Appendix A Instructions

A.1 Instruction Set

Operation Notation

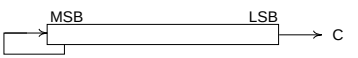
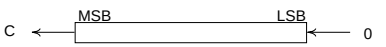
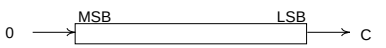
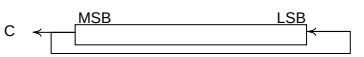
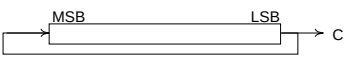
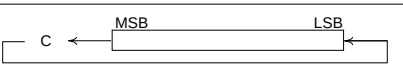
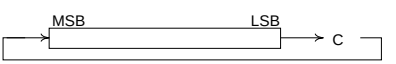
Rd	General register (destination operand)
Rs	General register (source operand)
Rn	General register
(EAd)	Destination operand
(EAs)	Source operand
CCR	Condition code register
N	N (Negative) flag in CCR
Z	Z (Zero) flag in CCR
V	V (Overflow) flag in CCR
C	C (Carry) flag in CCR
CR	Control register
PC	Program counter
CP	Code page register
SP	Stack pointer

FP	Frame pointer
#IMM	Immediate data
disp	Displacement
+	Add
–	Subtract
×	Multiply
÷	Divide
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
↔	Swap
¬	Logical NOT

Condition Code Notation

↕	Changed after instruction execution
0	Cleared to 0
1	Set to 1
—	Value before operation is retained
Δ	Changed depending on condition

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Data transfer	MOV: G	(EAs) ⟶ Rd Rs ⟶ (EAd) #IMM ⟶ (EAd)	B/W	↕	↕	0	—
	MOV: E	#IMM ⟶ Rd (short format)	B	↕	↕	0	—
	MOV: F	@ (d: 8, FP) ⟶ Rd Rs ⟶ @ (d: 8, FP) (short format)	B/W	↕	↕	0	—
	MOV: I	#IMM ⟶ Rd (short format)	W	↕	↕	0	—
	MOV: L	(@aa: 8) ⟶ Rd (short format)	B/W	↕	↕	0	—
	MOV: S	Rs ⟶ (@aa: 8) (short format)	B/W	↕	↕	0	—
	LDM	@ SP + ⟶ Rn (register list)	W	—	—	—	—
	STM	Rn (register list) ⟶ @ – SP	W	—	—	—	—
	XCH	Rs ⟷ Rd	W	—	—	—	—
	SWAP	Rd (upper byte) ⟷ Rd (lower byte)	B	↕	↕	0	—
	MOVTPE	Rs ⟶ (EAd) Synchronized with E clock	B	—	—	—	—
	MOVFPE	(EAs) ⟶ Rd Synchronized with E clock	B	—	—	—	—
Arithmetic operations	ADD: G	Rd + (EAs) ⟶ Rd	B/W	↕	↕	↕	↕
	ADD: Q	(EAd) + #IMM ⟶ (EAd) (#IMM = ±1, ±2) (short format)	B/W	↕	↕	↕	↕
	ADDS	Rd + (EAs) ⟶ Rd (Rd is always word size)	B/W	—	—	—	—
	ADDX	Rd + (EAs) + C ⟶ Rd	B/W	↕	↕	↕	↕
	DADD	$(Rd)_{10}$ + $(Rs)_{10}$ + C ⟶ $(Rd)_{10}$	B	—	↕	—	↕
	SUB	Rd – (EAs) ⟶ Rd	B/W	↕	↕	↕	↕
	SUBS	Rd – (EAs) ⟶ Rd	B/W	—	—	—	—
	SUBX	Rd – (EAs) – C ⟶ Rd	B/W	↕	↕	↕	↕
	DSUB	$(Rd)_{10}$ – $(Rs)_{10}$ – C ⟶ $(Rd)_{10}$	B	—	↕	—	↕
	MULXU	Rd × (EAs) ⟶ Rd 8 × 8 (Unsigned) 16 × 16	B/W	↕	↕	0	0
	DIVXU	Rd ÷ (EAs) ⟶ Rd 16 ÷ 8 (Unsigned) 32 ÷ 16	B/W	↕	↕	↕	0
	CMP: G	Rd – (EAs), Set CCR (EAd) – #IMM, Set CCR	B/W	↕	↕	↕	↕
	CMP: E	Rd – #IMM, Set CCR (short format)	B	↕	↕	↕	↕
	CMP: I	Rd – #IMM, Set CCR (short format)	W	↕	↕	↕	↕

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Arithmetic operations	EXTS	(< Bit 7 > of < Rd >) ⟶ (< Bit 15 to 8 > of < Rd >)	B	↕	↕	0	0
	EXTU	0 ⟶ (<Bit 15 to 8 > of < Rd >)	B	0	↕	0	0
	TST	(EAd) – 0, Set CCR	B/W	↕	↕	0	0
	NEG	0 – (EAd) ⟶ (EAd)	B/W	↕	↕	0	↕
	CLR	0 ⟶ (EAd)	B/W	0	1	0	0
	TAS	(EAd) – 0, Set CCR $(1)_2$ ⟶ (< Bit 7 > of < EAd >)	B	↕	↕	0	0
Shift operations	SHAL		B/W	↕	↕	↕	↕
	SHAR		B/W	↕	↕	0	↕
	SHLL		B/W	↕	↕	0	↕
	SHLR		B/W	0	↕	0	↕
	ROTL		B/W	↕	↕	0	↕
	ROTR		B/W	↕	↕	0	↕
	ROTXL		B/W	↕	↕	0	↕
	ROTXR		B/W	↕	↕	0	↕
Logic operations	AND	Rd ∧ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	OR	Rd ∨ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	XOR	Rd ⊕ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	NOT	¬ (EAd) ⟶ (EAd)	B/W	↕	↕	0	—
Bit manipulations	BSET	¬ (< Bit number > of < EAd >) ⟶ Z 1 ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—
	BCLR	¬ (< Bit number > of < EAd >) ⟶ Z 0 ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—
	BTST	¬ (< Bit number > of < EAd >) ⟶ Z	B/W	—	↕	—	—
	BNOT	¬ (< Bit number > of < EAd >) ⟶ Z ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Branching instructions	Bcc	If condition is true then PC + disp ⟶ PC else next;	—	—	—	—	—

Mnemonic		Description	Condition
BRA	(BT)	Always (True)	True
BRN	(BF)	Never (False)	False
BHI		HIgh	C ∨ Z = 0
BLS		Low or Same	C ∨ Z = 0
Bcc	(BHS)	Carry Clear (High or Same)	C = 0
BCS	(BLO)	Carry Set (LOw)	C = 1
BNE		Not Equal	Z = 0
BEQ		EQual	Z = 1
BVC		oVerflow Clear	V = 0
BVS		oVerflow Set	V = 1
BPL		PLus	N = 0
BMI		MInus	N = 1
BGE		Greater or Equal	N ⊕ V = 0
BLT		Less Than	N ⊕ V = 1
BGT		Greater Than	Z ∨ (N ⊕ V) = 0
BLE		Less or Equal	Z ∨ (N ⊕ V) = 1

