CRAY X-MP	CAL	Unit	Description
013ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A3 (bit 2^2 of i = 1)
014ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A4 (bit 2^2 of i = 1)
015ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A5 (bit 2^2 of i = 1)
016ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A6 (bit 2^2 of i = 1)
017ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A7 (bit 2^2 of i = 1
†††020ijkm	Ai *exp*	N/A	Transmit *exp* = jkm to Ai
†††021ijkm	Ai *exp*	N/A	Transmit *exp* = (ones complement of jkm) to Ai
†††022ijk	Ai *exp*	N/A	Transmit *exp* = jk to Ai
023ij0	Ai Sj	N/A	Transmit = (Sj) to Ai
023i01	Ai VL	N/A	Transmit (VL) to Ai
024ijk	Ai Bjk	N/A	Transmit (Bjk) to Ai
025ijk	Bjk Ai	N/A	Transmit (Ai) to Bjk
026ij0	Ai PSj	S Pop	Population count of (Sj) to Ai
026ij1	Ai QSj	S Pop	Population count parity of (Sj) to Ai
026ij7	Ai SBj	N/A	Transmit (SBj) to Ai
027ij0	Ai ZSj	S/LZ	Leading zero count of (Sj) to Ai
027ij7	SBj Ai	N/A	Transmit (Ai) to SBj
030ijk	Ai Aj + Ak	A Int Add	Integer sum of (Aj) and (Ak) to Ai
†030i00	Ai 1	A Int Add	Transmit 1 to Ai
†030i0k	Ai Ak	A Int Add	Transmit (Ak) to Ai
†030ij0	Ai Aj + 1	A Int Add	Integer sum of (Aj) and 1 to Ai
031ijk	Ai Aj-Ak	A Int Add	Integer difference of (Aj) less (Ak) to Ai
† †††031i00	Ai -1	A Int Add	Transmit -1 to Ai; (Ai = 77777777)
†031i0k	Ai -Ak	A Int Add	Transmit the negative of (Ak) to Ai
†031ij0	Ai Aj-1	A Int Add	Integer difference of (Aj) less 1 to Ai
032ijk	Ai Aj*Ak	A Int Mult	Integer product of (Aj) and (Ak) to Ai
†032ij0	Ai Aj*1	A Int Mult	Transmit (Aj) to Ai
†0320k	Ai Ak*0	A Int Mult	Zero fill Ai; (Ai = 00000000)
033i00	Ai CI	N/A	Channel number to Ai (j = 0)
033ij0	Ai CA,Aj	N/A	Address of channel (Aj) to Ai (j ≠ 0; k = 0)
033ij1	Ai CE,Aj	N/A	Error flag of channel (Aj) to Ai (j ≠ 0; k = 1)
034ijk	Bjk,Ai ,A0	Memory	Read (Ai) words to B register jk from memory address ((A0) + (DBA))
†034ijk	Bjk,Ai 0,A0	Memory	Read (Ai) words to B register jk from memory address ((A0) + (DBA))
035ijk	,A0 Bjk,Ai	Memory	Write (Ai) words at B register jk into memory address ((A0) + (DBA))
†035ijk	0,A0 Bjk,A1	Memory	Write (Ai) words at B register jk into memory address ((A0) + (DBA))
036ijk	Tjk,Ai ,A0	Memory	Read (Ai) words to T register jk from memory Address ((A0) + (DBA))
†036ijk	Tjk,Ai 0,A0	Memory	Read (Ai) words to T register jk from memory address ((A0) + (DBA))
037ijk	,A0 Tjk,Ai	Memory	Write (Ai) words at T register jk into memory address ((A0) + (DBA))
†037ijk	0,A0 Tjk,A1	Memory	Write (Ai) words at T register jk into memory address ((A0) + (DBA))
040ijkm	Si *exp*	N/A	Transmit *exp* = jkm to Si
041ijkm	Si *exp*	N/A	Transmit *exp* = ones complement of jkm to Si
042ijk	Si < *exp*	S Logical	Form zeros mask *exp* bits in Si from the left; jk field gets *exp*
†042ijk	Si # > *exp*	S Logical	Form zeros mask *exp* bits in Si from the left; jk field get *exp*
†042i77	Si 1	S Logical	Enter 1 into Si
†042i00	Si -1	S Logical	Enter -1 into Si; (Si = 177777 177777 177777 177777)
043ijk	Si > *exp*	S Logical	Form ones mask *exp* bits in Si from the left; jk field gets *exp*
†043ijk	Si # < *exp*	S Logical	Form ones mask *exp* bits in Si from the left; jk field gets *exp*
†043i00	Si 0	S Logical	Clear Si
044ijk	Si Sj&Sk	S Logical	Logical product of (Sj) and (Sk) to Si
†044ij0	Si Sj&SB	S Logical	Sign bit of (Sj) to Si
†044ij0	Si SB&Sj	S Logical	Sign bit of (Sj) to Si (j ≠ 0)

