Instructions [E]

This appendix lists symbolic machine instructions for Cray PVP systems. The notes for Table 17 through Table 23 are as follows:

Notes	Meaning
C	Instruction is valid on CRAY C90 systems.
D	Difference in operation between CRAY T90 mode and CRAY C90 mode.
E	Generation depends on the value of *exp*.
F	Functionality of instruction is different on CRAY T90 systems with IEEE floating-point hardware.
I	Instruction is only valid on CRAY T90 systems with IEEE floating-point hardware.
J	Instruction is valid on CRAY J90 systems.
M	Privileged to monitor mode.
N	New instruction.
R	Revised instruction for IEEE floating-point format.
T	Instruction is valid on CRAY T90 systems.
X	Valid only on CRAY Y-MP systems running in X-mode (X-mode not available on CRAY C90 systems).
Y	Instruction is valid on CRAY Y-MP systems running in Y-mode.

Common instructions
F.13

The instructions listed in Table 17 are available on Cray PVP systems as specified in the notes column. If the notes column is empty, the instruction is valid on all Cray PVP systems.

Table 17. Common symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
000000		ERR		–	Error exit.
0010*jk*	M	CA,A*j*	A*k*	–	Sets Current Address (CA) register for channel indicated by (A*j*) to the value specified in (A*k*), activates channel.
001000		PASS		–	Pass instruction.
0011*jk*	M	CL,A*j*	A*k*	_	Sets Channel Limit (CL) register for channel specified by (A*j*) to address specified by (A*k*).
0012*j*0	M	CI,A*j*		_	Clears interrupt flag and error flag for channel specified by (A*j*).
0012*j*1	M	MC,A*j*		_	Clears interrupt and error flags for channel indicated by (A*j*). If (A*j*) represents an output channel, sets device master clear. If (A*j*) represents an input channel, clears device ready-held.
0013*j*0	M	XA	A*j*	_	Enters XA register with (A*j*).
0014*j*0	M	RT	S*j*	–	Loads RTC register with (S*j*).
0014*j*1	M	SIPI	A*j*	–	Sets interprocessor interrupt request to CPU (A*j*); 0≤ (A*j*)≤ 7 on CRAY Y-MP systems.
001401	M	SIPI		–	Sets interprocessor interrupt request of CPU 0.
001402	M	CIPI		–	Clears interprocessor interrupt.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
0014*j*3	M	CLN	A*j*	–	Loads Cluster Number (CLN) register with (A*j*); $0 \leq (Aj) < 9_{10}$ on CRAY Y-MP systems, $0 \leq (Aj) \leq 17_{10}$ on CRAY C90 systems.
0014*j*4	M	PCI	S*j*	–	Loads Interrupt Interval (II) register with (S*j*).
001405	M	CCI		–	Clears clock interrupt.
001406	M	ECI		–	Enables programmable clock interrupt.
001407	M	DCI		–	Disables programmable clock interrupt.
001500	M,C,J, Y			–	Clears all performance monitor counters.
00200*k*		VL	A*k*	–	Transmits (A*k*) to VL (maximum VL = 128 in C90-mode, 64 in Y-mode).
002000		VL	1	–	Enters 1 into VL.
002100		EFI		–	Enables interrupt on floating-point error.
002200		DFI		–	Disables interrupt on floating-point error.
002300		ERI		–	Enables interrupt on operand range error.
002400		DRI		–	Disables interrupt on operand range error.
002500		DBM		–	Disables bidirectional memory transfers.
002600		EBM		–	Enables bidirectional memory transfers.
002700		CMR		–	Completes memory references.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
0030*j*0		VM	S*i*	–	Transmits (S*j*) to VM register.
003000		VM	0	–	Clears VM register.
0034*jk*		SM*jk*	1,TS	–	Tests and sets semaphore *jk*, $0 \leq jk \leq 37_8$. (Bit 2^2 of *j*=0.)
0036*jk*		SM*jk*	0	–	Clears semaphore *jk*, $0 \leq jk \leq 37_8$. (bit 2^2 of *j*=0.)
0037*jk*		SM*jk*	1	–	Sets semaphore *jk*, $0 \leq jk \leq 37_8$. (Bit 2^2 of *j*=0.)
004000		EX		–	Normal exit.
0050*jk*		J	B*jk*	–	Jumps to (B*jk*).
006*ijkm*		J	*exp*	–	Jumps to *exp*.
007*ijkm*	C,J,Y	R	*exp*	–	Return jump to *exp*; set B00 to (*P*)+2.
010*ijkm*	C,J,Y	JAZ	*exp*	–	Jumps to *exp* if (A0)=0.
011*ijkm*	C,J,Y	JAN	*exp*	–	Jumps to *exp* if (A0)≠0.
012*ijkm*	C,J,Y	JAP	*exp*	–	Jumps to *exp* if (A0) positive; includes (A0)=0.
013*ijkm*	C,J,Y	JAM	*exp*	–	Jumps to *exp* if (A0) negative.
014*ijkm*	C,J,Y	JSZ	*exp*	–	Jumps to *exp* if (S0)=0.
015*ijkm*	C,J,Y	JSN	*exp*	–	Jumps to *exp* if (S0)≠0.
016*ijkm*	C,J,Y	JSP	*exp*	–	Jumps to *exp* if (S0) positive; includes (S0)=0.
017*ijkm*	C,J,Y	JSM	*exp*	–	Jumps to *exp* if (S0) negative.
020*i*00*nm*	E	A*i*	*exp*	–	Transmits *exp* to A*i*.
021*i*00*nm*		A*i*	*exp*	–	Transmits ones complement of *exp* to A*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
022*ijk*	E	A*i*	*exp*	–	Transmits *exp* to A*i*.
023*ij*0		A*i*	S*j*	–	Transmits (S*j*) to A*i*.
023*i*01		A*i*	VL	–	Transmits (VL) to A*i*.
024*ijk*		A*i*	B*jk*	–	Transmits (B*jk*) to A*i*.
025*ijk*		B*jk*	A*i*	–	Transmits (A*i*) to B*jk*.
026*ij*0		A*i*	PS*j*	Pop/LZ	Population count of (S*j*) to A*i*.
026*ij*1		A*i*	QS*j*	Pop/LZ	Population count parity of (S*j*) to A*i*.
026*ij*7		A*i*	SB*j*	–	Transmits (SB*j*) to A*i*.
027*ij*0		A*i*	ZS*j*	Pop/LZ	Leading zero count of (S*j*) to A*i*.
027*ij*7		SB*j*	A*i*	–	Transmits (A*i*) to SB*j*.
030*ijk*		A*i*	A*j*+A*k*	A Int Add	Integer sum of (A*j*) and (A*k*) to A*i*.
030*ij*0		A*i*	A*j*+1	A Int Add	Integer sum of (A*j*) and 1 to A*i*.
030*i*0*k*		A*i*	A*k*	A Int Add	Transmits (A*k*) to A*i*.
031*ijk*		A*i*	A*j*–A*k*	A Int add	Integer difference of (A*j*) less (Ak) to A*i*.
031*ij*0		A*i*	A*j*–1	A Int Add	Integer difference of (A*j*) less 1 to A*i*.
031*i*00		A*i*	–1	A Int Add	Enters –1 into A*i*.
031*i*0*k*		A*i*	–A*k*	A Int Add	Transmits the negative of (A*k*) to A*i*.
032*ijk*		A*i*	A*j**A*k*	A Int Mult	Integer product of (A*j*) and (A*k*) to A*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
033*i*00		A*i*	CI	–	Channel number of highest priority interrupt request to A*i*.
033*ij*0		A*i*	CA,A*j*	–	Address of channel (A*j*) to A*i* (*j*≠0).
033*ij*1		A*i*	CE,Aj	–	Error flag of channel (A*j*) to A*i* (*j*≠0, *k*=1); if C90 mode, include Done flag.
034*ijk*		B*jk*,A*i*	,A0	Memory	Loads (A*i*) words from memory starting at address (A0) to B registers starting at register *jk*.
034*ijk*		B*jk*,A*i*	0,A0	Memory	Loads (A*i*) words from memory starting at address (A0) to B registers starting at register *jk*.
035*ijk*		,A0	B*jk*,A*i*	Memory	Stores (A*i*) words from B registers starting at register *jk* to memory starting at address (A0).
035*ijk*		0,A0	B*jk*,A*i*	Memory	Stores (A*i*) words from B registers starting at register *jk* to memory starting at address (A0).
036*ijk*		T*jk*,A*i*	,A0	Memory	Loads (A*i*) words from memory starting at address (A0) to T registers starting at register *jk*.
036*ijk*		T*jk*,A*i*	0,A0	Memory	Loads (A*i*) words from memory starting at address (A0) to T registers starting at register *jk*.
037*ijk*		,A0	T*jk*,A*i*	Memory	Stores (A*i*) words from T registers starting at register *jk* to memory starting at address (A0).
037*ijk*		0,A0	T*jk*,A*i*	Memory	Stores (A*i*) words from T registers starting at register *jk* to memory starting at address (A0).
040*i*00*nm*		S*i*	*exp*	–	Enters *exp* into S*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
041*i*00*nm*		S*i*	#*exp*	–	Enter ones complement of *exp* into S*i*.
041*i*00*nm*		S*i*	–*exp*	–	Enters twos complement of *exp* into S*i*.
042*ijk*		S*i*	<*exp*	Logical	Forms ones mask in S*i* *exp* bits from the right; *jk* field gets 64 – *exp*.
042*ijk*		S*i*	#>*exp*	Logical	Forms zeros mask in S*i* *exp* bits from the left; *jk* field gets *exp*.
042*i*77		S*i*	1	Logical	Enters 1 into S*i*.
042*i*00		S*i*	-1	Logical	Enters -1 into S*i*.
043*i*00		S*i*	0	Logical	Clears S*i*.
043*ijk*		S*i*	>*exp*	Logical	Forms ones mask in S*i*, *exp* bits from the left; *jk* field gets *exp*.
043*ijk*		S*i*	#<*exp*	Logical	Forms zeros mask in S*i*, *exp* bits from the right; *jk* field gets 64 – *exp*.
044*ijk*		S*i*	S*j*&S*k*	Logical	Logical product of (S*j*) and (S*k*) to S*i*.
044*ij*0		S*i*	S*j*&SB	Logical	Sign bit of (S*j*) to S*i*; *j*≠0.
044*ij*0		S*i*	SB&S*j*	Logical	Sign bit of (S*j*) to S*i*; *j*≠0.
045*ijk*		S*i*	#S*k*&S*j*	Logical	Logical product of (S*j*) and ones complement of (S*k*) to S*i*.
045*ij*0		S*i*	#SB&S*j*	Logical	(S*j*) with sign bit cleared to S*i*.
046*ijk*		S*i*	S*j*\S*k*	Logical	Logical difference of (S*j*) and (S*k*) to S*i*.
046*ij*0		S*i*	S*j*\SB	Logical	Enters (S*j*) into S*i* with sign bit toggled; *j*≠0.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