	JMP	Effective address ⟶ PC	—	—	—	—	—
	PJMP	Effective address ⟶ CP, PC	—	—	—	—	—
	BSR	PC ⟶ @ – SP PC + disp ⟶ PC	—	—	—	—	—
	JSR	PC ⟶ @ – SP Effective address ⟶ PC	—	—	—	—	—
	PJSR	PC ⟶ @ – SP CP ⟶ @ – SP Effective address ⟶ CP, PC	—	—	—	—	—
	RTS	@ SP + ⟶ PC	—	—	—	—	—
	PRTS	@ SP + ⟶ CP @ SP + ⟶ PC	—	—	—	—	—
	RTD	@ SP + ⟶ PC SP + #IMM ⟶ SP	—	—	—	—	—
	PRTD	@ SP + ⟶ CP @ SP + ⟶ PC SP + #IMM ⟶ SP	—	—	—	—	—
	SCB SCB/F SCB/NE SCB/EQ	If condition is true then next; else Rn – 1 ⟶ Rn; If Rn = –1 then next; else PC + disp ⟶ PC;	—	—	—	—	—

Mnemonic	Description	Condition
SCB/F		False
SCB/NE	Not Equal	Z = 0
SCB/EQ	Equal	Z = 1

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
System control	TRAPA	PC ⟶ @ – SP (If MAX MODE CP ⟶ @ – SP) SR ⟶ @ – SP (If MAX MODE < vector > ⟶ CP) < vector > ⟶ PC	—	—	—	—	—
	TRAP/VS	If V bit = “1” then TRAP else next;	—	—	—	—	—
	RTE	@ SP + ⟶ SR (If MAX MODE @ SP + ⟶ CP) @ SP + ⟶ PC	—	↕	↕	↕	↕
	LINK	FP (R6) ⟶ @ – SP SP ⟶ FP (R6) SP + #IMM ⟶ SP	—	—	—	—	—
	UNLK	FP (R6) ⟶ SP @SP + ⟶ FP	—	—	—	—	—
	SLEEP	Normal running mode ⟶ power-down state	—	—	—	—	—
	LDC	(EAs) ⟶ CR	B/W*	△	△	△	△
	STC	CR ⟶ (EAd)	B/W*	—	—	—	—
	ANDC	CR ∧ #IMM ⟶ CR	B/W*	△	△	△	△
	ORC	CR ∨ #IMM ⟶ CR	B/W*	△	△	△	△
	XORC	CR ⊕ #IMM ⟶ CR	B/W*	△	△	△	△
	NOP	PC + 1 ⟶ PC	—	—	—	—	—

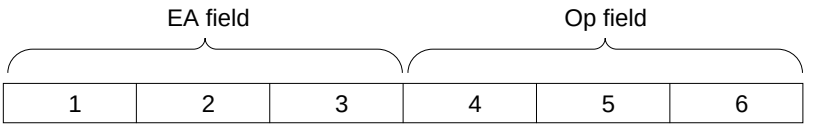
* Depends on the CR.

A.2 Instruction Codes

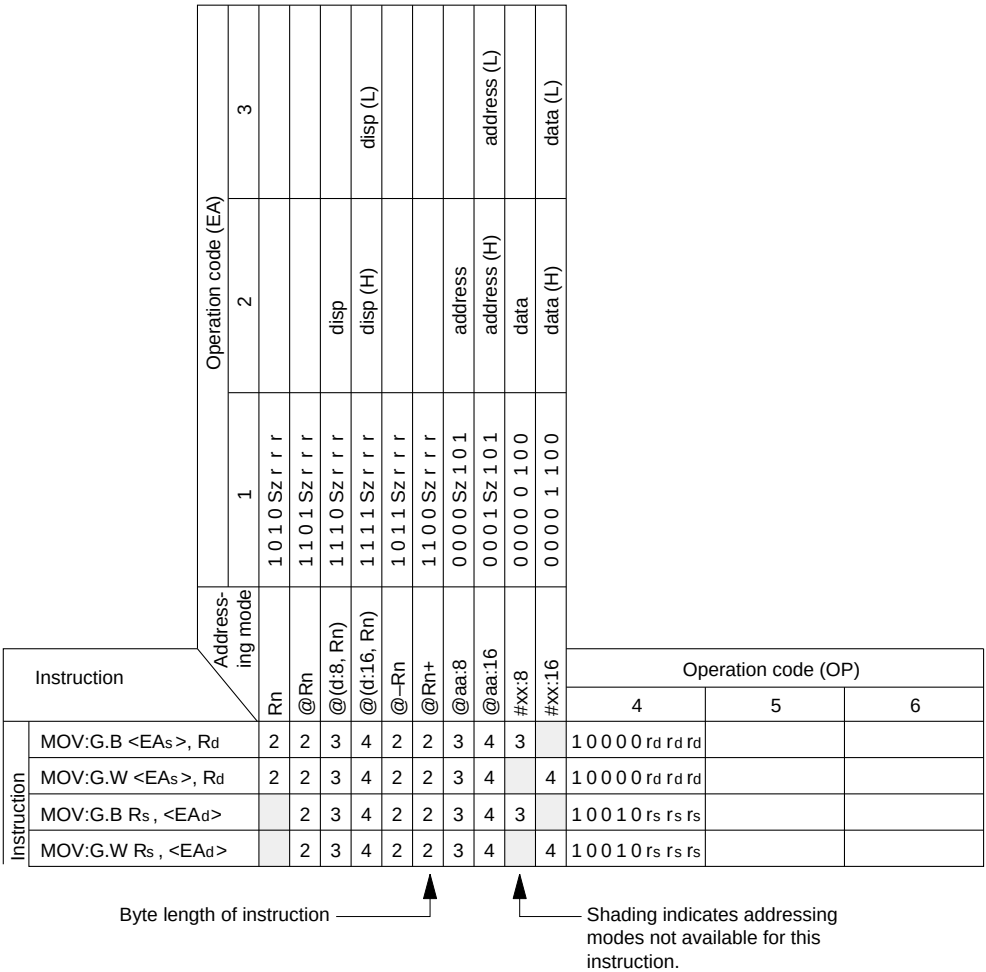
Table A-1 shows the machine-language coding of each instruction.

- **How to read table A-1 (a) to (d)**

The general operand format consists of an effective address (EA) field and operation-code (OP) field specified in the following order.



Bytes 2, 3, 5, 6 are not present in all instructions.



Some instructions have a special format in which the operation code comes first.

The following notation is used in the tables.

