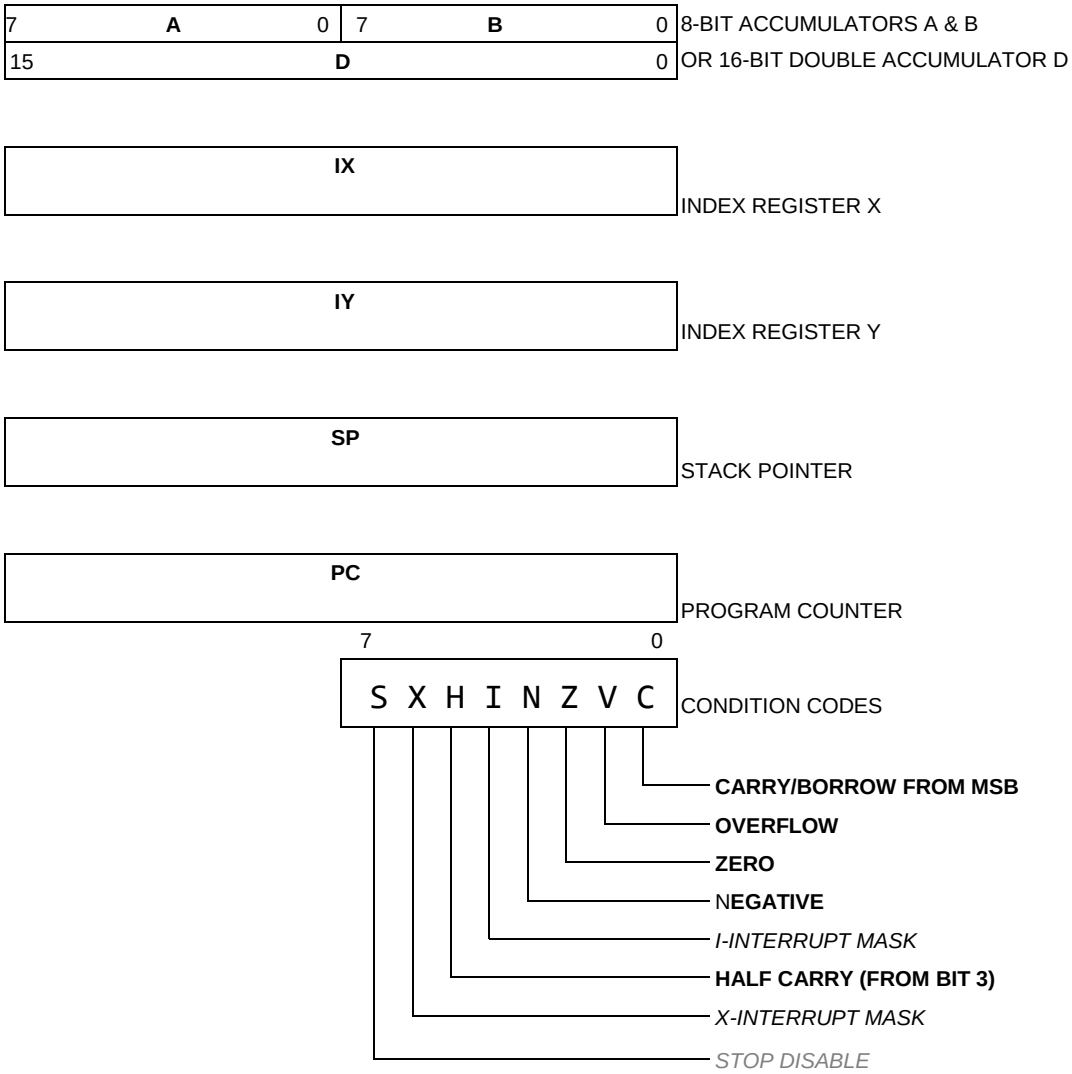
M68HC11E Series Programming Model



Crystal Dependent Timer Summary

	Selected Crystal	Common XTAL Frequencies		
		4.0 MHz	**8.0 MHz**	12.0 MHz
CPU Clock	**(E)**	1.0 MHz	**2.0 MHz**	3.0 MHz
Cycle Time	**(1/E)**	1000 ns	**500 ns**	333 ns

Interrupt Vector Assignments

Vector Address	Interrupt Source	CCR Mask Bit	Local Mask
FFC0, C1 – FFD4, D5	Reserved	—	—
FFD6, D7	SCI serial system[1] • SCI receive data register full • SCI receiver overrun • SCI transmit data register empty • SCI transmit complete • SCI idle line detect	I	 RIE RIE TIE TCIE ILIE
FFD8, D9	SPI serial transfer complete	I	SPIE
FFDA, DB	Pulse accumulator input edge	I	PAII
FFDC, DD	Pulse accumulator overflow	I	PAOVI
FFDE, DF	Timer overflow	I	TOI
FFE0, E1	Timer input capture 4/output compare 5	I	I4/O5I
FFE2, E3	Timer output compare 4	I	OC4I
FFE4, E5	Timer output compare 3	I	OC3I
FFE6, E7	Timer output compare 2	I	OC2I
FFE8, E9	Timer output compare 1	I	OC1I
FFEA, EB	Timer input capture 3	I	IC3I
FFEC, ED	Timer input capture 2	I	IC2I
FFEE, EF	Timer input capture 1	I	IC1I
FFF0, F1	Real-time interrupt	I	RTII
FFF2, F3	IRQ/ (external pin)	I	None
FFF4, F5	XIRQ/ (external pin)	X	None
FFF6, F7	Software interrupt	None	None
FFF8, F9	Illegal opcode trap	None	None
FFFA, FB	COP failure	None	NOCOP
FFFC, FD	Clock monitor fail	None	CME
FFFE, FF	RESET	None	None

NOTES:

1. Interrupts generated by SCI; read SCSR to determine source. Refer to HPRIO register to determine priority of interrupt.