CRAY X-MP	CAL	Unit	Description
045ijk	Si #Sk&Sj	S Logical	Logical product of (Sj) and ones complement of (Sk) to Si
†045ij0	Si #SB&Sj	S Logical	(Sj) with sign bit cleared to Si
046ijk	Si Sj/Sk	S Logical	Logical difference of (Sj) and (Sk) to Si
†046ij0	Si Sj/SB	S Logical	Toggle sign bit of Sj, then enter into Si
†046ij0	Si SB/Sj	S Logical	Toggle sign bit of Sj, then enter into Si (j ≠ 0)
047ijk	Si #Sj/Sk	S Logical	Logical equivalence of (Sk) and (Sj) to Si
†047i0k	Si #Sk	S Logical	Transmit ones complement of (Sk) to Si
†047ij0	Si #Sj/SB	S Logical	Logical equivalence of (Sj) and sign bit to Si
†047ij0	Si #SB/Sj	S Logical	Logical equivalence of (Sj) and sign bit to Si (j ≠ 0)
†047i00	Si #SB	S Logical	Enter ones complement of sign bit into Si
050ijk	Si Sj!Si&Sk	S Logical	Logical product of (Si) and (Sk) complement ORed with logical product of (Sj) and (Sk) to Si
†050ij0	Si Sj!Si&SB	S Logical	Scalar merge of (Si) and sign bit of (Sj) to Si
051ijk	Si Sj!Sk	S Logical	Logical sum of (Sj) and (Sk) to Si
†051i0k	Si Sk	S Logical	Transmit (Sk) to Si
†051ij0	Si Sj!SB	S Logical	Logical sum of (Sj) and sign bit to Si
†051ij0	Si SB!Sj	S Logical	Logical sum of (Sj) and sign bit to Si (j ≠ 0)
†051i00	Si SB	S Logical	Enter sign bit into Si
052ijk	S0 Si< *exp*	S Shift	Shift (Si) left *exp* = jk places to S0
053ijk	S0 Si> *exp*	S Shift	Shift (Si) right *exp* = 100_8 − jk places to S0
054ijk	Si Si< *exp*	S Shift	Shift (Si) left *exp* = jk places to Si
055ijk	Si Si> *exp*	S Shift	Shift (Si) right *exp* = 100_8 − jk places to Si
056ijk	Si Si,Sj< Ak	S Shift	Shift (Si) and (Sj) left (Ak) places to Si
†056ij0	Si Si,Sj< 1	S Shift	Shift (Si) and (Sj) left one place to Si
†056i0k	Si Si< Ak	S Shift	Shift (Si) left (Ak) places to Si
057ijk	Si Sj,Si > Ak	S Shift	Shift (Sj) and (Si) right (Ak) places to Si
†057ij0	Si Sj,Si > 1	S Shift	Shift (Sj and Si) right one place to Si
†057i0k	Si Si > Ak	S Shift	Shift (Si) fight (Ak) places to Si
060ijk	Si Sj + Sk	S Int Add	Integer sum of (Sj) and (Sk) to Si
†060i0k	Si Sk	S Int Add	Transmit (Sk) to Si
†060ij0	Si Sj + S0	S Int Add	Integer sum (2^{63}) and (Sj) to Si
061ijk	Si Sj-Sk	S Int Add	Integer difference of (Sj) and (Sk) to Si
†061i0k	Si -Sk	S Int Add	Transmit negative of (Sk) to Si
†061ij0	Si Sj-S0	S Int Add	Integer difference of (Sj) and 2^{63} to Si
062ijk	Si Sj + FSk	Fp Add	Floating-point sum of (Sj) and (Sk) to Si
†062i0k	Si + FSk	Fp Add	Normalize (Sk) to Si
063ijk	Si Sj-FSk	Fp Add	Floating-point difference of (Sj) and (Sk) to Si
†063i0k	Si -FSk	Fp Add	Transmit normalized negative of (Sk) to Si
064ijk	Si Sj*FSk	Fp Mult	Floating-point product of (Sj) and (Sk) to Si
065ijk	Si Sj*HSk	Fp Mult	Half-precision rounded floating-point product of (Sj) and (Sk) to Si
066ijk	Si Sj*RSk	Fp Mult	Full-precision rounded floating-point product of (Sj) and (Sk) to Si
067ijk	Si Sj*1Sk	Fp Mult	2 minus the floating-point product of (Sj) and (Sk) to Si
070ij0	Si /HSj	Fp Recp	Floating-point reciprocal approximation of (Sj) to Si
071i0k	Si Ak	N/A	Transmit (Ak) to Si with no sign extension
071i1k	Si + Ak	N/A	Transmit (Ak) to Si with sign extension
071i2k	Si *Fak	N/A	Transmit (Ak) to Si as unnormalized floating-point number (exponent equals 40060)
071i30	Si 0.6	N/A	Transmit constant 0.75*2**48 to Si (Si = 040060 140000 000000 000000)
071i40	Si 0.4	N/A	Transmit constant 0.5 to Si (Si = 040000 100000 000000 000000)
071i50	Si 1.0	N/A	Transmit constant 1.0 to Si (Si = 040001 100000 000000 000000)
071i60	Si 2.0	N/A	Transmit constant 2.0 to Si Si = 040002 100000 000000 000000)
071i70	Si 4.0	N/A	Transmit constant 4.0 to Si Si = 040003 100000 000000 000000)
072i00	Si RT	N/A	Transmit (RTC) to Si
072i02	Si SM	N/A	Transmit (SM) to Si
072ij3	Si STj	N/A	Transmit (STj) to Si
073i00	Si VM	N/A	Transmit (VM) to Si

CRAY X-MP	CAL	Unit	Description
073i01	Si SRj	N/A	Transmit (SRj) to Si (J = 0)
073i11	¶¶	N/A	Read performance counter to Si
073i21	¶¶	N/A	Increment performance counter
073i31	¶¶	SECDED	Clear all maintenance modes; must be in MM
073i02	SM Si	N/A	Transmit (Si) to SM
073ij3	STj Si	N/A	Transmit (Si) to STj
074ijk	Si Tjk	N/A	Transmit (Tjk) to Si
075ijk	Tjk Si	N/A	Transmit (Si) to Tjk
076ijk	Si Vj,Ak	N/A	Transmit (Vj, element (Ak)) to Si
077ijk	Vi,Ak Sj	N/A	Transmit (Sj) to Vi element (Ak)
†077i0k	Vi,Ak 0	N/A	Clear Vi element (Ak)
10hijkm	Ai *exp*,Ah	Memory	Read from memory address ((Ah) + (jkm) + (DBA)) to Ai (Ah ≠ 0)
†100ijkm	Ai *exp*,0	Memory	Read from memory address ((jkm) + (DBA)) to Ai (Ah = 0)
†100ijkm	Ai *exp*,	Memory	Read from memory address ((jkm) + (DBA)) to Ai (Ah = 0)
†10hi000	Ai ,Ah	Memory	Read from memory ((Ah) + (DBA)) to Ai (Ah ≠ 0)
11hijkm	*exp*,Ah Ai	Memory	Write (Ai) to memory address ((Ah) + (jkm) + (DBA)) (Ah ≠ 0)
†110ijkm	*exp*,0 Ai	Memory	Write (Ai) to memory address ((jkm) + (DBA)) (Ah = 0)
†110ijkm	*exp*, Ai	Memory	Write (Ai) to memory address ((jkm) + (DBA)) (Ah = 0)
†11hi000	,Ah Ai	Memory	Write (Ai) to memory address ((Ah) + (DBA)) (Ah ≠ 0)
12hijkm	Si *exp*,Ah	Memory	Read from memory address ((Ah) + (jkm) + (DBA)) to Si (Ah ≠ 0)
†120ijkm	Si *exp*,0	Memory	Read from memory address ((jkm) + (DBA)) to Si (Ah = 0)
†120ijkm	Si *exp*,	Memory	Read from memory address ((jkm) + (DBA)) to Si (Ah = 0)
†12hi000	Si ,Ah	Memory	Read from memory address ((Ah) + (DBA)) to Si (Ah ≠ 0)
13hijkm	*exp*,Ah Si	Memory	Write (Si) to memory address ((Ah) + (jkm) + (DBA)) (Ah ≠ 0)
†130ijkm	*exp*,0 Si	Memory	Write (Si) to memory address ((jkm) + (DBA)) (Ah = 0)
†130ijkm	*exp*, Si	Memory	Write (Si) to memory address ((jkm) + (DBA)) (Ah = 0)
13hi000	,Ah Si	Memory	Write (Si) to memory address ((Ah) + (DBA)) (Ah ≠ 0)
140ijk	Vi Sj&Vk	V Logical	Logical products of (Sj) and (Vk) to Vi
†140i0k	Vi 0	V Logical	Clear Vi
141ijk	Vi Vj&Vk	V Logical	Logical products of (Vj) and (Vk) to Vi
142ijk	Vi Sj!Vk	V Logical	Logical sums of (Sj) and (Vk) to Vi
†142i0k	Vi Vk	V Logical	Transmit (Vk) to Vi
143ijk	Vi Vj!Vk	V Logical	Logical sums of (Vj) and (Vk) to Vi
144ijk	Vi Sj/Vk	V Logical	Logical differences of (Sj) and (Vk) to Vi
†144i0k	Vi Vk	V Logical	Transmit (Vk) to Vi
145ijk	Vi Vj/Vk	V Logical	Logical differences of (Vj) and (Vk) to Vi
†145iii	Vi 0	V Logical	Clear Vi
146ijk	Vi Sj!Vk&VM	V Logical	Transmit (Sj) if VM bit = 1; (Vk) if VM bit = 0 to Vi
†146i0k	Vi #VM&Vk	V Logical	Vector merge of (Vk) and 0 to Vi
147ijk	Vi Vj!Vk&VM	V Logical	Transmit (Vj) if VM bit = 1; (Vk) if VM bit = 0 to Vi
150ijk	Vi Vj< Ak	V Shift	Shift (Vj) left (Ak) places to Vi
†150ij0	Vi Vj< 1	V Shift	Shift (Vj) left one place to Vi
151ijk	Vi Vj> Ak	V Shift	Shift (Vj) right (Ak) places to Vi
†151ij0	Vi Vj> 1	V Shift	Shift (Vj) right one place to Vi
152ijk	Vi Vj,Vj< Ak	V Shift	Double shift (Vj) left (Ak) places to Vi
†152ij0	Vi Vj,Vj< 1	V Shift	Double shift (Vj) left one place to Vi
153ijk	Vi Vj,Vj> Ak	V Shift	Double shift (Vj) right (Ak) places to Vi
†153ij0	Vi Vj,Vj> 1	V Shift	Double shift (Vj) right one place to Vi
154ijk	Vi Sj + Vk	V Int Add	Integer sums of (Sj) and (Vk) to Vi
†154i0k	Vi Vk	V Int Add	Transmit (Vk) to Vi
155ijk	Vi Vj + Vk	V Int Add	Integer sums of (Vj) and (Vk) to Vi
156ijk	Vi Sj-Vk	V Int Add	Integer differences of (Sj) and (Vk) to Vi

