUST READ FROM THE CHANNEL.  THIS BLOCK IS  *   ISPDRV.561
* EXPECTED TO BE AN ISP HEADER BLOCK, THE FIRST 24 PARCELS OF WHICH *   ISPDRV.562
* IS A LINK PROTOCOL HEADER.  THE HEADER CONTAINS THE LENGTH IN BITS *   ISPDRV.563
* OF THE HEADER BLOCK AND THE DATA BLOCK THAT MAY FOLLOW.  GETL GETS *   ISPDRV.564
* THESE LENGTHS AND CONVERTS THEM TO THE EXPECTED TRANSMISSION      *   ISPDRV.565
* LENGTHS IN WORDS OF THE CURRENT AND NEXT BLOCKS ON THE CHANNEL.   *   ISPDRV.566
*                                                                   *   ISPDRV.567
* ENTRY CONDITIONS:                                                 *   ISPDRV.568
*                                                                   *   ISPDRV.569
* R!BUF = ADDRESS OF BLOCK TO EXAMINE.                              *   ISPDRV.570
*                                                                   *   ISPDRV.571
* EXIT:                                                             *   ISPDRV.572
*                                                                   *   ISPDRV.573
* R!L1U, R!L1L = LENGTH IN WORDS OF THE CURRENT BLOCK ON THE CHANNEL *   ISPDRV.574
*                IF IT IS A HEADER BLOCK WITH VALID CHECKSUM, OTHER- *   ISPDRV.575
*                WISE L1U, L1L, L2U AND L2L ARE ALL RETURNED ZERO.  *   ISPDRV.576
*                                                                   *   ISPDRV.577
* R!L2U, R!L2L = LENGTH IN WORDS OF THE DATA BLOCK TO FOLLOW, IF ONE *   ISPDRV.578
*                IS EXPECTED, OTHERWISE L2U = L2L = 0.              *   ISPDRV.579
*                                                                   *   ISPDRV.580
* THERE IS NO ATTEMPT TO ENSURE THAT WHAT IS RECEIVED IS ACTUALLY AN *   ISPDRV.581
* ISP HEADER BLOCK.  IT ONLY HAS TO HAVE A 16-BIT CHECKSUM OF ZERO  *   ISPDRV.582
* OVER THE FIRST 24 PARCELS.  THE MOST LIKELY NON-HEADER WITH SUCH A *   ISPDRV.583
* CHECKSUM IS 24 PARCELS OF ZERO, WHICH RESULT IN L1U AND L1L BEING *   ISPDRV.584
* RETURNED WITH ZERO.  THIS CAUSES WAITIO TO CHECK FOR A HEADER    *   ISPDRV.585
* AGAIN IN THE NEXT BUFFER-FULL -- EVENTUALLY IT WILL FIND ONE, AND *   ISPDRV.586
* UNTIL THERE IS AN ERROR ON THE CHANNEL, EVERYTHING WILL BE IN     *   ISPDRV.587
* SYNCH.  THE DRIVER IS NOT RESPONSIBLE FOR DETECTING PROTOCOL      *   ISPDRV.588
* ERRORS -- THERE IS EXTENSIVE CHECKING IN IQM FOR ALL OF THESE.    *   ISPDRV.589
* GETL IS ONLY EXPECTED TO WORK CORRECTLY IF THERE ARE NO ERRORS IN *   ISPDRV.590
* HARDWARE OR PROTOCOL.                                             *   ISPDRV.591
*                                                                   *   ISPDRV.592
* ON COMPUTATION OF THE BLOCK LENGTHS:                              *   ISPDRV.593
*                                                                   *   ISPDRV.594
* ALL BLOCKS ARE A MINIMUM OF 24 PARCELS LONG (IH@LE AND DDB@LE).   *   ISPDRV.595
* THERE ARE TWO TYPES OF LINK HEADERS, DISTINGUISHED BY THE FUNCTION *   ISPDRV.596
* CODE (IH@FC):  RESET AND DATA.  THERE IS ALWAYS A DATA BLOCK AFTER *   ISPDRV.597
* A DATA HEADER, AND NEVER ONE AFTER A RESET.  THE DATA BLOCK LENGTH *   ISPDRV.598
```

```
                                 * IN THE HEADER MAY BE LESS ...AN DDB@LE, SINCE IT INDICATES THE        *   ISPDRV.599
                                 * NUMBER OF VALID BITS IN THE BLOCK, NOT THE LENGTH.  THE HEADER       *   ISPDRV.600
                                 * BLOCK LENGTH FIELD CONTAINS THE NUMBER OF VALID BITS FOLLOWING       *   ISPDRV.601
                                 * THE LINK HEADER (24 PARCELS), NOT INCLUDING IT.                      *   ISPDRV.602
                                 *                                                                     *   ISPDRV.603
                                 * SO, THE HEADER BLOCK LENGTH IN WORDS IS DERIVED FROM THE LENGTH      *   ISPDRV.604
                                 * FIELD IH@HB(1,2,3):                                                  *   ISPDRV.605
                                 *                                                                     *   ISPDRV.606
                                 *      LENGTH = (IH@HB+63)/64 + 6:                                    *   ISPDRV.607
                                 *                                                                     *   ISPDRV.608
                                 * THE DATA BLOCK LENGTH IS:                                           *   ISPDRV.609
                                 *                                                                     *   ISPDRV.610
                                 *      LENGTH = MIN(6,(IH@DB+63)/64)                                  *   ISPDRV.611
                                 *                                                                     *   ISPDRV.612
                                 ***********************************************************************   ISPDRV.613
                                                                                                           ISPDRV.614
1211                             GETL     *                                                               ISPDRV.615
1211   010000  024437                     R!L1U = 0                    .CLEAR 1ST DATA LENGTH             ISPDRV.616
1213   024440                             R!L1L = A                                                       ISPDRV.617
1214   024441                             R!L2U = A                    .CLEAR 2ND DATA LENGTH             ISPDRV.618
1215   024442                             R!L2L = A                                                       ISPDRV.619
                                                                                                           ISPDRV.620
                                          *                                                               ISPDRV.621
                                          *          CHECKSUM THE HEADER                                  ISPDRV.622
                                          *                                                               ISPDRV.623
                                                                                                           ISPDRV.624
1216   010000  024423                     R!%W1 = 0                    .CHECKSUM ACCUMULATOR              ISPDRV.625
1220   010030  024424                     R!%W2 = IH@LE                .PARCEL COUNTER                    ISPDRV.626
1222   020433  024425                     R!%W3 = R!BUF                .BUFFER ADDRESS                    ISPDRV.627
1224    <macro>                           $UNTIL (R!%W2 = 0)                                              ISPDRV.628
1226   030425  024426                       R!%W4 = (R!%W3)                                               ISPDRV.629
1230    <macro>                             XOR R!%W1,R!%W4,R!%W1      .FORM THE CHECKSUM                 ISPDRV.630
1243   026425                               R!%W3 = R!%W3 + 1                                             ISPDRV.631
1244   027424                               R!%W2 = R!%W2 - 1                                             ISPDRV.632
1245    <macro>                           $ENDTIL                                                         ISPDRV.633
1246   020423  102002  001000             EXIT, R!%W1 # 0              .EXIT IF BAD CHECKSUM              ISPDRV.634
                                                                                                           ISPDRV.635
                                          *                                                               ISPDRV.636
                                          *          VALIDATE HEADER FIELDS                               ISPDRV.637
                                          *                                                               ISPDRV.638
1251    <macro>                           GET R!%W1,IH@HLI,R!BUF       .HEADER LENGTH FIELD               ISPDRV.639
1257   010030  005001  024410             EXIT, R!%W1 # IH@LE < BYT$PAR .IF NOT 48 BYTES                  ISPDRV.640
```

```
        020423  023410  102002
        001000
                                                                                ISPDRV.641
1266     <macro>                    GET R!%W2,IH@FC,R!BUF        .GET LINK HEADER FUNCTION CODE   ISPDRV.642
1274     <macro>                    PUT R!%W2,APF@FC,R!%W1       .SAVE FUNCTION CODE IN REQUEST   ISPDRV.643
1301    020424  102002  001000      EXIT, R!%W2 # 0              .SKIP THE REST IF NOT DATA       ISPDRV.644
                                                                                ISPDRV.645
                                    *                                           ISPDRV.646
                                    *        GET THE DATA LENGTHS AND CONVERT TO WORDS          ISPDRV.647
                                    *                                           ISPDRV.648
                                                                                ISPDRV.649
1304     <macro>                    GET R!%W1,IH@HB1,R!BUF       .HEADER BLOCK LENGTH (3 BYTES)   ISPDRV.650
1312     <macro>                    GET R!%W2,IH@HB2,R!BUF                                        ISPDRV.651
1320     <macro>                    GET R!%W3,IH@HB3,R!BUF                                        ISPDRV.652
1326    010077  025425              R!%W3 = R!%W3 + O'77         .ROUND UP TO WORD MULTIPLE       ISPDRV.653
1330    020425  004010  022424      R!%W2 = R!%W3 > D'8 + R!%W2                                   ISPDRV.654
        024424
1334    020424  004010  022423      R!%W1 = R!%W2 > D'8 + R!%W1                                   ISPDRV.655
        024423
1340    020425  004006  011003      R!L1L = R!%W3 > 6 & O'3      .CONVERT BITS TO WORDS           ISPDRV.656
        024440
1344    020424  011377  005002      R!L1L = R!%W2 & O'377 < 2 + R!L1L                             ISPDRV.657
        022440  024440
1351    020423  011077  005012      R!L1L = R!%W1 & O'77 < D'10 + R!L1L                           ISPDRV.658
        022440  024440
1356    020423  004006  011003      R!L1U = R!%W1 > 6 & 3                                         ISPDRV.659
        024437
1362     <macro>                    GET R!%W1,UC@REQ,R!TBL       .REQUEST PACKET ADDRESS          ISPDRV.660
1367     <macro>                    PUT R!L1L,APF@HZ,R!%W1       .STORE HEADER LENGTH IN REQUEST  ISPDRV.661
                                                                                ISPDRV.662
1374     <macro>                    GET R!%W1,IH@DB1,R!BUF       .DATA BLOCK LENGTH (3 BYTES)     ISPDRV.663
1402     <macro>                    GET R!%W2,IH@DB2,R!BUF                                        ISPDRV.664
1410     <macro>                    GET R!%W3,IH@DB3,R!BUF                                        ISPDRV.665
1416    020423  022424              A = R!%W1 + R!%W2                                             ISPDRV.666
1420     <macro>                    $IF (A # 0) OR (R!%W3 >= DDB@LE)  .IF 6 WORDS OF DATA OR MORE ISPDRV.667
                                                                          .>>>>> 0          $IF      1
1425    010077  025425                R!%W3 = R!%W3 + O'77       .ROUND UP TO FULL WORD OF DATA   ISPDRV.668
1427    020425  004010  022424        R!%W2 = R!%W3 > D'8 + R!%W2                                  ISPDRV.669
        024424
1433    020424  004010  022423        R!%W1 = R!%W2 > D'8 + R!%W1                                  ISPDRV.670
        024423
1437     <macro>                    $ELSE                        .ELSE LESS THAN 6 WORDS          ISPDRV.671
```

```
1440   010600  024425                    R!%W3 = DDB@LE                  .MAKE DATA LENGTH 6 WORDS          ISPDRV.672
1442   020425  004010  024424            R!%W2 = R!%W3 > D'8                                                ISPDRV.673
1445    <macro>                         $ENDIF                                                             ISPDRV.674
                                                                                        .<<<<< 0                $ENDIF   1
1445   020425  004006  011003           R!L2L = R!%W3 > 6 & O'3         .CONVERT BITS TO WORDS             ISPDRV.675
       024442
1451   020424  011377  005002           R!L2L = R!%W2 & O'377 < 2 + R!L2L                                  ISPDRV.676
       022442  024442
1456   020423  011077  005012           R!L2L = R!%W1 & O'77 < D'10 + R!L2L                                ISPDRV.677
       022442  024442
1463   020423  004006  011003           R!L2U = R!%W1 > 6 & 3                                              ISPDRV.678
       024441
1467    <macro>                         GET R!%W1,UC@REQ,R!TBL          .REQUEST PACKET ADDRESS            ISPDRV.679
1474    <macro>                         PUT R!L2U,APF@DZ,R!%W1          .STORE HEADER LENGTH IN REQUEST    ISPDRV.680
1501    <macro>                         PUT R!L2L,APF@DL,R!%W1                                             ISPDRV.681
                                                                                                           ISPDRV.682
1506   001000                           EXIT                                                               ISPDRV.683
                                                                                                           ISPDRV.684
                                                                                                           ISPDRV.685
                                                                                                           ISPDRV.686
                                        END                                                                ISPDRV.687
```

0X	%ACTIVE		5:20 D	5:20	9:41 D	9:41	10:26 D	10:26	18:15 D	18:15		
0X	%B		1: 4 D	18: 4 B								
0X	%EX		5:20 D	5:20	6: 8 D	6: 8	6:11 D	6:11	9:40 D	9:40		
			9:41 D	9:41	10:25 D	10:25	10:26 D	10:26	12:39 D	12:39		
			18:15 D	18:15								
10	%GBLNUM	$APTEXT	2:11 F	2:11								
400	%GBLREG	$APTEXT	2:11									
0X	%PUNT		13: 9 D	13: 9								
20	AD@RC@N	$APTEXT	5:20	5:20 F	9:41	9:41 F	10:26	10:26 F	18:15	18:15 F		
13	AD@RC@P	$APTEXT	5:20 F	5:20	9:41 F	9:41	10:26 F	10:26	18:15 F	18:15		
0	AD@RC@S	$APTEXT	5:20	5:20 F	9:41	9:41 F	10:26	10:26 F	18:15	18:15 F		
1	BYT$PAR	$APTEXT	13:30	19:14	22:42							
3	BYT$WRD	$APTEXT	20: 4									
7	F$PAUSE	$APTEXT	12:39									
2	F$POP	$APTEXT	6: 8	9:40	10:25							
1	F$PUSH	$APTEXT	5:20	9:41	10:26							
3	F$TERM	$APTEXT	6:11									
11	F$TPUSH	$APTEXT	18:15									
0	FS$PRI	$APTEXT	5:20	9:41	10:26							
0X	O$ISPDRV		1: 4 D	1: 4								
17	OV@NUM@N	$APTEXT	1: 4									
1	OV@TYP@N	$APTEXT	1: 4									
0	PT$HUH	$APTEXT	13: 9									
11	SM@FST@N	$APTEXT	1: 4									
7	SM@NUM@N	$APTEXT	1: 4									
4	UC$CLS	$APTEXT	5:26									
0	UC$CMPT	$APTEXT	7:15	8: 9	9:34	10: 2	10:16	19:16	20: 3	20:15		
1	UC$CONT	$APTEXT	9:42	10:14	10:23							
40	UC$DRV	$APTEXT	5:41									
40	UC$ERR	$APTEXT	4: 2	9:34	9:37	10:20						
2	UC$NOOP	$APTEXT	9:33									
3	UC$OPN	$APTEXT	5:23									
5	UC$RD	$APTEXT	5:29	9:39	10:13							
6	UC$RDI	$APTEXT	5:32									
7	UC$WRT	$APTEXT	5:35	9:39	10:13	13:22						
10	UC$WRTL	$APTEXT	5:38	13:25								
377	UC@CCN@M	$APTEXT	14:19									
10	UC@CCN@N	$APTEXT	14:19									
0	UC@CCN@P	$APTEXT	14:19 F									
10	UC@CCN@S	$APTEXT	14:19	14:19 F								
10	UC@CHN@N	$APTEXT	5:12	5:12 F	15: 2	15: 2 F						
0	UC@CHN@P	$APTEXT	5:12 F	15: 2 F								

0	UC@CHN@S	$APTEXT	5:12	5:12 F	15: 2	15: 2 F						
20	UC@DSB@N	$APTEXT	19:11 F									
37	UC@DSB@P	$APTEXT	19:11 F	19:11								
0	UC@DSB@S	$APTEXT	19:11 F									
34	UC@DSR@P	$APTEXT	6: 8	9:40	10:25							
20	UC@DSZ@N	$APTEXT	19:15 F									
40	UC@DSZ@P	$APTEXT	19:15 F	19:15								
0	UC@DSZ@S	$APTEXT	19:15 F									
20	UC@IBI@N	$APTEXT	19:12	19:12 F								
50	UC@IBI@P	$APTEXT	19:12 F	19:12								
0	UC@IBI@S	$APTEXT	19:12	19:12 F								
1	UC@IPN@M	$APTEXT	13:34 F									
1	UC@IPN@N	$APTEXT	13:34 F	13:34								
46	UC@IPN@P	$APTEXT	13:34 F	13:34								
0	UC@IPN@S	$APTEXT	13:34 F	13:34								
1	UC@IRI@M	$APTEXT	13:20 F	18: 2	18:19							
1	UC@IRI@N	$APTEXT	13:20 F	18: 2	18: 2 F	18:19	18:19 F					
46	UC@IRI@P	$APTEXT	13:20 F	13:20	18: 2 F	18: 2	18:19 F	18:19				
1	UC@IRI@S	$APTEXT	13:20 F	18: 2	18: 2 F	18:19	18:19 F					
137777	UC@IRI@X	$APTEXT	13:20									
20	UC@ISI@N	$APTEXT	19:24	19:24 F								
47	UC@ISI@P	$APTEXT	19:24 F	19:24								
0	UC@ISI@S	$APTEXT	19:24	19:24 F								
44	UC@IWQ@P	$APTEXT	18: 6									
20	UC@REQ@N	$APTEXT	7:16	7:16 F	23:28	23:28 F	24:13	24:13 F				
31	UC@REQ@P	$APTEXT	7:16 F	7:16	23:28 F	23:28	24:13 F	24:13				
0	UC@REQ@S	$APTEXT	7:16	7:16 F	23:28	23:28 F	24:13	24:13 F				
20	UC@SDB@N	$APTEXT	13: 8	13: 8 F	13:10 F							
27	UC@SDB@P	$APTEXT	13: 8 F	13: 8	13:10 F	13:10						
0	UC@SDB@S	$APTEXT	13: 8	13: 8 F	13:10 F							
24	UC@SDR@P	$APTEXT	5:20	9:41	10:26							
20	UC@SDZ@N	$APTEXT	13:11	13:11 F	13:12 F							
30	UC@SDZ@P	$APTEXT	13:11 F	13:11	13:12 F	13:12						
0	UC@SDZ@S	$APTEXT	13:11	13:11 F	13:12 F							
20	UC@IMO@N	$APTEXT	13: 1	13: 1 F	18:10	18:10 F	18:38	18:38 F				
41	UC@IMO@P	$APTEXT	13: 1 F	13: 1	18:10 F	18:10	18:38 F	18:38				
0	UC@IMO@S	$APTEXT	13: 1	13: 1 F	18:10	18:10 F	18:38	18:38 F				
20	UC@ZBI@N	$APTEXT	7:18 F	9:29	9:29 F							
15	UC@ZBI@P	$APTEXT	7:18 F	7:18	9:29 F	9:29						
0	UC@ZBI@S	$APTEXT	7:18 F	9:29	9:29 F							

SYMBOL QUALIFIER IS ISPDRV

1	$CONST	13:10 D	13:10 F	13:12 D	13:12 F	13:20 D	13:20 F	13:34 D	13:34 F
		13:34							
45	$NEXT	4: 2 D	4: 2	4: 3	4: 3 D	4: 4	4: 4 D	4: 5	4: 5 D
		4: 6	4: 6 D						
446	$REGORG	2:11 D	2:11 F	2:12	2:12 D	2:13	2:13 D	2:14	2:14 D
		2:15	2:15 D	2:16	2:16 D	2:16 F	2:17	2:17 D	2:18
		2:18 D	2:19	2:19 D	2:20	2:20 D	2:21	2:21 D	2:22
		2:22 D	2:22 F	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D
		2:26	2:26 D	2:27	2:27 D	2:27 F	2:28	2:28 D	2:28 F
		2:29	2:29 D	2:30	2:30 D	2:31	2:31 D	2:32	2:32 D
		2:33	2:33 D	2:34	2:34 D	2:35	2:35 D	2:36	2:36 D
		2:37	2:37 D	2:38	2:38 D	2:39	2:39 D	2:40	2:40 D
		2:41	2:41 D	2:41 F	5:20	5:20 F	6: 8	6:11	9:40
		9:41	9:41 F	10:25	10:26	10:26 F	12:39	12:39 F	18:15
		18:15 F							
1	%NP	1: 4	2:28 D						
430	%P	1: 4	2:27 D	2:28	5:20 F	5:20	9:41 F	9:41	10:26 F
		10:26	12:39 F	12:39	18:15 F	18:15			
410	%S1	2:11	2:12 D	5:20	7:16	7:17	7:18	9:29	9:41
		10:26	12:34	13: 1	13: 8	13:10	13:11	13:12	13:20
		13:34	14:18	14:24	18: 2	18:10	18:15	18:19	18:37
		18:38	19:11	19:12	19:15	19:22	19:24	19:26	22:41
		22:42	23: 1	23: 4	23: 5	23:12	23:13	23:14	23:28
		23:29	23:31	23:32	23:33	24:13	24:14	24:15	
411	%S2	2:11	2:13 D						
412	%S3	2:11	2:14 D						
413	%S4	2:11	2:15 D						
414	%S5	2:11	2:16 D						
415	%T1	2:17 D	5:20	6: 8	9:40	9:41	10:25	10:26	22:32
416	%T2	2:18 D							
417	%T3	2:19 D	20: 6						
420	%T4	2:20 D							
421	%T5	2:21 D							
422	%T6	2:22 D							
423	%W1	2:23 D	7:16	7:17	9:29	9:30	18: 2	18: 4	18: 6
		18:15	18:19	18:21	19:24	19:26	20: 4	20: 6	22:27
		22:32	22:36	22:41	22:42	23: 5	23:12	23:18	23:24
		23:26	23:28	23:29	23:31	23:34	23:40	24: 9	24:11
		24:13	24:14	24:15					
424	%W2	2:24 D	7:17	7:18	18: 7	18: 9	18:10	18:15	22:28
		22:30	22:34	23: 4	23: 5	23: 6	23:13	23:16	23:18

		23:22	23:32	2[illegible]	23:38	23:40	24: 2	24: 7	
425	%W3	2:25 D	22:29	22:31	22:33	23:14	23:15	23:16	23:20
		23:33	23:36	23:37	23:38	24: 1	24: 2	24: 5	
426	%W4	2:26 D	22:31	22:32					
427	%W5	2:27 D							
20	APF@DL@N	5: 5 D	24:15 F						
37	APF@DL@P	5: 5 D	24:15 F	24:15					
0	APF@DL@S	5: 5 D	24:15 F						
20	APF@DZ@N	5: 4 D	24:14 F						
36	APF@DZ@P	5: 4 D	24:14 F	24:14					
0	APF@DZ@S	5: 4 D	24:14 F						
20	APF@FC@N	5: 2 D	23: 5 F						
34	APF@FC@P	5: 2 D	23: 5 F	23: 5					
0	APF@FC@S	5: 2 D	23: 5 F						
20	APF@HZ@N	5: 3 D	23:29 F						
35	APF@HZ@P	5: 3 D	23:29 F	23:29					
0	APF@HZ@S	5: 3 D	23:29 F						
1	APF@XB@M	4:38 D	4:38						
1	APF@XB@N	4:38 D	7:17	7:17 F					
34	APF@XB@P	4:38 D	7:17 F	7:17					
0	APF@XB@S	4:38 D	7:17	7:17 F					
77777	APF@XB@X	4:38 D							
433	BUF	2:31 D	13: 8	13: 9	13:32	18:26	19:11	19:13	22:29
		22:41	23: 4	23:12	23:13	23:14	23:31	23:32	23:33
436	CHN	2:34 D	5:12	12:31	13:16	14:16	14:19	14:20	14:33
		15: 2	17:42	18:18	19:22				
646	CLEAR	7:25	12:35	14:14 D					
431	CMD	2:29 D	5:20	5:23	5:26	5:29	5:32	5:35	5:38
		5:41	9:39	9:41	10:13	10:26	13:22	13:25	
600	DDB@LE	4:32 D	23:36	24: 1					
220	DOCLS	5:27	8: 8 D						
472	DODRV	5:42	11: 9 D						
225	DOTO	5:30	5:33	5:36	5:39	9:28 D			
166	DOOPN	5:24	7:13 D						
435	END	2:33 D	5:11	5:18	8:10				
1211	GETL	19:38	22:17 D						
42	IE$CTY	4: 4 D							
43	IE$FCN	4: 5 D	6: 3	11:10					
41	IE$HWE	4: 3 D	19:29						
44	IE$LEN	4: 6 D	13:36						
40	IE$TMO	4: 2 D	13: 2	18:42					
377	IH@DB1@M	4:24 D	4:24	23:31					

10	IH@DB1@N	4:24	D	23:31			
6	IH@DB1@P	4:24	D	23:31	F	23:31	
10	IH@DB1@S	4:24	D	23:31		23:31	F
177400	IH@DB1@X	4:24	D				
377	IH@DB2@M	4:25	D	4:25			
10	IH@DB2@N	4:25	D	23:32		23:32	F
7	IH@DB2@P	4:25	D	23:32	F	23:32	
0	IH@DB2@S	4:25	D	23:32		23:32	F
377	IH@DB2@X	4:25	D				
377	IH@DB3@M	4:26	D	4:26		23:33	
10	IH@DB3@N	4:26	D	23:33			
7	IH@DB3@P	4:26	D	23:33	F	23:33	
10	IH@DB3@S	4:26	D	23:33		23:33	F
177400	IH@DB3@X	4:26	D				
377	IH@FC@M	4:18	D	4:18		23: 4	
10	IH@FC@N	4:18	D	23: 4			
3	IH@FC@P	4:18	D	23: 4	F	23: 4	
10	IH@FC@S	4:18	D	23: 4		23: 4	F
177400	IH@FC@X	4:18	D				
377	IH@HB1@M	4:21	D	4:21			
10	IH@HB1@N	4:21	D	23:12		23:12	F
5	IH@HB1@P	4:21	D	23:12	F	23:12	
0	IH@HB1@S	4:21	D	23:12		23:12	F
377	IH@HB1@X	4:21	D				
377	IH@HB2@M	4:22	D	4:22		23:13	
10	IH@HB2@N	4:22	D	23:13			
5	IH@HB2@P	4:22	D	23:13	F	23:13	
10	IH@HB2@S	4:22	D	23:13		23:13	F
177400	IH@HB2@X	4:22	D				
377	IH@HB3@M	4:23	D	4:23			
10	IH@HB3@N	4:23	D	23:14		23:14	F
6	IH@HB3@P	4:23	D	23:14	F	23:14	
0	IH@HB3@S	4:23	D	23:14		23:14	F
377	IH@HB3@X	4:23	D				
20	IH@HCS@N	4:20	D				
4	IH@HCS@P	4:20	D				
0	IH@HCS@S	4:20	D				
377	IH@HL1@M	4:15	D	4:15			
10	IH@HL1@N	4:15	D	22:41		22:41	F
2	IH@HL1@P	4:15	D	22:41	F	22:41	
0	IH@HL1@S	4:15	D	22:41		22:41	F
377	IH@HL1@X	4:15	D				

30	THRLE	4:12 D	22:28	2[illegible]					
0	THFCSDAT	4:19 D							
440	L11	2:36 D	7:20	19:28	19:37	20: 6	20: 9	20:17	22:19
		23:20	23:22	23:23	23:24	23:25	23:29		
437	L10	2:35 D	7:19	19:27	19:37	20: 6	20:10	20:16	22:18
		23:26							
442	L21	2:38 D	7:22	20:17	20:19	22:21	24: 5	24: 7	24: 8
		24: 9	24:10	24:15					
441	L20	2:37 D	7:21	20:16	20:18	22:20	24:11	24:14	
355	S10	9:30	10:11 D	10:27					
434	S1Z	2:32 D	13:11	13:15	13:30	19:16			
475	START10	9:35	10:19	12:27 D	12:41				
537	START102	12:29	12:36	13: 5 D					
430	TBL	2:28 D	5:12	5:20	6: 8	7:16	7:18	9:29	9:40
		9:41	10:25	10:26	13: 1	13: 8	13:10	13:11	13:12
		13:20	13:34	14:19	15: 2	18: 2	18: 6	18:10	18:19
		18:38	19:11	19:12	19:15	19:24	23:28	24:13	
443	TLB	2:39 D	19:12	19:13	19:14	19:15	19:16	20: 4	
445	TM0	2:41 D	7:23	12:29	12:40	12:41	13: 1	13: 6	18:38
444	T02	2:40 D	17:37	18: 7	18:23	18:27			
432	US1	2:30 D	6: 3	6: 8	7:15	8: 9	9:33	9:34	9:37
		9:40	9:42	10: 2	10:14	10:16	10:20	10:23	10:25
		11:10	13: 2	13:36	18:42	19:17	19:29	20: 3	20:15
722	WAIT10	10: 5	10:22	17:36 D					
724	WAIT102	17:41 D	18:28						
234	X10	9:32 D							