- Sz: Operand size (byte or word)
  Byte: Sz = 0
  Word: Sz = 1

• rrr : General register number field

rrr	Sz = 0 (Byte)		Sz = 1 (Word)
	15 8	7 0	15 0
000	Not used	R0	R0
001	Not used	R1	R1
010	Not used	R2	R2
011	Not used	R3	R3
100	Not used	R4	R4
101	Not used	R5	R5
110	Not used	R6	R6
111	Not used	R7	R7

• ccc : Control register number field

ccc	Sz = 0 (Byte)	Sz = 1 (Word)
000	(Not allowed*)	15 0 SR
	7 0	
001	CCR	(Not allowed)
010	(Not allowed)	(Not allowed)
011	BR	(Not allowed)
100	EP	(Not allowed)
101	DP	(Not allowed)
110	(Not allowed)	(Not allowed)
111	TP	(Not allowed)

* “Disallowed” means that this combination of bits must not be specified. Specifying a disallowed combination may cause abnormal results.

• register list: A byte in which bits indicate general registers as follows

Bit	7	6	5	4	3	2	1	0
	R7	R6	R5	R4	R3	R2	R1	R0

• #VEC: Four bits designating a vector number from 0 to 15. The vector numbers correspond to addresses of entries in the exception vector table as follows:

	Vector Address	
#VEC	**Minimum Mode**	**Maximum Mode**
0	H'0020 – H'0021	H'0040 – H'0043
1	H'0022 – H'0023	H'0044 – H'0047
2	H'0024 – H'0025	H'0048 – H'004B
3	H'0026 – H'0027	H'004C – H'004F
4	H'0028 – H'0029	H'0050 – H'0053
5	H'002A – H'002B	H'0054 – H'0057
6	H'002C – H'002D	H'0058 – H'005B
7	H'002E – H'002F	H'005C – H'005F

	Vector Address	
#VEC	**Minimum Mode**	**Maximum Mode**
8	H'0030 – H'0031	H'0060 – H'0063
9	H'0032 – H'0033	H'0064 – H'0067
10	H'0034 – H'0035	H'0068 – H'006B
11	H'0036 – H'0037	H'006C – H'006F
12	H'0038 – H'0039	H'0070 – H'0073
13	H'003A – H'003B	H'0074 – H'0077
14	H'003C – H'003D	H'0078 – H'007B
15	H'003E – H'003F	H'007C – H'007F

• **Examples of machine-language coding**

Example 1: ADD:G.B @R0, R1

	EA Field	**OP Field**
Table A-1 (a)	1101Szrrr	00100rdrdrd
Machine code	11010000	00100 0 0 1
	H'D021	

Example 2: ADD:G.W @H'11:8, R1

	EA Field		**OP Field**
Table A-1 (a)	0000Sz101	00010001	00100rdrdrd
Machine code	0000 1 101	00010001	00100 0 0 1
	H'0D1121		

Table A-1 (a) Machine Language Coding [General Format]

Instruction		Addressing mode	Operation code (EA) 1	2	3	Operation code (OP) 4	5	6
		Rn	1010Szrrr			10000rdrdrd		
		@Rn	1101Szrrr			10000rdrdrd		
		@(d:8, Rn)	1110Szrrr	disp	disp (L)	10010rsrsrs		
		@(d:16, Rn)	1111Szrrr	disp (H)	disp (L)	10010rsrsrs		
		@–Rn	1011Szrrr			00000110	data	
		@Rn+	1100Szrrr			00000110	data	
		@aa:8	0000Sz101	address		00000111	data (H)	data (L)
		@aa:16	0001Sz101	address (H)	address (L)	00000010	register list	
		#xx:8	0000 0 100	data		00000010	register list	
		#xx:16	0000 1 100	data (H)	data (L)	10010rdrdrd		
Data transfer instruction	MOV:G.B <EAs>, Rd	2	2	3	4	00000000	10010rsrsrs	
	MOV:G.W <EAs>, Rd	2	2	3	4	00000000	10010rdrdrd	
	MOV:G.B Rs , <EAd>		2	3	4	00100rdrdrd		
	MOV:G.W Rs , <EAd>		2	3	4	00100rdrdrd		
	MOV:G.B #xx:8, <EAd>		3	4	5	00001000		
	MOV:G.W #xx:8, <EAd>		3	4	5	00001000		
	MOV:G.W #xx:16, <EAd>		4	5	6	00001001		
	LDM.W @SP+, <register list>					00001001		
	STM.W <register list>,@–SP					00001100		
	XCH.W Rs ,Rd	2				00001100		
	SWAP.B Rd	2				00001101		
	MOVTPE.B Rs , <EAd>		3	4	5	00001101		
	MOVTPE.B <EAs>, Rd		3	4	5	00001101		
Arithmetic operation instruction	ADD:G.B <EAs>, Rd	2	2	3	4	00100rdrdrd		
	ADD:G.W <EAs>, Rd	2	2	3	4	00100rdrdrd		
	ADD:Q.B #1, <EAd>*	2	2	3	4	00001000		
	ADD:Q.W #1, <EAd>*	2	2	3	4	00001000		
	ADD:Q.B #2, <EAd>*	2	2	3	4	00001001		
	ADD:Q.W #2, <EAd>*	2	2	3	4	00001001		
	ADD:Q.B #-1, <EAd>*	2	2	3	4	00001100		
	ADD:Q.W #-1, <EAd>*	2	2	3	4	00001100		
	ADD:Q.B #-2, <EAd>*	2	2	3	4	00001101		
	ADD:Q.W #-2, <EAd>*	2	2	3	4	00001101		
	ADDS.B <EAs>, Rd	2	2	3	4	00101rdrdrd		
	ADDS.W <EAs>, Rd	2	2	3	4	00101rdrdrd		
	ADDX.B <EAs>, Rd	2	2	3	4	10100rdrdrd		
	ADDX.W <EAs>, Rd	2	2	3	4	10100rdrdrd		

Note: *Short format instruction

Table A-1 (a) Machine Language Coding [General Format] (cont)

	Instruction	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16	Operation code (OP) 4	5	6
		Operation code (EA) 1: 1 0 1 0 Sz r r r	1 1 0 1 Sz r r r	1 1 1 0 Sz r r r	1 1 1 1 Sz r r r	1 0 1 1 Sz r r r	1 1 0 0 Sz r r r	0 0 0 0 Sz 1 0 1	0 0 0 1 Sz 1 0 1	0 0 0 0 0 1 0 0	0 0 0 0 1 1 0 0			
		Operation code (EA) 2	disp	disp (H)				address	address (H)	data	data (H)			
		Operation code (EA) 3		disp (L)					address (L)		data (L)			
Arithmetic operation instruction	DADD.B Rs ,Rd											0 0 0 0 0 0 0 0	1 0 1 0 0 rd rd rd	
	SUB.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0 0 1 1 0 rd rd rd		
	SUB.W <EAs>, R d	2	2	3	4	2	2	3	4		4	0 0 1 1 0 rd rd rd		
	SUBS.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0 0 1 1 1 rd rd rd		
	SUBS.W <EAs>,Rd	2	2	3	4	2	2	3	4		4	0 0 1 1 1 rd rd rd		
	SUBX.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 1 0 rd rd rd		
	SUBX.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 1 0 rd rd rd		
	DSUB.B Rs, Rd	3										0 0 0 0 0 0 0 0	1 0 1 1 0 rd rd rd	
	MULXU.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 0 1 rd rd rd		
	MULXU.X <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 0 1 rd rd rd		
	DIVXU.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 1 1 rd rd rd		
	DIVXU.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 1 1 rd rd rd		
	CMP:G.B <EAs>, Rd	2	3	4	5	3	3	4	5	3		0 1 1 1 0 rd rd rd		
	CMP:G.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0 1 1 1 0 rd rd rd		
	CMP:G.B #xx, <EAd>		3	4	5	3	3	4	5			0 0 0 0 0 1 0 0	data	
	CMP:G.W #xx, <EAd>		4	5	6	4	4	5	6			0 0 0 0 0 1 0 1	data (H)	data (L)
	EXTS.B Rd	2										0 0 0 1 0 0 0 1		
	EXTU.B Rd	2										0 0 0 1 0 0 1 0		
	TST.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 0		
	TST.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 0		
	NEG.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 0 0		
	NEG.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 0 0		
	CLR.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 0 1 1		
	CLR.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 0 1 1		
	TAS.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 1		