046*ij*0		S*i*	SB\S*j*	Logical	Enters (S*j*) into S*i* with sign bit toggled; *j*≠0.
047*ijk*		S*i*	#S*j*\S*k*	Logical	Logical equivalence of (S*j*) and (S*k*) to S*i*.
047*ij*0		S*i*	#S*j*\SB	Logical	Logical equivalence of (S*j*) and sign bit to S*i*; *j*≠0.
047*ij*0		Si	#SB\S*j*	Logical	Logical equivalence of sign bit and (S*j*) to S*i*; *j*≠0.
047*i*0*k*		S*i*	#S*k*	Logical	Transmits ones complement of (S*k*) to S*i*.
047*i*00		S*i*	#SB	Logical	Enters ones complement of sign bit into S*i*.
050*ijk*		S*i*	S*j*!S*i*&S*k*	Logical	Scalar merge of (S*i*) and (S*j*) to S*i*.
050*ij*0		S*i*	S*j*!S*i*&SB	Logical	Scalar merge of (S*i*) and sign bit of (S*j*) to S*i*.
051*ijk*		S*i*	S*j*!S*k*	Logical	Logical sum of (S*j*) and (S*k*) to S*i*.
051*ij*0		S*i*	S*j*!SB	Logical	Logical sum of (S*j*) and sign bit to S*i*; *j*≠0.
051*ij*0		S*i*	SB!S*j*	Logical	Logical sum of sign bit and (S*j*) to S*i*; *j*≠0.
051*i*0k		S*i*	S*k*	Logical	Transmits (S*k*) to S*i*.
051*i*00		S*i*	SB	Logical	Enters sign bit into S*i*.
052*ijk*		S0	S*i*<*exp*	Shift	Shifts (S*i*) left *exp* places to S0.
053*ijk*		S0	S*i*>*exp*	Shift	Shifts (S*i*) right *exp* places to S0.
054*ijk*		S*i*	S*i*<*exp*	Shift	Shifts (S*i*) left *exp* places to S*i*.
055*ijk*		S*i*	S*i*>*exp*	Shift	Shifts (S*i*) right *exp* places to S*i*.
056*ijk*		S*i*	S*i*,S*j*<A*k*	Shift	Shifts (S*i*) and (S*j*) left (A*k*) places to S*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
056*ij*0		S*i*	S*i*,S*j*<1	Shift	Shifts (S*i*) and (S*j*) left one place to S*i*.
056*i*0*k*		S*i*	S*i*<A*k*	Shift	Shifts (S*i*) left (A*k*) places to S*i*.
057*ijk*		S*i*	S*j*,S*i*>A*k*	Shift	Shifts (S*j*) and (S*i*) right (A*k*) places to S*i*.
057*ij*0		S*i*	S*j*,S*i*>1	Shift	Shifts (S*j*) and (S*i*) right one place to S*i*.
057*i*0*k*		S*i*	S*i*>A*k*	Shift	Shifts (S*i*) right (A*k*) places to S*i*.
060*ijk*		S*i*	S*j*+S*k*	Int Add	Integer sum of (S*j*) and (S*k*) to S*i*.
060*ij*0	C,Y	S*i*	S*j*+S0	Int Add	Integer sum of (S*j*) and the sign bit to S*i*.
061*ijk*		S*i*	S*j*−S*k*	Int Add	Integer difference of (S*j*) less (S*k*) to S*i*.
061*ij*0	C,Y	S*i*	S*j*−S0	Int Add	Integer difference of (S*j*) less the sign bit to S*i*.
061*i*0*k*		S*i*	−S*k*	Int Add	Transmits negative of (S*k*) to S*i*.
062*ijk*		S*i*	S*j*+FS*k*	Fp Add	Floating-point sum of (S*j*) and (S*k*) to S*i*.
062*i*0*k*		S*i*	+FS*k*	Fp Add	Normalizes (S*k*) to S*i*.
063*ijk*		S*i*	S*j*−FS*k*	Fp Add	Floating-point difference of (S*j*) less (S*k*) to S*i*.
063*i*0*k*		S*i*	−FS*k*	Fp Add	Transmits the negative of (S*k*) as a normalized floating-point value to S*i*.
064*ijk*		S*i*	S*j**FS*k*	Fp Mult	Floating-point product of (S*j*) and (S*k*) to S*i*.
065*ijk*		S*i*	S*j**HS*k*	Fp Mult	Half-precision, rounded, floating-point product of (S*i*) and (S*k*) to S*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
066*ijk*		S*i*	S*j**RS*k*	Fp Mult	Rounded, floating-point product of (S*j*) and (S*k*) to S*i*.
067*ijk*		S*i*	S*j**IS*k*	Fp Mult	2 minus floating-point product of (S*j*) and (S*k*) to S*i*.
070*ij*0		S*i*	/HS*j*	Fp Rcpl	Floating-point reciprocal approximation of (S*j*) to S*i*.
071*i*0*k*		S*i*	A*k*	–	Transmits (A*k*) to S*i* with no sign extension.
071*i*1*k*		S*i*	+A*k*	–	Transmits (A*k*) to S*i* with sign extension.
071*i*2*k*		S*i*	+FA*k*	–	Transmits (A*k*) to S*i* as an unnormalized floating-point value.
071*i*30		S*i*	0.6	–	Transmits 0.75*(2^{48}) to S*i* as a normalized floating-point constant.
071*i*40		S*i*	0.4	–	Transmits 0.5 to S*i* as a normalized floating-point constant.
071*i*50		S*i*	1.	–	Transmits 1.0 to S*i* as a normalized floating-point constant.
071*i*60		S*i*	2.	–	Transmits 2.0 to S*i* as a normalized floating-point constant.
071*i*70		S*i*	4.	–	Transmits 4.0 to S*i* as a normalized floating-point constant.
072*i*00		S*i*	RT	–	Transmits (RTC) to S*i*.
072*i*02		S*i*	SM	–	Transmits semaphores to S*i*.
072*ij*3		S*i*	ST*j*	–	Transmits (ST*j*) register to S*i*.
073*i*00		S*i*	VM	–	Transmits (VM) to S*i*.
073*i*02		SM	S*i*	–	Loads semaphores from S*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
073*i*01		S*i*	SR0	–	Transmits (SR0) to S*i*. This is the only form of the SR*j* syntax valid on CRAY Y-MP systems.
073*i*05		SR0	S*i*	–	Transmits (S*i*) bits 2^{48} through 2^{52} to SR0.
073*i*21	M,C,Y	S*i*	SR2	–	Reads PM counters 00 through 17 and increment pointer. If in Y-mode, increments performance counter.
073*i*31	M,C,Y	S*i*	SR3	–	Reads PM counters 20 through 37 and increment pointer. If in Y-mode, clears all maintenance modes.
074*ijk*		S*i*	T*jk*	–	Transmits (T*jk*) to S*i*.
075*ijk*		T*jk*	S*i*	–	Transmits (S*i*) to T*jk*.
076*ijk*		S*i*	V*j*, A*k*	–	Transmits (V*j*, element (A*k*)) to S*i*.
077*ijk*		V*i*, A*k*	S*j*	–	Transmits (S*j*) to V*i* element (A*k*).
077*i*0*k*		V*i*, A*k*	0	–	Clears element (A*k*) of register V*i*.
100*i*00*nm*		A*i*	*exp*, 0	Memory	Loads from (*exp*) to A*i*.
100*i*00*nm*		A*i*	*exp*,	Memory	Loads from (*exp*) to A*i*.
10*hi*0000		A*i*	, A*h*	Memory	Loads from (A*h*) to A*i*.
11*hi*00*nm*		*exp*, A*h*	A*i*	Memory	Stores (A*i*) to (A*h*)+*exp*; A*h*≠0.
110*i*00*nm*		*exp*, 0	A*i*	Memory	Stores (A*i*) to *exp*.
110*i*00*nm*		*exp*,	A*i*	Memory	Stores (A*i*) to *exp*.
11*hi*0000		, A*h*	A*i*	Memory	Stores (A*i*) to (A*h*).
12*hi*00*nm*		S*i*	*exp*, A*h*	Memory	Loads from ((A*h*)+*exp*) to S*i*; A*h*≠0.
120*i*00*nm*		S*i*	*exp*, 0	Memory	Loads from (*exp*) to S*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
120*i*00*nm*		S*i*	*exp*,	Memory	Loads from (*exp*) to S*i*.
12*hi*0000		S*i*	,A*h*	Memory	Loads from (A*h*) to S*i*.
13*hi*00*nm*		*exp*,A*h*	S*i*	Memory	Stores (S*i*) to (A*h*)+*exp*; A*h*≠0.
130*i*00*nm*		*exp*,0	S*i*	Memory	Stores (S*i*) to *exp*.
130*i*00*nm*		*exp*,	S*i*	Memory	Stores (S*i*) to *exp*.
13*h*i0000		,A*h*	S*i*	Memory	Stores (S*i*) to (A*h*).
140*ijk*		V*i*	S*j*&V*k*	Logical	Logical products of (S*j*) and (V*k*) to V*i*.
141*ijk*		V*i*	V*j*&V*k*	Logical	Logical products of (V*j*) and (V*k*) to V*i*.
142*ijk*		V*i*	S*j*!V*k*	Logical	Logical sums of (S*j*) and (V*k*) to V*i*.
142*i*0*k*		V*i*	V*k*	Logical	Transmits (V*k*) to V*i*.
143*ijk*		V*i*	V*j*!V*k*	Logical	Logical sums of (V*j*) and (V*k*) to V*i*.
144*ijk*		V*i*	S*j*\V*k*	Logical	Logical differences of (S*j*) and (V*k*) to V*i*.
145*ijk*		V*i*	V*j*\V*k*	Logical	Logical differences of (V*j*) and (V*k*) to V*i*.
145*iii*		V*i*	0	Logical	Clears V*i*.
146*ijk*		V*i*	S*j*!V*k*&VM	Logical	Vector merge of (S*j*) and (V*k*) to V*i*.
146*i*0*k*		V*i*	#VM&V*k*	Logical	Vector merge of (V*k*) and zero to V*i*.
147*ijk*		V*i*	V*j*!V*k*&VM	Logical	Vector merge of (V*j*) and (V*k*) to V*i*.
150*ijk*		V*i*	V*j*<A*k*	Shift	Shifts (V*j*) left (A*k*) places to V*i*.
150*ij*0		V*i*	V*j*<1	Shift	Shifts (V*j*) left one place to V*i*.
151*ijk*		V*i*	V*j*>A*k*	Shift	Shifts (V*j*) right (A*k*) places to V*i*.
151*ij*0		V*i*	V*j*>1	Shift	Shifts (V*j*) right one place to V*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
152*ijk*		V*i*	V*j*,V*j*<A*k*	Shift	Double-shifts (V*j*) left (A*k*) places to V*i*.
152*ij*0		V*i*	V*j*,V*j*<1	Shift	Double-shifts (V*j*) left one place to V*i*.
153*ijk*		V*i*	V*j*,V*j*>A*k*	Shift	Double-shifts (V*j*) right (A*k*) places to V*i*.
153*ij*0		V*i*	V*j*,V*j*>1	Shift	Double-shifts (V*j*) right one place to V*i*.
154*ijk*		V*i*	S*j*+V*k*	Int Add	Integer sums of (S*j*) and (V*k*) to V*i*.
155*ijk*		V*i*	V*j*+V*k*	Int Add	Integer sums of (V*j*) and (V*k*) to V*i*.
156*ijk*		V*i*	S*j*–V*k*	Int Add	Integer differences of (S*j*) and (V*k*) to V*i*.
156*i*0*k*		V*i*	–V*k*	Int Add	Transmits twos complement of (V*k*) to V*i*
157*ijk*		V*i*	V*j*–V*k*	Int Add	Integer differences of (V*j*) less (V*k*) to V*i*.
160*ijk*		V*i*	S*j**FV*k*	Fp Mult	Floating-point products of (S*j*) and (V*k*) to V*i*.
161*ijk*		V*i*	V*j**FV*k*	Fp Mult	Floating-point products of (V*j*) and (V*k*) to V*i*.
162*ijk*		V*i*	S*j**HV*k*	Fp Mult	Half-precision, rounded, floating-point products of (S*j*) and (V*k*) to V*i*.
163*ijk*		V*i*	V*j**HV*k*	Fp Mult	Half-precision, rounded, floating-point products of (V*j*) and (V*k*) to V*i*.
164*ijk*		V*i*	S*j**RV*k*	Fp Mult	Rounded floating-point products of (S*j*) and (V*k*) to V*i*.
165*ijk*		V*i*	V*j**RV*k*	Fp Mult	Rounded floating-point products of (V*j*) and (V*k*) to V*i*.