Opcode Maps Page 1

DIR

MSB / LSB		INH	INH	REL	INH	ACCA	ACCB	IND,X	EXT	ACCA IMM	ACCA DIR	ACCA IND,X	ACCA EXT	ACCB IMM	ACCB DIR	ACCB IND,X	ACCB EXT	
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0	TEST	SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2	IDIV	BRSET	BHI	PULA					SBC								2
0011	3	EDIV	BRCLR	BLS	PULB	COM				SUBD				ADDD				3
0100	4	LSRD	BSET	BCC	DES	LSR				AND								4
0101	5	ASLD	BCLR	BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LDA								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX	PAGE 2	BVC	PULX	ASL				EOR								8
1001	9	DEX	DAA	BVS	RTS	ROL				ADC								9
1010	A	CLV	PAGE 3	BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	RTI					ADD								B
1100	C	CLC	BSET	BGE	PSHX	INC				CPX				LDD				C
1101	D	SEC	BCLR	BLT	MUL	TST				BSR	JSR			PAGE 4	STD			D
1110	E	CLI	BRSET	BGT	WAI			JMP		LDS				LDX				E
1111	F	SEI	BRCLR	BLE	SWI	CLR				XGDX	STS			STOP	STX			F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

IND,X

Page 2 (18XX)

										ACCA				ACCB				
		INH			INH			IND,Y		IMM	DIR	IND,X	EXT	IMM	DIR	IND,X	EXT	
MSB / LSB		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0				TSY			NEG				SUB				SUB		0
0001	1											CMP				CMP		1
0010	2											SBC				SBC		2
0011	3							COM				SUBD				ADDD		3
0100	4							LSR				AND				AND		4
0101	5				TYS							BIT				BIT		5
0110	6							ROR				LDA				LDA		6
0111	7							ASR				STA				STA		7
1000	8	INY			PULY			ASL				EOR				EOR		8
1001	9	DEY						RDL				ADC				ADC		9
1010	A				ABY			DEC				ORA				ORA		A
1011	B											ADD				ADD		B
1100	C		BSET		PSHY			INC		CPY						LDD		C
1101	D		BCLR					TST				JSR				STD		D
1110	E		BRSET					JMP				LDS		LDY				E
1111	F		BRCLR					CLR		XGDY		STS			STY			F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

IND,Y

Page 3 (1AXX)

										ACCA				ACCB			
										IMM	DIR	IND,X	EXT			IND,X	
MSB / LSB		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0																	0
0001	1																	1
0010	2																	2
0011	3									CPD								3
0100	4																	4
0101	5																	5
0110	6																	6
0111	7																	7
1000	8																	8
1001	9																	9
1010	A																	A
1011	B																	B
1100	C											CPY						C
1101	D																	D
1110	E															LDY		E
1111	F															STY		F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Page 4 (CDXX)

										ACCA				ACCB			
												IND,Y				IND,Y	
LSB \ MSB		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0																	0
0001	1																	1
0010	2																	2
0011	3											CPD						3
0100	4																	4
0101	5																	5
0110	6																	6
0111	7																	7
1000	8																	8
1001	9																	9
1010	A																	A
1011	B																	B
1100	C											CPX						C
1101	D																	D
1110	E															LDX		E
1111	F															STX		F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Branches

Simple Branches

Mnemonic	Opcode	Cycles
BRA	20	3
BRN	21	3
BSR	8D	7

Simple Conditional Branches

Test	True		False	
	Instruction	Opcode	Instruction	Opcode
N = 1	BMI	2B	BPL	2A
Z = 1	BEQ	27	BNE	26
V = 1	BVS	29	BVC	28
C = 1	BCS	25	BCC	24

Signed Conditional Branches

Test	True		False	
	Instruction	Opcode	Instruction	Opcode
r > m	BGT	2E	BLE	2F
r ≥ m	BGE	2C	BLT	2D
r = m	BEQ	27	BNE	26
r ≤ m	BLE	2F	BGT	2E
r < m	BLT	2D	BGE	2C

Unsigned Conditional Branches

Test	True		False	
	Instruction	Opcode	Instruction	Opcode
r > m	BHI	22	BLS	23
r ≥ m	BHS/BCC	24	BL0/BCS	25
r = m	BEQ	27	BNE	26
r ≤ m	BLS	23	BHI	22
r < m	BLO/BCS	25	BHS/BCC	24

Bit Manipulation Branches

BRCLR

Branch if all selected bits are clear (opcode) (operand addr) (mask) (rel offset)
M • mm = 0? M = operand in memory; mm = mask