```
                IDENT TAPELOAD                                                  TAPELOAD.5
                ABS                                                             TAPELOAD.6
                                                                                TAPELOAD.7
                COMMENT   'Copyright (C) Cray Research, Inc., 1984'               COPYRT.2
                                                                                TAPELOAD.9
            ***********************************************************************TAPELOAD.10
            *                                                                   *TAPELOAD.11
            *                        T A P E   L O A D                          *TAPELOAD.12
            *                                                                   *TAPELOAD.13
            *  TAPELOAD is a stand-alone program which deadstarts the I/O       *TAPELOAD.14
            *  Subsystem from an expander tape file.                            *TAPELOAD.15
            *                                                                   *TAPELOAD.16
            *  TAPELOAD is deadstarted into the MIOP from the first file of a   *TAPELOAD.17
            *  deadstart tape (TAPELOAD must be contained in one tape block).   *TAPELOAD.18
            *                                                                   *TAPELOAD.19
            *  TAPELOAD saves information required by the DMP program in buffer *TAPELOAD.20
            *  memory.  The location is an assembly constant (REG$MOSU, REG$MOSL). *TAPELOAD.21
            *  This value is passed to DMP in registers %MOSU and %MOSL.        *TAPELOAD.22
            *                                                                   *TAPELOAD.23
            *  TAPELOAD prompts the user for the number of the file to be loaded *TAPELOAD.24
            *  with the 'FILE @MT0:' message. The response must be either:      *TAPELOAD.25
            *                                                                   *TAPELOAD.26
            *    Control-D  - TAPELOAD displays the message 'DEBUG' on the console *TAPELOAD.27
            *                 and will pass the value ON$DBUG to the loaded     *TAPELOAD.28
            *                 program in address DEBUG.  TAPELOAD will reprompt *TAPELOAD.29
            *                 with the 'FILE @MT0:' message.                    *TAPELOAD.30
            *                                                                   *TAPELOAD.31
            *    <file-num> - TAPELOAD loads the specifed file into MIOP local  *TAPELOAD.32
            *                 memory at address 0 and passes control to it.  The *TAPELOAD.33
            *                 records in the loaded file may be variable length up *TAPELOAD.34
            *                 4096 parcels.                                     *TAPELOAD.35
            *                                                                   *TAPELOAD.36
            *  TAPELOAD passes the value XMT$STRT to the DMP program in register *TAPELOAD.37
            *  %STRT to indicate that TAPELOAD is the caller.                   *TAPELOAD.38
            *                                                                   *TAPELOAD.39
            ***********************************************************************TAPELOAD.40
                                                                                TAPELOAD.41
                SCRATCH   %T0,%T1,%T2,%T3                                       TAPELOAD.42
                                                                                TAPELOAD.43
0   <macro>     %DUMP     REGISTER  (%DMP,%STRT,%MOSL,%MOSU)    .FOR DMP        TAPELOAD.44
      716       %DMP      EQUALS    $REGORG                                           $REG     2
      717       %STRT     EQUALS    $REGORG                                           $REG     2
```

```
                    720     %MOSL    EQUALS    $REGORG                                                    $REG    2
                    721     %MOSU    EQUALS    $REGORG                                                    $REG    2
0     <macro>               %DUMP    REGISTER  (BASE,SCR1,SCR2,SCR3)        .FOR TAPELOAD          TAPELOAD.45
                    716     BASE     EQUALS    $REGORG                                                    $REG    2
                    717     SCR1     EQUALS    $REGORG                                                    $REG    2
                    720     SCR2     EQUALS    $REGORG                                                    $REG    2
                    721     SCR3     EQUALS    $REGORG                                                    $REG    2
0     <macro>                        REGISTER  (DBG,FID,CNT,BUF)                                    TAPELOAD.46
                    722     DBG      EQUALS    $REGORG                                                    $REG    2
                    723     FID      EQUALS    $REGORG                                                    $REG    2
                    724     CNT      EQUALS    $REGORG                                                    $REG    2
                    725     BUF      EQUALS    $REGORG                                                    $REG    2
0     <macro>                        REGISTER  (MSG,CHAR,CHARS)                                     TAPELOAD.47
                    726     MSG      EQUALS    $REGORG                                                    $REG    2
                    727     CHAR     EQUALS    $REGORG                                                    $REG    2
                    730     CHARS    EQUALS    $REGORG                                                    $REG    2
0     <macro>                        REGISTER  (EOF,STAT)                                           TAPELOAD.48
                    731     EOF      EQUALS    $REGORG                                                    $REG    2
                    732     STAT     EQUALS    $REGORG                                                    $REG    2
0     <macro>                        REGISTER  (%T0,%T1,%T2,%T3)                                    TAPELOAD.49
                    733     %T0      EQUALS    $REGORG                                                    $REG    2
                    734     %T1      EQUALS    $REGORG                                                    $REG    2
                    735     %T2      EQUALS    $REGORG                                                    $REG    2
                    736     %T3      EQUALS    $REGORG                                                    $REG    2
                                     EXT       DEBUG                                                  TAPELOAD.50
                                                                                                      TAPELOAD.51
                                                                                                      TAPELOAD.52
```

```
                                                                                TAPELOAD.58
                                 ***********************************************************TAPELOAD.59
                                 *                                                         *TAPELOAD.60
                                 *       The initial section of code saves A, C, B, E, the exit stack, *TAPELOAD.61
                                 *       operand registers 700-777 in local memory beginning at loc    *TAPELOAD.62
                                 *       zero (overwriting the code there), and clears channels 3-47.  *TAPELOAD.63
                                 *       This block of information is ultimately written to buffer     *TAPELOAD.64
                                 *       memory for use by DMP, if required.  To minimize the amount   *TAPELOAD.65
                                 *       of information lost, the code is restricted to using operand  *TAPELOAD.66
                                 *       registers BASE through SCR3.                                  *TAPELOAD.67
                                 *                                                         *TAPELOAD.68
                                 ***********************************************************TAPELOAD.69
                                                                                TAPELOAD.70
                                         BASEREG                                                TAPELOAD.71
                                                                                TAPELOAD.72
 0   024717                              R!SCR1 = A                  .GET A                    TAPELOAD.73
 1   004020  024720                      R!SCR2 = A > D'16           .GET C                    TAPELOAD.74
 3   010000  024721                      R!SCR3 = 0                  .START ADDRESS            TAPELOAD.75
 5   020717  034721                      (R!SCR3) = R!SCR1           .SAVE A                   TAPELOAD.76
 7   026721                              R!SCR3 = R!SCR3 + 1         .NEXT                     TAPELOAD.77
10   020720  034721                      (R!SCR3) = R!SCR2           .SAVE C                   TAPELOAD.78
12   026721                              R!SCR3 = R!SCR3 + 1         .NEXT POSITION            TAPELOAD.79
13   050000  034721                      (R!SCR3) = B                .SAVE B                   TAPELOAD.80
                                                                                TAPELOAD.81
15   026721                              R!SCR3 = R!SCR3 + 1         .NEXT POSITION            TAPELOAD.82
16    <macro>                            EGET ((R!SCR3))             .SAVE E REGISTER          TAPELOAD.83
23    <macro>                            EPUT 0                                                TAPELOAD.84
                                                                                TAPELOAD.85
30   010000  024716                      R!BASE = 0                  .SET UP A BASE REGISTER   TAPELOAD.86
                                         BASEREG  BASE                                         TAPELOAD.87
                                                                                TAPELOAD.88
32                               SAVE    *                                                     TAPELOAD.89
                                                                                TAPELOAD.90
32   026721                              R!SCR3 = R!SCR3 + 1         .NEXT SAVE LOCATION       TAPELOAD.91
33    <macro>                            EXSGET ((R!SCR3))           .SAVE EXIT STACK ENTRY    TAPELOAD.92
40    <macro>                            EINCR                                                 TAPELOAD.93
46   041000  013020  104016              P = SAVE, A < D'16          .LOOP UNTIL ENTIRE STACK SAVED TAPELOAD.94
                                                                                TAPELOAD.95
                                                                                TAPELOAD.98
                                         *                                                     TAPELOAD.99
                                         * Clear channels 3 - 47.                              TAPELOAD.100
                                         *                                                     TAPELOAD.101
```

```
                                                                                                TAPELOAD.102
 51   010003  054000                    B = 3                          .FIRST CHANNEL NUMBER     TAPELOAD.103
                                                                                                TAPELOAD.104
 53    <macro>                          $UNTIL (B = 50)                                         TAPELOAD.105
 56   160000                              IOB:0                        .CLEAR CHANNEL            TAPELOAD.106
 57   166000                              IOB:6                        .DISABLE CHANNEL INTERRUPTS TAPELOAD.107
 60   056000                              B = B + 1                                              TAPELOAD.108
 61    <macro>                          $ENDTIL                                                 TAPELOAD.109
                                                                                                TAPELOAD.110
                                        *                                                       TAPELOAD.111
                                        * Save operand registers 700 - 777                      TAPELOAD.112
                                        *                                                       TAPELOAD.113
                                                                                                TAPELOAD.114
 62   010700  054000                    B = 700                        .FIRST REGISTER TO SAVE   TAPELOAD.115
 64  .075716 /000124                    P = OPSAVE                                             TAPELOAD.116
                                                                                                TAPELOAD.117
 25w                                    ORG        124/4               .SPACE FOR REGISTER DUMP  TAPELOAD.118
                                                                                                TAPELOAD.119
124                             OPSAVE  *                                                       TAPELOAD.120
                                                                                                TAPELOAD.121
124    <macro>                          $UNTIL (B = 0)                                          TAPELOAD.122
126   026721                              R!SCR3 = R!SCR3 + 1          .NEXT SAVE LOCATION       TAPELOAD.123
127   060000  034721                      (R!SCR3) = (B)               .SAVE NEXT OPERAND REGISTER TAPELOAD.124
131   056000                              B = B + 1                                              TAPELOAD.125
132    <macro>                          $ENDTIL                                                 TAPELOAD.126
                                                                                                TAPELOAD.127
                                        *                                                       TAPELOAD.128
                                        * Write first 2000 (octal) words of memory to buffer memory TAPELOAD.129
                                        *                                                       TAPELOAD.130
                                                                                                TAPELOAD.131
133   140005                            MOS:0                          .CLEAR                    TAPELOAD.132
134   010000  011777                    A = REG$MOSL & O'777           .LOWER ADDRESS            TAPELOAD.133
136   143005                            MOS:3                          .ENTER                    TAPELOAD.134
137   010000  004011  024717            R!SCR1 = REG$MOSL > D'9        .UNPACK ADDRESS           TAPELOAD.135
142   010000  005007  022717            A = REG$MOSU < D'7 + R!SCR1    .UPPER ADDRESS            TAPELOAD.136
145   142005                            MOS:2                          .ENTER                    TAPELOAD.137
146   010000                            A = 0                          .LOCAL ADDRESS            TAPELOAD.138
147   141005                            MOS:1                                                    TAPELOAD.139
150   014000 /002000                    A = O'2000                                               TAPELOAD.140
152   145005                            MOS:5                          .ENTER TRANSFER WORD COUNT TAPELOAD.141
153   041005  105001                    WAIT, MOS = BZ                 .WAIT UNTIL DONE          TAPELOAD.142
                                                                                                TAPELOAD.143
```