Table A-1 (a) Machine Language Coding [General Format] (cont)

	Instruction	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16	Operation code (EA) 1	2	3	Operation code (OP) 4	5	6
Shift instruction	SHAL.B <EAd>	2	2	3	4	2	2	3	4			1010Szrrr			00011000		
	SHAL.W <EAd>	2	2	3	4	2	2	3	4			1101Szrrr			00011000		
	SHAR.B <EAd>	2	2	3	4	2	2	3	4			1110Szrrr	disp		00011001		
	SHAR.W <EAd>	2	2	3	4	2	2	3	4			1111Szrrr	disp (H)	disp (L)	00011001		
	SHLL.B <EAd>	2	2	3	4	2	2	3	4			1011Szrrr			00011010		
	SHLL.W <EAd>	2	2	3	4	2	2	3	4			1100Szrrr			00011010		
	SHLR.B <EAd>	2	2	3	4	2	2	3	4			0000Sz101	address		00011011		
	SHLR.W <EAd>	2	2	3	4	2	2	3	4			0001Sz101	address (H)	address (L)	00011011		
	ROTL.B <EAd>	2	2	3	4	2	2	3	4			0000 0100	data		00011100		
	ROTL.W <EAd>	2	2	3	4	2	2	3	4			0000 1100	data (H)	data (L)	00011100		
	ROTR.B <EAd>	2	2	3	4	2	2	3	4						00011101		
	ROTR.W <EAd>	2	2	3	4	2	2	3	4						00011101		
	ROTXL.B <EAd>	2	2	3	4	2	2	3	4						00011110		
	ROTXL.W <EAd>	2	2	3	4	2	2	3	4						00011110		
	ROTXR.B <EAd>	2	2	3	4	2	2	3	4						00011111		
	ROTXR.W <EAd>	2	2	3	4	2	2	3	4						00011111		
Logic operation instruction	AND.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0101 0rdrdrd					
	AND.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0101 0rdrdrd					
	OR.B.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0100 0rdrdrd					
	OR.B.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0100 0rdrdrd					
	XOR.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0110 0rdrdrd					
	XOR.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0110 0rdrdrd					
	NOT.B <EAd>	2	2	3	4	2	2	3	4			00010101					
	NOT.W <EAd>	2	2	3	4	2	2	3	4			00010101					

Table A-1 (a) Machine Language Coding [General Format] (cont)

Instruction		Addressing mode	Operation code (EA) 1	Operation code (EA) 2	Operation code (EA) 3	Operation code (OP) 4	Operation code (OP) 5	Operation code (OP) 6
		Rn	1 0 1 0 Sz r r r			1 1 0 0 (data)		
		@Rn	1 1 0 1 Sz r r r			1 1 0 0 (data)		
		@(d:8, Rn)	1 1 1 0 Sz r r r	disp		0 1 0 0 1 rs rs rs		
		@(d:16, Rn)	1 1 1 1 Sz r r r	disp (H)	disp (L)	0 1 0 0 1 rs rs rs		
		@–Rn	1 0 1 1 Sz r r r			1 1 0 1 (data)		
		@Rn+	1 1 0 0 Sz r r r			1 1 0 1 (data)		
		@aa:8	0 0 0 0 Sz 1 0 1	address		0 1 0 1 1 rs rs rs		
		@aa:16	0 0 0 1 Sz 1 0 1	address (H)	address (L)	0 1 0 1 1 rs rs rs		
		#xx:8	0 0 0 0 0 1 0 0	data		1 1 1 1 (data)		
		#xx:16	0 0 0 0 1 1 0 0	data (H)	data (L)	1 1 1 1 (data)		
Bit manipulate instruction	BSET.B #xx, <EAd>	2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BSET.W #xx, <EAd>	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BSET.B Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 0		
	BSET.W Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1		
	BCLR.B #xx, <EAd>	2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BCLR.W #xx, <EAd>	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BCLR.B Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 0		
	BCLR.W Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1		
	BTST.B #xx, <EAd>	2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BTST.W #xx, <EAd>	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BTST.B Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 0		
	BTST.W Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1		
	BNOT.B #xx, <EAd>	2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BNOT.W #xx, <EAd>	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BNOT.B Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 0		
	BNOT.W Rs, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1		
System control instruction	LDC.B <EAs>, CR	2	2	3	4	2	2	3	4	3		1 0 0 0 1 c c c		
	LDC.W <EAs>, CR	2	2	3	4	2	2	3	4		4	1 0 0 0 1 c c c		
	STC.B CR, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1 1 c c c		
	STC.W CR, <EAd>	2	2	3	4	2	2	3	4			1 0 0 1 1 c c c		
	ANDC.B #xx:8, CR									3		0 1 0 1 1 c c c		
	ANDC.W #xx:16, CR										4	0 1 0 1 1 c c c		
	ORC.B #xx:8, CR									3		0 1 0 0 1 c c c		
	ORC.W #xx:16, CR										4	0 1 0 0 1 c c c		
	XORC.B #xx:8, CR									3		0 1 1 0 1 c c c		
	XORC.W #xx:16, CR										4	0 1 1 0 1 c c c		

Table A-1 (b) Machine Language Coding [Special Format: Short Format]

Instruction	Byte	Operation code			
		1	2	3	4
MOV:E,B #xx:8,Rd	2	01010rdrdrd	data		
MOV:I.W #xx:16,Rd	3	01011rdrdrd	data (H)	data (L)	
MOV:L.B @aa:8,Rd	2	01100rdrdrd	address (L)		
MOV:L.W @aa:8,Rd	2	01101rdrdrd	address (L)		
MOV:S.B Rs,@aa:8	2	01110rsrsrs	address (L)		
MOV:S.W Rs,@aa:8	2	01111rsrsrs	address (L)		
MOV:F.B @(d:8,R6),Rd	2	10000rdrdrd	disp		
MOV:F.W @(d:8,R6),Rd	2	10001rdrdrd	disp		
MOV:F.B Rs @(d:8,R6)	2	10010rsrsrs	disp		
MOV:F.W Rs,@(d:8,R6)	2	10011rsrsrs	disp		
CMP:E.B #xx:8,Rd	2	01000rdrdrd	data		
CMP:I.W #xx:16,Rd	3	01001rdrdrd	data (H)	data (L)	

