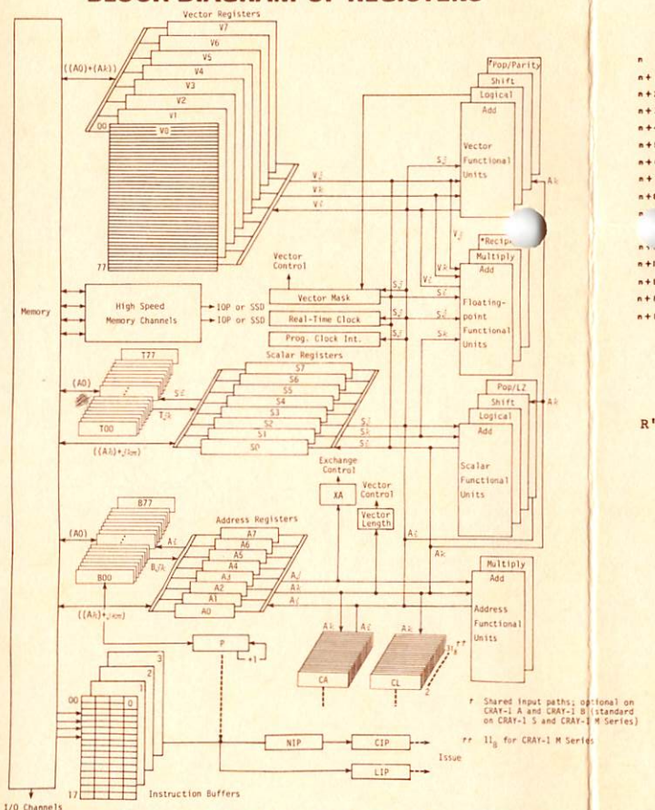
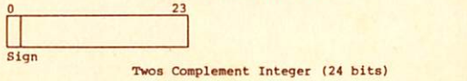
BLOCK DIAGRAM OF REGISTERS



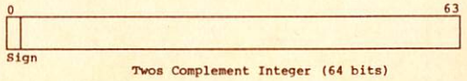
† Shared input paths; optional on CRAY-1 A and CRAY-1 B (standard on CRAY-1 S and CRAY-1 M Series)

†† 11_8 for CRAY-1 M Series

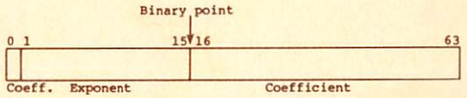
DATA FORMATS



Twos Complement Integer (24 bits)

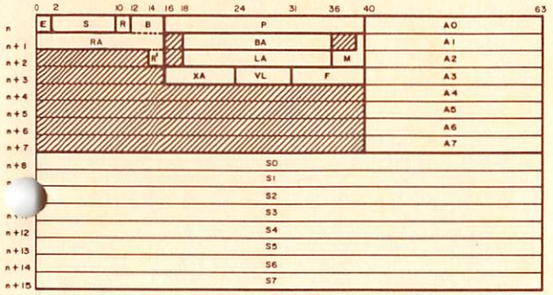


Twos Complement Integer (64 bits)



Signed Magnitude Floating-point (64 bits)

EXCHANGE PACKAGE



Registers

S	Syndrome bits
R'RAB	Read address for error (where B is bank)
P	Program Address, 24 bits
BA	Base Address, 18 bits
LA	Limit Address, 18 bits
XA	Exchange Address, 8 bits
VL	Vector Length, 7 bits

E - Error type (bits 0,1 of n)

10	Uncorrectable memory
01	Correctable memory

R - Read mode (bits 10,11 of n)

00	Scalar
01	I/O
10	Vector
11	Fetch

Word Offset	Bit	M - Modes
n+1	39	Interrupt monitor mode†
n+2	36	Interrupt on correctable memory error
n+2	37	Interrupt on floating-point error
n+2	38	Interrupt on uncorrectable memory error
n+2	39	Monitor mode

Word Offset	Bit	F - Flags
n+3	31	Programmable Clock Interrupt (PCI)††
n+3	32	MCU interrupt
n+3	33	Floating-point error
n+3	34	Operand range error
n+3	35	Program range error
n+3	36	Memory error
n+3	37	I/O interrupt
n+3	38	Error exit
n+3	39	Normal exit