```
                                                                                    TAPELOAD.145
                                 ***********************************************************TAPELOAD.146
                                 *                                                         *TAPELOAD.147
                                 *        All necessary resources have been saved in buffer memory.  *TAPELOAD.148
                                 *        The following sections of code may use other operand       *TAPELOAD.149
                                 *        registers besides BASE through SCR3.                       *TAPELOAD.150
                                 *                                                         *TAPELOAD.151
                                 ***********************************************************TAPELOAD.152
                                                                                    TAPELOAD.153
                                          *                                         TAPELOAD.154
                                          * Prompt for and input the tape file number (1-99)  TAPELOAD.155
                                          *                                         TAPELOAD.156
                                                                                    TAPELOAD.157
155   010000  024722                     R!DBG = OFF$DBUG              .DEFAULT DEBUG OPTION TO OFF  TAPELOAD.158
                                                                                    TAPELOAD.159
157                              GETFID   *                                         TAPELOAD.160
                                                                                    TAPELOAD.161
157   014000 /000343  024726             R!MSG = MSGFILE                            TAPELOAD.162
162   072314                             R = SEND                      .OUTPUT PROMPT MESSAGE  TAPELOAD.163
                                                                                    TAPELOAD.164
163   010000  024723                     R!FID = 0                     .FILE NUMBER PRESET  TAPELOAD.165
165   010000  024724                     R!CNT = 0                     .FILE NUMBER CHAR COUNTER  TAPELOAD.166
                                                                                    TAPELOAD.167
167    <macro>                           $UNTIL (R!CNT > 2)            .MAX LENGTH EXCEEDED  TAPELOAD.168
174   072322                               R = RECV                    .INPUT NEXT CHARACTER  TAPELOAD.169
175    <macro>                             $IF (R!CHAR = CC$CTD)       .IF CONTROL-D ENTERED  TAPELOAD.170
                                                                                  .>>>>> 0          $IF      1
201   014000 /000353  024726               R!MSG = MSGDBUG                        TAPELOAD.171
204   072272                               R = SEND                    .OUTPUT DEBUG MESSAGE  TAPELOAD.172
205   010001  024722                       R!DBG = ON$DBUG             .SET DEBUG REQUEST FLAG  TAPELOAD.173
207   071030                               P = GETFID                  .TRY AGAIN  TAPELOAD.174
210    <macro>                             $ELSEIF (R!CHAR = CC$CR) OR (R!CHAR = CC$LF) .IF TERMINATOR  TAPELOAD.175
220    <macro>                               $IF (R!FID = 0) OR (R!FID > D'99) .IF ILLEGAL FILE NUMBER  TAPELOAD.176
                                                                                  .>>>>> 1          $IF      1
230   071051                                 P = GETFID                  .TRY AGAIN  TAPELOAD.177
231    <macro>                               $ENDIF                                 TAPELOAD.178
                                                                                  .<<<<< 1          $ENDIF   1
231   070031                                 P = FIND                    .FIND FILE  TAPELOAD.179
232    <macro>                             $ELSEIF (R!CHAR < '0'R) OR (R!CHAR > '9'R) .IF NON NUMERIC  TAPELOAD.180
244   071065                                 P = GETFID                  .TRY AGAIN  TAPELOAD.181
245    <macro>                             $ELSE                                  TAPELOAD.182
246   026724                                 R!CNT = R!CNT + 1           .INC CHAR COUNTER  TAPELOAD.183
```

```
247   020723  005001  024717          R!SCR1 = R!FID - 1                                    TAPELOAD.184
252   020723  005003  022717          R!FID = R!FID < 3 + R!SCR1 + R!CHAR - '0'R .10X + DIGIT  TAPELOAD.185
      022727  013060  024723
260    <macro>                       $ENDIF                                                  TAPELOAD.186
                                                                          .<<<<< 0                $ENDIF    1
260    <macro>                     $ENDTIL                                                   TAPELOAD.187
261   071102                       P = GETFID                  .TRY AGAIN                    TAPELOAD.188
                                                                                             TAPELOAD.189
                                   *                                                         TAPELOAD.190
                                   * Find requested file on the tape                        TAPELOAD.191
                                   *                                                         TAPELOAD.192
                                                                                             TAPELOAD.193
262                        FIND    *                                                         TAPELOAD.194
                                                                                             TAPELOAD.195
262   140017                       EXB:0                       .CLEAR THE CHANNEL            TAPELOAD.196
263   072334                       R = CHWT                                                  TAPELOAD.197
                                                                                             TAPELOAD.198
264    <macro>                     DEVICE I@XMI                .SET DEVICE CODE              TAPELOAD.199
266   072304                       R = REWIND                  .REWIND TAPE                  TAPELOAD.200
                                                                                             TAPELOAD.201
267    <macro>                     $UNTIL (R!FID = 0)          .TAPE POSITIONED TO FILE      TAPELOAD.202
271   010001  024724                R!CNT = 1                                                TAPELOAD.203
273   072310                        R = FORWARD                 .SPACE FORWARD 1 RECORD       TAPELOAD.204
274    <macro>                      $IF (R!EOF # 0)             .IF DOUBLE EOF FOUND          TAPELOAD.205
                                                                                  .>>>>> 0          $IF       1
277   014000 /000401  024726         R!MSG = MSGNOFL                                          TAPELOAD.206
302   072174                         R = SEND                    .FILE NOT FOUND MESSAGE      TAPELOAD.207
303   071124                         P = GETFID                  .TRY AGAIN                   TAPELOAD.208
304    <macro>                      $ENDIF                                                   TAPELOAD.209
                                                                                  .<<<<< 0          $ENDIF    1
304    <macro>                      $UNTIL (R!EOF # 0)          .EOF REACHED                 TAPELOAD.210
306   010000  024724                 R!CNT = 0                                               TAPELOAD.211
310   072273                         R = FORWARD                 .SPACE FORWARD 65536 RECORDS TAPELOAD.212
311    <macro>                      $ENDTIL                                                  TAPELOAD.213
312   027723                        R!FID = R!FID - 1           .DEC FILE COUNTER            TAPELOAD.214
313    <macro>                     $ENDTIL                                                   TAPELOAD.215
                                                                                             TAPELOAD.216
                                   ***********************************************************TAPELOAD.217
                                   *                                                         *TAPELOAD.218
                                   *        The tape is positioned to the proper file. The file will *TAPELOAD.219
                                   *        be read into location zero.  In order to avoid overwriting *TAPELOAD.220
                                   *        the loader code, we must relocate it to high memory.  *TAPELOAD.221
```

```
                                 *                                                                    *TAPELOAD.222
                                 ************************************************************************TAPELOAD.223
                                                                                                       TAPELOAD.224
                                                                                                       TAPELOAD.225
                      -1000      $$X      SET  -XMT$SZ                                                 R14673BB.1
314     <macro>                          COPY 0,$$X,LOADEND            .RELOCATE                      R14673BB.2
                                                                                                       TAPELOAD.227
334   010000  017000 /001000             R!%T0 = 0 - XMT$SZ + LOAD2    .SET ENTRY ADDRESS             R14673BB.3
      016000 /000413  024733
342   074733                             P = R!%T0                     .ENTER AT LOAD2                TAPELOAD.229
                                                                                                       TAPELOAD.230
                                 ************************************************************************TAPELOAD.231
                                                                                                       TAPELOAD.232
                                          *                                                            TAPELOAD.233
                                          * Prompt message                                            TAPELOAD.234
                                          *                                                            TAPELOAD.235
                                                                                                       TAPELOAD.236
343                              MSGFILE  *                                                            TAPELOAD.237
 70w                    005015           VWD   D'8/CC$LF,D'8/CC$CR                                    TAPELOAD.238
344                                       'FILE @MT0:'                                                 TAPELOAD.239
352   000000                              0                                                            TAPELOAD.240
                                                                                                       TAPELOAD.241
353                              MSGDBUG  *                                                            TAPELOAD.242
 72w                    005015           VWD   D'8/CC$LF,D'8/CC$CR                                    TAPELOAD.243
354                                       'DEBUG'                                                      TAPELOAD.244
357   000000                              0                                                            TAPELOAD.245
                                                                                                       TAPELOAD.246
                                          *                                                            TAPELOAD.247
                                          * Error messages                                            TAPELOAD.248
                                          *                                                            TAPELOAD.249
                                                                                                       TAPELOAD.250
360                              MSGABRT  *                                                            TAPELOAD.251
 74w   005015                             VWD   D'8/CC$LF,D'8/CC$CR                                    TAPELOAD.252
361                                       'TAPE ERROR: DEADSTART ABORTED'                              TAPELOAD.253
400   000000                              0                                                            TAPELOAD.254
                                                                                                       TAPELOAD.255
401                              MSGNOFL  *                                                            TAPELOAD.256
100w          024064                     VWD   D'8/CC$LF,D'8/CC$CR                                    TAPELOAD.257
402                                       'FILE NOT FOUND'                                             TAPELOAD.258
412   000000                              0                                                            TAPELOAD.259
                                                                                                       TAPELOAD.260
                                 ************************************************************************TAPELOAD.261
```

```
                                                                                                TAPELOAD.262
413                                 LOAD2    *                                                  TAPELOAD.263
                                                                                                TAPELOAD.264
413   010000  017000 /001000                 R!BASE = 0 - XMT$SZ          .SET BASE REGISTER TO HIGH MEMORYR14673BB.4
      024716
                                             BASEREG BASE                                       TAPELOAD.266
                                                                                                TAPELOAD.267
417   010000  024725                         R!BUF = 0                    .BEGIN LOADING AT LOCATION 0  TAPELOAD.268
421   014000 /010000  024724                 R!CNT = O'10000              .WORD COUNT           TAPELOAD.269
                                                                                                TAPELOAD.270
424   010000  024731                         R!EOF = 0                    .EOF PRESET           TAPELOAD.271
426    <macro>                               $UNTIL (R!EOF # 0)           .ALL LOADED           TAPELOAD.272
430   072172                                   R = RDYWT                  .WAIT FOR DEVICE READY TAPELOAD.273
                                                                                                TAPELOAD.274
431    <macro>                                 DOC R!CNT,C                .ENTER WORD COUNT (COMPLEMENT) TAPELOAD.275
434   072163                                   R = CHWT                                           TAPELOAD.276
435    <macro>                                 DOB R!BUF                  .ENTER LOCAL MEMORY ADDRESS TAPELOAD.277
437   072160                                   R = CHWT                                           TAPELOAD.278
440    <macro>                                 DOA MU$RD                  .ENTER READ FUNCTION  TAPELOAD.279
442   072155                                   R = CHWT                                           TAPELOAD.280
443    <macro>                                 STR                        .ISSUE START          TAPELOAD.281
445   072152                                   R = CHWT                                           TAPELOAD.282
                                                                                                TAPELOAD.283
446   072171                                   R = INTWT                  .WAIT FOR INTERRUPT   TAPELOAD.284
                                                                                                TAPELOAD.285
447    <macro>                                 DIB                        .REQUEST BUFFER ADDRESS TAPELOAD.286
450   072147                                   R = CHWT                                           TAPELOAD.287
451    <macro>                                 BUS R!BUF                  .READ ADDRESS         TAPELOAD.288
453    <macro>                               $ENDTIL                                            TAPELOAD.289
                                                                                                TAPELOAD.290
                                             *                                                  TAPELOAD.291
                                             * Set up registers for loaded program             TAPELOAD.292
                                             *                                                  TAPELOAD.293
                                                                                                TAPELOAD.294
454    <macro>                               $IF (R!DBG = ON$DBUG)        .IF DEBUG FLAG FOR KERNEL TAPELOAD.295
                                                                                        .>>>>> 0        $IF      1
460   014000 /000000  024733                   (DEBUG) = ON$DBUG          .SET FLAG ON          TAPELOAD.296
      010001  034733
465    <macro>                               $ENDIF                                             TAPELOAD.297
                                                                                        .<<<<< 0        $ENDIF   1
465   010000  024717                         R!%STRT = XMT$STRT           .TAPELOAD IS CALLER FOR DMP TAPELOAD.298
467   010000  024720                         R!%MOSL = REG$MOSL           .REG SAVE ADDR IN MOS FOR DMP TAPELOAD.299
```

```
471   010000  024721                R!%MOSU = REG$MOSU            .                          TAPELOAD.300
                                                                                             TAPELOAD.301
473   010000  024733                R!%TO = 0                     .SET ENTRY ADDRESS         TAPELOAD.302
475   074733                        P = R!%TO                     .ENTER ROUTINE             TAPELOAD.303
                                                                                             TAPELOAD.304
                            ***********************************************************************TAPELOAD.305
                                                                                             TAPELOAD.306
                                    *                                                        TAPELOAD.307
                                    * Send message to console                                TAPELOAD.308
                                    *                                                        TAPELOAD.309
                                                                                             TAPELOAD.310
476                         SEND    *                                                        TAPELOAD.311
                                                                                             TAPELOAD.312
476   030726  024730                R!CHARS = (R!MSG)             .GET CHARACTERS            TAPELOAD.313
500   020730  004010  024727        R!CHAR = R!CHARS > D'8        .CHECK HIGH BYTE           TAPELOAD.314
503   103002  001000                EXIT, A = 0                   .IF END OF MESSAGE         TAPELOAD.315
505   072060                        R = ECHO                                                 TAPELOAD.316
506   020730  011377  024727        R!CHAR = R!CHARS & O'377      .CHECK LOW BYTE            TAPELOAD.317
511   103002  001000                EXIT, A = 0                   .IF END OF MESSAGE         TAPELOAD.318
513   072052                        R = ECHO                                                 TAPELOAD.319
514   026726                        R!MSG = R!MSG + 1             .NEXT PARCEL ADDRESS       TAPELOAD.320
515   071017                        P = SEND                                                 TAPELOAD.321
                                                                                             TAPELOAD.322
                                                                                             TAPELOAD.323
                                    *                                                        TAPELOAD.324
                                    * Input a character and echo it on the screen.  On exit, TAPELOAD.325
                                    * the character is in register CHAR.                    TAPELOAD.326
                                    *                                                        TAPELOAD.327
                                                                                             TAPELOAD.328
516                         RECV    *                                                        TAPELOAD.329
                                                                                             TAPELOAD.330
516   040046  104001                WAIT, TID # DN                .WAIT UNTIL DONE           TAPELOAD.331
520   150046                        TID:10                        .READ INPUT                TAPELOAD.332
521   024727                        R!CHAR = A                    .SAVE                      TAPELOAD.333
522    <macro>                      $IF (R!CHAR = CC$BS) OR (R!CHAR = CC$DEL) .IF BS OR DEL  TAPELOAD.334
                                                                                .>>>>> 0          $IF      1
531    <macro>                        $IF (R!CNT # 0)             .IF NOT FIRST CHAR         TAPELOAD.335
                                                                                .>>>>> 1          $IF      1
534   010010  024727                   R!CHAR = CC$BS              .SEND BACKSPACE            TAPELOAD.336
536   072027                           R = ECHO                                               TAPELOAD.337
537   010040  024727                   R!CHAR = ' 'R               .SEND SPACE                TAPELOAD.338
541   072024                           R = ECHO                                               TAPELOAD.339
```

```
542   010010  024727                    R!CHAR = CC$BS            .SEND BACKSPACE                TAPELOAD.340
544   072021                            R = ECHO                                                 TAPELOAD.341
545   010000  024717                    R!SCR1 = 0                .DIVIDE FID BY 10              TAPELOAD.342
547    <macro>                          $UNTIL (R!FID < D'10)     .REMAINDER LESS THAN 10        TAPELOAD.343
552   020723  013012  024723              R!FID = R!FID - D'10    .DIVIDE                        TAPELOAD.344
555   026717                              R!SCR1 = R!SCR1 + 1     .INC QUOTIENT                  TAPELOAD.345
556    <macro>                          $ENDTIL                                                  TAPELOAD.346
557   020717  024723                    R!FID = R!SCR1            .SET FID TO QUOTIENT           TAPELOAD.347
561   027724                            R!CNT = R!CNT - 1         .DEC CHARACTER COUNTER         TAPELOAD.348
562    <macro>                        $ENDIF                                                     TAPELOAD.349
                                                                                   .<<<<< 1            $ENDIF   1
562   071044                          P = RECV                    .TRY AGAIN                     TAPELOAD.350
563    <macro>                      $ENDIF                                                       TAPELOAD.351
                                                                                   .<<<<< 0            $ENDIF   1
563   072002                        R = ECHO                      .ECHO THE CHARACTER            TAPELOAD.352
564   001000                        EXIT                                                         TAPELOAD.353
                                                                                                 TAPELOAD.354
                                                                                                 TAPELOAD.355
                                    *                                                            TAPELOAD.356
                                    * Send character to console                                  TAPELOAD.357
                                    *                                                            TAPELOAD.358
                                                                                                 TAPELOAD.359
565                         ECHO    *                                                            TAPELOAD.360
                                                                                                 TAPELOAD.361
565   041047  105001                WAIT, TOD = BZ                .WAIT UNTIL CHANNEL NOT BUSY   TAPELOAD.362
567   020727                        A = R!CHAR                                                   TAPELOAD.363
570   154047                        TOD:14                        .WRITE OUT CHARACTER           TAPELOAD.364
571   001000                        EXIT                                                         TAPELOAD.365
                                                                                                 TAPELOAD.366
                                                                                                 TAPELOAD.367
                                    *                                                            TAPELOAD.368
                                    * Issue rewind function to tape                              TAPELOAD.369
                                    *                                                            TAPELOAD.370
                                                                                                 TAPELOAD.371
572                         REWIND  *                                                            TAPELOAD.372
                                                                                                 TAPELOAD.373
572    <macro>                      DOA MU$RWND                   .ENTER REWIND                  TAPELOAD.374
574   072023                        R = CHWT                                                     TAPELOAD.375
575    <macro>                      SIR                           .ISSUE START                   TAPELOAD.376
577   072020                        R = CHWT                                                     TAPELOAD.377
                                                                                                 TAPELOAD.378
600   072071                        R = BOTWT                     .WAIT FOR BEGINNING OF TAPE    TAPELOAD.379
```

```
601   072021                          R = RDYWT                    .WAIT FOR UNIT READY            TAPELOAD.380
602   001000                          EXIT                                                         TAPELOAD.381
                                                                                                   TAPELOAD.382
                                                                                                   TAPELOAD.383
                                      *                                                            TAPELOAD.384
                                      * Issue space forward function to tape                      TAPELOAD.385
                                      *                                                            TAPELOAD.386
                                                                                                   TAPELOAD.387
603                           FORWARD *                                                            TAPELOAD.388
                                                                                                   TAPELOAD.389
603    <macro>                        DOC R!CNT,C                  .ENTER RECORD COUNT (COMPLEMENT) TAPELOAD.390
606   072011                          R = CHWT                                                     TAPELOAD.391
607    <macro>                        DOA MU$SPFW                  .ENTER SPACE FORWARD            TAPELOAD.392
611   072006                          R = CHWT                                                     TAPELOAD.393
612    <macro>                        STR                          .ISSUE START                    TAPELOAD.394
614   072003                          R = CHWT                                                     TAPELOAD.395
                                                                                                   TAPELOAD.396
615   072022                          R = INTWT                    .WAIT FOR INTERRUPT             TAPELOAD.397
616   001000                          EXIT                                                         TAPELOAD.398
                                                                                                   TAPELOAD.399
                                                                                                   TAPELOAD.400
                                      *                                                            TAPELOAD.401
                                      * Wait for channel to accept command                        TAPELOAD.402
                                      *                                                            TAPELOAD.403
                                                                                                   TAPELOAD.404
617                           CHWT    *                                                            TAPELOAD.405
                                                                                                   TAPELOAD.406
617   041017  105001                  WAIT, EXB = BZ               .WAIT FOR CHANNEL               TAPELOAD.407
621   001000                          EXIT                                                         TAPELOAD.408
                                                                                                   TAPELOAD.409
                                                                                                   TAPELOAD.410
                                      *                                                            TAPELOAD.411
                                      * Wait for device ready                                      TAPELOAD.412
                                      *                                                            TAPELOAD.413
                                                                                                   TAPELOAD.414
622                           RDYWT   *                                                            TAPELOAD.415
                                                                                                   TAPELOAD.416
622    <macro>                        DIA                          .REQUEST STATUS WORD            TAPELOAD.417
623   073004                          R = CHWT                                                     TAPELOAD.418
624    <macro>                        BUS R!STAT                   .READ STATUS                    TAPELOAD.419
                                                                                                   TAPELOAD.420
626    <macro>                        $IF (0 = R!STAT & 1)         .IF NOT READY                   TAPELOAD.421
```