Table 17. Common symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
166*ijk*		V*i*	S*j**V*k*	Fp Mult	32-bit integer product of (S*j*) and (V*k*) to V*i*.
167*ijk*		V*i*	V*j**IV*k*	Fp Mult	Two minus floating-point products of (V*j*) and (V*k*) to V*i*.
170*ijk*		V*i*	S*j*+FV*k*	Fp Add	Floating-point sums of (S*j*) and (V*k*) to V*i*.
170*i*0*k*		V*i*	+FV*k*	Fp Add	Normalizes (V*k*) to V*i*.
171*ijk*		V*i*	V*j*+FV*k*	Fp Add	Floating-point sums of (V*j*) and (V*k*) to V*i*.
172*ijk*		V*i*	S*j*–FV*k*	Fp Add	Floating-point differences of (S*j*) less (V*k*) to V*i*.
172*i*0*k*		V*i*	–FV*k*	Fp Add	Transmits normalized negative of (V*k*) to V*i*.
173*ijk*		V*i*	V*j*–FV*k*	Fp Add	Floating-point differences of (V*j*) less (V*k*) to V*i*.
174*ij*0		V*i*	/HV*j*	Fp Rcpl	Floating-point reciprocal approximations of (V*j*) to V*i*.
174*ij*1		V*i*	PV*j*	V Pop	Population count of (V*j*) to V*i*.
174*ij*2		V*i*	QV*j*	V Pop	Population count parities of (V*j*) to V*i*.
1750*j*0		VM	V*j*,Z	V Logical	Sets VM bits for zero elements of V*j*.
1750*j*1		VM	V*j*,N	V Logical	Sets VM bits for nonzero elements of V*j*.
1750*j*2		VM	V*j*,P	V Logical	Sets VM bits for positive elements of V*j*.
1750*j*3		VM	V*j*,M	V Logical	Sets VM bits for negative elements of V*j*.