CRAY X-MP	CAL	Unit	Description
†156i0k	Vi -Vk	V Int Add	Transmit negative of (Vk) to Vi
157ijk	Vi Vj-Vk	V Int Add	Integer differences of (Vj) and (Vk) to Vi
160ijk	Vi Sj*FVk	Fp Mult	Floating-point products of (Sj) and (Vk) to Vi
†160i0k	Vi 0	Fp Mult	Clear Vi
161ijk	Vi Vj*FVk	Fp Mult	Floating-point products of (Vj) and (Vk) to Vi
162ijk	Vi Sj*HVk	Fp Mult	Half-precision rounded floating-point products of (Sj) and (Vk) to Vi
163ijk	Vi VJ*HVk	Fp Mult	Half-precision rounded floating-point products of (Vj) and (Vk) to Vi
164ijk	Vi Sj*RVk	Fp Mult	Rounded floating-point products of (Sj) and (Vk) to Vi
165ijk	Vi Vj*RVk	Fp Mult	Rounded floating-point products of (Vj) and (Vk) to Vi
166ijk	Vi Sj*1Vk	Fp Mult	2 minus the floating-point products of (Sj) and (Vk) to Vi
167ijk	Vi Vj*1Vk	Fp Mult	2 minus the floating-point products of (Vj) and (Vk) to Vi
170ijk	Vi Sj + FVk	Fp Add	Floating-point sums of (Sj) and (Vk) to Vi
†170i0k	Vi + FVk	Fp Add	Normalize (Vk) to Vi
171ijk	Vi Vj + FVk	Fp Add	Floating-point sums of (Vj) and (Vk) to Vi
172ijk	Vi Sj-FVk	Fp Add	Floating-point differences of (Sj) and (Vk) to Vi
†172i0k	Vi -FVk	Fp Add	Transmit normalized negatives of (Vk) to Vi
173ijk	Vi Vj-FVk	Fp Add	Floating-point differences of (Vj) and (Vk) to Vi
174ij0	Vi /HVj	Fp Recp	Floating-point reciprocal approximations of (Vj) to Vi
174ij1	Vi PVj	V Pop	Population counts of (Vj) to Vi
174ij2	Vi QVj	V Pop	Population count parities of (Vj) to Vi
1750j0	VM Vj,Z	V Logical	VM = 1, where (Vj) = 0
1750j1	VM Vj,N	V Logical	VM = 1, where (Vj) ≠ 0
1750j2	VM Vj,P	V Logical	VM = 1 if (VJ) positive; 0 is positive
1750j3	VM Vj,M	V Logical	VM = 1 if (VJ) negative; 1 is negative
175ij4	¶Vi, VM Vj,Z	V Logical	VM = 1, where (Vj) = 0 and element index is loaded into (compressed Vi)
175ij5	¶Vi, VM Vj,N	V Logical	VM = 1, where (Vj) ≠ 0 and element index is loaded into (compressed Vi)
175ij6	¶Vi, VM Vj,P	V Logical	VM = 1 if (Vj) positive and element index is loaded into (compressed Vi) 0 is positive
175ij7	¶Vi, VM Vj,M	V Logical	VM = 1 if (Vj) negative and element index is loaded into (compressed Vi)
176i0k	Vi ,A0,Ak	Memory	Read (VL) words to Vi from memory address ((A0) + (DBA)) incremented by (Ak)
†176i00	Vi ,A0,1	Memory	Read (VL) words to Vi from memory address ((A0) + (DBA)) incremented by 1
176i1k	¶Vi ,A0,Vk	Memory	Read (VL) words to Vi from memory address ((A0) + (Vk) + (DBA)) (gather)
1770jk	,A0,Ak Vj	Memory	Write (VL) words from Vj to memory address ((A0) + (DBA)) incremented by (Ak)
†1770j0	,A0, 1 Vj	Memory	Write (VL) words from Vj to memory address ((A0) + (DBA)) incremented by 1
1771jk	¶,A0, Vk Vj	Memory	Write (VL) words from Vj to memory address ((AO) + (Vk) + (DBA)) (scatter)

† Special syntax form
†† Privileged to monitor mode
††† Generated depending on the value of *exp*
¶ Supported by CAL version 1 with macro or opdef
¶¶ Not supported by CAL version 1
( ) Read as the contents of. . .

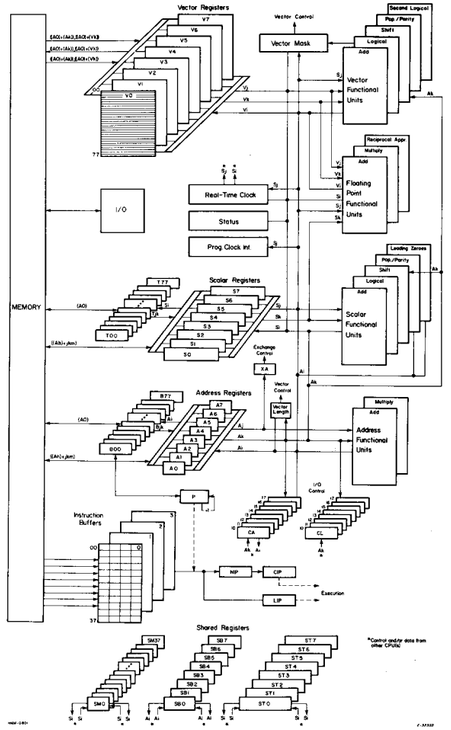
Register	Value
Ah, h = 0	0
Ai, i = 0	(A0)
Aj, j = 0	0
Ak, k = 0	1
Si, i = 0	(S0)
Sj, j = 0	0
Sk, k = 0	2^{63}

Logical Operators	
& AND	0101 1100 0100
! OR	0101 1100 1101
\ XOR	0101 1100 1001

CRAY RESEARCH, INC.