```
                                                                                .>>>>> 0          $IF      1
635   071013                          P = RDYWT                   .TRY AGAIN                    TAPELOAD.422
636    <macro>                       $ENDIF                                                     TAPELOAD.423
                                                                                .<<<<< 0          $ENDIF   1
636   001000                          EXIT                                                       TAPELOAD.424
                                                                                                 TAPELOAD.425
                                                                                                 TAPELOAD.426
                                      *                                                          TAPELOAD.427
                                      * Wait for interrupt                                       TAPELOAD.428
                                      *                                                          TAPELOAD.429
                                                                                                 TAPELOAD.430
637                           INTWT   *                                                          TAPELOAD.431
                                                                                                 TAPELOAD.432
637    <macro>                        STAT1                                                      TAPELOAD.433
640   005001                          A = A < 1                                                  TAPELOAD.434
641   104002                          P = INTWT, C = 0            .WAIT FOR FUNCTION COMPLETION  TAPELOAD.435
                                                                                                 TAPELOAD.436
                                      * Get device status                                        TAPELOAD.437
                                                                                                 TAPELOAD.438
642    <macro>                        DIA                         .REQUEST DEVICE STATUS         TAPELOAD.439
643   073024                          R = CHWT                                                   TAPELOAD.440
644    <macro>                        BUS R!STAT                  .READ STATUS                   TAPELOAD.441
                                                                                                 TAPELOAD.442
646   020732  005001                  A = R!STAT < 1                                             TAPELOAD.443
650    <macro>                        $IF (C = 0)                 .IF NOT ERROR                  TAPELOAD.444
                                                                                .>>>>> 0          $IF      1
652   010000  024731                    R!EOF = 0                 .STATUS                        TAPELOAD.445
654    <macro>                        $ELSEIF (0 # R!STAT & O'400) .IF EOF                       TAPELOAD.446
664   010001  024731                    R!EOF = 1                 .STATUS                        TAPELOAD.447
666    <macro>                        $ELSE                       .OTHER ERROR                   TAPELOAD.448
667   070034                            P = ABORT                                                TAPELOAD.449
670    <macro>                        $ENDIF                                                     TAPELOAD.450
                                                                                .<<<<< 0          $ENDIF   1
670   001000                          EXIT                                                       TAPELOAD.451
                                                                                                 TAPELOAD.452
                                                                                                 TAPELOAD.453
                                      *                                                          TAPELOAD.454
                                      * Wait for beginning of tape                               TAPELOAD.455
                                      *                                                          TAPELOAD.456
                                                                                                 TAPELOAD.457
671                           BOTWT   *                                                          TAPELOAD.458
                                                                                                 TAPELOAD.459
```

```
671    <macro>                          DIA                          .REQUEST DEVICE STATUS          TAPELOAD.460
672    073053                           R = CHWT                                                     TAPELOAD.461
673    <macro>                          BUS R!STAT                   .READ STATUS                    TAPELOAD.462
                                                                                                     TAPELOAD.463
675    <macro>                          $IF (0 # R!STAT & O'20000)   .IF STILL REWINDING             TAPELOAD.464
                                                                                         .>>>>> 0          $IF      1
705    071014                             P = BOTWT                  .TRY AGAIN                      TAPELOAD.465
706    <macro>                          $ENDIF                                                       TAPELOAD.466
                                                                                         .<<<<< 0          $ENDIF   1
                                                                                                     TAPELOAD.467
                                        * Wait for load point                                        TAPELOAD.468
                                                                                                     TAPELOAD.469
706                            LOADPT   *                                                            TAPELOAD.470
                                                                                                     TAPELOAD.471
706    <macro>                          DIA                                                          TAPELOAD.472
707    073070                           R = CHWT                                                     TAPELOAD.473
710    <macro>                          BUS R!STAT                                                   TAPELOAD.474
                                                                                                     TAPELOAD.475
712    <macro>                          $IF (0 = R!STAT & 200)                                       TAPELOAD.476
                                                                                         .>>>>> 0          $IF      1
721    071013                             P = LOADPT                                                 TAPELOAD.477
722    <macro>                          $ENDIF                                                       TAPELOAD.478
                                                                                         .<<<<< 0          $ENDIF   1
722    001000                           EXIT                                                         TAPELOAD.479
                                                                                                     TAPELOAD.480
                                                                                                     TAPELOAD.481
                                        *                                                            TAPELOAD.482
                                        * Abort deadstart                                            TAPELOAD.483
                                        *                                                            TAPELOAD.484
                                                                                                     TAPELOAD.485
723                            ABORT    *                                                            TAPELOAD.486
                                                                                                     TAPELOAD.487
723    020716  016000 /000360           R!MSG = R!BASE + MSGABRT     .ABORT MESSAGE                  TAPELOAD.488
       024726
727    073231                           R = SEND                     .SEND                           TAPELOAD.489
730    070000                           WAIT                         .HANG                           TAPELOAD.490
                                                                                                     TAPELOAD.491
                                                                                                     TAPELOAD.492
731                            LOADEND  *                                                            TAPELOAD.493
                                        ERRIF  LOADEND,GT,XMT$SZ     .LOADER EXCEEDS MAX SIZE        R14673BB.5
                                                                                                     TAPELOAD.495
                                        END                                                          TAPELOAD.496
```

-1000	$$X		7: 5 D	7: 6						
737	$REGORG		1:40 D	1:41	1:41 D	1:42	1:42 D	2: 1	2: 1 D	2: 2
			2: 2 D	2: 3 D	2: 4	2: 4 D	2: 5	2: 5 D	2: 6	2: 6 D
			2: 7	2: 7 D	2: 8 F	2: 9	2: 9 D	2:10	2:10 D	2:11
			2:11 D	2:12	2:12 D	2:13 F	2:14	2:14 D	2:15	2:15 D
			2:16	2:16 D	2:17 F	2:18	2:18 D	2:19	2:19 D	2:20 F
			2:21	2:21 D	2:22	2:22 D	2:23	2:23 D	2:24	2:24 D
716	%DMP		1:41 D							
716	%DUMP	$APTEXT	1:40	2: 3						
720	%MOSL	$APTEXT	2: 1 D	8:42						
721	%MOSU	$APTEXT	2: 2 D	9: 1						
717	%STRT	$APTEXT	1:42 D	8:41						
733	%T0		1:38	2:21 D	7: 9	7:10	8:37	8:38	9: 3	9: 4
			12: 1	12:28	13: 6	13:20				
734	%T1		1:38	2:22 D	7: 6					
735	%T2		1:38	2:23 D	7: 6					
736	%T3		1:38	2:24 D	7: 6					
0	@EGET		3:26 D							
0	@EPUT		3:27 D	3:36 D						
0	@EXSGET		3:35 D							
723	ABORT		12:31	13:31 D						
716	BASE		2: 4 D	3:29	3:30	4:15 B	8: 4	8: 6	13:33	
671	BOTWT		10:42	12:41 D	13: 7					
725	BUF		2:12 D	8: 8	8:17	8:28				
10	CC$BS	$APTEXT	9:36	9:39	10: 1					
15	CC$CR	$APTEXT	5:32	7:19	7:24	7:33	7:38			
4	CC$CTD	$APTEXT	5:27							
177	CC$DEL	$APTEXT	9:36							
12	CC$LF	$APTEXT	5:32	7:19	7:24	7:33	7:38			
727	CHAR		2:15 D	5:27	5:32	5:39	6: 2	9:15	9:18	9:34
			9:36	9:39	9:41	10: 1	10:26			
730	CHARS		2:16 D	9:14	9:15	9:18				
617	CHWT		6:16	8:16	8:18	8:20	8:22	8:27	10:38	10:40
			11:12	11:14	11:16	11:26 D	11:39	12:21	13: 2	13:16
724	CNT		2:11 D	5:22	5:24	5:42	6:22	6:32	8: 9	8:15
			9:38	10: 9	11:11					
722	DBG		2: 9 D	5:14	5:30	8:36				
0X	DEBUG		2:26 D	8:37						
565	ECHO		9:17	9:20	9:40	9:42	10: 2	10:15	10:23 D	
731	EOF		2:18 D	6:25	6:31	8:11	8:12	12:27	12:29	
17	EXB	$APTEXT	6:15	6:18	8:15	8:17	8:19	8:21	8:26	8:28
			10:37	10:39	11:11	11:13	11:15	11:28	11:38	11:40

			12:14	12:20	12:22	13: 1	13: 3	13:15	13:17	
723	FID		2:10 D	5:21	5:34	6: 1	6: 2	6: 3	6:21	6:35
			10: 4	10: 5	10: 8					
262	FIND		5:38	6:13 D						
603	FORWARD		6:23	6:33	11: 9 D					
157	GETFID		5:16 D	5:31	5:35	5:40	6: 7	6:28		
22	I@XMT	$APTEXT	6:18							
637	INTWT		8:24	11:18	12:12 D	12:16				
413	LOAD2		7: 9	8: 2 D						
731	LOADEND		7: 6	13:39 D	13:40 F					
706	LOADPT		13:13 D	13:21						
5	MOS	$APTEXT	4:31	4:33	4:36	4:38	4:40	4:41		
726	MSG		2:14 D	5:18	5:28	6:26	9:14	9:21	13:33	
360	MSGABRT		7:32 D	13:33						
353	MSGDBUG		5:28	7:23 D						
343	MSGFILE		5:18	7:18 D						
401	MSGNOFL		6:26	7:37 D						
0	MU$RD	$APTEXT	8:19							
10	MU$RWND	$APTEXT	10:37							
30	MU$SPEW	$APTEXT	11:13							
0	OFF$DBUG	$APTEXT	5:14							
1	ON$DBUG	$APTEXT	5:30	8:36	8:37					
124	OPSAVE		4:15	4:19 D						
2	PXS	$APTEXT	3:26	3:27	3:35	3:36				
622	RDYWT		8:13	11: 1	11:36 D	12: 2				
516	RECV		5:25	9:30 D	10:12					
0	REG$MOSL	$APTEXT	4:32	4:34	8:42					
0	REG$MOSU	$APTEXT	4:35	9: 1						
572	REWIND		6:19	10:35 D						
32	SAVE		3:32 D	3:37						
717	SCR1		2: 5 D	3:16	3:19	4:34	4:35	6: 1	6: 2	10: 3
			10: 6	10: 8						
720	SCR2		2: 6 D	3:17	3:21					
721	SCR3		2: 7 D	3:18	3:19	3:20	3:21	3:22	3:23	3:25
			3:26	3:34	3:35	4:22	4:23			
476	SEND		5:19	5:29	6:27	9:12 D	9:22	13:35		
732	STAT		2:19 D	11:40	12: 1	12:22	12:24	12:28	13: 3	13: 6
			13:17	13:20						
46	TID	$APTEXT	9:32	9:33						
47	TOD	$APTEXT	10:25	10:27						
0	XMT$STRT	$APTEXT	8:41							
1000	XMT$SZ	$APTEXT	7: 5	7: 8	8: 4	13:40 F				

```
      IDENT DISKLOAD                                                    DISKLOAD.5
      ABS                                                               DISKLOAD.6
                                                                        DISKLOAD.7
      COMMENT   'Copyright (C) Cray Research, Inc., 1984'                 COPYRT.2
                                                                        DISKLOAD.9
*********************************************************************DISKLOAD.10
*                                                                   *DISKLOAD.11
*                        D I S K   L O A D                          *DISKLOAD.12
*                                                                   *DISKLOAD.13
*  DISKLOAD is a stand-alone program which deadstarts the I/O       *DISKLOAD.14
*  Subsystem from an expander disk file.                            *DISKLOAD.15
*                                                                   *DISKLOAD.16
*  DISKLOAD is deadstarted into the MIOP from the first 64 words of *DISKLOAD.17
*  block 0 on disk.                                                 *DISKLOAD.18
*                                                                   *DISKLOAD.19
*  DISKLOAD saves information required by the DMP program in buffer *DISKLOAD.20
*  memory.  The location is an assembly constant (REG$MOSU, REG$MOSL). *DISKLOAD.21
*  This value is passed to DMP in registers %MOSU and %MOSL.        *DISKLOAD.22
*                                                                   *DISKLOAD.23
*  DISKLOAD prompts the user for the name of the file to be loaded  *DISKLOAD.24
*  with the 'FILE @DK0:' message. The response must be either:      *DISKLOAD.25
*                                                                   *DISKLOAD.26
*    Control-D  - DISKLOAD displays the message 'DEBUG' on the console *DISKLOAD.27
*                 and will pass the value ON$DBUG to the loaded     *DISKLOAD.28
*                 program in address DEBUG.  DISKLOAD will reprompt *DISKLOAD.29
*                 with the 'FILE @DK0:' message.                    *DISKLOAD.30
*                                                                   *DISKLOAD.31
*    DUMP       - DISKLOAD loads the DMP program into MIOP local   *DISKLOAD.32
*                 memory at address 0 and passes control to it.     *DISKLOAD.33
*                                                                   *DISKLOAD.34
*    <dir-name> - DISKLOAD displays a menu of all files in the      *DISKLOAD.35
*                 requested directory.  DISKLOAD will reprompt with *DISKLOAD.36
*                 the 'FILE @DK0:' message.                         *DISKLOAD.37
*                                                                   *DISKLOAD.38
*    <file-name>- DISKLOAD loads the specifed file into MIOP local  *DISKLOAD.39
*                 memory at address 0 and passes control to it.  It *DISKLOAD.40
*                 checks for an associated file of overlays with the *DISKLOAD.41
*                 name <file-name>.OV that is loaded to buffer memory, *DISKLOAD.42
*                 if found.                                         *DISKLOAD.43
*                                                                   *DISKLOAD.44
*  DISKLOAD passes the value XDK$STRT to the DMP program in register *DISKLOAD.45
*  %STRT to indicate that DISKLOAD is the caller.  It sets the flag in *DISKLOAD.46
```

```
                                 *  address STRT in the KERNEL to XDK$STRT to indicate that the caller  *DISKLOAD.47
                                 *  is DISKLOAD.                                                        *DISKLOAD.48
                                 *                                                                      *DISKLOAD.49
                                 ************************************************************************DISKLOAD.50
                                                                                                         DISKLOAD.51
                                          SCRATCH    %T0,%T1,%T2,%T3                                     DISKLOAD.52
                                                                                                         DISKLOAD.53
0     <macro>                    %DUMP    REGISTER   (%DMP,%STRT,%MOSL,%MOSU)  .FOR DMP                 DISKLOAD.54
                        716      %DMP     EQUALS     $REGORG                                                   $REG     2
                        717      %STRT    EQUALS     $REGORG                                                   $REG     2
                        720      %MOSL    EQUALS     $REGORG                                                   $REG     2
                        721      %MOSU    EQUALS     $REGORG                                                   $REG     2
0     <macro>                    %DUMP    REGISTER   (BASE,SCR1,SCR2,SCR3)     .FOR DISKLOAD            DISKLOAD.55
                        716      BASE     EQUALS     $REGORG                                                   $REG     2
                        717      SCR1     EQUALS     $REGORG                                                   $REG     2
                        720      SCR2     EQUALS     $REGORG                                                   $REG     2
                        721      SCR3     EQUALS     $REGORG                                                   $REG     2
0     <macro>                              REGISTER   (MOSL,MOSU,WRDS)          .LOAD1                   DISKLOAD.56
                        722      MOSL     EQUALS     $REGORG                                                   $REG     2
                        723      MOSU     EQUALS     $REGORG                                                   $REG     2
                        724      WRDS     EQUALS     $REGORG                                                   $REG     2
0     <macro>                              REGISTER   (CYL,HD,SCT,CNT,BUF)                                 DISKLOAD.57
                        725      CYL      EQUALS     $REGORG                                                   $REG     2
                        726      HD       EQUALS     $REGORG                                                   $REG     2
                        727      SCT      EQUALS     $REGORG                                                   $REG     2
                        730      CNT      EQUALS     $REGORG                                                   $REG     2
                        731      BUF      EQUALS     $REGORG                                                   $REG     2
0     <macro>                              REGISTER   (MSG,CHAR,CHARS)                                     DISKLOAD.58
                        732      MSG      EQUALS     $REGORG                                                   $REG     2
                        733      CHAR     EQUALS     $REGORG                                                   $REG     2
                        734      CHARS    EQUALS     $REGORG                                                   $REG     2
0     <macro>                              REGISTER   (FID,DIR,DBG,PNT,FLG)     .LOAD2                   DISKLOAD.59
                        735      FID      EQUALS     $REGORG                                                   $REG     2
                        736      DIR      EQUALS     $REGORG                                                   $REG     2
                        737      DBG      EQUALS     $REGORG                                                   $REG     2
                        740      PNT      EQUALS     $REGORG                                                   $REG     2
                        741      FLG      EQUALS     $REGORG                                                   $REG     2
0     <macro>                              REGISTER   (EOF,LBN,BLK,NUM)                                    DISKLOAD.60
                        742      EOF      EQUALS     $REGORG                                                   $REG     2
                        743      LBN      EQUALS     $REGORG                                                   $REG     2
                        744      BLK      EQUALS     $REGORG                                                   $REG     2
                        745      NUM      EQUALS     $REGORG                                                   $REG     2
```

```
        <macro>                         REGISTER   (DRE,LIM,DID,MCH)                                  DISKLOAD.61
                   746     DRE          EQUALS     $REGORG                                                  $REG     2
                   747     LIM          EQUALS     $REGORG                                                  $REG     2
                   750     DID          EQUALS     $REGORG                                                  $REG     2
                   751     MCH          EQUALS     $REGORG                                                  $REG     2
0       <macro>                         REGISTER   (DGP,DGX,DGB,DGC)                                     DISKLOAD.62
                   752     DGP          EQUALS     $REGORG                                                  $REG     2
                   753     DGX          EQUALS     $REGORG                                                  $REG     2
                   754     DGB          EQUALS     $REGORG                                                  $REG     2
                   755     DGC          EQUALS     $REGORG                                                  $REG     2
0       <macro>                         REGISTER   (QTU,QTL,RMU,RML)                                     DISKLOAD.63
                   756     QTU          EQUALS     $REGORG                                                  $REG     2
                   757     QTL          EQUALS     $REGORG                                                  $REG     2
                   760     RMU          EQUALS     $REGORG                                                  $REG     2
                   761     RML          EQUALS     $REGORG                                                  $REG     2
0       <macro>                         REGISTER   (SRC,DST,SIZ,PAT)                                     DISKLOAD.64
                   762     SRC          EQUALS     $REGORG                                                  $REG     2
                   763     DST          EQUALS     $REGORG                                                  $REG     2
                   764     SIZ          EQUALS     $REGORG                                                  $REG     2
                   765     PAT          EQUALS     $REGORG                                                  $REG     2
0       <macro>                         REGISTER   (%W1,%W2,%W3)                                         DISKLOAD.65
                   766     %W1          EQUALS     $REGORG                                                  $REG     2
                   767     %W2          EQUALS     $REGORG                                                  $REG     2
                   770     %W3          EQUALS     $REGORG                                                  $REG     2
0       <macro>                         REGISTER   (%T0,%T1,%T2,%T3)                                     DISKLOAD.66
                   771     %T0          EQUALS     $REGORG                                                  $REG     2
                   772     %T1          EQUALS     $REGORG                                                  $REG     2
                   773     %T2          EQUALS     $REGORG                                                  $REG     2
                   774     %T3          EQUALS     $REGORG                                                  $REG     2
                                                                                                         DISKLOAD.67
                                                                                                         DISKLOAD.68
                                        EXT         SIRT                                                 DISKLOAD.69
                                        EXT         DEBUG                                                DISKLOAD.70
                                                                                                         DISKLOAD.71
                                        * Equates                                                        DISKLOAD.72
                                                                                                         DISKLOAD.73
                   400     BISIZ        EQUALS    O'400                 .BASIC BOOT SIZE LIMIT           DISKLOAD.74
                     5     LDSIZ        EQUALS    XDK$SZ/BISIZ          .LOADER SIZE IN SECTORS          R14673BC.1
                                                                                                         DISKLOAD.77
```