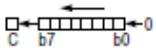
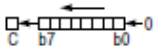
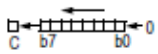
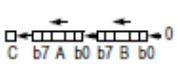
BRSET

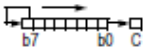
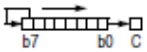
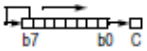
Branch if all selected bits are set (opcode) (operand addr) (rel offset)

$(\overline{M})$ • mm = 0? M = operand in memory; mm = mask

Instruction Set

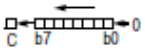
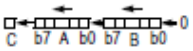
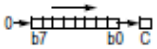
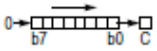
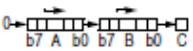
This table shows all the M68HC11 instructions in all possible addressing modes. For each instruction, the table shows the operand construction, the number of machine code bytes, and execution time in CPU E-clock cycles

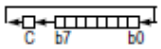
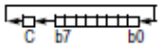
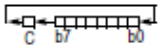
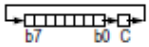
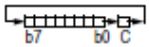
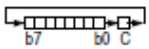
Mnemonic	Operation	Description	Addressing Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
ABA	Add Accumulators	A + B ⇒ A	INH	1B	—	2	—	—	Δ	—	Δ	Δ	Δ	Δ
ABX	Add B to X	IX + (00 : B) ⇒ IX	INH	3A	—	3	—	—	—	—	—	—	—	—
ABY	Add B to Y	IY + (00 : B) ⇒ IY	INH	18 3A	—	4	—	—	—	—	—	—	—	—
ADCA (opr)	Add with Carry to A	A + M + C ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	89 99 B9 A9 18 A9	ii dd hh ll ff ff	2 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
ADCB (opr)	Add with Carry to B	B + M + C ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C9 D9 F9 E9 18 E9	ii dd hh ll ff ff	2 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
ADDA (opr)	Add Memory to A	A + M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	8B 9B BB AB 18 AB	ii dd hh ll ff ff	2 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
ADDB (opr)	Add Memory to B	B + M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	CB DB FB EB 18 EB	ii dd hh ll ff ff	2 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
ADDD (opr)	Add 16-Bit to D	D + (M : M + 1) ⇒ D	IMM DIR EXT IND,X IND,Y	C3 D3 F3 E3 18 E3	jj kk dd hh ll ff ff	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ANDA (opr)	AND A with Memory	A • M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	84 94 B4 A4 18 A4	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
ANDB (opr)	AND B with Memory	B • M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C4 D4 F4 E4 18 E4	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
ASL (opr)	Arithmetic Shift Left		EXT IND,X IND,Y	78 68 18 68	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ASLA	Arithmetic Shift Left A		A INH	48	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLB	Arithmetic Shift Left B		B INH	58	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLD	Arithmetic Shift Left D		INH	05	—	3	—	—	—	—	Δ	Δ	Δ	Δ

Mnemonic	Operation	Description	Addressing Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
ASR	Arithmetic Shift Right		EXT IND,X IND,Y	77 67 18 67	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ASRA	Arithmetic Shift Right A		A INH	47	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRB	Arithmetic Shift Right B		B INH	57	—	2	—	—	—	—	Δ	Δ	Δ	Δ
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	3	—	—	—	—	—	—	—	—
BCLR (opr) (msk)	Clear Bit(s)	$M \bullet (\overline{mm}) \Rightarrow M$	DIR IND,X IND,Y	15 1D 18 1D	dd mm ff mm ff mm	6 7 8	—	—	—	—	Δ	Δ	0	—
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	3	—	—	—	—	—	—	—	—
BEQ (rel)	Branch if = Zero	? Z = 1	REL	27	rr	3	—	—	—	—	—	—	—	—
BGE (rel)	Branch if Δ Zero	? $N \oplus V = 0$	REL	2C	rr	3	—	—	—	—	—	—	—	—
BGT (rel)	Branch if > Zero	? $Z + (N \oplus V) = 0$	REL	2E	rr	3	—	—	—	—	—	—	—	—
BHI (rel)	Branch if Higher	? C + Z = 0	REL	22	rr	3	—	—	—	—	—	—	—	—
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	3	—	—	—	—	—	—	—	—
BITA (opr)	Bit(s) Test A with Memory	A • M	A IMM A DIR A EXT A IND,X A IND,Y	85 95 B5 A5 18 A5	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
BITB (opr)	Bit(s) Test B with Memory	B • M	B IMM B DIR B EXT B IND,X B IND,Y	C5 D5 F5 E5 18 E5	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
BLE (rel)	Branch if Δ Zero	? $Z + (N \oplus V) = 1$	REL	2F	rr	3	—	—	—	—	—	—	—	—
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	3	—	—	—	—	—	—	—	—
BLS (rel)	Branch if Lower or Same	? C + Z = 1	REL	23	rr	3	—	—	—	—	—	—	—	—
BLT (rel)	Branch if < Zero	? $N \oplus V = 1$	REL	2D	rr	3	—	—	—	—	—	—	—	—
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	3	—	—	—	—	—	—	—	—
BNE (rel)	Branch if not = Zero	? Z = 0	REL	26	rr	3	—	—	—	—	—	—	—	—
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	3	—	—	—	—	—	—	—	—
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	3	—	—	—	—	—	—	—	—
BRCLR(opr) (msk) (rel)	Branch if Bit(s) Clear	? M • mm = 0	DIR IND,X IND,Y	13 1F 18 1F	dd mm rr ff mm rr ff mm rr	6 7 8	—	—	—	—	—	—	—	—
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	3	—	—	—	—	—	—	—	—
BRSET(opr) (msk) (rel)	Branch if Bit(s) Set	? $(\overline{M}) \bullet mm = 0$	DIR IND,X IND,Y	12 1E 18 1E	dd mm rr ff mm rr ff mm rr	6 7 8	—	—	—	—	—	—	—	—
BSET (opr) (msk)	Set Bit(s)	$M + mm \Rightarrow M$	DIR IND,X IND,Y	14 1C 18 1C	dd mm ff mm ff mm	6 7 8	—	—	—	—	Δ	Δ	0	—
BSR (rel)	Branch to Subroutine	See Figure 3–2	REL	8D	rr	6	—	—	—	—	—	—	—	—
BVC (rel)	Branch if Overflow Clear	? V = 0	REL	28	rr	3	—	—	—	—	—	—	—	—