CRAY X-MP CAL VERSION 1 HARDWARE REFERENCE CARD

CRAY X-MP BLOCK DIAGRAM



GATE ARRAY PINOUTS

(A) 5/6, 8/7 = 13 14 15 16 + 9 10 11 + 1 2 3

(B) 10/9, 7/8, 6/5 = 11 13 + 14 15 + 16 1 + 2 3

(C) 7/8 = 5 6 + 9 10 + 11 13 + 14 15 + 16 1 + 2 3

(D) 7/8 = 2 6 5 1 16 9 13 14 11 15 10 3
Marco P6 = C5 C4 C3 C2 C1 C0 E5 E4 E3 E2 E1 G

(E) 5/6, 8/7, 9/10, 14/13, 15/16, 1/2 = 3 11

(F) 10 9 8 7 1 16 15 14 = DCD(2 3 5)/13 11 6 = (13 11)'

(G) 10/9, 7/8, 6/5 = SUM(11 13 14, –) /3 2 1
16/15 = 11 13 14

(H) 8/7 = 6 5 3 2 1 16
9/10 = 11 13 14 15

(I) 10/9 = 11 13 14
6/5 = 3 2 1
8/7 = 15 16

(J) 8/7 = 6 5
9/10 = 11 5
16/15 = 14 13
1/2 = 3 13

(K) 9/10 = 13 14 11 + 15 16 3
7/8 = 1 2 11 + 5 6 3

(L) 8/7 = 6 5;13
9/10 = 11 5;13
16/15 = 14 5;13
1/2 = 3 5;13

(M) 6/5, 7/8, 10/9 = 15 16 1 + 11 14 + 2 3;13

(N) 10/9 = 3 15 16 5 8 11 1 6 7 14 2;13
Marco P5 = C4 C3 C2 C1 C0 E4 E3 E2 E1 F G

(O) 15/16 = MUX (14 1):DCD(5);13
10/9 = MUX (11 8):DCD(5);13
7/6 = MUX (2 3):DCD(5);13

(P) 2/3 = 15 10 + 16 11 + 1 14;13
5/6 = 7 10 + 8 11 + 9 14;13

(Q) 7/8 = SUM (14, 15, 16): (3, 2, 1)
5/6 = CARY (14, 15, 16): (3, 2, 1)
10/9 = 11 13

(R) 14 = 12:DCD(6 7 9 10)/3 + 13
15 = 11:DCD(6 7 9 10)/3 + 13
1 = 4:DCD(6 7 9 10)/3 + 13
2 = 5:DCD(6 7 9 10)/3 + 13

(S) 1 = 17:DCD(14 13 12 11 10 8 7 6 5 4 3 2)/16 + 15

(T) 2/1 = SUM (8, 7, 3): (5, 14, 6);13
16/15 = CARY (8, 7, 3): (5, 14, 6);13
9/10 = 11;13

(U) 7/8 = MUX (1 16 13 5):DCD (3 2)/6
10/9 = MUX (1 16 13 5):DCD (14 15)/11

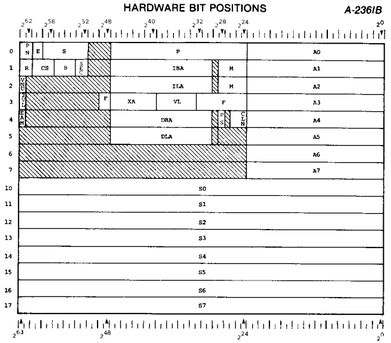
(W) 5/6, 8/7, 9/10, 14/13, 15/16, 1/2 = 3 11

(Y) 7/8, 6/5 = 10 9 + 14 15 + 16 1 + 2 3 + 11;13

(8) 3 = 1
2 = 16
11 = 14
10 = 13
7 = 9
6 = 8

(15) 6 = 7*8*9*10*11*13*15*16*14*1*2*3*

EXCHANGE PACKAGE



Register		Bits used with: 1 CPU	2 CPUs	4 CPUs
CSB	Read address for error, (CS)	61-59	61-57	61-58
	(B)	56-54	56-54	57-54
	(SEC)	53-52	53-52	53-52
IBA	Instruction Base Address	47-29	47-29	47-30
ILA	Instruction Limit Address	47-29	47-29	47-30
DBA	Data Base Address	47-29	47-29	47-30
CLN	Cluster Number	25-24	25-24	26-24
S	Syndrome bits	59-52	59-52	59-52
P	Program address	47-24	47-24	47-24
XA	Exchange address	47-40	47-40	47-40
VL	Vector Length	39-33	39-33	39-33
PS	Program State	28	28	28

PN Processor Number (If 2 bits in master select = 00)

00	Executed in CPU 0	10	Executed in CPU 2
01	Executed in CPU 1	11	Executed in CPU 3

E Error Type

10	Uncorrectable memory error
01	Correctable memory error

R Read Mode

00	I/O
01	Scalar reference 100-137
10	Vector, B, or T
11	Instruction fetch or exchange

VNU Vector Not Used

1 Instructions 076, 077, or 140-177 have not been used

ESVL Enable Second Vector Logical

1 Instructions 140-145 can select the Second Vector Logical unit

EAM Enhanced Addressing Mode

1 Instructions 100-137 will sign extend

M Mode Register

Word	Bit	Description
n+1	28	WS Waiting on Semaphore
n+1	27	FPS Floating-point Error Status
n+1	26	BDM Bidirectional Memory access
n+1	25	SEI Select for External Interrupt†
n+1	24	IMM Interrupt Monitor Mode
n+2	28	IOR Interrupt on Operand Range error
n+2	27	ICM Interrupt on Correctable Memory error
n+2	26	IFP Interrupt on Floating-point error
n+2	25	IUM Interrupt on Uncorrectable Memory error
n+2	24	MM Monitor Mode

F Flag Register (Word N+3)

Word	Bits	Description
n+3	49	ICP Interrupt from Internal CPU†
n+3	48	DL Deadlock†
n+3	32	PCI Programmable Clock Interrupt
n+3	31	MCU MCU Interrupt (MIOP)
n+3	30	FPE Floating-point Error
n+3	29	ORE Operand Range Error
n+3	28	PRE Program Range Error
n+3	27	ME Memory Error
n+3	26	IOI I/O Interrupt
n+3	25	EEX Error Exit (000)
n+3	24	NEX Normal Exit (004)
n+4	28	PS Program State
n+4	26-24	CLN Cluster Number register†

† Bits not used on single processors

FUNCTIONAL UNITS

Functional Unit	Unit Time (Clock Periods)	Instructions
Address Integer Add	1	030, 031
Address Integer Multiply	3	032
Scalar Integer Add	2	060, 061
Scalar Logical	< 1	042 - 051
Scalar Shift Single	1	052 - 055
Scalar Shift Double	2	056, 057
Scalar Pop/Parity/	3	026
Leading Zero	2	027
Vector Integer Add	2	154 - 157
Full Vector Logical	1	140 - 147, 175
Second Vector Logical	3	140 - 145
Vector Shift	2	150, 151, 153
Vector Shift Double Left	3	152
Vector Pop/Parity	4	174ij1, 174ij2
Floating-point Add	5	062, 063, 170 - 173
Floating-point Multiply	6	064 - 067, 160 - 167
Floating-point Reciprocal	13	070, 174ij0
Memory (Scalar)		
Two and four processor	13	100 - 130
Single processor	16	100 - 130

CRAY X-MP LOSP CHANNELS

LOSP	Cables	
10	LOSP 0 Input	} MCU
11	LOSP 0 Output control	} MCU
12	LOSP 1 Input	
13	LOSP 1 Output	
14	LOSP 2 Input	
15	LOSP 2 Output	
16	LOSP 3 Input	
17	LOSP 3 Output	

LOSP	1 CPU	2 CPUs	4 CPUs
Channels available	10 - 17 or 10 - 13	10 - 17	10 - 17

HISP 1 and 2 CPUs	Cables	
HISP A pair	HISP 0 In A1 Data	HISP 0 Out B1 Data
	In A2 Data	Out B2 Data
	In A3 Data	Out B3 Data
	In A4 Control	Out B4 Control
	In A5 Control	
HISP B pair	HISP 2 In E1 Data	HISP 2 Out F1 Data
	In E2 Data	Out F2 Data
	In E3 Data	Out F3 Data
	In E4 Control	Out F4 Control
	In E5 Control	