† Supports Monitor Mode Interrupt option

†† Supports Programmable Clock option (optional on CRAY-1 Models A and B; standard on CRAY-1 S Series and CRAY-1 M Series computers)

FATAL ERRORS

C	Name, symbol, constant or data item error
D	Double defined symbol or duplicate parameter name
E	Definition or conditional sequence illegally nested
F	Too many entries
I	Instruction placement error
L	Location field error
N	Relocatable field error
O	Operand field error
P	Programmer error
R	Result field error
S	Syntax error
T	Type error
U	Undefined symbol or operation
V	Register expression or field width error
X	Expression error

WARNING ERRORS

W	Programmer warning error
W1	Location field symbol ignored
W2	Bad location symbol
W3	Expression element type error
W4	Possible symbolic machine instruction error
W5	Truncation error
W6	Location field symbol not defined
W7	Micro substitution error
W8	Address counter boundary error
Y1	External declaration error
Y2	Macro or opdef redefined

CONSTANT AND DATA NOTATION

Integer constant

$$\begin{Bmatrix}'O'\\'D'\\'X'\end{Bmatrix} [integer] \begin{Bmatrix}S+n\\S-n\end{Bmatrix}$$

Floating-point constant

$$\begin{Bmatrix}'O'\\'D'\\'X'\end{Bmatrix} \begin{bmatrix}integer.\\integer.fraction\\.fraction\end{bmatrix} \begin{Bmatrix}E+n\\En\\D+n\\Dn\end{Bmatrix} \begin{Bmatrix}S+n\\Sn\end{Bmatrix}$$

or

$$\begin{Bmatrix}'O'\\'D'\\'X'\end{Bmatrix} [integer] \begin{Bmatrix}E+n\\En\\D+n\\Dn\end{Bmatrix} \begin{Bmatrix}S+n\\Sn\end{Bmatrix}$$

Character constant

$$\begin{Bmatrix}'A'\\'C'\\'E'\end{Bmatrix} ['character\ string'] \begin{Bmatrix}'H'\\'L'\\'R'\\'Z'\end{Bmatrix}$$

Character data

$$\begin{Bmatrix}'A'\\'C'\\'E'\end{Bmatrix} ['character\ string'] [count] \begin{Bmatrix}'H'\\'L'\\'R'\\'Z'\end{Bmatrix}$$

Numeric data

Same as constant but may be preceded by $\begin{Bmatrix}+\\-\end{Bmatrix}$

CRAY RESEARCH, INC.

CRAY-1 CAL REFERENCE CARD

CAL CONTROL STATEMENT

CAL,CPU=*type*,I=*idn*,L=*ldn*,B=*bdn*,E=*edn*,ABORT,DEBUG,*options*,

LIST=*name*,S=*sdn*,SYM=*sym*,T=*bst*,X=*xdn*.

CPU	Omitted	Machine currently executing CAL
	CPU=*type*	Specify CRAY-1 or CRAY-XMP
I	Omitted	Source on $IN
	I=*idn*	Source on *idn*
L	Omitted	List output on $OUT
	L=0	No list output
	L=*ldn*	List output on *ldn*
B	Omitted	Binary on $BLD
	B=0	No binary
	B=*bdn*	Binary on *bdn*
E	Omitted	No error listing
	E	Error list on $OUT
	E=*edn*	Error list on *edn* unless *edn*=*ldn*, then *ldn*
ABORT	Omitted	Do not abort
	ABORT	Abort on fatal error during assembly
DEBUG	Omitted	Write binary record on fatal error and set fatal error flag
	DEBUG	Write binary record with fatal error flag clear
options	See *options* under CAL control statement in CAL Reference Manual (*options* overrides the LIST pseudo.)	
LIST	Omitted	LIST pseudos with a null (empty) location field processed
	LIST	All LIST pseudos processed
	LIST=*name*	LIST pseudo instructions with a location field matching name processed
S	Omitted	$SYSTXT
	S=0	No system text
	S=*sdn*	System text on *sdn*
SYM	Omitted	No symbol table
	SYM	Symbol table on dataset holding binary load data
	SYM=*sym*	Symbol table on *sym*
T	Omitted	No binary system text written
	T=0	No binary system text written
	T	Binary dataset written to $BST
	T=*bst*	Binary system text written to *bst*
X	Omitted	No global cross-reference records written
	X=0	No global cross-reference records written
	X	Global cross-reference records written to $XRF
	X=*xdn*	Global cross-reference records written to *xdn*