Table 17. Common symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
175*ij*4		V*i*,VM	V*j*,Z	V Logical	Sets VM bits for zero elements of V*j*, and register V*i* to the compressed indices of V*j* for zero elements of V*j*.
175*ij*5		V*i*,VM	V*j*,N	V Logical	Sets VM bits for nonzero elements of V*j*, and register V*i* to the compressed indices of V*j* for nonzero elements of V*j*.
175*ij*6		Vi,VM	V*j*,P	V Logical	Sets VM bits for positive elements of V*j*, and register V*i* to the compressed indices of V*j* for positive elements of V*j*.
175*ij*7		V*i*,VM	V*j*,M	V Logical	Sets VM bits for negative elements of V*j*, and register V*i* to the compressed indices of V*j* for negative elements of V*j*.
176*i*0*k*		V*i*	,A0,A*k*	Memory	Loads (VL) words from memory starting at (A0) incrementing by (A*k*) and load into V*i*.
176*i*00		V*i*	,A0,1	Memory	Loads (VL) words from consecutive memory addresses starting with (A0) and load into V*i*.
176*i*1*k*		V*i*	,A0,V*k*	Memory	Reads (VL) words to V*i* from (A0) + (V*k*)
1770*jk*		,A0,A*k*	V*j*	Memory	Stores (VL) words from (V*j*) to memory starting at (A0) incrementing by (A*k*).
1770*j*0		,A0,1	V*j*	Memory	Stores (V*j*) to memory in consecutive addresses starting with (A0).
1771*jk*		,A0,V*k*	V*j*	Memory	Stores (VL) words from V*j* to (A0) + (V*k*).