HISP 4 CPUs	Cables	
HISP C pair	HISP 4 In I1 Data	HISP 4 Out J1 Data
	In I2 Data	Out J2 Data
	In I3 Data	Out J3 Data
	In I4 Control	Out J4 Control
	In I5 Control	
HISP D pair	HISP 6 In M1 Data	HISP 6 Out N1 Data
	In M2 Data	Out N2 Data
	In M3 Data	Out N3 Data
	In M4 Control	Out N4 Control
	In M5 Control	

VARIOUS MCU BASIC COMMANDS

DS	Dead Start	:FI	Display All Files
MC	Master Clear	:LO	Load File Or (/)
DR	Display Right	:SA	Save File
DL	Display Left	:DE	Delete File
DX	Display Exchange Package	:RE	Stop Refresh
DT	Display Text	:SN	Snap Display
DF	Display Forward	:KI	Kill Mode
DB	Display Backward	:TD	Test Dead Mode
DE	Display Memory Error	:DD	Dead Dump Mode
S	Set Memory/Parcel	:TB	Test Basic Mode
S+	Set Memory/Parcel + 1	:RU	Run Mode

MONITORS

Sample MTA / M1 Scope Loop	MTA Parameters
/MTA P = 200-0 200 060123 7, 40-0 6, 200-0 S2 = 000000 000017 177777 177777 S3 = 000000 000000 000000 000001	DX 0 = Uses Exchange Package 0 40-0 = Subroutine to load A and S register into memory and output 4K to MCU 70-0 = Subroutine to clear A and S register 200-0 = Start user code All codes execute in monitor mode

Sample MTI Scope Loop	MTI Parameters
/MTI P = 200-0 200 060123 030770 6,200-0 S2 = 000000 000017 177777 177777 S3 = 000000 000000 000000 000001	DX 120 = Current Ex Pkg DX 140 = Initial Ex Pkg DX 160 = Active Ex Pkg Loc 27 = 1 Allows current Ex Pkg to become initial Ex Pkg during DS

Sample MTX Scope Loop	MTX Parameters
/MTX P = 1000-0 1000 060134 046012 014,1010-0 1001 030660 6,1000-0 1010 030770 6,1000-0 S2 = 000000 000020 000000 000000 S3 = 000000 000017 177777 177777 S4 = 000000 000000 000000 000001	Loc 60 = Monitor Type 0 = On-Line 1 = MTA P = 200-0 in DX 0 2 = MTI P = 300-0 in DX 0 Loc 70 = Memory size Loc 71 = Number of banks Loc 400 = Memory Error Table

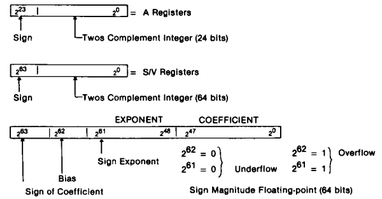
Sample AMP2 Scope Loop	AMP2 Parameters
/AMP2 P = 200-0 200 060123 030770 6,200-0 S2 = 000000 000017 177777 177777 S3 = 000000 000000 000000 000001 Running in CPU 0 and CPU 1	DX 100 = Test Ex Pkg CPU 0 DX 120 = Test Ex Pkg CPU 1 DX 140 = Initial Ex Pkg CPU 0 DX 160 = Initial Ex Pkg CPU 1

SAMPLE AMP4 Scope Loop	AMP4 Parameters
/AMP4 /MTX 1000 P = 1000-0 IBA = 1000 2000 060123 030770 6,1000-0 S2 = 000000 000017 177777 177777 S3 = 000000 000000 000000 000001 Running in all CPUs	DX 220, 320, 420, 520 CPU 0-3 Test Ex Pkg DX 240, 340, 440, 540, CPU 0-3 initial Ex Pkg

CRAY RESEARCH, INC. *Technical Operations Building* 890 Industrial Blvd., C.F., WI 54729

Run Monitor	Diagnostic Codes
1. :KI 2. :RU :Run (varies with system) 3. DT 7600 4. PROCA (for 4 CPUs) or PROC 1 (for 2 CPUs) 5. CPU 0 - Select CPU (0-3) 6. TL1, TL2, and TL3 7. Start 8. Repeat step 5 - 7 for all CPUs 9. Drop or Drop 'n' (unloads 'n' or all from TL1, TL2, and TL3) 10. Stop or Stop 'n' (stops execution of 'n' or all from TL1, TL2, and TL3) 11. TLA 3SR3 (or diagnostic name) loads all cells with that diagnostic (some tests are too long for cell 17 and will fall out) Note: $ Precedes all commands on Pitts	1 = MTA 2 = Amp2 or 2 CPU version 3 = MTI Monitor 4 = Amp4 or 4 CPU version 5 = MTX or "Special" Monitor / = Command Buffer :GO / 1-5 ARA 2/4 MEB 1-16 1 = 1 Mwords 2 = 2 Mwords 4 = 4 Mwords 8 = 8 Mwords 16 = 16 Mwords Processor Number

DATA FORMATS



EXPONENT RANGE

-20000	Negative	0	Positive	+17777
20000	Range	40000	Range	57777

FLOATING-POINT RANGE ERRORS

Floating Add or Floating Multiply

Underflow =	0	0– – – 0	0– – – – – – – 0
	Sign	Exponent	Coefficient, no flag
Overflow =	0	60000	Calculated
	Sign	Exponent	Coefficient, flag set

Floating-Point Reciprocal

Underflow =	0	60000	Calculated
	Sign	Exponent	Coefficient, 2^{47} = 0 and flag set
Overflow =	0	60000	Calculated
	Sign	Exponent	Coefficient, 2^{47} = 0 and flag set