Mnemonic	Operation	Description	Addressing Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
BVS (rel)	Branch if Overflow Set	? V = 1	REL	29	rr	3	—	—	—	—	—	—	—	—
CBA	Compare A to B	A – B	INH	11	—	2	—	—	—	—	Δ	Δ	Δ	Δ
CLC	Clear Carry Bit	0 ⇒ C	INH	0C	—	2	—	—	—	—	—	—	—	0
CLI	Clear Interrupt Mask	0 ⇒ I	INH	0E	—	2	—	—	—	0	—	—	—	—
CLR (opr)	Clear Memory Byte	0 ⇒ M	EXT IND,X IND,Y	7F 6F 18 6F	hh ll ff ff	6 6 7	—	—	—	—	0	1	0	0
CLRA	Clear Accumulator A	0 ⇒ A	A INH	4F	—	2	—	—	—	—	0	1	0	0
CLRB	Clear Accumulator B	0 ⇒ B	B INH	5F	—	2	—	—	—	—	0	1	0	0
CLV	Clear Overflow Flag	0 ⇒ V	INH	0A	—	2	—	—	—	—	—	—	0	—
CMPA (opr)	Compare A to Memory	A – M	A IMM A DIR A EXT A IND,X A IND,Y	81 91 B1 A1 18 A1	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
CMPB (opr)	Compare B to Memory	B – M	B IMM B DIR B EXT B IND,X B IND,Y	C1 D1 F1 E1 18 E1	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
COM (opr)	Ones Complement Memory Byte	$FF – M ⇒ M	EXT IND,X IND,Y	73 63 18 63	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	0	1
COMA	Ones Complement A	$FF – A ⇒ A	A INH	43	—	2	—	—	—	—	Δ	Δ	0	1
COMB	Ones Complement B	$FF – B ⇒ B	B INH	53	—	2	—	—	—	—	Δ	Δ	0	1
CPD (opr)	Compare D to Memory 16-Bit	D – M : M + 1	IMM DIR EXT IND,X IND,Y	1A 83 1A 93 1A B3 1A A3 CD A3	jj kk dd hh ll ff ff	5 6 7 7 7	—	—	—	—	Δ	Δ	Δ	Δ
CPX (opr)	Compare X to Memory 16-Bit	IX – M : M + 1	IMM DIR EXT IND,X IND,Y	8C 9C BC AC CD AC	jj kk dd hh ll ff ff	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
CPY (opr)	Compare Y to Memory 16-Bit	IY – M : M + 1	IMM DIR EXT IND,X IND,Y	18 8C 18 9C 18 BC 1A AC 18 AC	jj kk dd hh ll ff ff	5 6 7 7 7	—	—	—	—	Δ	Δ	Δ	Δ
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19	—	2	—	—	—	—	Δ	Δ	Δ	Δ
DEC (opr)	Decrement Memory Byte	M – 1 ⇒ M	EXT IND,X IND,Y	7A 6A 18 6A	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	—
DECA	Decrement Accumulator A	A – 1 ⇒ A	A INH	4A	—	2	—	—	—	—	Δ	Δ	Δ	—
DECB	Decrement Accumulator B	B – 1 ⇒ B	B INH	5A	—	2	—	—	—	—	Δ	Δ	Δ	—