CRAY J90 and CRAY Y-MP specific instructions

F.14

The instructions listed in Table 18 are available on all CRAY J90 and CRAY Y-MP systems.

Table 18. CRAY J90 and CRAY Y-MP symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
0015*j*0	M			–	Selects performance monitor.
001501	M			–	Disables Port A error correction.
001511	M			–	Disables Port B error correction.
001521	M			–	Disables Port D error correction.
001531	M			–	Enables T register data to be routed through Port D error correction, rather than Port B.
001541	M			–	Enables replacement of check byte with data on Ports C and D writes, and replacement of data with check bytes on Ports A, B, and D reads.
001551	M			–	Enables replacement of check byte with V*k* data on Port C during execution of 1771*jk*.
01*hijkm*	X,E	A*h*	*exp*	–	Transmits *exp* to A*h* (bit 2^2 of *i*=1).
020*ijkm*	X,E	A*i*	*exp*	–	Transmits *exp* to A*i*.
021*ijkm*	X	A*i*	*#exp*	–	Transmits ones complement of *exp* to A*i*.
021*ijkm*	X	A*i*	*–exp*	–	Transmits twos complement of *exp* to A*i*.
040*ijkm*	X	S*i*	*exp*	–	Enters *exp* into S*i*.
041*ijkm*	X	S*i*	*#exp*	–	Enters ones complement of *exp* into S*i*.
041*ijkm*	X	S*i*	*–exp*	–	Enters twos complement of *exp* into S*i*.

Table 18. CRAY J90 and CRAY Y-MP symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
10h*ijkm*	X	A*i*	*exp*,A*h*	Memory	Loads from ((A*h*)+*exp*) to A*i*; A*h*≠0.
100*ijkm*	X	A*i*	*exp*,0	Memory	Loads from (*exp*) to A*i*.
100*ijkm*	X	A*i*	*exp*,	Memory	Loads from (*exp*) to A*i*.
10*hi*000	X	A*i*	,A*h*	Memory	Loads from (A*h*) to A*i*.
10*hi*00*nm*	Y	A*i*	*exp*,A*h*	Memory	Loads A*i* from ((A*h*)+*exp*)
11*hijkm*	X	*exp*,A*h*	A*i*	Memory	Stores (A*i*) to (A*h*)+*exp*; A*h*≠0.
110*ijkm*	X	*exp*,0	A*i*	Memory	Stores (A*i*) to *exp*.
110*ijkm*	X	*exp*,	A*i*	Memory	Stores (A*i*) to *exp*.
11*hi*000	X	,A*h*	A*i*	Memory	Stores (A*i*) to (A*h*).
12*hijkm*	X	S*i*	*exp*,A*h*	Memory	Loads from ((A*h*)+*exp*) to S*i*; A*h*≠0.
120*ijkm*	X	S*i*	*exp*,0	Memory	Loads from (*exp*) to S*i*.
120*ijkm*	X	S*i*	*exp*,	Memory	Loads from (*exp*) to S*i*.
12*hi*000	X	S*i*	,A*h*	Memory	Loads from (A*h*) to S*i*.
13*hijkm*	X	*exp*,A*h*	S*i*	Memory	Stores (S*i*) to (A*h*)+*exp*; A*h*≠0.
130*ijkm*	X	*exp*,0	S*i*	Memory	Stores (S*i*) to *exp*.
130*ijkm*	X	*exp*,	S*i*	Memory	Stores (S*i*) to *exp*.
13*hi*000	X	,A*h*	S*i*	Memory	Stores (S*i*) to (A*h*).
166*ijk*	X	V*i*	S*j**IV*k*	Fp Mult	Two minus floating-point products of (S*j*) and (V*k*) to V*i*.

CRAY C90 specific instructions

F.15

The instructions listed in Table 19 are specific to CRAY C90 systems and CRAY T90 systems running in C90 mode.

Table 19. CRAY C90 symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
0012*j*2	M	DI,A*j*		_	Disables channel (A*j*) interrupts.
0012*j*3	M	EI,A*j*		_	Enables channel (A*j*) interrupts.
001302	M	EMI		_	Enables monitor mode interrupt modes.
001303	M	DMI		_	Disables monitor mode interrupt modes.
001600	M	ESI		–	Enables system I/O interrupts.
0017*jk*	M	BP,*k*	A*j*	–	Transmits (A*j*) to breakpoint address *k*. *k*=0 sets lower-address limit; *k*=1 sets upper-address limit.
002301		EBP		–	Enables interrupt on breakpoint.
002401		DBP		–	Disables interrupt on breakpoint.
002704		CPA		–	Complete port reads and writes.
002705		CPR		–	Completes port reads.
002706		CPW		–	Completes port writes.
0030*j*0		VM0	S*j*	–	Transmits (S*j*) to VM register.
003000		VM0	0	–	Clears VM register.
0030*j*1		VM1	S*j*	–	Transmits (S*j*) to VM upper register.
003001		VM1	0	–	Clears VM1 register.