Table A-1 (c) Machine Language Coding [Special Format: Branch Instruction]

Instruction		Byte	Operation code			
			1	2	3	4
Bcc d:8	BRA (BT)	2	00100000	disp		
	BRN (BF)		00100001	disp		
	BHI		00100010	disp		
	BLS		00100011	disp		
	BCC (BHS)		00100100	disp		
	BCS (BLO)		00100101	disp		
	BNE		00100110	disp		
	BEQ		00100111	disp		
	BVC		00101000	disp		
	BVS		00101001	disp		
	BPL		00101010	disp		
	BMI		00101011	disp		
	BGE		00101100	disp		
	BLT		00101101	disp		
	BGT		00101110	disp		
	BLE		00101111	disp		
Bcc d:16	BRA (BT)	3	00110000	disp (H)	disp (L)	
	BRN (BF)		00110001	disp (H)	disp (L)	
	BHI		00110010	disp (H)	disp (L)	
	BLS		00110011	disp (H)	disp (L)	
	BCC (BHS)		00110100	disp (H)	disp (L)	
	BCS (BLO)		00110101	disp (H)	disp (L)	
	BNE		00110110	disp (H)	disp (L)	
	BEQ		00110111	disp (H)	disp (L)	
	BVC		00111000	disp (H)	disp (L)	
	BVS		00111001	disp (H)	disp (L)	
	BPL		00111010	disp (H)	disp (L)	
	BMI		00111011	disp (H)	disp (L)	
	BGE		00111100	disp (H)	disp (L)	
	BLT		00111101	disp (H)	disp (L)	
	BGT		00111110	disp (H)	disp (L)	
	BLE		00111111	disp (H)	disp (L)	
JMP @Rn		2	00010001	11010rrr		
JMP @aa:16		3	00010000	address (H)	address (L)	

Table A-1 (c) Machine Language Coding [Special Format: Branch Instruction]

Instruction		Byte	Operation code			
			1	2	3	4
JMP @(d:8,Rn)		3	00010001	11100rrr	disp	
JMP @(d:16,Rn)		4	00010001	11110rrr	disp (H)	disp (L)
BSR d:8		2	00001110	disp		
BSR d:16		3	00011110	disp (H)	disp (L)	
JSR @Rn		2	00010001	11011rrr		
JSR @aa:16		3	00011000	address (H)	address (L)	
JSR @(d:8,Rn)		3	00010001	11101rrr	disp	
JSR @(d:16,Rn)		4	00010001	11111rrr	disp (H)	disp (L)
RTS		1	00011001			
RTD #xx:8		2	00010100	data		
RTD #xx:16		3	00011100	data (H)	data (L)	
SCB/cc Rn,disp	SCB/F	3	00000001	10111rrr	disp	
	SCB/NE		00000110	10111rrr	disp	
	SCB/EQ		00000111	10111rrr	disp	
PJMP @aa:24		4	00010011	page	address (H)	address (L)
PJMP @Rn		2	00010001	11000rrr		
PJSR @aa:24		4	00000011	page	address (H)	address (L)
PJSR @Rn		2	00010001	11001rrr		
PRTS		2	00010001	00011001		
PRTD #xx:8		3	00010001	00010100	data	
PRTD #xx:16		4	00010001	00011100	data (H)	data (L)

Table A-1 (d) Machine Language Coding [Special Format: System Control Instructions]

Instruction	Byte	Operation code			
		1	2	3	4
TRAPA #xx	2	00001000	0001 #VEC		
TRAP/VS	1	00001001			
RTE	1	00001010			
LINK FP,#xx:8	2	00010111	data		
LINK FP,#xx:16	3	00011111	data (H)	data (L)	
UNLK FP	1	00001111			
SLEEP	1	00011010			
NOP	1	00000000			

A.3 Operation Code Map

Tables A-2 through A-6 are maps of the operation codes. Table A-2 shows the meaning of the first byte of the instruction code, indicating both operation codes and addressing modes. Tables A-2 through A-6 indicate the meanings of operation codes in the second and third bytes.

Table A-2 Operation Codes in Byte 1

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SCB/F See Tbl. A-6	LDM	PJSR @aa:24	#xx:8 See Tbl. A-5	#aa:8.B See Tbl. A-4	SCB/NE See Tbl. A-6	SCB/EQ See Tbl. A-6	TRAPA	TRAPVS	RTE		#xx:16 See Tbl. A-5	@aa:8.W See Tbl. A-4	BSR d:8	UNLK
1	JMP	See Tbl. A-6 *	STM	PJMP @aa:24	RTD #xx:8	@aa:16.B See Tbl. A-4		LINK #xx:8	JSR	RTS	SLEEP		RTD #xx:16	@aa:16.W See Tbl. A-4	BSR d:16	LINK #xx:16
2	BRA d:8	BRN	BHI	BLS	Bcc	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
3	BRA d:16	BRN	BHI	BLS	Bcc	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
4	CMP:E #xx:8, Rn								CMP:I #xx:16, Rn							
	R0	R1	R2	R3	R4	R5	R6	R7	R0	R1	R2	R3	R4	R5	R6	R7
5	MOV:E #xx:8, Rn								MOV:I #xx:16, Rn							
6	MOV:L.B @aa:8, Rn								MOV:L.W @aa:8, Rn							
7	MOV:S.B Rn, @aa:8								MOV:S.W Rn, @aa:8							
8	MOV:F.B @ (d:8, R6), Rn								MOV:F.W @ (d:8, R6), Rn							
9	MOV:F.B Rn, @ (d:8, R6)								MOV:F.W Rn, @ (d:8,R6)							
A	Rn					(Byte)	See Table A-3		Rn				(Word)		See Table A-3	
B	@–Rn					(Byte)	See Table A-4		@–Rn				(Word)		See Table A-4	
C	@Rn+					(Byte)	See Table A-4		@Rn+				(Word)		See Table A-4	
D	@Rn					(Byte)	See Table A-4		@Rn				(Word)		See Table A-4	
E	@(d:8,Rn)					(Byte)	See Table A-4		@(d:8,Rn)				(Word)		See Table A-4	
F	@(d:16,Rn)					(Byte)	See Table A-4		@(d:16,Rn)				(Word)		See Table A-4	

Notes:

References to tables A-3 through A-6 indicate that the instruction code has one or more additional bytes, described in those tables.