INSTRUCTIONS

CRAY-1	CAL	UNIT	DESCRIPTION
000*zxx*	ERR	-	Error exit
†000*ijk*	ERR *exp*	-	Error exit
††0010*jk*	CA,A*j* Ak	-	Set the channel (A*j*) current address to (Ak) and begin the I/O sequence
††0011*jk*	CL,A*j* Ak	-	Set the channel (A*j*) limit address to (Ak)
††0012*jz*	CI,A*j*	-	Clear channel (A*j*) interrupt flag
††0013*jz*	XA A*j*	-	Enter XA register with (A*j*)
††0014*j*0	RT S*j*	-	Enter RTC register with (S*j*)
††§0014*j*4	PCI S*j*	-	Enter II register with (S*j*)
††§0014*z*5	CCI	-	Clear PCI request
††§0014*z*6	ECI	-	Enable PCI request
††§0014*z*7	DCI	-	Disable PCI request
0020*zk*	VL Ak	-	Transmit (Ak) to VL register
†0020*z*0	VL 1	-	Transmit 1 to VL register
0021*zz*	EFI	-	Enable interrupt on floating-point error
0022*zz*	DFI	-	Disable interrupt on floating-point error
003*zjz*	VM S*j*	-	Transmit (S*j*) to VM register
†003*z*0*z*	VM 0	-	Clear VM register
004*zzz*	EX	-	Normal exit
†004*ijk*	EX *exp*	-	Normal exit
005*zjk*	J B*jk*	-	Jump to (B*jk*)
006*ijkm*	J *exp*	-	Jump to *exp*
007*ijkm*	R *exp*	-	Return jump to *exp*; set B00 to P.
010*ijkm*	JAZ *exp*	-	Branch to *exp* if (A0)=0
011*ijkm*	JAN *exp*	-	Branch to *exp* if (A0)≠0
012*ijkm*	JAP *exp*	-	Branch to *exp* if (A0)≥0
013*ijkm*	JAM *exp*	-	Branch to *exp* if (A0)<0
014*ijkm*	JSZ *exp*	-	Branch to *exp* if (S0)=0
015*ijkm*	JSN *exp*	-	Branch to *exp* if (S0)≠0
016*ijkm*	JSP *exp*	-	Branch to *exp* if (S0)≥0
017*ijkm*	JSM *exp*	-	Branch to *exp* if (S0)<0
†††020*ijkm*	Ai *exp*	-	Transmit *exp*=*jkm* to Ai
†††021*ijkm*	Ai *exp*	-	Transmit *exp*=ones complement of *jkm* to Ai
†††022*ijk*	Ai *exp*	-	Transmit *exp*=*jk* to Ai
023*ijz*	Ai S*j*	-	Transmit (S*j*) to Ai
024*ijk*	Ai B*jk*	-	Transmit (B*jk*) to Ai
025*ijk*	B*jk* Ai	-	Transmit (Ai) to B*jk*
0266*ij*0	Ai PS*j*	Pop/LZ	Population count of (S*j*) to Ai
§§0026*ij*1	Ai QS*j*	Pop/LZ	Population count parity of (S*j*) to Ai
027*ijz*	Ai ZS*j*	Pop/LZ	Leading zero count of (S*j*) to Ai
030*ijk*	Ai A*j*+Ak	A Int Add	Integer sum of (A*j*) and (Ak) to Ai
†030*i*0*k*	Ai Ak	A Int Add	Transmit (Ak) to Ai
†030*ij*0	Ai A*j*+1	A Int Add	Integer sum of (A*j*) and 1 to Ai
031*ijk*	Ai A*j*-Ak	A Int Add	Integer difference of (A*j*) less (Ak) to Ai
† †††031*i*00	Ai -1	A Int Add	Transmit -1 to Ai
†031*i*0*k*	Ai -Ak	A Int Add	Transmit the negative of (Ak) to Ai
†031*ij*0	Ai A*j*-1	A Int Add	Integer difference of (A*j*) less 1 to Ai
032*ijk*	Ai A*j**Ak	A Int Mult	Integer product of (A*j*) and (Ak) to Ai
0336*i*0*z*	Ai CI	-	Channel number to Ai (*j*=0)
033*ij*0	Ai CA,A*j*	-	Address of channel (A*j*) to Ai (*j*≠0; *k*=0)
033*ij*1	Ai CE,A*j*	-	Error flag of channel (A*j*) to Ai (*j*≠0; *k*=1)
034*ijk*	B*jk*,Ai ,A0	Memory	Read (Ai) words to B register *jk* from (A0)
†034*ijk*	B*jk*,Ai 0,A0	Memory	Read (Ai) words to B register *jk* from (A0)
035*ijk*	,A0 B*jk*,Ai	Memory	Store (Ai) words at B register *jk* to (A0)
†035*ijk*	0,A0 B*jk*,Ai	Memory	Store (Ai) words at B register *jk* to (A0)
036*ijk*	T*jk*,Ai ,A0	Memory	Read (Ai) words to T register *jk* from (A0)
†036*ijk*	T*jk*,Ai 0,A0	Memory	Read (Ai) words to T register *jk* from (A0)
037*ijk*	,A0 T*jk*,Ai	Memory	Store (Ai) words at T register *jk* to (A0)
†037*ijk*	0,A0 T*jk*,Ai	Memory	Store (Ai) words at T register *jk* to (A0)