Table 19. CRAY C90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
0034*jk*		SM,A*k*	1,TS	–	Test and set semaphore (A*k*) (Bit 2^2 of *j*=1.)
0036*jk*		SM,A*k*	0	–	Clears semaphore (A*k*). (Bit 2^2 of *j*=1.)
0037*jk*		SM,A*k*	1	–	Sets semaphore (A*k*). (Bit 2^2 of *j*=1.)
00400*k*		EX*k*		–	Exit *k*.
0051*jk*		JINV	B*jk*	–	Jumps to (B*jk*) (maintenance only, invalidate instruction buffers).
006000*nm*		J	*exp*	–	Jumps to *exp*.
0064*jknm*		JTS*jk*	*exp*	–	Jumps to *exp* if SM*jk*=1; else, set SM*jk* = 1 (bit 2^2 of *j*=0).
0064*jknm*		JTS,A*k*	*exp*	–	Jump to *exp* if SM,A*k*=1; else set SM,A*k* = 1 (bit 2^2 of *j*=1).
026*ij*2	D	A*i*	PA*j*	Pop/LZ	Population count of (A*j*) to A*i*.
026*ij*3	D	A*i*	QA*j*	Pop/LZ	Population count parity of (A*j*) to A*i*.
026*ij*4		A*i*	SB,A*j*,+1	–	Transmits (SB) designated by (A*j*) to A*i*, increment by 1.
026*ij*5		A*i*	SB*j*,+1	–	Transmits (SB*j*) to A*i*; increment by 1.
026*ij*6		A*i*	SB,A*j*	–	Transmits (SB) designated by (A*j*) to A*i*.
027*i*j6		SB,A*j*	A*i*	–	Transmits (A*i*) to SB designated by (A*j*).
040*i*20*nm*		S*i*	S*i*:*exp*	–	Transmits *exp* into S*i* bits 2^0-2^{31} bits 2^{32}–2^{63} are unchanged.

Table 19. CRAY C90 symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
040*i*40*nm*		S*i*	*exp*:S*i*	–	Transmits *exp* into S*i* bits 2^{32}-2^{63} bits 2^{0}–2^{31} are unchanged.
072*ij*6		S*i*	ST,A*j*	–	Transmits (ST) designated by (A*j*) to S*i*.
073*i*00		S*i*	VM0	–	Transmits (VM) to S*i*.
073*i*10		S*i*	VM1	–	Transmits (VM1) to S*i*.
073*ij*1	M	S*i*	SR*j*	–	Transmits (SR*j*) to S*i*.
073*ij*6		ST,A*j*	S*i*	–	Transmits (S*i*) to ST designated by (A*j*).
073*ij*5		SR*j*	S*i*	–	Transmits (S*i*) to SR*j*.
073*i*25	M	SR2	S*i*	–	Issues PM maintenance advance.
073*i*75	M	SR7	S*i*	–	Transmits (S*i*) to maintenance mode register.
10*hi*20*nm*	D	A*i*	*exp*,A*h*,BC	Memory	Loads A*i* from ((A*h*)+*exp*) bypassing data cache and invalidating cache line
005400 150*ij*0		V*i*	V*j*<V0	Shift	Shifts (V*j*) left (V0) places to V*i*.
005400 151*ij*0		V*i*	V*j*>V0	Shift	Shifts (V*j*) right (V0) places to V*i*.
005400 152*ijk*		V*i*	V*j*,A*k*	Shift	Transfers (V*j*) starting at element (A*k*) to (V*i*) starting at element 0.
174*ij*3		V*i*	ZV*j*	V Pop	Leading zero count of (V*j*) to V*i*.

CRAY J90 specific instructions

F.16

The instructions listed in Table 20 are specific to CRAY J90 systems. All of the instructions listed in this table are new instructions.

Table 20. CRAY J90 symbolic machine instructions

Opcode	Notes	CAL	Unit	Function
`0015`*j*`0`	M	N/A	_	Selects performance monitor.
`001501`	M	N/A	–	Disables port A error correction.
`001511`	M	N/A	–	Disables port B error correction.
`001521`	M	N/A	–	Disables port D I/O error correction.
`001541`	M	N/A	–	Enables replacement of checkbyte with data on ports for writes and the replacement of data with checkbytes on ports for reads.
`001551`	M	N/A	–	Replaces checkbits with V*k* data bits on the path to the VA ASIC during execution of instruction `1771`*jk*.
`0016`*j*`1`		`IVC`	–	Send invalidate cache request to CPU (A*j*).

CRAY T90 specific instructions

F.17

The instructions listed in Table 21 are specific to CRAY T90 systems. The instructions listed in Table 19, page 386, are available on CRAY T90 systems running in C90 mode.