INSTRUCTIONS

CRAY X-MP	CAL	Unit	Description
000000	ERR	N/A	Error exit
††0010jk	CA,Aj Ak	N/A	Set channel (Aj) CA register to (Ak) and begin I/O sequence
††0011jk	CI,Aj Ak	N/A	Set channel (Aj) CL register to (Ak)
††0012j0	CI,Aj	N/A	Clear channel (Aj) Interrupt and Error flags; clear device Master Clear (output channel)
††0012j1	MC,Aj	N/A	Clear channel (Aj) Interrupt and Error flags, set device Master Clear (output channel); clear device ready-held (input channel)
††0013j0	XA Aj	N/A	Enter XA register with (Aj)
††0014j0	RT Sj	N/A	Enter RTC register with (Sj)
††0014j1	¶SIPI *exp*	N/A	Set interprocessor interrupt request of CPU *exp*; $0 \le exp \le 5$ (0 = no-op)
††001402	¶CIPI	N/A	Clear interprocessor interrupt
††0014j3	CLN *exp*	N/A	Select CLN register *exp*; $0 \le exp \le 5$ (0 = no-op)
††0014j4	PCI Sj	N/A	Enter II register with (Sj)
††001405	CCI	N/A	Clear programmable clock interrupt (PCI) request
††001406	ECI	N/A	Enable PCI request
††001407	DCI	N/A	Disable PCI request
††0015j0	¶¶	N/A	Select performance monitor
††001501	¶¶	SECDED	Set maintenance read mode; must be in maintenance mode (MM)
††001511	¶¶	SECDED	Load diagnostic check byte with S1; must be in MM
††001521	¶¶	SECDED	Set maintenance write mode 1; must be in MM
††001531	¶¶	SECDED	Set maintenance write mode 2; must be in MM
00200k	VL Ak	N/A	Transmit (Ak) to VL register
††002000	VL 1	N/A	Transmit 1 to VL register
002100	EFI	N/A	Enable interrupt on floating-point error
002200	DFI	N/A	Disable interrupt on floating-point error
002300	ERI	N/A	Enable operand range error interrupts
002400	DRI	N/A	Disable operand range error interrupts
002500	DBM	N/A	Disable bidirectional memory transfers
002600	EBM	N/A	Enable bidirectional memory transfers
002700	CMR	N/A	Complete memory references
0030j0	VM Sj	N/A	Transmit (S_j) to VM register
††003000	VM 0	N/A	Clear VM register
0034jk	SMjk 1,TS	N/A	Test & set semaphore jk $0 \le jk \le 37_8$
0036jk	SMjk 0	N/A	Clear semaphore jk $0 \le jk \le 37_8$
0037jk	SMjk 1	N/A	Set semaphore jk $0 \le jk \le 37_8$
004000	EX	N/A	Normal exit
0050jk	J Bjk	N/A	Jump to (Bjk)
006ijkm	J *exp*	N/A	Jump to *exp*
007ijkm	R *exp*	N/A	Return jump to *exp*; set B00 to (P) + 2
010ijkm	JAZ *exp*	N/A	Branch to *exp* if (A0) = 0 (bit 2^2 of i = 0)
011ijkm	JAN *exp*	N/A	Branch to *exp* if (A0) ≠ 0 (bit 2^2 of i = 0)
012ijkm	JAP *exp*	N/A	Branch to *exp* if (A0) positive; 0 is positive (bit 2^2 of i = 0)
013ijkm	JAM *exp*	N/A	Branch to *exp* if (A0) negative (bit 2^2 of i = 0)
014ijkm	JSZ *exp*	N/A	Branch to *exp* if (S0) = 0 (bit 2^2 of i = 0)
015ijkm	JSN *exp*	N/A	Branch to *exp* if (S0) ≠ 0 (bit 2^2 of i = 0)
016ijkm	JSP *exp*	N/A	Branch to *exp* if (S0) positive; 0 is positive (bit 2^2 of i = 0)
017ijkm	JSM *exp*	N/A	Branch to *exp* if (S0) negative (bit 2^2 of i = 0)
010ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A0 (bit 2^2 of i = 1)
011ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A1 (bit 2^2 of i = 1)
012ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A2 (bit 2^2 of i = 1)

CRAY X-MP	CAL	Unit	Description
013ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A3 (bit 2^2 of i = 1)
014ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A4 (bit 2^2 of i = 1)
015ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A5 (bit 2^2 of i = 1)
016ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A6 (bit 2^2 of i = 1)
017ijkm	¶Ah *exp*	N/A	Transmit *exp* = ijkm to A7 (bit 2^2 of i = 1
†††020ijkm	Ai *exp*	N/A	Transmit *exp* = jkm to Ai
†††021ijkm	Ai *exp*	N/A	Transmit *exp* = ones complement of jkm) to Ai
†††022ijk	Ai *exp*	N/A	Transmit *exp* = jk to Ai
023ij0	Ai Sj	N/A	Transmit = (Sj) to Ai
023i01	Ai VL	N/A	Transmit (VL) to Ai
024ijk	Ai Bjk	N/A	Transmit (Bjk) to Ai
025ijk	Bjk Ai	N/A	Transmit (Ai) to Bjk
026ij0	Ai PSj	S Pop	Population count of (Sj) to Ai
026ij1	Ai QSj	S Pop	Population count parity of (Sj) to Ai
026ij7	Ai SBj	N/A	Transmit (SBj) to Ai
027ij0	Ai ZSj	S/LZ	Leading zero count of (Sj) to Ai
027ij7	SBj Ai	N/A	Transmit (Ai) to SBj
030ijk	Ai Aj + Ak	A Int Add	Integer sum of (Aj) and (Ak) to Ai
†030i00	Ai 1	A Int Add	Transmit 1 to Ai
†030i0k	Ai Ak	A Int Add	Transmit (Ak) to Ai
†030ij0	Ai Aj + 1	A Int Add	Integer sum of (Aj) and 1 to Ai
031ijk	Ai Aj-Ak	A Int Add	Integer difference of (Aj) less (Ak) to Ai
† †††031i00	Ai -1	A Int Add	Transmit -1 to Ai; (Ai = 77777777)
†031i0k	Ai -Ak	A Int Add	Transmit the negative of (Ak) to Ai
†031ij0	Ai Aj-1	A Int Add	Integer difference of (Aj) less 1 to Ai
032ijk	Ai Aj*Ak	A Int Mult	Integer product of (Aj) and (Ak) to Ai
†032ij0	Ai Aj*1	A Int Mult	Transmit (Aj) to Ai
†032i0k	Ai Ak*0	A Int Mult	Zero fill Ai; (Ai = 00000000)
033i00	Ai CI	N/A	Channel number to Ai (j = 0)
033ij0	Ai CA,Aj	N/A	Address of channel (Aj) to Ai (j ≠ 0; k = 0)
033ij1	Ai CE,Aj	N/A	Error flag of channel (Aj) to Ai (j ≠ 0; k = 1)
034ijk	Bjk,Ai ,A0	Memory	Read (Ai) words to B register jk from memory address ((A0) + (DBA))
†034ijk	Bjk,Ai 0,A0	Memory	Read (Ai) words to B register jk from memory address ((A0) + (DBA))
035ijk	,A0 Bjk,Ai	Memory	Write (Ai) words at B register jk into memory address ((A0) + (DBA))
†035ijk	0,A0 Bjk,A1	Memory	Write (Ai) words at B register jk into memory address ((A0) + (DBA))
036ijk	Tjk,Ai ,A0	Memory	Read (Ai) words to T register jk from memory Address ((A0) + (DBA))
†036ijk	Tjk,Ai 0,A0	Memory	Read (Ai) words to T register jk from memory address ((A0) + (DBA))
037ijk	,A0 Tjk,Ai	Memory	Write (Ai) words at T register jk into memory address ((A0) + (DBA))
†037ijk	0,A0 Tjk,A1	Memory	Write (Ai) words at T register jk into memory address ((A0) + (DBA))
040ijkm	Si *exp*	N/A	Transmit *exp* = jkm to Si
041ijkm	Si *exp*	N/A	Transmit *exp* = ones complement of jkm to Si
042ijk	Si *< exp*	S Logical	Form zeros mask *exp* bits in Si from the left; jk field gets *exp*
†042ijk	Si *#> exp*	S Logical	Form zeros mask *exp* bits in Si from the left; jk field get *exp*
†042ij77	Si -1	S Logical	Enter 1 into Si
†042i00	Si -1	S Logical	Enter -1 into Si; (Si = 777777 177777 177777 177777)
043ijk	Si *> exp*	S Logical	Form ones mask *exp* bits in Si from the left; jk field gets *exp*
†043ijk	Si *#< exp*	S Logical	Form ones mask *exp* bits in Si from the left; jk field gets *exp*
†043i00	Si 0	S Logical	Clear Si
044ijk	Si Sj&Sk	S Logical	Logical product of (Sj) and (Sk) to Si
†044ij0	Si Sj&SB	S Logical	Sign bit of (Sj) to Si
†044ij0	Si SB&Sj	S Logical	Sign bit of (Sj) to Si (j ≠ 0)