CRAY-1	CAL	UNIT	DESCRIPTION
040*ijkm*	Si *exp*	-	Transmit *jkm* to Si
041*ijkm*	Si *exp*	-	Transmit *exp*=ones complement of *jkm* to Si
042*ijk*	Si <*exp*	S Logical	Form ones mask *exp* bits in Si from the right; *jk* field gets 64-*exp*.
†042*ijk*	Si #>*exp*	S Logical	Form zeros mask *exp* bits in Si from the left; *jk* field gets *exp*.
†042*i*77	Si 1	S Logical	Enter 1 into Si
†042*i*00	Si -1	S Logical	Enter -1 into Si
043*ijk*	Si >*exp*	S Logical	Form ones mask *exp* bits in Si from the left; *jk* field gets *exp*.
†043*ijk*	Si #<*exp*	S Logical	Form zeros mask *exp* bits in Si from the right; *jk* field gets 64-*exp*.
†044*i*00	Si 0	S Logical	Clear Si
044*ijk*	Si Sj&Sk	S Logical	Logical product of (S*j*) and (Sk) to Si
†044*ij*0	Si Sj&SB	S Logical	Sign bit of (S*j*) to Si
†044*ij*0	Si SB&S*j*	S Logical	Sign bit of (S*j*) to Si (*j*≠0)
045*ijk*	Si #Sk&S*j*	S Logical	Logical product of (S*j*) and ones complement of (Sk) to Si
†045*ij*0	Si #SB&S*j*	S Logical	(S*j*) with sign bit cleared to Si
046*ijk*	Si S*j*\Sk	S Logical	Logical difference of (S*j*) and (Sk) to Si
†046*ij*0	Si S*j*\SB	S Logical	Toggle sign bit of S*j*, then enter into Si
†046*ij*0	Si SB\S*j*	S Logical	Toggle sign bit of S*j*, then enter into Si (*j*≠0)
047*ijk*	Si #S*j*\Sk	S Logical	Logical equivalence of (Sk) and (S*j*) to Si
†047*i*0*k*	Si #Sk	S Logical	Transmit ones complement of (Sk) to Si
†047*ij*0	Si #S*j*\SB	S Logical	Logical equivalence of (S*j*) and sign bit to Si
†047*ij*0	Si #SB\S*j*	S Logical	Logical equivalence of (S*j*) and sign bit to Si (*j*≠0)
†047*i*00	Si #SB	S Logical	Enter ones complement of sign bit into Si
050*ijk*	Si S*j*!Si&Sk	S Logical	Logical product of (Si) and (Sk) complement ORed with logical product of (S*j*) and (Sk) to Si
†050*ij*0	Si S*j*!Si&SB	S Logical	Scalar merge of (Si) and sign bit of (S*j*) to Si
051*ijk*	Si S*j*!Sk	S Logical	Logical sum of (S*j*) and (Sk) to Si