Table 21. CRAY T90 symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
0013*j*1	M	A*j*	XA	_	Enters A*j* register with (XA).
001640	M	BCD		–	Broadcasts cluster detach.
0016*j*1	M,N	IVCP	A*j*	–	Invalidate cache in CPU (A*j*).
0016*j*2	M,N	IVCL	A*j*	–	Invalidate cache in CPUs in cluster (A*j*).
002101	N,I	EFI	INV	–	Enables floating-point invalid interrupts.
002102	N,I	EFI	DIV	–	Enables floating-point divide by zero interrupts.
002103	N,I	EFI	OVF	–	Enables floating-point overflow interrupts.
002104	N,I	EFI	UNF	–	Enables floating-point underflow interrupts.
002105	N,I	EFI	INX	–	Enables floating-point inexact interrupts.
002106	N,I	EFI	INP	–	Enables floating-point exceptional input interrupts.
002201	N,I	DFI	INV	–	Disables floating-point invalid interrupts.
002202	N,I	DFI	DIV	–	Disables floating-point divide by zero interrupts.
002203	N,I	DFI	OVF	–	Disables floating-point overflow interrupts.
002204	N,I	DFI	UNF	–	Disables floating-point underflow interrupts.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
002205	N,I	DFI	INX	–	Disables floating-point inexact interrupts.
002206	N,I	DFI	INP	–	Disables floating-point exceptional input interrupts.
002501		ESC		–	Enables scalar cache (sets SCE to 1).
002601		DSC		–	Disables and invalidates scalar cache (clears SCE to 0).
002707	N,I	CFP		–	Completes all floating-point operations.
0030*j*2		VM0	A*j*	–	Transmits (A*j*) to VM register.
0030*j*3		VM1	A*j*	–	Transmits (A*j*) to VM upper register.
003004	N,I	RNM		–	Sets round to nearest mode.
003005	N,I	RUM		–	Sets round up mode.
003006	N,I	RZM		–	Sets round to zero mode.
003007	N,I	RDM		–	Sets round down mode.
006100*nm*		IJ	*exp*	–	Jumps to address in *exp*.
007100*nm*		IR	*exp*	–	Return jump to address in *exp*; set B00 to (*P*)+3.
020*i*20*nm*		A*i*	*Ai:exp*	–	Transmits *exp* to low-order 32 bits of A*i*.
020*i*40*nm*		A*i*	*exp:*A*i*	–	Transmits *exp* to high-order 32 bits of A*i*.
023*ij*6	M	A*i*	EA,*j*	–	Transmits exit address *j* to A*i*.
023*ij*7	M	A*i*	EA,A*j*	–	Transmits exit address (A*j*) to A*i*.
026*ij*2	D	A*i*	PA*j*	Pop/LZ	Population count of (A*j*) to A*i*.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
026*ij*3	D	A*i*	QA*j*	Pop/LZ	Population count parity of (A*j*) to A*i*.
027*ij*1		A*i*	ZA*j*	Pop/LZ	Leading zero count of (A*j*) to A*i*.
027*ij*2	M	EA*j*	A*i*	–	Transmits (A*j*) to exit address *j*.
027*ij*3	M	EA,A*j*	A*i*	–	Transmits (A*i*) to exit address (A*j*).
005400 042*ijk*		A*i*	<*exp*	Logical	Forms ones mask in A*i* *exp* bits from the right; *jk* field gets 64 – *exp*.
005400 043*ijk*		A*i*	>*exp*	Logical	Forms ones mask in A*i*, *exp* bits from the left; *jk* field gets *exp*.
005400 044*ijk*		A*i*	*A*j&A*k*	Logical	Logical product of (A*j*) and (A*k*) to A*i*.
005400 045*ijk*		A*i*	#A*k*&A*j*	Logical	Logical product of (A*j*) and (A*k*) to A*i*.
005400 046*ijk*		A*i*	A*j*\A*k*	Logical	Logical difference of (A*j*) and (A*k*) to A*i*.
005400 047*ijk*		*Ai*	#A*j*\A*k*	Logical	Logical equivalence of (A*j*) and (A*k*) to A*i*.
005400 050*ijk*		A*i*	A*j*!A*i*&A*k*	Logical	Merge A*i* and A*j* to A*i* by using (A*k*) as mask.
005400 051*ijk*		A*i*	A*j*!A*k*	Logical	Logical sum of (A*j*) and (A*k*) to A*i*.
005400 052*ijk*		A0	A*i*<*exp*	Shift	Shifts (A*i*) left *exp* places to A0.
005400 053*ijk*		A0	A*i*>*exp*	Shift	Shifts (A*i*) right *exp* places to A0.
005400 054*ijk*		A*i*	A*i*<*exp*	Shift	Shifts (A*i*) left *exp* places to A*i*.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
005400 055*ijk*		A*i*	A*i*>*exp*	Shift	Shifts (A*i*) right *exp* places to A*i*.
005400 056*ijk*		A*i*	A*i*,A*j*<A*k*	Shift	Shifts (A*i*) and (A*j*) left (A*k*) places to A*i*.
005400 057*ijk*		A*i*	A*j*,A*i*>A*k*	Shift	Shifts (A*j*) and (A*i*) right (A*k*) places to A*i*.
062*ijk*	R	S*i*	S*j*+FS*k*	–	Floating-point S*j* plus S*k* to S*i*.
063*ijk*	R	S*i*	S*j*–FS*k*	–	Floating-point S*j* minus S*k* to S*i*.
064*ijk*	R	S*i*	S*j**FS*k*	–	Floating-point S*j* times S*k* to S*i*.
065*ijk*	F	S*i*	S*j*/FS*j*	–	Floating-point S*k* divided by S*j* to S*i*.
066*ijk*	F	S*i*	S*j**LS*k*	–	Integer S*j* times S*k* to S*i*, returning lower.
005400 066*ijk*	N,I	S*i*	S*j**US*k*	–	Integer S*j* times S*k* to S*i*, returning upper.
070*ij*0	F	S*i*	SQRT,S*j*	–	Floating-point square root of S*j* to S*i*.
070*ij*1		V*i*	CI,S*j*&VM	–	Transmit compressed index of (S*j*) controlled by (VM) to V*i*
070*ij*2	N,I	S*i*	INT,S*j*	–	Floating-point S*j* to integer S*i*.
070*ij*3	N,I	S*i*	RINT,S*j*	–	Floating-point S*j* to rounded integer S*i*.
070*ij*4	N,I	S*i*	FLT,S*j*	–	Integer S*j* to floating-point S*i*.
073*ij*3		ST*j*	S*i*	–	Transmits (S*i*) to ST*j*.
005400 073*i*05	N,I	SETRM	S*i*	–	Set rounding mode from S*i*.
073*i*20		A*i*	VM0	–	Transmits (VM0) to A*i*.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
073*i*30		A*i*	VM1	–	Transmits (VM1) to A*i*.
10*hi*20*nm*	D	A*i*	*exp*,A*h*,BC	Memory	Loads A*i* from ((A*h*)+*exp*) bypassing data cache and invalidating cache line
10*hi*40*pnm*		A*i*	*exp*,A*h*	Memory	Loads A*i* from ((A*h*)+*exp*)
10*hi*60*pnm*		A*i*	*exp*,A*h*,BC	Memory	Loads A*i* from ((A*h*)+*exp*) bypassing data cache and invalidating cache line
11*hi*40*pnm*		*exp*,A*h*	A*i*	Memory	Stores (A*i*) to ((A*h*)+*exp*)
12*hi*20*nm*		S*i*	*exp*,A*h*,BC	Memory	Loads S*i* from ((A*h*)+*exp*) bypassing data cache and invalidating cache line
12*hi*40*pnm*		S*i*	*exp*,A*h*	Memory	Loads S*i* from ((A*h*)+*exp*)
12*hi*60*pnm*		S*i*	*exp*,A*h*,BC	Memory	Loads S*i* from ((A*h*)+*exp*) bypassing data cache and invalidating cache line
13*hi*40*pnm*		*exp*,A*h*	S*i*	Memory	Stores (S*i*) to ((A*h*)+*exp*)
005400 153*ij*0		V*i*	V*j*,[VM]	–	Compress V*j* by (VM) to V*i*.
005400 153*ij*1		V*j*,[VM]	V*i*	–	Expand V*j* by [VM] to V*i*.
160*ijk*	R	V*i*	S*j**FV*k*	–	Floating-point S*j* times V*k* to V*i*.
161*ijk*	R	V*i*	V*j**FV*k*	–	Floating-point V*j* times V*k* to V*i*.
162*ijk*	F	V*i*	V*k*/FS*j*	–	Floating-point V*k* divided by S*j* to V*i*.
163*ijk*	F	V*i*	V*k*/FV*j*	–	Floating-point V*k* divided by V*j* to V*i*.
005501 164*ijk*	N,I	S*i*	S*j*,EQ,S*k*	–	Floating-point compare equal.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
005502 164*ijk*	N,I	S*i*	S*j*,NE,S*k*	–	Floating-point compare not equal.
005503 164*ijk*	N,I	S*i*	S*j*,GT,S*k*	–	Floating-point compare greater than.
005504 164*ijk*	N,I	S*i*	S*j*,LE,S*k*	–	Floating-point compare less than or equal.
005505 164*ijk*	N,I	S*i*	S*j*,LT,S*k*	–	Floating-point compare less than.
005506 164*ijk*	N,I	S*i*	S*j*,GE,S*k*	–	Floating-point compare greater than or equal.
005507 164*ijk*	N,I	S*i*	S*j*,UN,S*k*	–	Floating-point compare unordered.
005521 1640*jk*	N,I	VM	S*j*,EQ,V*k*	–	Floating-point compare equal.
005522 1640*jk*	N,I	VM	S*j*,NE,V*k*	–	Floating-point compare not equal.
005523 1640*jk*	N,I	VM	S*j*,GT,V*k*	–	Floating-point compare greater than.
005524 1640*jk*	N,I	VM	S*j*,LE,V*k*	–	Floating-point compare less than or equal.
005525 1640*jk*	N,I	VM	S*j*,LT,V*k*	–	Floating-point compare less than.
005526 1640*jk*	N,I	VM	S*j*,GE,V*k*	–	Floating-point compare greater than or equal.
005527 1640*jk*	N,I	VM	S*j*,UN,V*k*	–	Floating-point compare unordered.
005541 1640*jk*	N,I	VM	V*j*,EQ,V*k*	–	Floating-point compare equal.
005542 1640*jk*	N,I	VM	V*j*,NE,V*k*	–	Floating-point compare not equal.
005543 1640*jk*	N,I	VM	V*j*,GT,V*k*	–	Floating-point compare greater than.