Mnemonic	Operation	Description	Addressing Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
DES	Decrement Stack Pointer	SP – 1 ⇒ SP	INH	34	—	3	—	—	—	—	—	—	—	—
DEX	Decrement Index Register X	IX – 1 ⇒ IX	INH	09	—	3	—	—	—	—	—	Δ	—	—
DEY	Decrement Index Register Y	IY – 1 ⇒ IY	INH	18 09	—	4	—	—	—	—	—	Δ	—	—
EORA (opr)	Exclusive OR A with Memory	A ⊕ M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	88 98 B8 A8 18 A8	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
EORB (opr)	Exclusive OR B with Memory	B ⊕ M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C8 D8 F8 E8 18 E8	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
FDIV	Fractional Divide 16 by 16	D / IX ⇒ IX; r ⇒ D	INH	03	—	41	—	—	—	—	—	Δ	Δ	Δ
IDIV	Integer Divide 16 by 16	D / IX ⇒ IX; r ⇒ D	INH	02	—	41	—	—	—	—	—	Δ	0	Δ
INC (opr)	Increment Memory Byte	M + 1 ⇒ M	EXT IND,X IND,Y	7C 6C 18 6C	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	—
INCA	Increment Accumulator A	A + 1 ⇒ A	A INH	4C	—	2	—	—	—	—	Δ	Δ	Δ	—
INCB	Increment Accumulator B	B + 1 ⇒ B	B INH	5C	—	2	—	—	—	—	Δ	Δ	Δ	—
INS	Increment Stack Pointer	SP + 1 ⇒ SP	INH	31	—	3	—	—	—	—	—	—	—	—
INX	Increment Index Register X	IX + 1 ⇒ IX	INH	08	—	3	—	—	—	—	—	Δ	—	—
INY	Increment Index Register Y	IY + 1 ⇒ IY	INH	18 08	—	4	—	—	—	—	—	Δ	—	—
JMP (opr)	Jump	See Figure 3–2	EXT IND,X IND,Y	7E 6E 18 6E	hh ll ff ff	3 3 4	—	—	—	—	—	—	—	—
JSR (opr)	Jump to Subroutine	See Figure 3–2	DIR EXT IND,X IND,Y	9D BD AD 18 AD	dd hh ll ff ff	5 6 6 7	—	—	—	—	—	—	—	—
LDAA (opr)	Load Accumulator A	M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	86 96 B6 A6 18 A6	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
LDAB (opr)	Load Accumulator B	M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C6 D6 F6 E6 18 E6	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
LDD (opr)	Load Double Accumulator D	M ⇒ A,M + 1 ⇒ B	IMM DIR EXT IND,X IND,Y	CC DC FC EC 18 EC	jj kk dd hh ll ff ff	3 4 5 5 6	—	—	—	—	Δ	Δ	0	—

Mnemonic	Operation	Description	Addressing Mode		Instruction				Condition Codes							
					Opcode		Operand	Cycles	S	X	H	I	N	Z	V	C
LDS (opr)	Load Stack Pointer	M : M + 1 ⇒ SP		IMM		8E	jj kk	3	—	—	—	—	Δ	Δ	0	—
				DIR		9E	dd	4								
				EXT		BE	hh ll	5								
				IND,X		AE	ff	5								
				IND,Y	18	AE	ff	6								
LDX (opr)	Load Index Register X	M : M + 1 ⇒ IX		IMM		CE	jj kk	3	—	—	—	—	Δ	Δ	0	—
				DIR		DE	dd	4								
				EXT		FE	hh ll	5								
				IND,X		EE	ff	5								
				IND,Y	CD	EE	ff	6								
LDY (opr)	Load Index Register Y	M : M + 1 ⇒ IY		IMM	18	CE	jj kk	4	—	—	—	—	Δ	Δ	0	—
				DIR	18	DE	dd	5								
				EXT	18	FE	hh ll	6								
				IND,X	1A	EE	ff	6								
				IND,Y	18	EE	ff	6								
LSL (opr)	Logical Shift Left			EXT		78	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
				IND,X		68	ff	6								
				IND,Y	18	68	ff	7								
LSLA	Logical Shift Left A		A	INH		48	—	2	—	—	—	—	Δ	Δ	Δ	Δ
LSLB	Logical Shift Left B		B	INH		58	—	2	—	—	—	—	Δ	Δ	Δ	Δ
LSLD	Logical Shift Left Double			INH		05	—	3	—	—	—	—	Δ	Δ	Δ	Δ
LSR (opr)	Logical Shift Right			EXT		74	hh ll	6	—	—	—	—	0	Δ	Δ	Δ
				IND,X		64	ff	6								
				IND,Y	18	64	ff	7								
LSRA	Logical Shift Right A		A	INH		44	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRB	Logical Shift Right B		B	INH		54	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRD	Logical Shift Right Double			INH		04	—	3	—	—	—	—	0	Δ	Δ	Δ
MUL	Multiply 8 by 8	A * B ⇒ D		INH		3D	—	10	—	—	—	—	—	—	—	Δ
NEG (opr)	Two's Complement Memory Byte	0 – M ⇒ M		EXT		70	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
				IND,X		60	ff	6								
				IND,Y	18	60	ff	7								
NEGA	Two's Complement A	0 – A ⇒ A	A	INH		40	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Two's Complement B	0 – B ⇒ B	B	INH		50	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NOP	No operation	No Operation		INH		01	—	2	—	—	—	—	—	—	—	—
ORAA (opr)	OR Accumulator A (Inclusive)	A + M ⇒ A	A	IMM		8A	ii	2	—	—	—	—	Δ	Δ	0	—
			A	DIR		9A	dd	3								
			A	EXT		BA	hh ll	4								
			A	IND,X		AA	ff	4								
			A	IND,Y	18	AA	ff	5								
ORAB (opr)	OR Accumulator B (Inclusive)	B + M ⇒ B	B	IMM		CA	ii	2	—	—	—	—	Δ	Δ	0	—
			B	DIR		DA	dd	3								
			B	EXT		FA	hh ll	4								
			B	IND,X		EA	ff	4								
			B	IND,Y	18	EA	ff	5								