†051*i*0*k*	Si Sk	S Logical	Transmit (Sk) to Si
†051*ij*0	Si S*j*!SB	S Logical	Logical sum of (S*j*) and sign bit to Si
†051*ij*0	Si SB!S*j*	S Logical	Logical sum of (S*j*) and sign bit to Si (*j*≠0)
†051*i*00	Si SB	S Logical	Enter sign bit into Si
052*ijk*	Si Si<*exp*	S Shift	Shift (Si) left *exp*=*jk* places to Si
053*ijk*	Si Si>*exp*	S Shift	Shift (Si) right *exp*=64-*jk* places to Si
054*ijk*	Si Si<*exp*	S Shift	Shift (Si) left *exp*=*jk* places
055*ijk*	Si Si>*exp*	S Shift	Shift (Si) right *exp*=64-*jk* places
056*ijk*	Si Si,Sj<Ak	S Shift	Shift (Si and S*j*) left (Ak) places to Si
†056*ij*0	Si Si,Sj<1	S Shift	Shift (Si and S*j*) left one place to Si
†056*i*0*k*	Si Si<Ak	S Shift	Shift (Si) left (Ak) places to Si
057*ijk*	Si Sj,Si>Ak	S Shift	Shift (S*j* and Si) right (Ak) places to Si
†057*ij*0	Si Sj,Si>1	S Shift	Shift (S*j* and Si) right one place to Si
†057*i*0*k*	Si Si>Ak	S Shift	Shift (Si) right (Ak) places to Si
060*ijk*	Si Sj+Sk	S Int Add	Integer sum of (S*j*) and (Sk) to Si
061*ijk*	Si Sj-Sk	S Int Add	Integer difference of (S*j*) and (Sk) to Si
†061*i*0*k*	Si -Sk	S Int Add	Transmit negative of (Sk) to Si
062*ijk*	Si Sj+FSk	Fp Add	Floating-point sum of (S*j*) and (Sk) to Si
†062*i*0*k*	Si +FSk	Fp Add	Normalize (Sk) to Si
063*ijk*	Si Sj-FSk	Fp Add	Floating-point difference of (S*j*) and (Sk) to Si
†063*i*0*k*	Si -FSk	Fp Add	Transmit normalized negative of (Sk) to Si
064*ijk*	Si Sj*FSk	Fp Mult	Floating-point product of (S*j*) and (Sk) to Si
065*ijk*	Si Sj*HSk	Fp Mult	Half-precision rounded floating-point product of (S*j*) and (Sk) to Si
066*ijk*	Si Sj*RSk	Fp Mult	Full-precision rounded floating-point product of (S*j*) and (Sk) to Si
067*ijk*	Si Sj*ISk	Fp Mult	2-floating-point product of (S*j*) and (Sk) to Si
070*ijz*	Si /HS*j*	Fp Rcpl	Floating-point reciprocal approximation of (S*j*) to Si
071*i*0*k*	Si Ak	-	Transmit (Ak) to Si with no sign extension