```
                                                                                DISKLOAD.79
                        ***********************************************************************DISKLOAD.80
                        *                                                                     *DISKLOAD.81
                        *  The initial section of code saves A, C, B, E, the exit stack, and  *DISKLOAD.82
                        *  operand registers 700-777 in local memory starting at address 400. *DISKLOAD.83
                        *  This block of information is ultimately written to buffer memory   *DISKLOAD.84
                        *  for use by DMP, if required.  To minimize the amount of information *DISKLOAD.85
                        *  lost, the code is restricted to using operand registers BASE thru   *DISKLOAD.86
                        *  SCR3.                                                              *DISKLOAD.87
                        *                                                                     *DISKLOAD.88
                        ***********************************************************************DISKLOAD.89
                                                                                DISKLOAD.90
                                BASEREG                                         DISKLOAD.91
                                                                                DISKLOAD.92
 0   024717                     R!SCR1 = A                  .GET A              DISKLOAD.93
 1   004020  024720             R!SCR2 = A > D'16           .GET C              DISKLOAD.94
 3   010000  024721             R!SCR3 = 0                  .START ADDRESS      R14673BC.2
 5   020717  034721             (R!SCR3) = R!SCR1           .SAVE A             DISKLOAD.96
 7   026721                     R!SCR3 = R!SCR3 + 1         .NEXT               DISKLOAD.97
10   020720  034721             (R!SCR3) = R!SCR2           .SAVE C             DISKLOAD.98
12   026721                     R!SCR3 = R!SCR3 + 1         .NEXT POSITION      DISKLOAD.99
13   050000  034721             (R!SCR3) = B                .SAVE B             DISKLOAD.100
                                                                                DISKLOAD.101
15   026721                     R!SCR3 = R!SCR3 + 1         .NEXT POSITION      DISKLOAD.102
16    <macro>                   EGET ((R!SCR3))             .SAVE E REGISTER    DISKLOAD.103
23    <macro>                   EPUT 0                                          DISKLOAD.104
                                                                                DISKLOAD.105
30   010000  024716             R!BASE = 0                  .SET UP A BASE REGISTER DISKLOAD.106
                                BASEREG  BASE                                   DISKLOAD.107
                                                                                DISKLOAD.108
32                      SAVE    *                                               DISKLOAD.109
                                                                                DISKLOAD.110
32   026721                     R!SCR3 = R!SCR3 + 1         .NEXT SAVE LOCATION DISKLOAD.111
33    <macro>                   EXSGET ((R!SCR3))           .SAVE EXIT STACK ENTRY DISKLOAD.112
40    <macro>                   EINCR                                           DISKLOAD.113
46   041000  013020  104016     P = SAVE, A < D'16          .LOOP UNTIL ENTIRE STACK SAVED DISKLOAD.114
                                                                                DISKLOAD.115
                                *                                               R14673BC.3
                                * Clear channels 3 - 47                         R14673BC.4
                                *                                               R14673BC.5
                                                                                R14673BC.6
51   010003  054000             B = 3                       .FIRST CHANNEL NUMBER R14673BC.7
```

```
                                                                                R14673BC.8
 53    <macro>                        $UNTIL (B = 50)                          R14673BC.9
 56    160000                           IOB:0                  .CLEAR CHANNEL  R14673BC.10
 57    166000                           IOB:6                  .DISABLE CHANNEL INTERRUPTS  R14673BC.11
 60    056000                           B = B + 1                               R14673BC.12
 61    <macro>                        $ENDTIL                                  R14673BC.13
                                                                                R14673BC.14
                                      *                                         DISKLOAD.116
                                      * Save operand registers 700 - 777       DISKLOAD.117
                                      *                                         DISKLOAD.118
                                                                                DISKLOAD.119
 62    010700  054000                 B = 700                  .FIRST REGISTER TO SAVE  DISKLOAD.120
 64    075716 /000124                 P = OPSAVE                              R14673BC.15
                                                                                R14673BC.16
 25w                                  ORG       124/4          .SPACE FOR REGISTER DUMP  R14673BC.17
                                                                                R14673BC.18
124                          OPSAVE  *                                         R14673BC.19
                                                                                R14673BC.20
124    <macro>                        $UNTIL (B = 0)                           DISKLOAD.121
126    026721                           R!SCR3 = R!SCR3 + 1    .NEXT SAVE LOCATION  DISKLOAD.122
127    060000  034721                   (R!SCR3) = (B)         .SAVE NEXT OPERAND REGISTER  DISKLOAD.123
131    056000                           B = B + 1                               DISKLOAD.124
132    <macro>                        $ENDTIL                                  DISKLOAD.125
                                                                                DISKLOAD.126
```

```
                                                                                DISKLOAD.128
                                 ************************************************************DISKLOAD.129
                                 *                                                          *DISKLOAD.130
                                 *        All necessary resources have been saved in local memory.  *DISKLOAD.131
                                 *        The following sections of code may use other operand      *DISKLOAD.132
                                 *        registers besides BASE through SCR3.                      *DISKLOAD.133
                                 *                                                          *DISKLOAD.134
                                 ************************************************************DISKLOAD.135
                                          *                                                 DISKLOAD.136
                                          * Write first 2000 (octal) words of memory to buffer memory  DISKLOAD.137
                                          *                                                 DISKLOAD.138
                                                                                            DISKLOAD.139
                                                                                            DISKLOAD.140
133   010000  024722                      R!MOSL = REG$MOSL          .LOWER ADDRESS         DISKLOAD.141
135   010000  024723                      R!MOSU = REG$MOSU          .UPPER ADDRESS         DISKLOAD.142
137   014000 /002000  024724              R!WRDS = 0'2000            .TRANSFER SIZE         DISKLOAD.143
142   010000  024731                      R!BUF = 0                  .BUFFER ADDRESS        R14673BC.21
144   072167                              R = MOSIO                  .CALL MOS DRIVER       DISKLOAD.145
                                                                                            DISKLOAD.146
                                          *                                                 DISKLOAD.147
                                          * Read in all of LOAD to last 2400 (octal) parcels of memory  DISKLOAD.148
                                          *                                                 DISKLOAD.149
                                                                                            DISKLOAD.150
145   010000  024725                      R!CYL = 0                  .CYLINDER NUMBER 0     DISKLOAD.151
147   010000  024726                      R!HD  = 0                  .HEAD NUMBER 0         DISKLOAD.152
151   010000  024727                      R!SCT = 0                  .PHYSICAL SECTOR NUMBER 0  DISKLOAD.153
153   010005  024730                      R!CNT = LDSIZ              .FOR SIZE              DISKLOAD.154
155   010000  017000 /002400              R!BUF = 0 - XDK$SZ         .START OF CODE         R14673BC.22
      024731
161   072006                              R = XDISK                  .CALL DISK DRIVER      DISKLOAD.156
                                                                                            DISKLOAD.157
162   020731  016000 /000647              R!%TO = R!BUF + LOAD2      .SET ENTRY ADDRESS     DISKLOAD.158
      024771
166   074771                              P = R!%TO                  .ENTER AT LOAD2        DISKLOAD.159
                                                                                            DISKLOAD.160
                                 ************************************************************DISKLOAD.161
                                                                                            DISKLOAD.162
                                          *                                                 DISKLOAD.163
                                          * Mini device driver for Expander Disk            DISKLOAD.164
                                          *                                                 DISKLOAD.165
                                                                                            DISKLOAD.166
167                              XDISK    *                                                 DISKLOAD.167
```

```
                                                                                DISKLOAD.168
                                  * Initialize                                  DISKLOAD.169
                                                                                DISKLOAD.170
167     <macro>                   DEVICE I@XDK                .SET DEVICE CODE  DISKLOAD.171
                                                                                DISKLOAD.172
171     <macro>                   CLR                         .CLEAR DEVICE     DISKLOAD.173
173   072066                      R = CHWT                                      DISKLOAD.174
                                                                                DISKLOAD.175
                          AMPEX   IFE I@XDK,EQ,DX$AMPEX       .IF AMPEX DISK    R14673BC.23
        <skipped>                   DOA DU$NS                 .SET NO SEEK      DISKLOAD.177
        <skipped>                   R = CHWT                  .                 DISKLOAD.178
        <skipped>                                                               DISKLOAD.179
        <skipped>                   DOB R!CYL                 .SET CYLINDER     DISKLOAD.180
        <skipped>                   R = CHWT                  .                 DISKLOAD.181
        <skipped>                   DOC DU$CYL                .STEER            DISKLOAD.182
        <skipped>                   R = CHWT                  .                 DISKLOAD.183
        <skipped>                                                               DISKLOAD.184
        <skipped>                   DOB R!HD                  .SET HEAD         DISKLOAD.185
        <skipped>                   R = CHWT                  .                 DISKLOAD.186
        <skipped>                   DOC DU$HEAD               .STEER            DISKLOAD.187
        <skipped>                   R = CHWT                  .                 DISKLOAD.188
        <skipped>                                                               DISKLOAD.189
        <skipped>                   DOB R!SCT                 .SET SECTOR       DISKLOAD.190
        <skipped>                   R = CHWT                  .                 DISKLOAD.191
        <skipped>                   DOC DU$SECT               .STEER            DISKLOAD.192
        <skipped>                   R = CHWT                  .                 DISKLOAD.193
        <skipped>                                                               DISKLOAD.194
        <skipped>                   DOB R!CNT                 .SET COUNT        DISKLOAD.195
        <skipped>                   R = CHWT                  .                 DISKLOAD.196
        <skipped>                   DOC DU$COUNT              .STEER            DISKLOAD.197
        <skipped>                   R = CHWT                  .                 DISKLOAD.198
        <skipped>                                                               DISKLOAD.199
        <skipped>                   DOB R!BUF                 .SET BUFFER ADDR  DISKLOAD.200
        <skipped>                   R = CHWT                  .                 DISKLOAD.201
        <skipped>                                                               DISKLOAD.202
        <skipped>                   DOC DU$READ               .SET COMMAND      DISKLOAD.203
        <skipped>                   R = CHWT                  .                 DISKLOAD.204
                          AMPEX   ENDIF                                         R14673BC.24
                                                                                R14673BC.25
                          CDC     IFE I@XDK,EQ,DX$CDC         .IF CDC DISK      R14673BC.26
174     <macro>                     DOA DV$SEEK               .SEEK TO CYLINDER DISKLOAD.207
176   072063                        R = CHWT                  .                 DISKLOAD.208
```

```
                                                                                DISKLOAD.209
177    <macro>                  DOC R!CYL                  .SET CYLINDER       DISKLOAD.210
201    072060                   R = CHWT                   .                   DISKLOAD.211
                                                                                DISKLOAD.212
202    <macro>                  PLS                        .ISSUE PULSE        DISKLOAD.213
204    072055                   R = CHWT                                       DISKLOAD.214
205    072064                   R = CTLWT                  .WAIT FOR CONTROL FULL TO CLEAR  DISKLOAD.215
                                                                                DISKLOAD.216
206    <macro>                  DOA DV$READ                .SET COMMAND        DISKLOAD.217
210    072051                   R = CHWT                                       DISKLOAD.218
                                                                                DISKLOAD.219
211    <macro>                  DOC1 R!HD,R!SCT,R!CNT      .SET HEAD, SECT AND COUNT (MSB)  DISKLOAD.220
222    072037                   R = CHWT                   .                   DISKLOAD.221
                                                                                DISKLOAD.222
223    <macro>                  DOC2 R!HD,R!SCT,R!CNT      .SET HEAD, SECT AND COUNT  DISKLOAD.223
241    072020                   R = CHWT                   .                   DISKLOAD.224
                                                                                DISKLOAD.225
242    <macro>                  DOB R!BUF                  .SET BUFFER ADDRESS DISKLOAD.226
244    072015                   R = CHWT                   .                   DISKLOAD.227
                       CDC    ENDIF                                            R14673BC.27
                                                                                DISKLOAD.229
245    <macro>                STR                          .ISSUE START        DISKLOAD.230
247    072012                 R = CHWT                                         DISKLOAD.231
                                                                                DISKLOAD.232
                              * Wait for interrupt                              DISKLOAD.233
                                                                                DISKLOAD.234
250                    INTWT  *                                                DISKLOAD.235
                                                                                DISKLOAD.236
250    <macro>                STAT1                                            DISKLOAD.237
251    005001                 A = A < 1                                        DISKLOAD.238
252    104002                 P = INTWT, C = 0             .WAIT FOR FUNCTION COMPLETION  DISKLOAD.239
                                                                                DISKLOAD.240
                              * Get device status                               DISKLOAD.241
                                                                                DISKLOAD.242
253    <macro>                DIA                          .REQUEST DEVICE STATUS DISKLOAD.243
254    072005                 R = CHWT                                         DISKLOAD.244
255    <macro>                BUS                          .READ STATUS        DISKLOAD.245
                                                                                DISKLOAD.246
256    011001                 A = A & 1                                        DISKLOAD.247
257    103021                 P = ABORT, A # 0             .IF ERROR           DISKLOAD.248
260    001000                 EXIT                                             DISKLOAD.249
                                                                                DISKLOAD.250
```

```
                                        *                                                       DISKLOAD.251
                                        *                                                       DISKLOAD.252
                                        * Wait for channel to accept command                   DISKLOAD.253
                                        *                                                       DISKLOAD.254
                                                                                                DISKLOAD.255
261                             CHWT    *                                                       DISKLOAD.256
                                                                                                DISKLOAD.257
261     <macro>                         CHWT                            .WAIT FOR CHANNEL       DISKLOAD.258
267     102011                          P = ABORT, A = 0                .IF TIMEOUT             DISKLOAD.259
270     001000                          EXIT                                                    DISKLOAD.260
                                                                                                DISKLOAD.261
                                                                                                DISKLOAD.262
                                        *                                                       DISKLOAD.263
                                        * Wait for controller to accept command                DISKLOAD.264
                                        *                                                       DISKLOAD.265
                                                                                                DISKLOAD.266
271                             CTLWT   *                                                       DISKLOAD.267
                                                                                                DISKLOAD.268
271     <macro>                         DIA                             .READ STATUS            DISKLOAD.269
272     073011                          R = CHWT                        .                       DISKLOAD.270
273     <macro>                         BUS                                                     DISKLOAD.271
                                                                                                DISKLOAD.272
274     015000 /100000                  A = A & O'100000                .CONTROL CLEAR FLAG     DISKLOAD.273
276     107005                          P = CTLWT, A # 0                .AGAIN                  DISKLOAD.274
277     001000                          EXIT                                                    DISKLOAD.275
                                                                                                DISKLOAD.276
                                                                                                DISKLOAD.277
                                        *                                                       DISKLOAD.278
                                        * Abort deadstart                                       DISKLOAD.279
                                        *                                                       DISKLOAD.280
                                                                                                DISKLOAD.281
300                             ABORT   *                                                       DISKLOAD.282
                                                                                                DISKLOAD.283
300     020716  016000 /000355          R!MSG = R!BASE + MSGABRT        .ABORT MESSAGE          DISKLOAD.284
        024732
304     072002                          R = SEND                        .SEND                   DISKLOAD.285
305     070000                          WAIT                            .HANG                   DISKLOAD.286
                                                                                                DISKLOAD.287
                                                                                                DISKLOAD.288
                                        *                                                       DISKLOAD.289
                                        * Send message to console                               DISKLOAD.290
                                        *                                                       DISKLOAD.291
```

```
                                                                                DISKLOAD.292
306                         SEND    *                                           DISKLOAD.293
                                                                                DISKLOAD.294
306   030732  024734                R!CHARS = (R!MSG)           .GET CHARACTERS DISKLOAD.295
310   020734  004010  024733        R!CHAR = R!CHARS > D'8      .CHECK HIGH BYTE DISKLOAD.296
313   103002  001000                EXIT, A = 0                 .IF END OF MESSAGE DISKLOAD.297
315   072011                        R = ECHO                                    DISKLOAD.298
316   020734  011377  024733        R!CHAR = R!CHARS & O'377    .CHECK LOW BYTE DISKLOAD.299
321   103002  001000                EXIT, A = 0                 .IF END OF MESSAGE DISKLOAD.300
323   072003                        R = ECHO                                    DISKLOAD.301
324   026732                        R!MSG = R!MSG + 1           .NEXT PARCEL ADDRESS DISKLOAD.302
325   071017                        P = SEND                                    DISKLOAD.303
                                                                                DISKLOAD.304
                                                                                DISKLOAD.305
                                    *                                           DISKLOAD.306
                                    * Send character to console                DISKLOAD.307
                                    *                                           DISKLOAD.308
                                                                                DISKLOAD.309
326                         ECHO    *                                           DISKLOAD.310
                                                                                DISKLOAD.311
326   041047  105001                WAIT, TOD = BZ              .WAIT UNTIL CHANNEL NOT BUSY DISKLOAD.312
330   020733                        A = R!CHAR                                  DISKLOAD.313
331   154047                        TOD:14                      .WRITE OUT CHARACTER DISKLOAD.314
332   001000                        EXIT                                        DISKLOAD.315
                                                                                DISKLOAD.316
                                                                                DISKLOAD.317
                                    *                                           DISKLOAD.318
                                    * Mini device driver for MOS IO             DISKLOAD.319
                                    *                                           DISKLOAD.320
                                                                                DISKLOAD.321
333                         MOSIO   *                                           DISKLOAD.322
                                                                                DISKLOAD.323
333   140005                        MOS:0                       .CLEAR          DISKLOAD.324
334   020722  011777                A = R!MOSI & O'777          .LOWER ADDRESS DISKLOAD.325
336   143005                        MOS:3                       .ENTER          DISKLOAD.326
337   020722  004011  024766        R!%W1 = R!MOSL > D'9        .UNPACK ADDRESS DISKLOAD.327
342   020723  005007  022766        A = R!MOSU < D'7 + R!%W1    .UPPER ADDRESS DISKLOAD.328
345   142005                        MOS:2                       .ENTER          DISKLOAD.329
346   020731                        A = R!BUF                   .BUFFER ADDRESS DISKLOAD.330
347   141005                        MOS:1                                       DISKLOAD.331
350   020724                        A = R!WRDS                                  DISKLOAD.332
351   145005                        MOS:5                       .ENTER TRANSFER WORD COUNT DISKLOAD.333
```

```
352   041005  105001          WAIT, MOS = BZ              .WAIT UNTIL DONE          DISKLOAD.334
354   001000                  EXIT                                                  DISKLOAD.335
                                                                                    DISKLOAD.336
                                                                                    DISKLOAD.337
                        ***********************************************************DISKLOAD.338
                                                                                    DISKLOAD.339
                                *                                                   DISKLOAD.340
                                * Error messages                                    DISKLOAD.341
                                *                                                   DISKLOAD.342
                                                                                    DISKLOAD.343
355                     MSGABRT *                                                   DISKLOAD.344
 73w        024064              VWD   D'8/CC$LF,D'8/CC$CR                           DISKLOAD.345
356                             'DISK ERROR: DEADSTART ABORTED'                     DISKLOAD.346
375   000000                    0                                                   DISKLOAD.347
                                                                                    DISKLOAD.348
376                     BOOTEND *                                                   DISKLOAD.349
                                ERRIF     BOOTEND,GT,BTSIZ    .BASIC BOOT MUST FIT IN DISKLOAD.350
                                                              .ONE PHYSICAL DISK SECTOR DISKLOAD.351
                                                                                    DISKLOAD.352
376                     MSGNOFL *                                                   DISKLOAD.353
 77w            012032          VWD   D'8/CC$LF,D'8/CC$CR                           DISKLOAD.354
377                             'FILE NOT FOUND'                                    DISKLOAD.355
407   000000                    0                                                   DISKLOAD.356
                                                                                    DISKLOAD.357
410                     MSGFLW  *                                                   DISKLOAD.358
102w  005015                    VWD   D'8/CC$LF,D'8/CC$CR                           DISKLOAD.359
411                             'FILE FLAWED'                                       DISKLOAD.360
417   000000                    0                                                   DISKLOAD.361
                                                                                    DISKLOAD.362
420                     MSGNODR *                                                   DISKLOAD.363
104w  005015                    VWD   D'8/CC$LF,D'8/CC$CR                           DISKLOAD.364
421                             'DIRECTORY NOT FOUND'                               DISKLOAD.365
433   000000                    0                                                   DISKLOAD.366
                                                                                    DISKLOAD.367
                                *                                                   DISKLOAD.368
                                * Prompt messages                                   DISKLOAD.369
                                *                                                   DISKLOAD.370
                                                                                    DISKLOAD.371
434                     MSGFILE *                                                   DISKLOAD.372
107w  005015                    VWD   D'8/CC$LF,D'8/CC$CR                           DISKLOAD.373
435                             'FILE @DK0:'                                        DISKLOAD.374
443   000000                    0                                                   DISKLOAD.375
```