Table 21. CRAY T90 symbolic machine instructions (continued)

Opcode	Notes	CAL		Unit	Function
005544 1640*jk*	N,I	VM	V*j*,LE,V*k*	–	Floating-point compare less than or equal.
005545 1640*jk*	N,I	VM	V*j*,LT,V*k*	–	Floating-point compare less than.
005546 1640*jk*	N,I	VM	V*j*,GE,V*k*	–	Floating-point compare greater than or equal.
005547 1640*jk*	N,I	VM	V*j*,UN,V*k*	–	Floating-point compare unordered.
165*ijk*	F	V*i*	V*j**LV*k*	–	Integer V*j* times V*k* to V*i* returning lower.
005400 165*ijk*	N,I	V*i*	V*j**UV*k*	–	Integer V*j* times V*k* to V*i* returning upper.
166*ijk*	F	V*i*	S*j**LV*k*	–	Integer S*j* times V*k* to V*i* returning lower.
005400 166*ijk*	N,I	V*i*	S*j**UV*k*	–	Integer S*j* times V*k* to V*i* returning upper.
167*ij*0	F	V*i*	INT,V*j*	–	Floating-point V*j* to integer V*i*.
167*ij*1	F	V*i*	RINT,V*j*	–	Floating-point V*j* to rounded integer V*i*.
167*ij*2	F	V*i*	FLT,V*j*	–	Integer V*j* to floating-point V*i*.
170*ijk*	R	V*i*	S*j*+FV*k*	–	Floating-point S*j* plus V*k* to V*i*.
171*ijk*	R	V*i*	V*j*+FV*k*	–	Floating-point V*j* plus V*k* to V*i*.
172*ijk*	R	V*i*	S*j*–FV*k*	–	Floating-point S*j* minus V*k* to V*i*.
173*ijk*	R	V*i*	V*j*–FV*k*	–	Floating-point V*j* minus V*k* to V*i*.

Table 21. CRAY T90 symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
174*ij*0	F	V*i*	SQRT,V*j*	–	Floating-point square root of V*j* to V*i*.
005400 176*ijk*		V*i* :V*j*	,A0:A*k*,V*k*	Memory	Loads V*i* from memory using addresses (A0) + (V*k*) and load V*j* from memory using addresses (A*k*) + (V*k*).

Bit Matrix multiply instructions

F.18

The instructions listed in Table 22 are available only on systems that support the bit matrix multiply (BMM) function.

Table 22. Bit matrix multiply symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
002210		CBL		–	Clears the B matrix loaded bit in the exchange package and the status register.
070*ij*6	N	S*i*	S*j**BT	BMM	Load single bit matrix element S*j* into the BMM functional unit. Generate results of S*j**BT and store the results in S*i*. S*j* must be left-justified and zero-filled.
1740*j*5	N	BMM	UV*j*	–	Transmits V*j* elements 64 through 127 to B matrix.

Table 22. Bit matrix multiply symbolic machine instructions

Opcode	Notes	CAL		Unit	Function
174*ij*4		BMM	V*j*	BMM	Load elements 0 through VL of V*j* into the BMM functional unit as B^t. Matrix B must be stored in V*j*.
174*ij*6		V*i*	V*j**BT	BMM	Logical bit matrix multiply of V*j* and elements 0 through VL of matrix B^t to V*i*. Matrix A must be left-justified and zero-filled in V*j*; result matrix C is stored in V*i* left-justified to bit 63.

Special register values and logical operators

F.19

Table 23 shows special register values and logical operators.

Table 23. Special register values and logical operators

Register	Value	Logical operators	
A*h*, *h*=0	0		0101
A*i*, *i*=0	(A0)		1100
A*j*, *j*=0	0	(&, AND, Product)	0100
A*k*, *k*=0	1		
S*i*, *i*=0	(S0)		0101
S*j*, *j*=0	0		1100
S*k*, *k*=0	2^{63}	(!, OR, Sum)	1101
			0101
			1100
		(\, XOR, Difference)	1001