Mnemonic	Operation	Description	Addressing Mode		Instruction			Condition Codes							
					Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
PSHA	Push A onto Stack	A ⇒ Stk,SP = SP – 1	A	INH	36	—	3	—	—	—	—	—	—	—	—
PSHB	Push B onto Stack	B ⇒ Stk,SP = SP – 1	B	INH	37	—	3	—	—	—	—	—	—	—	—
PSHX	Push X onto Stack (Lo First)	IX ⇒ Stk,SP = SP – 2		INH	3C	—	4	—	—	—	—	—	—	—	—
PSHY	Push Y onto Stack (Lo First)	IY ⇒ Stk,SP = SP – 2		INH	18 3C	—	5	—	—	—	—	—	—	—	—
PULA	Pull A from Stack	SP = SP + 1, A ⇐ Stk	A	INH	32	—	4	—	—	—	—	—	—	—	—
PULB	Pull B from Stack	SP = SP + 1, B ⇐ Stk	B	INH	33	—	4	—	—	—	—	—	—	—	—
PULX	Pull X From Stack (Hi First)	SP = SP + 2, IX ⇐ Stk		INH	38	—	5	—	—	—	—	—	—	—	—
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY ⇐ Stk		INH	18 38	—	6	—	—	—	—	—	—	—	—
ROL (opr)	Rotate Left			EXT IND,X IND,Y	79 69 18 69	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ROLA	Rotate Left A		A	INH	49	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLB	Rotate Left B		B	INH	59	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROR (opr)	Rotate Right			EXT IND,X IND,Y	76 66 18 66	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
RORA	Rotate Right A		A	INH	46	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORB	Rotate Right B		B	INH	56	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RTI	Return from Interrupt	See Figure 3–2		INH	3B	—	12	Δ	↓	Δ	Δ	Δ	Δ	Δ	Δ
RTS	Return from Subroutine	See Figure 3–2		INH	39	—	5	—	—	—	—	—	—	—	—
SBA	Subtract B from A	A – B ⇒ A		INH	10	—	2	—	—	—	—	Δ	Δ	Δ	Δ
SBCA (opr)	Subtract with Carry from A	A – M – C ⇒ A	A A A A A	IMM DIR EXT IND,X IND,Y	82 92 B2 A2 18 A2	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SBCB (opr)	Subtract with Carry from B	B – M – C ⇒ B	B B B B B	IMM DIR EXT IND,X IND,Y	C2 D2 F2 E2 18 E2	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SEC	Set Carry	1 ⇒ C		INH	0D	—	2	—	—	—	—	—	—	—	1
SEI	Set Interrupt Mask	1 ⇒ I		INH	0F	—	2	—	—	—	1	—	—	—	—
SEV	Set Overflow Flag	1 ⇒ V		INH	0B	—	2	—	—	—	—	—	—	1	—

Mnemonic	Operation	Description	Addressing Mode		Instruction			Condition Codes							
					Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
STAA (opr)	Store Accumulator A	A ⇒ M	A A A A	DIR EXT IND,X IND,Y	97 B7 A7 18 A7	dd hh ll ff ff	3 4 4 5	—	—	—	—	Δ	Δ	0	—
STAB (opr)	Store Accumulator B	B ⇒ M	B B B B	DIR EXT IND,X IND,Y	D7 F7 E7 18 E7	dd hh ll ff ff	3 4 4 5	—	—	—	—	Δ	Δ	0	—
STD (opr)	Store Accumulator D	A ⇒ M, B ⇒ M + 1		DIR EXT IND,X IND,Y	DD FD ED 18 ED	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STOP	Stop Internal Clocks	—		INH	CF	—	2	—	—	—	—	—	—	—	—
STS (opr)	Store Stack Pointer	SP ⇒ M : M + 1		DIR EXT IND,X IND,Y	9F BF AF 18 AF	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STX (opr)	Store Index Register X	IX ⇒ M : M + 1		DIR EXT IND,X IND,Y	DF FF EF CD EF	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STY (opr)	Store Index Register Y	IY ⇒ M : M + 1		DIR EXT IND,X IND,Y	18 DF 18 FF 1A EF 18 EF	dd hh ll ff ff	5 6 6 6	—	—	—	—	Δ	Δ	0	—
SUBA (opr)	Subtract Memory from A	A – M ⇒ A	A A A A A	IMM DIR EXT IND,X IND,Y	80 90 B0 A0 18 A0	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SUBB (opr)	Subtract Memory from B	B – M ⇒ B	A A A A A	IMM DIR EXT IND,X IND,Y	C0 D0 F0 E0 18 E0	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SUBD (opr)	Subtract Memory from D	D – M : M + 1 ⇒ D		IMM DIR EXT IND,X IND,Y	83 93 B3 A3 18 A3	jj kk dd hh ll ff ff	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
SWI	Software Interrupt	See Figure 3–2		INH	3F	—	14	—	—	—	1	—	—	—	—
TAB	Transfer A to B	A ⇒ B		INH	16	—	2	—	—	—	—	Δ	Δ	0	—
TAP	Transfer A to CC Register	A ⇒ CCR		INH	06	—	2	Δ	↓	Δ	Δ	Δ	Δ	Δ	Δ
TBA	Transfer B to A	B ⇒ A		INH	17	—	2	—	—	—	—	Δ	Δ	0	—
TEST	TEST (Only in Test Modes)	Address Bus Counts		INH	00	—	*	—	—	—	—	—	—	—	—
TPA	Transfer CC Register to A	CCR ⇒ A		INH	07	—	2	—	—	—	—	—	—	—	—
TST (opr)	Test for Zero or Minus	M – 0		EXT IND,X IND,Y	7D 6D 18 6D	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	0	0
TSTA	Test A for Zero or Minus	A – 0	A	INH	4D	—	2	—	—	—	—	Δ	Δ	0	0
TSTB	Test B for Zero or Minus	B – 0	B	INH	5D	—	2	—	—	—	—	Δ	Δ	0	0
TSX	Transfer Stack Pointer to X	SP + 1 ⇒ IX		INH	30	—	3	—	—	—	—	—	—	—	—