* H'11 is the first operation code byte of the following instructions:

JMP,JSR, PJSR (register indirect addressing mode)

JMP,JSR (register indirect addressing mode with displacement)

PRTS, PRTD (all addressing modes)

Table A-3 Operation Codes in Byte 2 (Axxx)

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	See Tbl. A-6*								ADD:Q #1	ADD:Q #2			ADD:Q #-1	ADD:Q #-2		
1	SWAP	EXTS	EXTU	CLR	NEG	NOT	TST	TAS	SHAL	SHAR	SHLL	SHLR	ROTL	ROTR	ROTXL	ROTXR
2	ADD R0	ADD R1	ADD R2	ADD R3	ADD R4	ADD R5	ADD R6	ADD R7	ADDS R0	ADDS R1	ADDS R2	ADDS R3	ADDS R4	ADDS R5	ADDS R6	ADDS R7
3	SUB								SUBS							
4	OR								BSET (Register indirect specification of bit number)							
5	AND								BCLR (Register indirect specification of bit number)							
6	XOR								BNOT (Register indirect specification of bit number)							
7	CMP								BTST (Register indirect specification of bit number)							
8	MOV								LDC							
9	XCH								STC							
A	ADDX								MULXU							
B	SUBX								DIVXU							
C	BSET (Immediate specification of bit number) b0	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	b13	b14	b15
D	BCLR (Immediate specification of bit number)															
E	BNOT (Immediate specification of bit number)															
F	BTST (Immediate specification of bit number)															

Note: * The operation code is in byte 3, given in table A-6.

Table A-4 Operation Codes in Byte 2 (05xx, 15xx, 0Dxx, 1Dxx, Bxxx, Cxxx, Dxxx, Exxx, Fxxx)

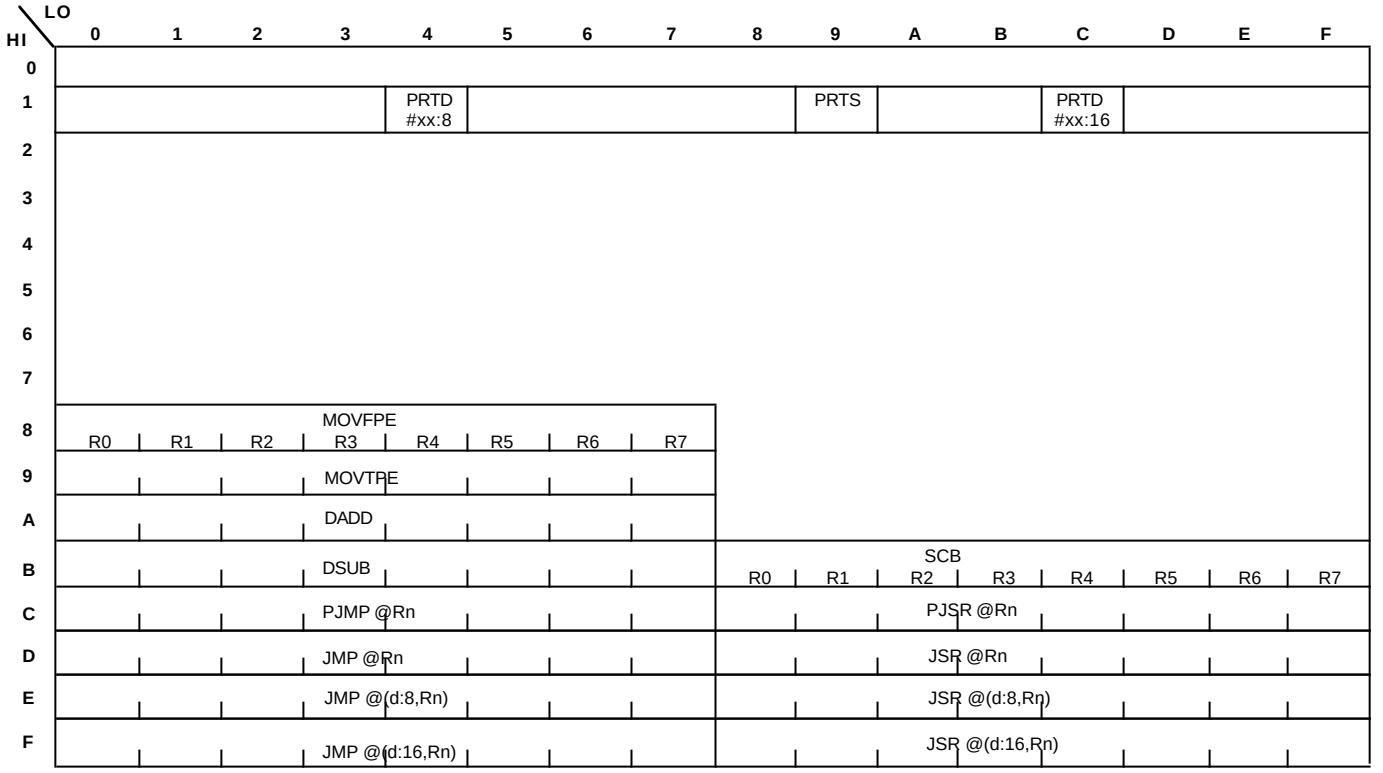
HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	See Tbl. A-6*				CMP #xx:8	CMP #xx:16	MOV #xx:8	MOV #xx:16	ADD:Q #1	ADD:Q #2			ADD:Q #-1	ADD:Q #-2		
1				CLR	NEG	NOT	TST	TAS	SHAL	SHAR	SHLL	SHLR	ROTL	ROTR	ROTXL	ROTXR
2	ADD								ADDS							
3	SUB								SUBS							
4	OR								BSET (Register indirect specification of bit number)							
5	AND								BCLR (Register indirect specification of bit number)							
6	XOR								BNOT (Register indirect specification of bit number)							
7	CMP								BTST (Register indirect specification of bit number)							
8	MOV (load)								LDC							
9	MOV (store)								STC							
A	ADDX								MULXU							
B	SUBX								DIVXU							
C	BSET (Immediate specification of bit number)															
D	BCLR (Immediate specification of bit number)															
E	BNOT (Immediate specification of bit number)															
F	BTST (Immediate specification of bit number)															

Note: * The operation code is in byte 3, given in table A-6.

Table A-5 Operation Codes in Byte 2 (04xx, 0Cxx)

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2				ADD								ADDS				
3				SUB								SUBS				
4				OR								ORC				
5				AND								ANDC				
6				XOR								XORC				
7				CMP												
8				MOV								LDC				
9																
A				ADDX								MULXU				
B				SUBX								DIVXU				
C																
D																
E																
F																

Table A-6 Operation Codes in Bytes 2 and 3 (11xx, 01xx, 06xx, 07xx, xx00xx)



A.4 Instruction Execution Cycles

Tables A-7 (1) through (6) list the number of cycles required by the CPU to execute each instruction in each addressing mode.

The meaning of the symbols in the tables is explained below. The values of I, J, and K are used to calculate the number of execution cycles when off-chip memory is accessed for an instruction fetch or operand read/write. The formulas for these calculations are given next.