```
                                                                                                DISKLOAD.376
444                         MSGDBUG  *                                                          DISKLOAD.377
111w   005015                        VWD   D'8/CC$LF,D'8/CC$CR                                  DISKLOAD.378
445                                  'DEBUG'                                                    DISKLOAD.379
450   000000                         0                                                          DISKLOAD.380
                                                                                                DISKLOAD.381
                                     *                                                          DISKLOAD.382
                                     * Menu display messages                                    DISKLOAD.383
                                     *                                                          DISKLOAD.384
                                                                                                DISKLOAD.385
451                         MENUHDR  *                                                          DISKLOAD.386
112w          024064                 VWD   D'8/CC$LF,D'8/CC$CR                                  DISKLOAD.387
452                                  '          FILE                    CREATED            WORDS ' DISKLOAD.388
511   000000                         0                                                          DISKLOAD.389
                                                                                                DISKLOAD.390
512                         MENULIN  *                                                          DISKLOAD.391
122w                012032           VWD   D'8/CC$LF,D'8/CC$CR                                  DISKLOAD.392
513                                  '                                                        ' DISKLOAD.393
552   000000                         0                                                          DISKLOAD.394
                                                                                                DISKLOAD.395
                              6      MENUFIL  EQUALS  D'6                  .FILE NAME OFFSET IN MESSAGE     DISKLOAD.396
                             17      MENUCRD  EQUALS  D'15                 .CREATION DATE OFFSET IN MESSAGE DISKLOAD.397
                             24      MENUCRT  EQUALS  D'20                 .CREATION TIME OFFSET IN MESSAGE DISKLOAD.398
                             75      MENUWRD  EQUALS  D'61                 .WORDS OFFSET IN MESSAGE         DISKLOAD.399
                             36      MENUSIZ  EQUALS  D'30                 .MENU LINE SIZE                  DISKLOAD.400
                                                                                                DISKLOAD.401
                                     *                                                          DISKLOAD.402
                                     * Directory entry buffer                                  DISKLOAD.403
                                     *                                                          DISKLOAD.404
                                                                                                DISKLOAD.405
553                         DIRBUF   *                                                          DISKLOAD.406
553                                  <<FD@END@P>>                  .FILE DESCRIPTION ENTRY     DISKLOAD.407
                                                                                                DISKLOAD.408
```

```
                                                                                                  DISKLOAD.410
                                 ***********************************************************************DISKLOAD.411
                                 *                                                                    *DISKLOAD.412
                                 *        Prompt for and input the disk file name.                    *DISKLOAD.413
                                 *                                                                    *DISKLOAD.414
                                 ***********************************************************************DISKLOAD.415
                                                                                                  DISKLOAD.416
647                              LOAD2    *                                                          DISKLOAD.417
                                                                                                  DISKLOAD.418
647   010000  017000 /002400             R!BASE = 0 - XDK$SZ          .SET BASE REGISTER TO HIGH MEMORYR14673BC.28
      024716
                                         BASEREG BASE                                             DISKLOAD.420
                                                                                                  DISKLOAD.421
653   020716  016000 /000553             R!DIR = R!BASE + DIRBUF      .DIRECTORY ENTRY BUFFER ADDRESS  DISKLOAD.422
      024736
657   020736  012001  024735             R!FID = R!DIR + FD@FID@P     .FILE ID BUFFER ADDRESS          DISKLOAD.423
                                                                                                  DISKLOAD.424
662   010000  024737                     R!DBG = OFF$DBUG             .DEFAULT DEBUG OPTION TO OFF     DISKLOAD.425
                                                                                                  DISKLOAD.426
664                              GETFID   *                                                          DISKLOAD.427
                                                                                                  DISKLOAD.428
664   020716  016000 /000434             R!MSG = R!BASE + MSGFILE                                 DISKLOAD.429
      024732
670   073362                             R = SEND                     .OUTPUT PROMPT MESSAGE           DISKLOAD.430
                                                                                                  DISKLOAD.431
671   020735  024763                     R!DST = R!FID                .CLEAR FILE ID BUFFER            DISKLOAD.432
673   010020  024764                     R!SIZ = FD@FID@Z             .                                DISKLOAD.433
675   010000  024765                     R!PAT = 0                    .CLEAR TO ZEROS                  DISKLOAD.434
677   077716 /002103                     R = CLEAR                    .                                DISKLOAD.435
                                                                                                  DISKLOAD.436
701   010000  024740                     R!PNT = 0                    .FILE ID CHAR POINTER            DISKLOAD.437
                                                                                                  DISKLOAD.438
703    <macro>                           $UNTIL (R!PNT = FD@FID@C)    .MAX LENGTH EXCEEDED             DISKLOAD.439
706   072063                               R = RECV                   .INPUT NEXT CHARACTER            DISKLOAD.440
707    <macro>                             $IF (R!CHAR = CC$CTD)      .IF CONTROL-D ENTERED            DISKLOAD.441
                                                                                        .>>>>> 0          $IF      1
713   020716  016000 /000444               R!MSG = R!BASE + MSGDBUG   .SEND DEBUG MESSAGE              DISKLOAD.442
      024732
717   073411                               R = SEND                   .TO CONSOLE                      DISKLOAD.443
720   010001  024737                       R!DBG = ON$DBUG            .SET DEBUG REQUEST FLAG          DISKLOAD.444
722   071036                               P = GETFID                 .TRY AGAIN                       DISKLOAD.445
723    <macro>                             $ELSEIF (R!CHAR = '/'R)    .IF DIRECTORY DELIMITER          DISKLOAD.446
```

```
 730    <macro>                          $IF (R!PNT = 0) OR (R!PNT >= FD@DIR@C) .IF FIRST OR > MAX  DISKLOAD.447
                                                                          .>>>>> 1          $IF      1
 736   071052                              P = GETFID                .TRY AGAIN                    DISKLOAD.448
 737    <macro>                          $ENDIF                                                    DISKLOAD.449
                                                                          .<<<<< 1          $ENDIF   1
 737   010020  024740                      R!PNT = FD@DIR@C          .SET POINTER TO FILE NAME OFFSET DISKLOAD.450
 741    <macro>                        $ELSEIF (R!CHAR = CC$CR) OR (R!CHAR = CC$LF) .IF TERMINATOR  DISKLOAD.451
 751    <macro>                          $IF (R!PNT > FD@DIR@C)    .IF FULL FILE ID SPECIFIED       DISKLOAD.452
                                                                          .>>>>> 1          $IF      1
 757   075716 /001125                      P = LOAD                  .LOAD FILE AND EXECUTE          DISKLOAD.453
 761    <macro>                          $ELSE                                                     DISKLOAD.454
 762   075716 /001345                      P = FSTAT                 .MENU DISPLAY OR DUMP REQUESTED DISKLOAD.455
 764    <macro>                          $ENDIF                                                    DISKLOAD.456
                                                                          .<<<<< 1          $ENDIF   1
 764    <macro>                        $ELSE                                                       DISKLOAD.457
 765   077716 /001073                      R = PUTCHAR               .ENTER                          DISKLOAD.458
 767    <macro>                        $ENDIF                                                      DISKLOAD.459
                                                                          .<<<<< 0          $ENDIF   1
 767    <macro>                      $ENDTIL                                                       DISKLOAD.460
 770   071104                        P = GETFID                      .FILE ID TOO LONG, TRY AGAIN    DISKLOAD.461
                                                                                                    DISKLOAD.462
                                                                                                    DISKLOAD.463
                              ************************************************************************DISKLOAD.464
                                                                                                    DISKLOAD.465
                                    *                                                               DISKLOAD.466
                                    * Input a character and echo it on the screen. Handle BackspaceDISKLOAD.467
                                    * and Delete as special cases.  On exit, the character is in   DISKLOAD.468
                                    * register CHAR.                                                DISKLOAD.469
                                    *                                                               DISKLOAD.470
                                                                                                    DISKLOAD.471
 771                          RECV   *                                                               DISKLOAD.472
                                                                                                    DISKLOAD.473
 771   040046  104001                WAIT, TID # DN                  .WAIT UNTIL DONE                DISKLOAD.474
 773   150046                        TID:10                          .READ INPUT                     DISKLOAD.475
 774   024733                        R!CHAR = A                      .SAVE                           DISKLOAD.476
 775    <macro>                      $IF (R!CHAR = CC$BS) OR (R!CHAR = CC$DEL) .IF BS OR DEL         DISKLOAD.477
                                                                          .>>>>> 0          $IF      1
1005    <macro>                        $IF (R!PNT # 0)               .IF NOT FIRST CHAR              DISKLOAD.478
                                                                          .>>>>> 1          $IF      1
1011   010010  024733                      R!CHAR = CC$BS            .SEND BACKSPACE                 DISKLOAD.479
1013   073465                              R = ECHO                                                  DISKLOAD.480
1014   010040  024733                      R!CHAR = ' 'R             .SEND SPACE                     DISKLOAD.481
```

```
1017   073470                          R = ECHO                                                  DISKLOAD.482
1021   010010  024733                  R!CHAR = CC$BS              .SEND BACKSPACE               DISKLOAD.483
1022   073473                          R = ECHO                                                  DISKLOAD.484
       <macro>                         $IF (R!PNT = FD@DIR@C)      .IF SLASH POSITION            DISKLOAD.485
                                                                                   .>>>>> 2          $IF      1
1026   010000  024740                    R!PNT = 0                 .SCAN DIRECTORY NAME FOR END  DISKLOAD.486
1030   010001  024733                    R!CHAR = 1                .PRESET                       DISKLOAD.487
1032   <macro>                           $UNTIL (R!CHAR = 0)       .END OF NAME                  DISKLOAD.488
1034   <macro>                             GETBYTE R!CHAR,R!PNT,R!FID,INC=1 .GET CHAR            DISKLOAD.489
1057   <macro>                           $ENDTIL                                                 DISKLOAD.490
1060   <macro>                         $ELSE                                                     DISKLOAD.491
1062   027740                            R!PNT = R!PNT - 1         .BACKSPACE                    DISKLOAD.492
1063   010000  024733                    R!CHAR = 0                .CLEAR CHAR TO ZERO           DISKLOAD.493
1065   077716 /001073                    R = PUTCHAR                                             DISKLOAD.494
1067   <macro>                         $ENDIF                                                    DISKLOAD.495
                                                                                   .<<<<< 2          $ENDIF   1
1067   027740                            R!PNT = R!PNT - 1         .BACKSPACE                    DISKLOAD.496
1070   <macro>                         $ENDIF                                                    DISKLOAD.497
                                                                                   .<<<<< 1          $ENDIF   1
1070   071077                          P = RECV                    .TRY AGAIN                    DISKLOAD.498
1071   <macro>                       $ENDIF                                                      DISKLOAD.499
                                                                                   .<<<<< 0          $ENDIF   1
1071   073543                        R = ECHO                      .ECHO THE CHARACTER           DISKLOAD.500
1072   001000                        EXIT                                                        DISKLOAD.501
                                                                                                   DISKLOAD.502
                                                                                                   DISKLOAD.503
                                     *                                                             DISKLOAD.504
                                     * Add char to FILE ID in next position                       DISKLOAD.505
                                     *                                                             DISKLOAD.506
                                                                                                   DISKLOAD.507
1073                        PUTCHAR  *                                                             DISKLOAD.508
                                                                                                   DISKLOAD.509
1073   <macro>                       PUTBYTE R!CHAR,R!PNT,R!FID,INC=1                            DISKLOAD.510
1124   001000                        EXIT                                                        DISKLOAD.511
                                                                                                   DISKLOAD.512
                                                                                                   DISKLOAD.513
                                     *                                                             DISKLOAD.514
                                     * Load requested file and execute it                         DISKLOAD.515
                                     *                                                             DISKLOAD.516
                                                                                                   DISKLOAD.517
1125                        LOAD     *                                                             DISKLOAD.518
                                                                                                   DISKLOAD.519
```

```
1125   010000  024741                R!FLG = 0                      .ASSUME FILE IS NOT KERNEL       DISKLOAD.520
                                                                                                    DISKLOAD.521
                                      * Check for possible KERNEL overlay file (fileid.OV)          DISKLOAD.522
                                                                                                    DISKLOAD.523
1127   010056  024733                R!CHAR = '.'R                  .APPEND .OV TO FILE ID           DISKLOAD.524
1131   073036                        R = PUTCHAR                                                     DISKLOAD.525
1132   010117  024733                R!CHAR = 'O'R                                                   DISKLOAD.526
1134   073041                        R = PUTCHAR                                                     DISKLOAD.527
1135   010126  024733                R!CHAR = 'V'R                                                   DISKLOAD.528
1137   073044                        R = PUTCHAR                                                     DISKLOAD.529
                                                                                                    DISKLOAD.530
1140   010020  024747                R!LIM = FD@FID@Z               .FILE ID SIZE                    DISKLOAD.531
1142   072275                        R = DIRSRC                     .SEARCH DISK FILE DIRECTORY      DISKLOAD.532
1143    <macro>                      $IF (R!DRE # 0)                .IF OVLY FILE FOUND              DISKLOAD.533
                                                                                        .>>>>> 0          $IF      1
                                                                                                    DISKLOAD.534
1146    <macro>                        GET R!%W1,FD@FLW,R!DIR       .CHECK FOR FLAWS                 DISKLOAD.535
1152    <macro>                        $IF (R!%W1 # 0)              .IF FLAWED                       DISKLOAD.536
                                                                                        .>>>>> 1          $IF      1
1155   020716  012410  024732          R!MSG = R!BASE + MSGFLW      .SEND FILE FLAWED MESSAGE        DISKLOAD.537
1160   073652                          R = SEND                     .TO CONSOLE                      DISKLOAD.538
1161   071275                          P = GETFID                   .TRY AGAIN                       DISKLOAD.539
1162    <macro>                        $ENDIF                                                        DISKLOAD.540
                                                                                        .<<<<< 1          $ENDIF   1
                                                                                                    DISKLOAD.541
1162   010001  024741                  R!FLG = 1                    .KERNEL TO BE LOADED FLAG        DISKLOAD.542
                                                                                                    DISKLOAD.543
                                       * Copy overlay file to MOS                                   DISKLOAD.544
                                                                                                    DISKLOAD.545
1164   014000 /100000  024722          R!MOSL = OVL$MOSL            .MOS OVLY ADDR                   DISKLOAD.546
1167   010000  024723                  R!MOSU = OVL$MOSU            .                                DISKLOAD.547
1171   014000 /001000  024724          R!WRDS = O'1000              .TRANSFER SIZE                   DISKLOAD.548
                                                                                                    DISKLOAD.549
1174   010001  024745                  R!NUM = 1                    .BLOCKS PER I/O                  DISKLOAD.550
1176   010000  024731                  R!BUF = 0                    .READ OVLY BLOCK TO ADDR 0       DISKLOAD.551
                                                                                                    DISKLOAD.552
1200    <macro>                        GET R!EOF,FD@EOF,R!DIR       .LOGICAL END OF FILE             DISKLOAD.553
1205   010001  024743                  R!LBN = 1                    .LOGICAL BLOCK NUMBER            DISKLOAD.554
1207    <macro>                        $UNTIL (R!LBN > R!EOF)       .ALL COPIED                      DISKLOAD.555
1214   072351                          R = GETBLK                   .GET PHYSICAL BLOCK NUMBER       DISKLOAD.556
1215   072424                          R = CONVERT                  .CONVERT TO PHYSICAL PARAMS      DISKLOAD.557
1216   077716 /000167                  R = XDISK                    .CALL DISK DRIVER                DISKLOAD.558
```

```
1225   073665                          R = MOSIO                    .CALL MOS DRIVER                DISKLOAD.559
                                                                                                     DISKLOAD.560
1221    <macro>                        DADD R!MOSU,R!MOSL,R!WRDS    .INC MOS ADDR                   DISKLOAD.561
1226   026743                          R!LBN = R!LBN + 1            .INC LOGICAL BLOCK NUMBER       DISKLOAD.562
1227    <macro>                      $ENDTIL                                                        DISKLOAD.563
                                                                                                     DISKLOAD.564
1230    <macro>                    $ENDIF                                                           DISKLOAD.565
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                     DISKLOAD.566
                                   *                                                                 DISKLOAD.567
                                   * Clear .OV from FILE ID                                          DISKLOAD.568
                                   *                                                                 DISKLOAD.569
                                                                                                     DISKLOAD.570
1230   020740  013003  024740      R!PNT = R!PNT - 3            .BACK UP CHAR POINTER           DISKLOAD.571
1233   010000  024733              R!CHAR = 0                   .SET TO TERMINATOR              DISKLOAD.572
1235   073142                      R = PUTCHAR                                                  DISKLOAD.573
1236   073143                      R = PUTCHAR                                                  DISKLOAD.574
1237   073144                      R = PUTCHAR                                                  DISKLOAD.575
                                                                                                     DISKLOAD.576
                                   *                                                                 DISKLOAD.577
                                   * Find requested file, load and execute it                       DISKLOAD.578
                                   *                                                                 DISKLOAD.579
                                                                                                     DISKLOAD.580
1240   072177                      R = DIRSRC                   .SEARCH DISK FILE DIRECTORY     DISKLOAD.581
1241    <macro>                    $IF (R!DRE = 0)              .IF REQUESTED FILE NOT FOUND    DISKLOAD.582
                                                                                .>>>>> 0                $IF       1
1244   020716  012376  024732        R!MSG = R!BASE + MSGNOFL     .FILE NOT FOUND MESSAGE         DISKLOAD.583
1247   073741                        R = SEND                     .SEND TO CONSOLE                DISKLOAD.584
1250   071364                        P = GETFID                   .GET NEW FILENAME               DISKLOAD.585
1251    <macro>                    $ENDIF                                                           DISKLOAD.586
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                     DISKLOAD.587
1251    <macro>                    GET R!%W1,FD@FLW,R!DIR       .CHECK FOR FLAWS                DISKLOAD.588
1255    <macro>                    $IF (R!%W1 # 0)              .IF FLAWED                      DISKLOAD.589
                                                                                .>>>>> 0                $IF       1
1260   020716  012410  024732        R!MSG = R!BASE + MSGFLW      .SEND FILE FLAWED MESSAGE       DISKLOAD.590
1263   073755                        R = SEND                     .TO CONSOLE                     DISKLOAD.591
1264   071400                        P = GETFID                   .TRY AGAIN                      DISKLOAD.592
1265    <macro>                    $ENDIF                                                           DISKLOAD.593
                                                                                .<<<<< 0                $ENDIF    1
                                                                                                     DISKLOAD.594
                                   *                                                                 DISKLOAD.595
```

```
                                 * Load and execute requested file                          DISKLOAD.596
                                 *                                                          DISKLOAD.597
                                                                                            DISKLOAD.598
1265   010001  024745            R!NUM = 1                    .BLOCKS PER I/O               DISKLOAD.599
1267   010000  024731            R!BUF = 0                    .READ BLOCK TO ADDR 0         DISKLOAD.600
                                                                                            DISKLOAD.601
1271    <macro>                  GET R!EOF,FD@EOF,R!DIR       .LOGICAL END OF FILE          DISKLOAD.602
1276   010001  024743            R!LBN = 1                    .LOGICAL BLOCK NUMBER         DISKLOAD.603
1300    <macro>                  $UNTIL (R!LBN > R!EOF)       .ALL COPIED                   DISKLOAD.604
1305   072260                      R = GETBLK                .GET PHYSICAL BLOCK NUMBER    DISKLOAD.605
1306   072333                      R = CONVERT               .CONVERT TO PHYSICAL PARAMS   DISKLOAD.606
1307   077716 /000167              R = XDISK                 .CALL DISK DRIVER             DISKLOAD.607
                                                                                            DISKLOAD.608
1311   014000 /004000  025731      R!BUF = R!BUF + FM$BFZP   .INC LOCAL BUFFER ADDRESS     DISKLOAD.609
1314   026743                      R!LBN = R!LBN + 1         .INC LOCAL BLOCK NUMBER       DISKLOAD.610
1315    <macro>                  $ENDTIL                                                   DISKLOAD.611
                                                                                            DISKLOAD.612
                                 *                                                          DISKLOAD.613
                                 * Set up registers for loaded program                      DISKLOAD.614
                                 *                                                          DISKLOAD.615
                                                                                            DISKLOAD.616
1316    <macro>                  $IF (R!FLG # 0)              .IF KERNEL                    DISKLOAD.617
                                                                                .>>>>> 0            $IF      1
1321   014000 /000000  024771      (STRT) = XDK$STRT         .CALLER IS DISKLOAD FLAG      DISKLOAD.618
       010001  034771
1326   014000 /000000  024771      (DEBUG) = R!DBG           .DEBUG OPTION                 DISKLOAD.619
       020737  034771
1333    <macro>                  $ELSE                        .OTHER BINARY                 DISKLOAD.620
1334   010001  024717              R!%STRT = XDK$STRT        .DISKLOAD IS CALLER           DISKLOAD.621
1336   010000  024720              R!%MOSL = REG$MOSL        .REG SAVE ADDR IN MOS FOR DMP DISKLOAD.622
1340   010000  024721              R!%MOSU = REG$MOSU        .                             DISKLOAD.623
1342    <macro>                  $ENDIF                                                    DISKLOAD.624
                                                                                .<<<<< 0            $ENDIF   1
                                                                                            DISKLOAD.625
1342   010000  024771            R!%TO = 0                    .SET ENTRY ADDRESS            DISKLOAD.626
1344   074771                    P = R!%TO                    .ENTER ROUTINE                DISKLOAD.627
                                                                                            DISKLOAD.628
                                 *                                                          DISKLOAD.629
                                 * Check file name for DUMP or menu display                 DISKLOAD.630
                                 *                                                          DISKLOAD.631
                                                                                            DISKLOAD.632
1345                     FSTAT   *                                                          DISKLOAD.633
```