CRAY-1	CAL	UNIT	DESCRIPTION
071*i*1*k*	Si +Ak	-	Transmit (Ak) to Si with sign extension
071*i*2*k*	Si +FAk	-	Transmit (Ak) to Si as unnormalized floating-point number
071*i*3*z*	Si 0.6	-	Transmit constant 0.75*2**48 to Si
071*i*4*z*	Si 0.4	-	Transmit constant 0.5 to Si
071*i*5*z*	Si 1.	-	Transmit constant 1.0 to Si
071*i*6*z*	Si 2.	-	Transmit constant 2.0 to Si
071*i*7*z*	Si 4.	-	Transmit constant 4.0 to Si
072*izz*	Si RT	-	Transmit (RTC) to Si
073*izz*	Si VM	-	Transmit (VM) to Si
074*ijk*	Si Tjk	-	Transmit (Tjk) to Si
075*ijk*	Tjk Si	-	Transmit (Si) to Tjk
076*ijk*	Si Vj,Ak	-	Transmit (Vj, element (Ak)) to Si
077*ijk*	Vi,Ak Sj	-	Transmit (Sj) to Vi element (Ak)
†077*i*0*k*	Vi,Ak 0	-	Clear Vi element (Ak)
10*hijkm*	Ai *exp*,Ah	Memory	Read from ((Ah)+*exp*) to Ai (A0=0)
†100*ijkm*	Ai *exp*,0	Memory	Read from (*exp*) to Ai
†100*ijkm*	Ai *exp*,	Memory	Read from (*exp*) to Ai
†10*hi*000	Ai ,Ah	Memory	Read from (Ah) to Ai
11*hijkm*	*exp*,Ah Ai	Memory	Store (Ai) to (Ah)+*exp* (A0=0)
†110*ijkm*	*exp*,0 Ai	Memory	Store (Ai) to *exp*
†110*ijkm*	*exp*, Ai	Memory	Store (Ai) to *exp*
†11*hi*000	,Ah Ai	Memory	Store (Ai) to (Ah)
12*hijkm*	Si *exp*,Ah	Memory	Read from ((Ah)+*exp*) to Si (A0=0)
†120*ijkm*	Si *exp*,0	Memory	Read from (*exp*) to Si
†120*ijkm*	Si *exp*,	Memory	Read from (*exp*) to Si
†12*hi*000	Si ,Ah	Memory	Read from (Ah) to Si
13*hijkm*	*exp*,Ah Si	Memory	Store (Si) to (Ah)+*exp* (A0=0)
†130*ijkm*	*exp*,0 Si	Memory	Store (Si) to *exp*
†130*ijkm*	*exp*, Si	Memory	Store (Si) to *exp*
†13*hi*000	,Ah Si	Memory	Store (Si) to (Ah)
140*ijk*	Vi Sj&Vk	V Logical	Logical products of (S*j*) and (Vk) to Vi
141*ijk*	Vi Vj&Vk	V Logical	Logical products of (V*j*) and (Vk) to Vi
142*ijk*	Vi Sj!Vk	V Logical	Logical sums of (S*j*) and (Vk) to Vi
†142*i*0*k*	Vi Vk	V Logical	Transmit (Vk) to Vi
143*ijk*	Vi Vj!Vk	V Logical	Logical sums of (V*j*) and (Vk) to Vi
144*ijk*	Vi Sj\Vk	V Logical	Logical differences of (S*j*) and (Vk) to Vi
145*ijk*	Vi Vj\Vk	V Logical	Logical differences of (V*j*) and (Vk) to Vi
†145*iii*	Vi 0	V Logical	Clear Vi
146*ijk*	Vi Sj!Vk&VM	V Logical	Transmit (S*j*) if VM bit=1; (Vk) if VM bit=0 to Vi.
†146*i*0*k*	Vi #VM&Vk	V Logical	Vector merge of (Vk) and 0 to Vi
147*ijk*	Vi Vj!Vk&VM	V Logical	Transmit (V*j*) if VM bit=1; (Vk) if VM bit=0 to Vi.
150*ijk*	Vi Vj<Ak	V Shift	Shift (V*j*) left (Ak) places to Vi
†150*ij*0	Vi Vj<1	V Shift	Shift (V*j*) left one place to Vi
151*ijk*	Vi Vj>Ak	V Shift	Shift (V*j*) right (Ak) places to Vi
†151*ij*0	Vi Vj>1	V Shift	Shift (V*j*) right one place to Vi
152*ijk*	Vi Vj,Vj<Ak	V Shift	Double shift (V*j*) left (Ak) places to Vi
†152*ij*0	Vi Vj,Vj<1	V Shift	Double shift (V*j*) left one place to Vi
153*ijk*	Vi Vj,Vj>Ak	V Shift	Double shift (V*j*) right (Ak) places to Vi
†153*ij*0	Vi Vj,Vj>1	V Shift	Double shift (V*j*) right one place to Vi
154*ijk*	Vi Sj+Vk	V Int Add	Integer sums of (S*j*) and (Vk) to Vi
155*ijk*	Vi Vj+Vk	V Int Add	Integer sums of (V*j*) and (Vk) to Vi
156*ijk*	Vi Sj-Vk	V Int Add	Integer differences of (S*j*) and (Vk) to Vi
†156*i*0*k*	Vi -Vk	V Int Add	Transmit negative of (Vk) to Vi
157*ijk*	Vi Vj-Vk	V Int Add	Integer differences of (V*j*) and (Vk) to Vi
160*ijk*	Vi Sj*FVk	Fp Mult	Floating-point products of (S*j*) and (Vk) to Vi