A.4.1 Calculation of Instruction Execution States

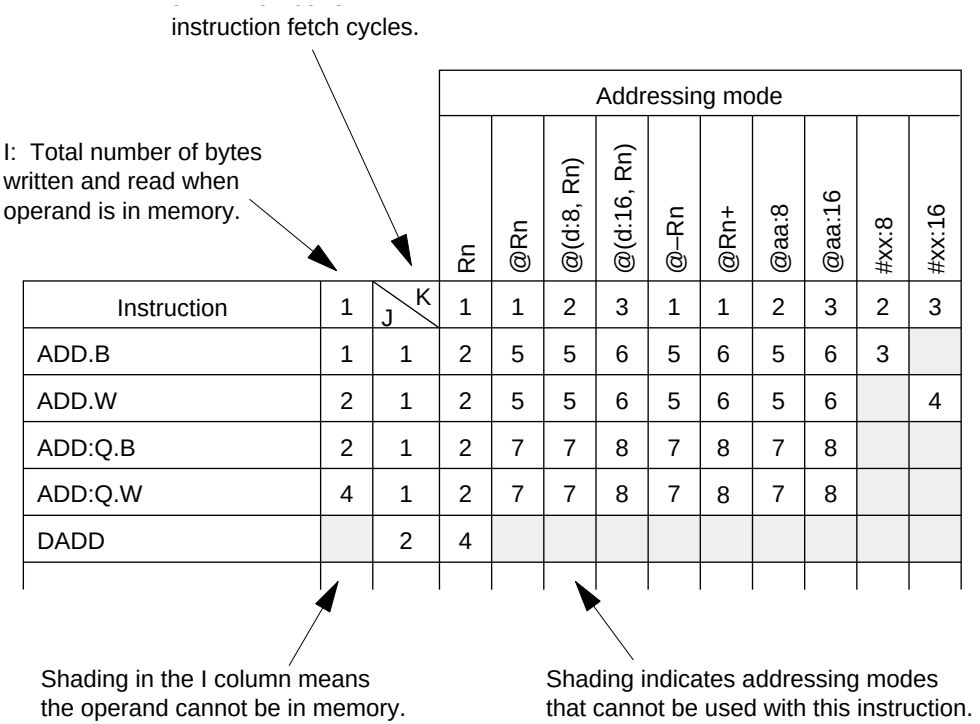
Instruction Fetch	Operand Read/Write	Number of States	
On-chip memory *1	On-chip memory	(Value given in table A-7) + (Value in table A-8)	
	On-chip memory module or off-chip memory *2	Byte	(Value in table A-7) + (Value in table A-8) + I
		Word	((Value in table A-7) + (Value in table A-8) + 2I
Off-chip memory *2	On-chip memory	(Value given in table A-7) + 2(J + K)	
	On-chip supporting module or off-chip memory *2	Byte	(Value in table A-7) + I + 2(J + K)
		Word	((Value in table A-7) + 2(I + J + K)

Notes: *1. When the instruction is fetched from on-chip memory (ROM or RAM), the number of execution states varies by 1 or 2 depending of whether the instruction is stored at an even or odd address. This difference must be noted when software is used for timing, and in other cases in which the exact number of states is important.

*2. If wait states are inserted in access to external memory, add the necessary number of cycles.

A.4.2 Tables of Instruction Execution Cycles

Tables A-7 (1) through (6) should be read as shown below:



- **Examples of Calculation of Number of States Required for Execution**

(Example 1) Instruction fetch from on-chip memory

Operand Read/Write	Start Addr.	Assembler Notation			Table A-7 + Table A-8	Number of States
		Address	Code	Mnemonic		
On-chip memory or general register	Even	H'0100	H'D821	ADD @R0, R1	5 + 1	6
	Odd	H'0101	H'D821	ADD @R0, R1	5 + 0	5

(Example 2) Instruction fetch from on-chip memory

Operand Read/Write	Start Addr.	Assembler Notation			Table A-7 + Table A-8 + 2I	Number of States
		Address	Code	Mnemonic		
On-chip supporting module or external memory (word)	Even	H'FC00	H'11D8	JSR @R0	9 + 0 + 2 × 2	13
	Odd	H'FC01	H'11D8	JSR @R0	9 + 1 + 2 × 2	14

(Example 3) Instruction fetch from external memory

Operand Read/Write	Assembler Notation			Table A-7 + 2(J + K)	Number of States
	Address	Code	Mnemonic		
On-chip memory or general register	H'9002	H'D821	ADD @R0, R1	5 + 2 × (1 + 1)	9

Table A-7 Instruction Execution Cycles (1)

			Addressing mode									
			Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
Instruction	1	J \ K	1	1	2	3	1	1	2	3	2	3
ADD:G.B	1	1	2	5	5	6	5	6	5	6	3	
ADD:G.W	2	1	2	5	5	6	5	6	5	6		4
ADD:Q.B	2	1	2	7	7	8	7	8	7	8		
ADD:Q.W	4	1	2	7	7	8	7	8	7	8		
ADDS.B	1	1	3	5	5	6	5	6	5	6	3	
ADDS.W	2	1	3	5	5	6	5	6	5	6		4
ADDX.B	1	1	2	5	5	6	5	6	5	6	3	
ADDX.W	2	1	2	5	5	6	5	6	5	6		4
AND.B	1	1	2	5	5	6	5	6	5	6	3	
AND.W	2	1	2	5	5	6	5	6	5	6		4
ANDC		1									5	9
BCLR.B	2	1	4	7	7	8	7	8	7	8		
BCLR.W	4	1	4	7	7	8	7	8	7	8		
BNOT.B	2	1	4	7	7	8	7	8	7	8		
BNOT.W	4	1	4	7	7	8	7	8	7	8		
BSET.B	2	1	4	7	7	8	7	8	7	8		
BSET.W	4	1	4	7	7	8	7	8	7	8		
BTST.B	1	1	3	5	5	6	5	6	5	6		
BTST.W	2	1	3	5	5	6	5	6	5	6		
CLR.B	1	1	2	5	5	6	5	6	5	6		
CLR.W	2	1	2	5	5	6	5	6	5	6		
CMP:G.B	1	1	2	5	5	6	5	6	5	6	3	
CMP:G.W	2	1	2	5	5	6	5	6	5	6		4
CMP:G.B #XX:8, <EA>	1	2		6	6	7	6	7	6	7		
CMP:G.B #XX:16, <EA>	2	3		7	7	8	7	8	7	8		

Table A-7 Instruction Execution Cycles (2)

			Addressing mode									
			Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
Instruction	1	J \ K	1	1	2	3	1	1	2	3	2	3
CMP:E #xx:8, Rd		0									2	
CMP:I #xx:16, Rd		0										3
DADD		2	4									
DIVXU.B	1	1	20	23	23	24	23	24	23	24	21	
DIVXU.W	2	1	26	29	29	30	29	30	29	30		28
DSUB		2	4									
EXTS		1	3									
EXTU		1	3									
LDC.B	1	1	3	6	6	7	6	7	6	7	4	
LDC.W	2	1	4	7	7	8	7	8	7	8		6
MOV.B	1	1	2	5	5	6	5	6	5	6	3	
MOV.W	2	1	2	5	5	6	5	6	5	6		4
MOV.B #xx:8, <EA>	1	2		7	7	8	7	8	7	8		
MOV.B #xx:16, <EA>	2	3		8	8	9	8	9	8	9		
MOV:E #xx:8, Rd		0									2	
MOV:I #xx:8, Rd		0										3
MOV:L.B @aa:8, Rd	1	0							5			
MOV:L.W @aa:8, Rd	2	0							5			
MOV:S.B Rd ,@aa:8	1	0							5			
MOV:S.W Rd ,@aa:8	2	0							5			
MOV:F.B @(d:8, R6), Rd	1	0			5							
MOV:F.W @(d:8, R6), Rd	2	0			5							
MOV:F.B Rd, @(d:8, R6)	1	0			5							
MOV:F.W Rd, @(d:8, R6)	2	0			5							