CRAY X-MP	CAL	Unit	Description
045ijk	Si #Sk&Sj	S Logical	Logical product of (S_j) and ones complement of (Sk) to Si
†045ij0	Si #SB&Sj	S Logical	(S_j) with sign bit cleared to Si
046ijk	Si Sj\Sk	S Logical	Logical difference of (S_j) and (Sk) to Si
†046ij0	Si Sj\SB	S Logical	Toggle sign bit of S_j, then enter into Si
†046ij0	Si SB\Sj	S Logical	Toggle sign bit of S_j, then enter into Si (j ≠ 0)
047ijk	Si #Sj\Sk	S Logical	Logical equivalence of (Sk) and (S_j) to Si
†047i0k	Si #Sk	S Logical	Transmit ones complement of (Sk) to Si
†047ij0	Si #Sj\SB	S Logical	Logical equivalence of (S_j) and sign bit to Si
†047ij0	Si #SB\Sj	S Logical	Logical equivalence of (S_j) and sign bit to Si (j ≠ 0)
†047i00	Si #SB	S Logical	Enter ones complement of sign bit into Si
050ijk	Si Sj\Si&Sk	S Logical	Logical product of (S_j) and (Sk) complement QRed with logical product of (S_j) and (Sk) to Si
†050ij0	Si Sj\Si&SB	S Logical	Scalar merge of (S_j) and sign bit of (S_j) to Si
051ijk	Si Sj!Sk	S Logical	Logical sum of (S_j) and (Sk) to Si
†051i0k	Si Sk	S Logical	Transmit (Sk) to Si
†051ij0	Si Sj!SB	S Logical	Logical sum of (S_j) and sign bit to Si
†051ij0	Si SB!Sj	S Logical	Logical sum of (S_j) and sign bit to Si (j ≠ 0)
†051i00	Si SB	S Logical	Enter sign bit into Si
052ijk	S0 Si< *exp*	S Shift	Shift (Si) left *exp* = jk places to S0
053ijk	S0 Si> *exp*	S Shift	Shift (Si) right *exp* = 100_8 − jk places to S0
054ijk	Si Si< *exp*	S Shift	Shift (Si) left *exp* = jk places to Si
055ijk	Si Si> *exp*	S Shift	Shift (Si) right *exp* = 100_8 − jk places to Si
056ijk	Si Si,Sj< Ak	S Shift	Shift (Si) and (Sj) left (Ak) places to Si
†056ij0	Si Si,Sj< 1	S Shift	Shift (Si) and (Sj) left one place to Si
†056i0k	Si Si< Ak	S Shift	Shift (Si) left (Ak) places to Si
057ijk	Si Sj,Si> Ak	S Shift	Shift (Sj) and (Si) right (Ak) places to Si
†057ij0	Si Sj,Si> 1	S Shift	Shift (S_j and Si) right one place to Si
†057i0k	Si Si> Ak	S Shift	Shift (Si) fight (Ak) places to Si
060ijk	Si Sj + Sk	S Int Add	Integer sum of (Sj) and (Sk) to Si
†060i0k	Si Sk	S Int Add	Transmit (Sk) to Si
†060ij0	Si Sj + S0	S Int Add	Integer sum (2^{63}) and (Sj) to Si
061ijk	Si Sj-Sk	S Int Add	Integer difference of (Sj) and (Sk) to Si
†061i0k	Si -Sk	S Int Add	Transmit negative of (Sk) to Si
†061ij0	Si Sj-S0	S Int Add	Integer difference of (Sj) and 2^{63} to Si
062ijk	Si Sj + FSk	Fp Add	Floating-point sum of (Sj) and (Sk) to Si
†062i0k	Si + FSk	Fp Add	Normalize (Sk) to Si
063ijk	Si Sj-FSk	Fp Add	Floating-point difference of (Sj) and (Sk) to Si
†063i0k	Si -FSk	Fp Add	Transmit normalized negative of (Sk) to Si
064ijk	Si Sj*FSk	Fp Mult	Floating-point product of (Sj) and (Sk) to Si
065ijk	Si Sj*HSk	Fp Mult	Half-precision rounded floating-point product of (Sj) and (Sk) to Si
066ijk	Si Sj*RSk	Fp Mult	Full-precision rounded floating-point product of (Sj) and (Sk) to Si
067ijk	Si Sj*ISk	Fp Mult	2 minus the floating-point product of (Sj) and (Sk) to Si
070ij0	Si /HSj	Fp Recp	Floating-point reciprocal approximation of (Sj) to Si
071i0k	Si Ak	N/A	Transmit (Ak) to Si with no sign extension
071i1k	Si + Ak	N/A	Transmit (Ak) to Si with sign extension
071i2k	Si *Fak	N/A	Transmit (Ak) to Si as unnormalized floating-point number (exponent equals 40060)
071i30	Si 0.6	N/A	Transmit constant 0.75*2**48 to Si (Si = 040060 140000 000000 000000)
071i40	Si 0.4	N/A	Transmit constant 0.5 to Si (Si = 040000 100000 000000 000000)
071i50	Si 1.0	N/A	Transmit constant 1.0 to Si (Si = 040001 100000 000000 000000)
071i60	Si 2.0	N/A	Transmit constant 2.0 to Si Si = 040002 100000 000000 000000)
071i70	Si 4.0	N/A	Transmit constant 4.0 to Si Si = 040003 100000 000000 000000)
072i00	Si RT	N/A	Transmit (RTC) to Si
072i02	Si SM	N/A	Transmit (SM) to Si
072ij3	Si STj	N/A	Transmit (STj) to Si
073i00	Si VM	N/A	Transmit (VM) to Si