```
                                                                                                  DISKLOAD.634
1345    <macro>                         $IF ((R!FID) = 'DU') AND ((R!FID + 1) = 'MP') .IF DUMP    DISKLOAD.635
                                                                                .>>>>> 0          $IF      1
                                                                                                  DISKLOAD.636
                                          * Load DMP program from block 1 on disk and execute it  DISKLOAD.637
                                                                                                  DISKLOAD.638
1362   010001  024744                     R!BLK = DX$DPBLK            .PRESET LOC OF DMP (BLOCK 1)  DISKLOAD.639
1364   010002  024745                     R!NUM = 2                   .SIZE (2 BLOCKS)              DISKLOAD.640
1366   010000  024731                     R!BUF = 0                   .READ TO ADDR 0               DISKLOAD.641
                                                                                                  DISKLOAD.642
1370   072251                             R = CONVERT                 .CONVERT TO PHYSICAL PARAMS   DISKLOAD.643
1371   077716 /000167                     R = XDISK                   .CALL DISK DRIVER             DISKLOAD.644
                                                                                                  DISKLOAD.645
                                          *                                                       DISKLOAD.646
                                          * Set up registers for loaded program                   DISKLOAD.647
                                          *                                                       DISKLOAD.648
                                                                                                  DISKLOAD.649
1373   010001  024717                     R!%STRT = XDK$STRT          .DISKLOAD IS CALLER           DISKLOAD.650
1375   010000  024720                     R!%MOSL = REG$MOSL          .REG SAVE ADDR FOR DMP        DISKLOAD.651
1377   010000  024721                     R!%MOSU = REG$MOSU                                        DISKLOAD.652
                                                                                                  DISKLOAD.653
1401   010000  024771                     R!%TO = 0                   .SET ENTRY ADDRESS            DISKLOAD.654
1403   074771                             P = R!%TO                   .ENTER ROUTINE                DISKLOAD.655
                                                                                                  DISKLOAD.656
1404    <macro>                         $ENDIF                                                    DISKLOAD.657
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                  DISKLOAD.658
                                        *                                                         DISKLOAD.659
                                        * Display menu of all files in requested directory        DISKLOAD.660
                                        *                                                         DISKLOAD.661
                                                                                                  DISKLOAD.662
1404   010010  024747                   R!LIM = FD@DIR@Z            .DIRECTORY NAME SIZE          DISKLOAD.663
1406   072031                           R = DIRSRC                  .SEARCH DISK FILE DIRECTORY   DISKLOAD.664
1407    <macro>                         $IF (R!DRE = 0)             .IF NO FILES FOUND            DISKLOAD.665
                                                                                .>>>>> 0          $IF      1
1412   020716  012420  024732            R!MSG = R!BASE + MSGNODR    .DIRECTORY NOT FOUND MESSAGE  DISKLOAD.666
1415   077716 /000306                    R = SEND                    .SEND TO CONSOLE              DISKLOAD.667
1417   071533                            P = GETFID                  .GET NEW FILE ID              DISKLOAD.668
1420    <macro>                         $ENDIF                                                    DISKLOAD.669
                                                                                .<<<<< 0          $ENDIF   1
                                                                                                  DISKLOAD.670
1420   020716  012451  024732           R!MSG = R!BASE + MENUHDR    .DISPLAY MENU HEADER          DISKLOAD.671
```

```
1423   077716 /000306                R = SEND                        .ON CONSOLE                      DISKLOAD.672
                                                                                                       DISKLOAD.673
1425    <macro>                      $UNTIL (R!DRE = 0)              .ALL FOUND                       DISKLOAD.674
1427   072263                          R = FORMAT                    .FORMAT FILE STATISTICS          DISKLOAD.675
1430   077716 /000306                  R = SEND                      .SEND TO CONSOLE                 DISKLOAD.676
                                                                                                       DISKLOAD.677
1432   010074  025746                  R!DRE = R!DRE + DR@DE@Z       .NEXT DIRECTORY ENTRY ADDR       DISKLOAD.678
1434   072017                          R = DIRFND                    .FIND NEXT FILE                  DISKLOAD.679
1435    <macro>                      $ENDTIL                                                         DISKLOAD.680
1436   071552                        P = GETFID                      .GET NEW FILE ID                 DISKLOAD.681
                                                                                                       DISKLOAD.682
                                                                                                       DISKLOAD.683
                              ***********************************************************************DISKLOAD.684
                                                                                                       DISKLOAD.685
                                     *                                                                 DISKLOAD.686
                                     * Search disk directory for FILE ID                              DISKLOAD.687
                                     *                                                                 DISKLOAD.688
                                                                                                       DISKLOAD.689
1437                          DIRSRC *                                                                DISKLOAD.690
                                                                                                       DISKLOAD.691
1437   010005  024744                R!BLK = DX$DRBLK                .FIRST DIRECTORY BLOCK ON DISK   DISKLOAD.692
1441   010001  024745                R!NUM = 1                       .                                DISKLOAD.693
1443   010000  024731                R!BUF = 0                       .READ DIRECTORY BLOCK TO ADDR 0  DISKLOAD.694
                                                                                                       DISKLOAD.695
1445                          DIRNXT *                                                                DISKLOAD.696
                                                                                                       DISKLOAD.697
1445   072174                        R = CONVERT                     .CONVERT TO PHYSICAL PARAMS      DISKLOAD.698
1446   077716 /000167                R = XDISK                       .READ                            DISKLOAD.699
                                                                                                       DISKLOAD.700
1450    <macro>                      ADDRESS R!DRE DR@DE,R!BUF       .FIRST DIRECTORY ENTRY ADDR      DISKLOAD.701
                                                                                                       DISKLOAD.702
1453                          DIRFND *                                                                DISKLOAD.703
                                                                                                       DISKLOAD.704
1453    <macro>                      $UNTIL (R!DRE >= R!BUF + FM$BFZP) .ALL SCANNED                  DISKLOAD.705
1462    <macro>                        GET R!%W1,FD@STT,R!DRE        .ENTRY STATUS                    DISKLOAD.706
1465    <macro>                        $IF (R!%W1 # 0)               .IF ENTRY CONTAINS AN FDE        DISKLOAD.707
                                                                                        .>>>>> 0            $IF      1
1470   072037                            R = MATCH                   .COMPARE DIR ENTRY ID TO FILE ID DISKLOAD.708
1471    <macro>                          $IF (R!MCH # 0)             .IF MATCH                        DISKLOAD.709
                                                                                        .>>>>> 1            $IF      1
1474    <macro>                            COPY R!DRE,R!DIR,FD@END@P .COPY DIRECTORY ENTRY TO BUFFERDISKLOAD.710
1512   001000                              EXIT                      .RETURN DIRECTORY ENTRY PTR      DISKLOAD.711
```

```
1513    <macro>                          $ENDIF                                                    DISKLOAD.712
                                                                                  .<<<<< 1                $ENDIF   1
1513    <macro>                        $ENDIF                                                      DISKLOAD.713
                                                                                  .<<<<< 0                $ENDIF   1
1513   010074  025746                    R!DRE = R!DRE + DR@DE@Z      .NEXT DIRECTORY ENTRY ADDR     DISKLOAD.714
1515    <macro>                        $ENDTIL                                                     DISKLOAD.715
                                                                                                     DISKLOAD.716
                                       * GET NEXT DIRECTORY BLOCK                                    DISKLOAD.717
                                                                                                     DISKLOAD.718
1516    <macro>                        GET R!BLK,DR@NXT,R!BUF       .NEXT LINK                     DISKLOAD.719
1520    <macro>                        $IF (R!BLK # 0)              .IF NOT NIL                    DISKLOAD.720
                                                                                  .>>>>> 0                $IF      1
1523   071056                            P = DIRNXT                   .SEARCH NEXT BLOCK             DISKLOAD.721
1524    <macro>                        $ENDIF                                                      DISKLOAD.722
                                                                                  .<<<<< 0                $ENDIF   1
                                                                                                     DISKLOAD.723
                                       * FILE NOT FOUND                                              DISKLOAD.724
                                                                                                     DISKLOAD.725
1524   010000  024746                  R!DRE = 0                    .FILE NOT FOUND                DISKLOAD.726
1526   001000                          EXIT                                                          DISKLOAD.727
                                                                                                     DISKLOAD.728
                                                                                                     DISKLOAD.729
                                       *                                                             DISKLOAD.730
                                       * Compare FILE ID to directory entry file id                  DISKLOAD.731
                                       *                                                             DISKLOAD.732
                                                                                                     DISKLOAD.733
1527                           MATCH   *                                                             DISKLOAD.734
                                                                                                     DISKLOAD.735
1527    <macro>                        ADDRESS R!DID,FD@FID,R!DRE   .DIRECTORY ENTRY FILE ID ADDR  DISKLOAD.736
1532   010000  024766                  R!%W1 = 0                    .PARCEL POINTER                DISKLOAD.737
1534    <macro>                        $UNTIL (R!%W1 = R!LIM)       .ALL COMPARED                  DISKLOAD.738
1537   020750  022766  024767            R!%W2 = R!DID + R!%W1        .                              DISKLOAD.739
1542   030767  024767                    R!%W2 = (R!%W2)              .NEXT TWO DIR ENTRY FILE ID CHARSDISKLOAD.740
1544   020735  022766  024770            R!%W3 = R!FID + R!%W1                                       DISKLOAD.741
1547   030770  024770                    R!%W3 = (R!%W3)              .NEXT TWO FILE ID CHARS        DISKLOAD.742
1551    <macro>                          $IF (R!%W2 # R!%W3)          .IF NO MATCH                   DISKLOAD.743
                                                                                  .>>>>> 0                $IF      1
1555   010000  024751                      R!MCH = 0                  .NO  MATCH                     DISKLOAD.744
1557   001000                              EXIT                                                      DISKLOAD.745
1560    <macro>                          $ENDIF                                                      DISKLOAD.746
                                                                                  .<<<<< 0                $ENDIF   1
1560   026766                            R!%W1 = R!%W1 + 1            .NEXT PARCEL                   DISKLOAD.747
```

```
1561    <macro>                          $ENDTIL                                                    DISKLOAD.748
1562   010001  024751                    R!MCH = 1                    .MATCH                        DISKLOAD.749
1564   001000                            EXIT                                                       DISKLOAD.750
                                                                                                     DISKLOAD.751
                                                                                                     DISKLOAD.752
                                         *                                                           DISKLOAD.753
                                         * Convert logical block in file to physical block on disk  DISKLOAD.754
                                         *                                                           DISKLOAD.755
                                                                                                     DISKLOAD.756
1565                            GETBLK   *                                                           DISKLOAD.757
                                                                                                     DISKLOAD.758
1565   020743  024744                    R!BLK = R!LBN                .LOGICAL BLOCK NUMBER         DISKLOAD.759
                                                                                                     DISKLOAD.760
                                                                                                     DISKLOAD.761
                                         * Find data group containing logical block                  DISKLOAD.762
                                                                                                     DISKLOAD.763
1567    <macro>                          GET R!DGP,FD@POP,R!DIR       .DATA MAP POPULATION          DISKLOAD.764
1574   010000  024753                    R!DGX = 0                    .LOOP COUNTER                 DISKLOAD.765
                                                                                                     DISKLOAD.766
1576    <macro>                          ADDRESS R!DGB,FD@DGB,R!DIR   .FIRST DATA GROUP BEGIN ADDR  DISKLOAD.767
1601    <macro>                          ADDRESS R!DGC,FD@DGC,R!DIR   .FIRST DATA GROUP COUNT ADDR  DISKLOAD.768
1604    <macro>                          $UNTIL (R!DGX = R!DGP)       .ALL SCANNED                  DISKLOAD.769
1607   030754  024766                      R!%W1 = (R!DGB)            .GROUP BEGIN                  DISKLOAD.770
1611   030755  024767                      R!%W2 = (R!DGC)            .GROUP COUNT                  DISKLOAD.771
1613    <macro>                            $IF (R!BLK <= R!%W2)       .IF BLOCK IN THIS GROUP       DISKLOAD.772
                                                                                        .>>>>> 0          $IF      1
1621   020766  022744  013001                R!BLK = R!%W1 + R!BLK - 1 .OFFSET FROM GROUP BEGIN BLOCK DISKLOAD.773
       024744
1625   001000                                EXIT                                                    DISKLOAD.774
1626    <macro>                            $ENDIF                                                    DISKLOAD.775
                                                                                        .<<<<< 0          $ENDIF   1
1626   020744  023767  024744               R!BLK = R!BLK - R!%W2      .SKIP GROUP                   DISKLOAD.776
                                                                                                     DISKLOAD.777
1631   026753                               R!DGX = R!DGX + 1          .NEXT DATA MAP GROUP          DISKLOAD.778
1632   010002  025754                       R!DGB = R!DGB + FD@DG@Z    .                             DISKLOAD.779
1634   010002  025755                       R!DGC = R!DGC + FD@DG@Z    .                             DISKLOAD.780
1636    <macro>                          $ENDTIL                                                    DISKLOAD.781
                                                                                                     DISKLOAD.782
1637   075716 /000300                    P = ABORT                    .FILE DESCRIPTION ENTRY HOSED DISKLOAD.783
                                                                                                     DISKLOAD.784
                                                                                                     DISKLOAD.785
                                         *                                                           DISKLOAD.786
```

```
                                        * Convert block number and number of blocks to physical params DISKLOAD.787
                                        *                                                              DISKLOAD.788
                                                                                                       DISKLOAD.789
1641                            CONVERT *                                                              DISKLOAD.790
                                                                                                       DISKLOAD.791
1641  010000  024726                    R!HD = 0                      .INITIAL HEAD                  DISKLOAD.792
1643  020744  024766                    R!%W1 = R!BLK                 .COUNTER                       DISKLOAD.793
1645   <macro>                          $UNTIL (R!%W1 < DX$MB80)                                     DISKLOAD.794
1650  020766  013004  024766              R!%W1 = R!%W1 - DX$MB80     .DIVIDE BY BLOCKS PER HEAD     DISKLOAD.795
1653  026726                              R!HD = R!HD + 1             .NEXT HEAD                     DISKLOAD.796
1654   <macro>                          $ENDTIL                                                      DISKLOAD.797
                                                                                                       DISKLOAD.798
1655  010000  024727                    R!SCT = 0                     .INITIAL SECTOR                DISKLOAD.799
1657   <macro>                          $UNTIL (R!%W1 = 0)                                           DISKLOAD.800
1661  027766                              R!%W1 = R!%W1 - 1           .DEC BLOCK COUNTER             DISKLOAD.801
1662  010010  025727                      R!SCT = R!SCT + DX$MS80     .MULTIPLY BY SECTORS PER BLOCK DISKLOAD.802
1664   <macro>                          $ENDTIL                                                      DISKLOAD.803
                                                                                                       DISKLOAD.804
1665  010000  024725                    R!CYL = 0                     .INITIAL CYLINDER              DISKLOAD.805
1667   <macro>                          $UNTIL (R!HD < DX$MH80)                                      DISKLOAD.806
1672  020726  013005  024726              R!HD = R!HD - DX$MH80       .DIVIDE BY HEADS PER CYLINDER  DISKLOAD.807
1675  026725                              R!CYL = R!CYL + 1           .NEXT CYLINDER                 DISKLOAD.808
1676   <macro>                          $ENDTIL                                                      DISKLOAD.809
                                                                                                       DISKLOAD.810
1677  010000  024730                    R!CNT = 0                     .INITIAL SIZE                  DISKLOAD.811
1701  020745  024766                    R!%W1 = R!NUM                 .COUNTER                       DISKLOAD.812
1703   <macro>                          $UNTIL (R!%W1 = 0)                                           DISKLOAD.813
1705  010010  025730                      R!CNT = R!CNT + DX$MS80     .MULTIPLY BY SECTORS PER BLOCK DISKLOAD.814
1707  027766                              R!%W1 = R!%W1 - 1           .DEC BLOCK COUNT               DISKLOAD.815
1710   <macro>                          $ENDTIL                                                      DISKLOAD.816
1711  001000                            EXIT                                                         DISKLOAD.817
                                                                                                       DISKLOAD.818
                                                                                                       DISKLOAD.819
                                        *                                                              DISKLOAD.820
                                        * Format file statistics and move to MENULIN                 DISKLOAD.821
                                        *                                                              DISKLOAD.822
                                                                                                       DISKLOAD.823
1712                            FORMAT  *                                                              DISKLOAD.824
                                                                                                       DISKLOAD.825
1712  020716  012512  024732            R!MSG = R!BASE + MENULIN    .MESSAGE BUFFER                DISKLOAD.826
                                                                                                       DISKLOAD.827
1715  020732  012001  024763            R!DST = R!MSG + 1           .CLEAR MENULIN                 DISKLOAD.828
```

```
1720   010036  024764                R!SIZ = MENUSIZ                .                                DISKLOAD.829
1722   014000 /020040  024765        R!PAT = '  '                   .SET TO BLANKS                   DISKLOAD.830
1725   077716 /002103                R = CLEAR                      .                                DISKLOAD.831
                                                                                                      DISKLOAD.832
                                     * File name                                                      DISKLOAD.833
                                                                                                      DISKLOAD.834
1727    <macro>                      ADDRESS R!SRC,FD@FIL,R!DIR     .FILE NAME ADDR                  DISKLOAD.835
1732   020732  012006  024763        R!DST = R!MSG + MENUFIL        .MOVE TO MESSAGE                 DISKLOAD.836
1735   010020  024764                R!SIZ = FD@FIL@C               .CHARS TO MOVE                   DISKLOAD.837
1737   077716 /002120                R = MOVE                                                        DISKLOAD.838
                                                                                                      DISKLOAD.839
                                     * Creation date                                                  DISKLOAD.840
                                                                                                      DISKLOAD.841
1741    <macro>                      ADDRESS R!SRC,FD@CRD,R!DIR     .CREATION DATE ADDR              DISKLOAD.842
1744   020732  012017  024763        R!DST = R!MSG + MENUCRD        .MOVE TO MESSAGE                 DISKLOAD.843
1747   010010  024764                R!SIZ = FD@CRD@C               .CHARS TO MOVE                   DISKLOAD.844
1751   077716 /002120                R = MOVE                                                        DISKLOAD.845
                                                                                                      DISKLOAD.846
                                     * Creation time                                                  DISKLOAD.847
                                                                                                      DISKLOAD.848
1753    <macro>                      ADDRESS R!SRC,FD@CRT,R!DIR     .CREATION TIME ADDR              DISKLOAD.849
1756   020732  012024  024763        R!DST = R!MSG + MENUCRT        .MOVE TO MESSAGE                 DISKLOAD.850
1761   010010  024764                R!SIZ = FD@CRT@C               .CHARS TO MOVE                   DISKLOAD.851
1763   077716 /002120                R = MOVE                                                        DISKLOAD.852
                                                                                                      DISKLOAD.853
                                     * File size                                                      DISKLOAD.854
                                                                                                      DISKLOAD.855
1765    <macro>                      GET R!QTU,FD@SZU,R!DIR         .FILE SIZE (UPPER)               DISKLOAD.856
1772    <macro>                      GET R!QTL,FD@SZL,R!DIR         .FILE SIZE (LOWER)               DISKLOAD.857
1777   010075  024740                R!PNT = MENUWRD                .BYTE OFFSET OF SIZE FIELD END   DISKLOAD.858
                                                                                                      DISKLOAD.859
                                     * Convert binary to decimal                                      DISKLOAD.860
                                                                                                      DISKLOAD.861
2001    <macro>                      $UNTIL (R!QTU = 0) AND (R!QTL = 0) .QUOTIENT ZERO               DISKLOAD.862
2007   020756  024760                  R!RMU = R!QTU                .REMAINDER                       DISKLOAD.863
2011   020757  024761                  R!RML = R!QTL                                                 DISKLOAD.864
2013   010000  024756                  R!QTU = 0                    .QUOTIENT PRESET                 DISKLOAD.865
2015   010000  024757                  R!QTL = 0                                                     DISKLOAD.866
2017    <macro>                        $UNTIL (R!RMU = 0) AND (R!RML < D'10) .SINGLE DIGIT           DISKLOAD.867
2025    <macro>                          DSUB R!RMU,R!RML,D'10      .DIVIDE REMAINDER BY 10          DISKLOAD.868
2036    <macro>                          DADD R!QTU,R!QTL,1         .                                DISKLOAD.869
2041    <macro>                        $ENDTIL                                                       DISKLOAD.870
```