Table A-7 Instruction Execution Cycles (3)

Instruction	1	J \ K	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
			1	1	2	3	1	1	2	3	2	3
MOVFPE *	0	2		13—20	13—20	14—21	13—20	14—21	13—20	14—21		
MOVTPE *	0	2		13—20	13—20	14—21	13—20	14—21	13—20	14—21		
MULXU.B	1	1	16	19	19	20	19	20	19	20	18	
MULXU.W	2	1	23	25	25	26	25	26	25	26		25
NEG.B	2	1	2	7	7	8	7	8	7	8		
NEG.W	4	1	2	7	7	8	7	8	7	8		
NOT.B	2	1	2	7	7	8	7	8	7	8		
NOT.W	4	1	2	7	7	8	7	8	7	8		
OR.B	1	1	2	5	5	6	5	6	5	6	3	
OR.W	2	1	2	5	5	6	5	6	5	6		4
ORC		1									5	9
ROTL.B	2	1	2	7	7	8	7	8	7	8		
ROTL.W	4	1	2	7	7	8	7	8	7	8		
ROTR.B	2	1	2	7	7	8	7	8	7	8		
ROTR.W	4	1	2	7	7	8	7	8	7	8		
ROTXL.B	2	1	2	7	7	8	7	8	7	8		
ROTXL.W	4	1	3	7	7	8	7	8	7	8		
ROTXR.B	2	1	2	7	7	8	7	8	7	8		
ROTXR.W	4	1	2	7	7	8	7	8	7	8		
SHAL.B	2	1	2	7	7	8	7	8	7	8		
SHAL.W	4	1	2	7	7	8	7	8	7	8		
SHAR.B	2	1	2	7	7	8	7	8	7	8		
SHAR.W	4	1	2	7	7	8	7	8	7	8		
SHLL.B	2	1	2	7	7	8	7	8	7	8		
SHLL.W	4	1	2	7	7	8	7	8	7	8		

* MOVFPE and MOVTPE are executed synchronous with the E-clock, so the number of execution states will change depending on timing of the execution.

Table A-7 Instruction Execution Cycles (4)

Instruction	1	J \ K	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
			1	1	2	3	1	1	2	3	2	3
SHLR.B	2	1	2	7	7	8	7	8	7	8		
SHLR.W	4	1	2	7	7	8	7	8	7	8		
STC.B	1	1	2	7	7	8	7	8	7	8		
STC.W	2	1	2	7	7	8	7	8	7	8		
SUB.B	1	1	2	5	5	6	5	6	5	6	3	
SUB.W	2	1	2	5	5	6	5	6	5	6		4
SUBS.B	1	1	3	5	5	6	5	6	5	6	3	
SUBS.W	2	1	3	5	5	6	5	6	5	6		4
SUBX.B	1	1	2	5	5	6	5	6	5	6	3	
SUBX.W	2	1	2	5	5	6	5	6	5	6		4
SWAP		1	3									
TAS	2	1	4	7	7	8	7	8	7	8		
TST.B	1	1	2	5	5	6	5	6	5	6		
TST.W	2	1	2	5	5	6	5	6	5	6		
XCH		1	4									
XOR.B	1	1	2	5	5	6	5	6	5	6	3	
XOR.W	4	1	4	5	5	6	5	6	5	6		4
XORC		1									5	9

		*	J \ K	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
DIVXU.B	Zero divide, minimum mode	6/7	1	20	23	23	24	23	24	23	24	21	
DIVXU.B	Zero divide, maximum mode	10/11	1	25	28	28	29	28	29	28	29	21	
DIVXU.W	Zero divide, minimum mode	6/8	1	20	23	23	24	23	24	23	24		27
DIVXU.W	Zero divide, maximum mode	10/12	1	25	28	28	29	28	29	28	29		27
DIVXU.B	Overflow	1	1	8	11	11	12	11	12	11	12	9	
DIVXU.W	Overflow	2	1	8	11	11	12	11	12	11	12		10

* For register and immediate operands

For memory operand

Table A-7 Instruction Execution Cycles (5)

Instruction	(Condition)	Execution Cycles	I	J + K
Bcc d:8	Condition false, branch not taken	3		2
	Condition true, branch taken	7		5
Bcc d:16	Condition false, branch not taken	3		3
	Condition true, branch taken	7		6
BSR	d:8	9	2	4
	d:16	9	2	5
JMP	@aa:16	7		5
	@Rn	6		5
	@(d:8, Rn)	7		5
	@(d:16, Rn)	8		6
JSR	@aa:16	9	2	5
	@Rn	9	2	5
	@(d:8, Rn)	9	2	5
	@(d:16, Rn)	10	2	6
LDM		6 + 4n*	2n	2
LINK	#xx:8	6	2	2
	#xx:16	7	2	3
NOP		2		1
RTD	#xx:8	9	2	4
	#xx:16	9	2	5
RTE	Minimum mode	13	4	4
	Maximum mode	15	6	4
RTS		8	2	4
SCB	Condition false, branch not taken	3		3
	Count = –1, branch not taken	4		3
	Other than the above, branch taken	8		6
SLEEP	Cycles preceding transition to power-down mode	2		0
STM		6 + 3n*	2n	2

* n is the number of registers specified in the register list.

Table A-7 Instruction Execution Cycles (6)

Instruction	(Condition)	Execution Cycles	I	J + K
TRAPA	Minimum mode	17	6	4
	Maximum mode	22	10	4
TRAP/VS	V = 0, trap not taken	3		1
	V = 1, trap taken, minimum mode	18	6	4
	V = 1, trap taken, maximum mode	23	10	4
UNLK		5	2	1
PJMP	@aa:24	9		6
	@Rn	8		5
PJSR	@aa:24	15	4	6
	@Rn	13	4	5
PRTS		12	4	5
PRTD	#xx:8	13	4	5
	#xx:16	13	4	6

Table A-8 (a) Adjusted Value (Branch Instruction)

Instruction	Address	Adjusted Value
BSR, JMP, JSR, RTS, RTD, RTE	even	0
TRAPA, PJMP, PJSR, PRTS, PRTD	odd	1
Bcc, SCB, TRAP/VS (When branches)	even	0
	odd	1

Table A-8 (b) Adjusted Value (Other Instructions by Addressing Modes)

Instruction	Start address	Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
MOV.B #xx:8, <EA>	even		1	1	1	1	1	1	1		
MOVTPE, MOVFPE	odd		1	1	1	1	1	1	1		
MOV.W #xx:16, <EA>	even		2	0	2	2	2	0	2		
	odd		0	2	0	0	0	2	0		
Instruction other than above	even	0	1	0	1	1	1	0	1	0	0
	odd	0	0	1	0	0	0	1	0	0	0