CRAY-1	CAL	UNIT	DESCRIPTION
161*ijk*	Vi Vj*FVk	Fp Mult	Floating-point products of (V*j*) and (Vk) to Vi
162*ijk*	Vi Sj*HVk	Fp Mult	Half-precision rounded floating-point products of (S*j*) and (Vk) to Vi
163*ijk*	Vi Vj*HVk	Fp Mult	Half-precision rounded floating-point products of (V*j*) and (Vk) to Vi
164*ijk*	Vi Sj*RVk	Fp Mult	Rounded floating-point products of (S*j*) and (Vk) to Vi
165*ijk*	Vi Vj*RVk	Fp Mult	Rounded floating-point products of (V*j*) and (Vk) to Vi
166*ijk*	Vi Sj*IVk	Fp Mult	2-floating-point products of (S*j*) and (Vk) to Vi
167*ijk*	Vi Vj*IVk	Fp Mult	2-floating-point products of (V*j*) and (Vk) to Vi
ijk	Vi Sj+FVk	Fp Add	Floating-point sums of (S*j*) and (Vk) to Vi
†170*i*0*k*	Vi +FVk	Fp Add	Normalize (Vk) to Vi
171*ijk*	Vi Vj+FVk	Fp Add	Floating-point sums of (V*j*) and (Vk) to Vi
172*ijk*	Vi Sj-FVk	Fp Add	Floating-point differences of (S*j*) and (Vk) to Vi
†172*i*0*k*	Vi -FVk	Fp Add	Transmit normalized negatives of (Vk) to Vi
173*ijk*	Vi Vj-FVk	Fp Add	Floating-point differences of (V*j*) and (Vk) to Vi
174*ij*0	Vi /HV*j*	Fp Rcpl	Floating-point reciprocal approximations of (V*j*) to Vi
§§174*ij*1	Vi PV*j*	V Pop	Population counts of (V*j*) to Vi
§§174*ij*2	Vi QV*j*	V Pop	Population count parities of (V*j*) to Vi
175*zj*0	VM V*j*,Z	V Logical	VM=1 where (V*j*)=0
175*zj*1	VM V*j*,N	V Logical	VM=1 where (V*j*)≠0
175*zj*2	VM V*j*,P	V Logical	VM=1 where (V*j*) positive
175*zj*3	VM V*j*,M	V Logical	VM=1 where (V*j*) negative
176*izk*	Vi ,A0,Ak	Memory	Read (VL) words to Vi from (A0) incremented by (Ak)
†176*iz*1	Vi ,A0,1	Memory	Read (VL) words to Vi from (A0) incremented by 1
177*zjk*	,A0,Ak V*j*	Memory	Store (VL) words from V*j* to (A0) incremented by (Ak)
†177*zj*0	,A0,1 V*j*	Memory	Store (VL) words from V*j* to (A0) incremented by 1

† Special syntax form
†† Privileged to monitor mode
††† Generated depending on value of *exp*
§ Programmable clock (optional on CRAY-1 Models A and B)
§§ Vector Population Count (optional on CRAY-1 Models A and B)
z Field not used by hardware; assembler generates zero in this position.

REGISTER	VALUE
Ah, *h*=0	0
Ai, *i*=0	(A0)
Aj, *j*=0	0
Ak, *k*=0	1
Si, *i*=0	(S0)
Sj, *j*=0	0
Sk, *k*=0	2^{63}

LOGICAL OPERATORS	
&	0101
AND	1100
	0100
!	0101
OR	1100
	1101
\	0101
XOR	1100
	1001

CRAY RESEARCH, INC.
Software Development Division
1440 Northland Drive, Mendota Heights, Minnesota 55120