Mnemonic	Operation	Description	Addressing Mode	Instruction			Condition Codes							
				Opcode	Operand	Cycles	S	X	H	I	N	Z	V	C
TSY	Transfer Stack Pointer to Y	SP + 1 ⇒ IY	INH	18 30	—	4	—	—	—	—	—	—	—	—
TXS	Transfer X to Stack Pointer	IX – 1 ⇒ SP	INH	35	—	3	—	—	—	—	—	—	—	—
TYS	Transfer Y to Stack Pointer	IY – 1 ⇒ SP	INH	18 35	—	4	—	—	—	—	—	—	—	—
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E	—	**	—	—	—	—	—	—	—	—
XGDX	Exchange D with X	IX ⇒ D, D ⇒ IX	INH	8F	—	3	—	—	—	—	—	—	—	—
XGDY	Exchange D with Y	IY ⇒ D, D ⇒ IY	INH	18 8F	—	4	—	—	—	—	—	—	—	—

Cycle

* Infinity or until reset occurs

** 12 cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPU E-clock cycles (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (14 + n total).

Operands

dd = 8-bit direct address ($0000–$00FF) (high byte assumed to be $00)

ff = 8-bit positive offset $00 (0) to $FF (255) (is added to index)

hh = High-order byte of 16-bit extended address

ii = One byte of immediate data

jj = High-order byte of 16-bit immediate data

kk = Low-order byte of 16-bit immediate data

ll = Low-order byte of 16-bit extended address

mm = 8-bit mask (set bits to be affected)

rr = Signed relative offset $80 (–128) to $7F (+127)
(offset relative to address following machine code offset byte))

Operators

( ) Contents of register shown inside parentheses

⇐ Is transferred to

⇑ Is pulled from stack

⇓ Is pushed onto stack

• Boolean AND

\+ Arithmetic addition symbol except where used as inclusive-OR symbol in Boolean formula

⊕ Exclusive-OR

* Multiply

: Concatenation

– Arithmetic subtraction symbol or negation symbol (two's complement)