CRAY X-MP	CAL	Unit	Description
073i01	Si SRj	N/A	Transmit (SRj) to Si (J = 0)
073ii1	¶¶	N/A	Read performance counter to Si
073i21	¶¶	N/A	Increment performance counter
073i31	¶¶ SECDED		Clear all maintenance modes; must be in MM
073i02	SM Si	N/A	Transmit (Si) to SM
073ij3	STj Si	N/A	Transmit (Si) to STj
074ijk	Si Tjk	N/A	Transmit (Tjk) to Si
075ijk	Tjk Si	N/A	Transmit (Si) to Tjk
076ijk	Si Vj,Ak	N/A	Transmit (Vj, element (Ak)) to Si
077ijk	Vi,Ak Sj	N/A	Transmit (Sj) to Vi element (Ak)
†077i0k	Vi,Ak 0	N/A	Clear Vi element (Ak)
10hijkm	Ai *exp*,Ah	Memory	Read from memory address ((Ah) + (jkm) + (DBA)) to Ai (Ah ≠ 0)
†100ijkm	Ai *exp*,0	Memory	Read from memory address ((jkm) + (DBA)) to Ai (Ah = 0)
†100ijkm	Ai *exp*,	Memory	Read from memory address ((jkm) + (DBA)) to Ai (Ah = 0)
†10hi000	Ai ,Ah	Memory	Read from memory ((Ah) + (DBA)) to Ai (Ah ≠ 0)
11hijkm	*exp*,Ah Ai	Memory	Write (Ai) to memory address ((Ah) + (jkm) + (DBA)) (Ah ≠ 0)
†110ijkm	*exp*,0 Ai	Memory	Write (Ai) to memory address ((jkm) + (DBA)) (Ah = 0)
†110ijkm	*exp*, Ai	Memory	Write (Ai) to memory address ((jkm) + (DBA)) (Ah = 0)
†11hi000	,Ah Ai	Memory	Write (Ai) to memory address ((Ah) + (DBA)) (Ah ≠ 0)
12hijkm	Si *exp*,Ah	Memory	Read from memory address ((Ah) + (jkm) + (DBA)) to Si (Ah ≠ 0)
†120ijkm	Si *exp*,0	Memory	Read from memory address ((jkm) + (DBA)) to Si (Ah = 0)
†120ijkm	Si *exp*,	Memory	Read from memory address ((jkm) + (DBA)) to Si (Ah = 0)
†12hi000	Si ,Ah	Memory	Read from memory address ((Ah) + (DBA)) to Si (Ah ≠ 0)
13hijkm	*exp*,Ah Si	Memory	Write (Si) to memory address ((Ah) + (jkm) + (DBA)) (Ah ≠ 0)
†130ijkm	*exp*,0 Si	Memory	Write (Si) to memory address ((jkm) + (DBA)) (Ah = 0)
†130ijkm	*exp*, Si	Memory	Write (Si) to memory address ((jkm) + (DBA)) (Ah = 0)
†13hi000	,Ah Si	Memory	Write (Si) to memory address ((Ah) + (DBA)) (Ah ≠ 0)
140ijk	Vi Sj&Vk	V Logical	Logical products of (S_j) and (Vk) to Vi
†140i0k	Vi 0	V Logical	Clear Vi
141ijk	Vi Vj&Vk	V Logical	Logical products of (Vj) and (Vk) to Vi
142ijk	Vi Sj!Vk	V Logical	Logical sums of (S_j) and (Vk) to Vi
†142i0k	Vi Vk	V Logical	Transmit (Vk) to Vi
143ijk	Vi Vj!Vk	V Logical	Logical sums of (Vj) and (Vk) to Vi
144ijk	Vi Sj\Vk	V Logical	Logical differences of (S_j) and (Vk) to Vi
†144i0k	Vi Vk	V Logical	Transmit (Vk) to Vi
145ijk	Vi Vj\Vk	V Logical	Logical differences of (Vj) and (Vk) to Vi
†145iii	Vi 0	V Logical	Clear Vi
146ijk	Vi Sj!Vk&VM	V Logical	Transmit (S_j) if VM bit = 1; (Vk) if VM bit = 0 to Vi
†146i0k	Vi #VM&Vk	V Logical	Vector merge of (Vk) and 0 to Vi
147ijk	Vi Vj!Vk&VM	V Logical	Transmit (Vj) if VM bit = 1; (Vk) if VM bit = 0 to Vi
150ijk	Vi Vj< Ak	V Shift	Shift (Vj) left (Ak) places to Vi
†150ij0	Vi Vj< 1	V Shift	Shift (Vj) left one place to Vi
151ijk	Vi Vj> Ak	V Shift	Shift (Vj) right (Ak) places to Vi
†151ij0	Vi Vj> 1	V Shift	Shift (Vj) right one place to Vi
152ijk	Vi Vj,Vj< Ak	V Shift	Double shift (Vj) left (Ak) places to Vi
†152ij0	Vi Vj,Vj< 1	V Shift	Double shift (Vj) left one place to Vi
153ijk	Vi Vj,Vj> Ak	V Shift	Double shift (Vj) right (Ak) places to Vi
†153ij0	Vi Vj,Vj> 1	V Shift	Double shift (Vj) right one place to Vi
154ijk	Vi Sj + Vk	V Int Add	Integer sums of (S_j) and (Vk) to Vi
†154i0k	Vi Vk	V Int Add	Transmit (Vk) to Vi
155ijk	Vi Vj + Vk	V Int Add	Integer sums of (Vj) and (Vk) to Vi
156ijk	Vi Sj-Vk	V Int Add	Integer differences of (S_j) and (Vk) to Vi

CRAY X-MP	CAL	Unit	Description
†156i0k	Vi -Vk	V Int Add	Transmit negative of (Vk) to Vi
157ijk	Vi Vj-Vk	V Int Add	Integer differences of (Vj) and (Vk) to Vi
160ijk	Vi Sj*FVk	Fp Mult	Floating-point products of (S_j) and (Vk) to Vi
†160i0k	Vi 0	Fp Mult	Clear Vi
161ijk	Vi Vj*FVk	Fp Mult	Floating-point products of (Vj) and (Vk) to Vi
162ijk	Vi Sj*HVk	Fp Mult	Half-precision rounded floating-point products of (S_j) and (Vk) to Vi
163ijk	Vi VJ*HVk	Fp Mult	Half-precision rounded floating-point products of (Vj) and (Vk) to Vi
164ijk	Vi Sj*RVk	Fp Mult	Rounded floating-point products of (S_j) and (Vk) to Vi
165ijk	Vi Vj*RVk	Fp Mult	Rounded floating-point products of (Vj) and (Vk) to Vi
166ijk	Vi Sj*IVk	Fp Mult	2 minus the floating-point products of (S_j) and (Vk) to Vi
167ijk	Vi Vj*IVk	Fp Mult	2 minus the floating-point products of (Vj) and (Vk) to Vi
170ijk	Vi Sj + FVk	Fp Add	Floating-point sums of (S_j) and (Vk) to Vi
†170i0k	Vi + FVk	Fp Add	Normalize (Vk) to Vi
171ijk	Vi Vj + FVk	Fp Add	Floating-point sums of (Vj) and (Vk) to Vi
172ijk	Vi Sj-FVk	Fp Add	Floating-point differences of (S_j) and (Vk) to Vi
†172i0k	Vi -FVk	Fp Add	Transmit normalized negatives of (Vk) to Vi
173ijk	Vi Vj-FVk	Fp Add	Floating-point differences of (Vj) and (Vk) to Vi
174ij0	Vi /HVj	Fp Recp	Floating-point reciprocal approximations of (Vj) to Vi
174ij1	Vi PVj	V Pop	Population counts of (Vj) to Vi
174ij2	Vi QVj	V Pop	Population count parities of (Vj) to Vi
1750j0	VM Vj,Z	V Logical	VM = 1, where (Vj) = 0
1750j1	VM Vj,N	V Logical	VM = 1, where (Vj) ≠ 0
1750j2	VM Vj,P	V Logical	VM = 1 if (VJ) positive; 0 is positive
1750j3	VM Vj,M	V Logical	VM = 1 if (VJ) negative; 1 is negative
175ij4	¶Vi, VM Vj,Z	V Logical	VM = 1, where (Vj) = 0 and element index is loaded into (compressed Vi)
175ij5	¶Vi, VM Vj,N	V Logical	VM = 1, where (Vj) ≠ 0 and element index is loaded into (compressed Vi)
175ij6	¶Vi, VM Vj,P	V Logical	VM = 1 if (Vj) positive and element index is loaded into (compressed Vi) 0 is positive
175ij7	¶Vi, VM Vj,M	V Logical	VM = 1 if (Vj) negative and element index is loaded into (compressed Vi)
176i0k	Vi ,A0,Ak	Memory	Read (VL) words to Vi from memory address ((A0) + (DBA)) incremented by (Ak)
†176i00	Vi ,A0,1	Memory	Read (VL) words to Vi from memory address ((A0) + (DBA)) incremented by 1
176i1k	¶Vi ,A0,Vk	Memory	Read (VL) words to Vi from memory address ((A0) + (Vk) + (DBA)) (gather)
1770jk	,A0,Ak Vj	Memory	Write (VL) words from Vj to memory address ((A0) + (DBA)) incremented by (Ak)
†1770j0	,A0, 1 Vj	Memory	Write (VL) words from Vj to memory address ((A0) + (DBA)) incremented by 1
1771jk	¶,A0, Vk Vj	Memory	Write (VL) words from Vj to memory address ((A0) + (Vk) + (DBA)) (scatter)

† Special syntax form
†† Privileged to monitor mode
††† Generated depending on the value of *exp*
¶ Supported by CAL version 1 with macro or opdef
¶¶ Not supported by CAL version 1
( ) Read as the contents of . . .

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

CRAY RESEARCH, INC.

CRAY X-MP CAL VERSION 1 HARDWARE REFERENCE CARD

CRAY X-MP BLOCK DIAGRAM