```
2042   020761  012060  024733          R!CHAR = R!RML +    R                    .ADD BIAS              DISKLOAD.871
2045    <macro>                         PUTBYTE R!CHAR,R!PNT,R!MSG,INC=O'177777 .ENTER CHAR          DISKLOAD.872
2101    <macro>                       $ENDTIL                                                        DISKLOAD.873
2102   .001000                        EXIT                                                           DISKLOAD.874
                                                                                                      DISKLOAD.875
                                                                                                      DISKLOAD.876
                                      *                                                              DISKLOAD.877
                                      * Clear data area                                              DISKLOAD.878
                                      *                                                              DISKLOAD.879
                                                                                                      DISKLOAD.880
2103                          CLEAR   *                                                              DISKLOAD.881
                                                                                                      DISKLOAD.882
2103   020763  024766                 R!%W1 = R!DST                  .POINTER                       DISKLOAD.883
2105    <macro>                       $UNTIL (R!%W1 = R!DST + R!SIZ) .ALL CLEARED                   DISKLOAD.884
2113   020765  034766                   (R!%W1) = R!PAT              .SET TO PATTERN                DISKLOAD.885
2115   026766                           R!%W1 = R!%W1 + 1            .NEXT PARCEL                   DISKLOAD.886
2116    <macro>                       $ENDTIL                                                        DISKLOAD.887
2117   001000                         EXIT                                                           DISKLOAD.888
                                                                                                      DISKLOAD.889
                                                                                                      DISKLOAD.890
                                      *                                                              DISKLOAD.891
                                      * MOVE STRING TO MESSAGE                                       DISKLOAD.892
                                      *                                                              DISKLOAD.893
                                                                                                      DISKLOAD.894
2120                          MOVE    *                                                              DISKLOAD.895
                                                                                                      DISKLOAD.896
2120   010000  024766                 R!%W1 = 0                      .SOURCE POINTER                DISKLOAD.897
2122    <macro>                       $UNTIL (R!%W1 = R!SIZ)         .ALL MOVED                     DISKLOAD.898
2125    <macro>                         GETBYTE R!CHAR,R!%W1,R!SRC   .NEXT CHAR                     DISKLOAD.899
2147    <macro>                         $IF (R!CHAR = 0)             .IF TERMINATOR                 DISKLOAD.900
                                                                                        .>>>>> 0          $IF      1
2152   001000                             EXIT                       .ALL DONE                      DISKLOAD.901
2153    <macro>                         $ENDIF                                                       DISKLOAD.902
                                                                                        .<<<<< 0          $ENDIF   1
2153    <macro>                         PUTBYTE R!CHAR,R!%W1,R!DST,INC=1 .PLACE CHAR IN MESSAGE      DISKLOAD.903
2204    <macro>                       $ENDTIL                                                        DISKLOAD.904
2205   001000                         EXIT                                                           DISKLOAD.905
                                                                                                      DISKLOAD.906
                                                                                                      DISKLOAD.907
2206                          LOADEND *                                                              DISKLOAD.908
                                      ERRIF  LOADEND,GT,XDK$SZ       .LOADER EXCEEDS MAX SIZE       R14673BC.29
                                                                                                      DISKLOAD.910
```

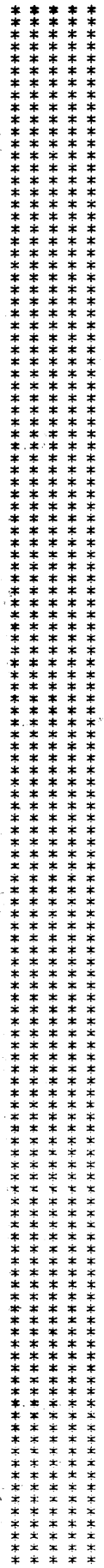
END

775	$REGORG		2: 8 D	2: 9	2: 9 D	2:10	2:10 D	2:11	2:11 D	2:12
			2:12 D	2:13 D	2:14	2:14 D	2:15	2:15 D	2:16	2:16 D
			2:17	2:17 D	2:18 F	2:19	2:19 D	2:20	2:20 D	2:21
			2:21 D	2:22 F	2:23	2:23 D	2:24	2:24 D	2:25	2:25 D
			2:26	2:26 D	2:27	2:27 D	2:28 F	2:29	2:29 D	2:30
			2:30 D	2:31	2:31 D	2:32 F	2:33	2:33 D	2:34	2:34 D
			2:35	2:35 D	2:36	2:36 D	2:37	2:37 D	2:38 F	2:39
			2:39 D	2:40	2:40 D	2:41	2:41 D	2:42	2:42 D	3: 1 F
			3: 2	3: 2 D	3: 3	3: 3 D	3: 4	3: 4 D	3: 5	3: 5 D
			3: 6 F	3: 7	3: 7 D	3: 8	3: 8 D	3: 9	3: 9 D	3:10
			3:10 D	3:11 F	3:12	3:12 D	3:13	3:13 D	3:14	3:14 D
			3:15	3:15 D	3:16 F	3:17	3:17 D	3:18	3:18 D	3:19
			3:19 D	3:20	3:20 D	3:21 F	3:22	3:22 D	3:23	3:23 D
			3:24	3:24 D	3:25 F	3:26	3:26 D	3:27	3:27 D	3:28
			3:28 D	3:29	3:29 D					
716	%DMP		2: 9 D							
716	%DUMP	$APTEXT	2: 8	2:13						
720	%MOSI	$APTEXT	2:11 D	18:30	19:19					
721	%MOSU	$APTEXT	2:12 D	18:31	19:20					
717	%STRT	$APTEXT	2:10 D	18:29	19:18					
771	%T0		2: 6	3:26 D	6:32	6:34	15: 9	15:33	16:37	18: 7
			18:24	18:25	18:26	18:27	18:35	18:36	19: 3	19:22
			19:23	20:34	22:17	24:28	24:29	25: 2	25:14	25:29
			25:35							
772	%T1		2: 6	3:27 D	8:12	8:15	15: 9	15:33	20:41	25: 2
			25:29	25:35						
773	%T2		2: 6	3:28 D	15:33	20:41	25: 2	25:35		
774	%T3		2: 6	3:29 D	20:41	24:40				
766	%W1		3:22 D	10:36	10:37	16:17	16:19	17:33	17:35	20:35
			20:37	21:30	21:31	21:32	21:34	21:42	22:23	22:27
			23: 7	23: 8	23: 9	23:14	23:15	23:26	23:27	23:29
			25:13	25:14	25:15	25:16	25:27	25:28	25:29	25:35
767	%W2		3:23 D	21:32	21:33	21:37	22:24	22:26	22:32	
770	%W3		3:24 D	21:34	21:35	21:37				
0	@EGET		4:25 D							
0	@EPUT		4:26 D	4:35 D						
0	@EXSGET		4:34 D							
300	ABORT		8:40	9: 9	9:32 D	22:39				
716	BASE		2:14 D	4:28	4:29	5:13 B	9:34	13:10	13:12	13:14
			13:22	13:29 B	13:37	14:10 B	14:12 B	14:16 B	14:37 B	14:39 B
			15:11 B	15:14 B	16:20	16:42 B	17:27	17:36	18:12 B	19:12 B
			19:36	19:37 B	19:42	20: 1 B	20: 5 B	20:28 B	22:39 B	23:40

			24: 3 B	24:10 B	24:17 B	24:24 B	24:34 B	25: 2 B		
744	BLK		2:41 D	19: 7	20:21	21:10	21:12	22:12	22:26	22:27
			22:32	23: 7						
376	BOOTEND		11:16 D	11:17 F						
400	BTSIZ		3:37 D	3:38	11:17 F					
731	BUF		2:27 D	6:17	6:28	6:32	8:18	10:39	16:35	18: 5
			18:14	19: 9	20:23	20:30	20:34	21:10		
10	CC$BS	$APTEXT	14:37	14:40	15: 2					
15	CC$CR	$APTEXT	11:12	11:21	11:26	11:31	11:40	12: 3	12:12	12:17
			14: 7							
4	CC$CTD	$APTEXT	13:36							
177	CC$DEL	$APTEXT	14:37							
12	CC$LF	$APTEXT	11:12	11:21	11:26	11:31	11:40	12: 3	12:12	12:17
			14: 7							
733	CHAR		2:30 D	10: 5	10: 8	10:22	13:36	13:42	14: 7	14:35
			14:37	14:40	14:42	15: 2	15: 7	15: 8	15: 9	15:13
			15:33	16: 5	16: 7	16: 9	17:15	25: 1	25: 2	25:29
			25:31	25:35						
734	CHARS		2:31 D	10: 4	10: 5	10: 8				
261	CHWT		7: 7	7:42	8: 3	8: 6	8:10	8:13	8:16	8:19
			8:23	8:36	9: 6 D	9:20				
2103	CLEAR		13:29	24: 3	25:11 D					
730	CNT		2:26 D	6:27	8:12	8:15	23:25	23:28		
1641	CONVERT		16:41	18:11	19:11	20:27	23: 4 D			
271	CTLWT		8: 7	9:17 D	9:24					
725	CYL		2:23 D	6:24	8: 2	23:19	23:22			
737	DBG		2:35 D	13:18	13:40	18:26				
0X	DEBUG		3:33 D	18:26						
754	DGB		3: 9 D	22:20	22:23	22:35				
755	DGC		3:10 D	22:21	22:24	22:36				
752	DGP		3: 7 D	22:17	22:22					
753	DGX		3: 8 D	22:18	22:22	22:34				
750	DID		3: 4 D	21:29	21:32					
736	DIR		2:34 D	13:14	13:16	16:17	16:37	17:33	18: 7	20:41
			22:17	22:20	22:21	24: 7	24:14	24:21	24:28	24:29
553	DIRBUF		12:31 D	13:14						
1453	DIREND		20: 8	20:32 D						
1445	DIRNXT		20:25 D	21:13						
1437	DIRSRC		16:13	17:24	19:33	20:19 D				
10	DR@DLOP	$APTEXT	20:30							
74	DR@DLOZ	$APTEXT	20: 7	21: 5						
20	DR@NXION	$APTEXT	21:10	21:10 F						

0	DR@NXT@P	$APTEXT	21:10 F							
0	DR@NXT@S	$APTEXT	21:10	21:10 F						
746	DRF		3: 2 D	16:15	17:26	19:35	20: 3	20: 7	20:30	20:34
			20:35	20:41	21: 5	21:19	21:29			
763	DST		3:18 D	13:26	23:42	24: 8	24:15	24:22	25:13	25:14
			25:35							
12	DV$HEAD	$APTEXT	8:15							
0	DV$READ	$APTEXT	8: 9							
5	DV$SELCT	$APTEXT	8:12	8:15						
400	DV$SEEK	$APTEXT	7:41							
60	DX$AMPEX	$APTEXT	7: 9 F							
27	DX$CDC	$APTEXT	7:40 F							
1	DX$DPBLK	$APTEXT	19: 7							
5	DX$DRBLK	$APTEXT	20:21							
4	DX$MB80	$APTEXT	23: 8	23: 9						
5	DX$MH80	$APTEXT	23:20	23:21						
10	DX$MS80	$APTEXT	23:16	23:28						
326	ECHO		10: 7	10:10	10:19 D	14:41	15: 1	15: 3	15:23	
742	EOI		2:39 D	16:37	16:39	18: 7	18: 9			
17	EXB	$APTEXT	7: 4	7: 6	7:41	8: 2	8: 5	8: 9	8:12	8:15
			8:18	8:22	8:29	8:35	8:37	9: 8	9:19	9:21
10	FD@CRD@C	$APTEXT	24:16							
21	FD@CRD@P	$APTEXT	24:14							
10	FD@CRT@C	$APTEXT	24:23							
25	FD@CRT@P	$APTEXT	24:21							
2	FD@DG@Z	$APTEXT	22:35	22:36						
40	FD@DGB@P	$APTEXT	22:20							
41	FD@DGC@P	$APTEXT	22:21							
20	FD@DTR@C	$APTEXT	14: 2	14: 6	14: 9	15: 5				
10	FD@DTR@Z	$APTEXT	19:32							
74	FD@END@P	$APTEXT	12:32	20:41						
20	FD@EOI@N	$APTEXT	16:37	16:37 F	18: 7	18: 7 F				
36	FD@EOI@P	$APTEXT	16:37 F	16:37	18: 7 F	18: 7				
0	FD@EOI@S	$APTEXT	16:37	16:37 F	18: 7	18: 7 F				
40	FD@FTD@C	$APTEXT	13:33							
1	FD@FTD@P	$APTEXT	13:16	21:29						
20	FD@FTD@Z	$APTEXT	13:27	16:12						
20	FD@FTL@C	$APTEXT	24: 9							
11	FD@FTL@P	$APTEXT	24: 7							
1	FD@FTW@M	$APTEXT	16:17	17:33						
1	FD@FTW@N	$APTEXT	16:17	16:17 F	17:33	17:33 F				
0	FD@FTW@P	$APTEXT	16:17 F	17:33 F						

2	FD@FLW@S	$APTEXT	16:17	16:17 F	17:33	17:33 F				
20	FD@POP@N	$APTEXT	22:17	22:17 F						
37	FD@POP@P	$APTEXT	22:17 F	22:17						
0	FD@POP@S	$APTEXT	22:17	22:17 F						
1	FD@STI@N	$APTEXT	20:35	20:35 F						
0	FD@STI@P	$APTEXT	20:35 F							
0	FD@STI@S	$APTEXT	20:35	20:35 F						
20	FD@SZL@N	$APTEXT	24:29	24:29 F						
32	FD@SZL@P	$APTEXT	24:29 F	24:29						
0	FD@SZL@S	$APTEXT	24:29	24:29 F						
20	FD@SZU@N	$APTEXT	24:28	24:28 F						
31	FD@SZU@P	$APTEXT	24:28 F	24:28						
0	FD@SZU@S	$APTEXT	24:28	24:28 F						
735	FID		2:33 D	13:16	13:26	15: 9	15:33	19: 3	21:34	
741	FLG		2:37 D	16: 1	16:26	18:23				
4000	FMSBFZP	$APTEXT	18:14	20:34						
1712	FORMAT		20: 4	23:38 D						
1345	FSTAT		14:12	18:42 D						
1565	GETBLK		16:40	18:10	22:10 D					
664	GETFID		13:20 D	13:41	14: 3	14:20	16:22	17:29	17:38	19:38
			20:10							
726	HD		2:24 D	6:25	8:15	23: 6	23:10	23:20	23:21	
27	I@XDK	$APTEXT	7: 4	7: 9 F	7:40 F					
250	INTWT		8:27 D	8:31						
743	LBN		2:40 D	16:38	16:39	17: 4	18: 8	18: 9	18:15	22:12
5	LDSIZ		3:38 D	6:27						
747	LIM		3: 3 D	16:12	19:32	21:31				
1125	LOAD		14:10	15:41 D						
647	LOAD2		6:32	13: 8 D						
2206	LOADEND		25:40 D	25:41 F						
1527	MATCH		20:38	21:27 D						
751	MCH		3: 5 D	20:40	21:38	22: 2				
17	MENUCRD		12:22 D	24:15						
24	MENUCRT		12:23 D	24:22						
6	MENUFIL		12:21 D	24: 8						
451	MENUHDR		12:11 D	19:42						
512	MENULIN		12:16 D	23:40						
36	MENUSIZ		12:25 D	24: 1						
75	MENUWRD		12:24 D	24:30						
5	MOS	$APTEXT	10:33	10:35	10:38	10:40	10:42	11: 1		
333	MOSIO		6:18	10:31 D	17: 1					
722	MOSL		2:19 D	6:14	10:34	10:36	16:30	17: 3		

723	MOSU		2:20 D	6:15	[illegible]	16:31	17: 3			
2120	MOVE		24:10	24:17	24:24	25:25 D				
732	MSG		2:29 D	9:34	10: 4	10:11	13:22	13:37	16:20	17:27
			17:36	19:36	19:42	23:40	23:42	24: 8	24:15	24:22
			25: 2							
355	MSGABRT		9:34	11:11 D						
444	MSGDBUG		12: 2 D	13:37						
434	MSGFILE		11:39 D	13:22						
410	MSGFLW		11:25 D	16:20	17:36					
420	MSGNODR		11:30 D	19:36						
376	MSGNOFI		11:20 D	17:27						
745	NUM		2:42 D	16:34	18: 4	19: 8	20:22	23:26		
0	OFFSDBUG	$APTEXT	13:18							
1	ONSDBUG	$APTEXT	13:40							
124	OPSAVE		5:13	5:17 D						
100000	OVLSMOSL	$APTEXT	16:30							
0	OVLSMOSU	$APTEXT	16:31							
765	PAT		3:20 D	13:28	24: 2	25:15				
740	PNT		2:36 D	13:31	13:33	14: 2	14: 6	14: 9	14:39	15: 5
			15: 6	15: 9	15:12	15:17	15:33	17:14	24:30	25: 2
1073	PUTCHAR		14:16	15:14	15:31 D	16: 6	16: 8	16:10	17:16	17:17
			17:18							
2	PXS	$APTEXT	4:25	4:26	4:34	4:35				
757	QTL		3:13 D	24:29	24:34	24:36	24:38	24:41		
756	QTU		3:12 D	24:28	24:34	24:35	24:37	24:41		
771	RECV		13:34	14:31 D	15:20					
0	REGSMOSL	$APTEXT	6:14	18:30	19:19					
0	REGSMOSU	$APTEXT	6:15	18:31	19:20					
761	RML		3:15 D	24:36	24:39	24:40	25: 1			
760	RMU		3:14 D	24:35	24:39	24:40				
32	SAVE		4:31 D	4:36						
717	SCR1		2:15 D	4:15	4:18					
720	SCR2		2:16 D	4:16	4:20					
721	SCR3		2:17 D	4:17	4:18	4:19	4:20	4:21	4:22	4:24
			4:25	4:33	4:34	5:20	5:21			
727	SCT		2:25 D	6:26	8:12	8:15	23:13	23:16		
306	SEND		9:36	10: 2 D	10:12	13:24	13:39	16:21	17:28	17:37
			19:37	20: 1	20: 5					
764	SIZ		3:19 D	13:27	24: 1	24: 9	24:16	24:23	25:14	25:28
762	SRC		3:17 D	24: 7	24:14	24:21	25:29			
0X	STRT		3:32 D	18:24						
46	TTD	$APTEXT	14:33	14:34						

47	TOD	$APTEXT	10:21	10:23				
724	WRDS		2:21 D	6:16	10:41	16:32	17: 3	
620	XD$TIME	$APTEXT	9: 8					
167	XDISK		6:30	6:42 D	16:42	18:12	19:12	20:28
1	XDK$STRT	$APTEXT	18:24	18:29	19:18			
2400	XDK$SZ	$APTEXT	3:38	6:28	13:10	25:41 F