Condition Codes

— Bit not changed

0 Bit always cleared

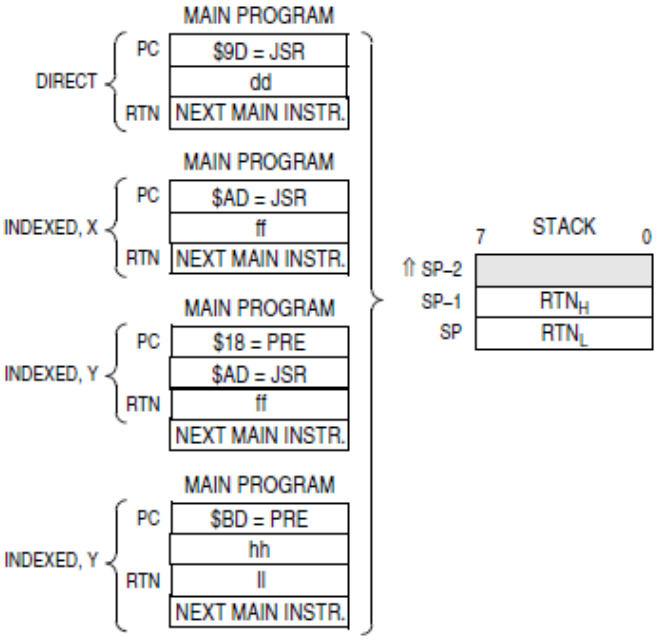
1 Bit always set

Δ Bit cleared or set, depending on operation

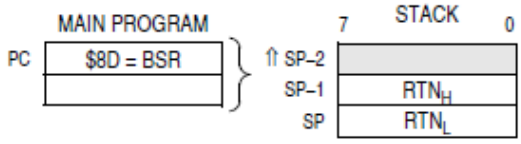
↓ Bit can be cleared, cannot become set

Special Operations

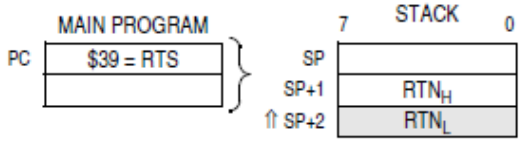
JSR, JUMP TO SUBROUTINE



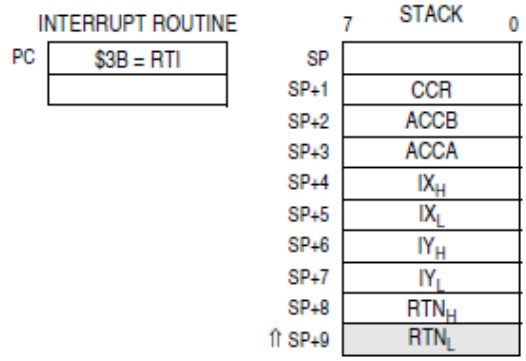
BSR, BRANCH TO SUBROUTINE



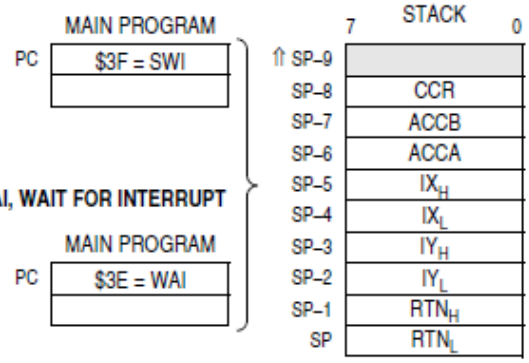
RTS, RETURN FROM SUBROUTINE



RTI, RETURN FROM INTERRUPT



SWI, SOFTWARE INTERRUPT



LEGEND:

RTN = ADDRESS OF NEXT INSTRUCTION IN MAIN PROGRAM TO BE EXECUTED UPON RETURN FROM SUBROUTINE
RTN$_H$ = MOST SIGNIFICANT BYTE OF RETURN ADDRESS
RTN$_L$ = LEAST SIGNIFICANT BYTE OF RETURN ADDRESS
⇑ = STACK POINTER POSITION AFTER OPERATION IS COMPLETE
dd = 8-BIT DIRECT ADDRESS ($0000–$00FF) (HIGH BYTE ASSUMED TO BE $00)
ff = 8-BIT POSITIVE OFFSET $00 (0) TO $FF (255) IS ADDED TO INDEX
hh = HIGH-ORDER BYTE OF 16-BIT EXTENDED ADDRESS
ll = LOW-ORDER BYTE OF 16-BIT EXTENDED ADDRESS
rr = SIGNED RELATIVE OFFSET $80 (–128) TO $7F (+127) (OFFSET RELATIVE TO THE ADDRESS FOLLOWING THE MACHINE CODE OFFSET BYTE)

M68HC11E Series Registers

Summary of the M68HC11E registers. Note that the 128-byte register block can be remapped to any 4K boundary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$1000	Port A Data Register (PORTA)	Read: Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
		Reset:	I	0	0	0	I	I	I	I
$1001	Reserved		R	R	R	R	R	R	R	R
$1002	Parallel I/O Control Register (PIOC)	Read: Write:	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB
		Reset:	0	0	0	0	0	U	1	1
$1003	Port C Data Register (PORTC)	Read: Write:	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
		Reset:	Indeterminate after reset							
$1004	Port B Data Register (PORTB)	Read: Write:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
		Reset:	0	0	0	0	0	0	0	0
$1005	Port C Latched Register (PORTCL)	Read: Write:	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
		Reset:	Indeterminate after reset							
$1006	Reserved		R	R	R	R	R	R	R	R
$1007	Port C Data Direction Register (DDRC)	Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$1008	Port D Data Register (PORTD)	Read: Write:	0	0	PD5	PD4	PD3	PD2	PD1	PD0
		Reset:	U	U	I	I	I	I	I	I
$1009	Port D Data Direction Register (DDRD)	Read: Write:			DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$100A	Port E Data Register (PORTE)	Read: Write:	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
		Reset:	Indeterminate after reset							

Hexadecimal to ASCII Conversion

Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII
$00	NUL	$20	SP *space*	$40	@	$60	` *grave*
$01	SOH	$21	!	$41	A	$61	a
$02	STX	$22	“ quote	$42	B	$62	b
$03	ETX	$23	#	$43	C	$63	c
$04	EOT	$24	$	$44	D	$64	d
$05	ENQ	$25	%	$45	E	$65	e
$06	ACK	$26	&	$46	F	$66	f
$07	BEL *beep*	$27	‘ *apost.*	$47	G	$67	g
$08	BS *back sp*	$28	(	$48	H	$68	h
$09	HT *tab*	$29	)	$49	I	$69	i
$0A	LF *linefeed*	$2A	*	$4A	J	$6A	j
$0B	VT	$2B	+	$4B	K	$6B	k
$0C	FF	$2C	, *comma*	$4C	L	$6C	l
$0D	CR *return*	$2D	- *dash*	$4D	M	$6D	m
$0E	SO	$2E	. *period*	$4E	N	$6E	n
$0F	SI	$2F	/	$4F	O	$6F	o
$10	DLE	$30	0	$50	P	$70	p
$11	DC1	$31	1	$51	Q	$71	q
$12	DC2	$32	2	$52	R	$72	r
$13	DC3	$33	3	$53	S	$73	s
$14	DC4	$34	4	$54	T	$74	t
$15	NAK	$35	5	$55	U	$75	u
$16	SYN	$36	6	$56	V	$76	v
$17	ETB	$37	7	$57	W	$77	w
$18	CAN	$38	8	$58	X	$78	x
$19	EM	$39	9	$59	Y	$79	y
$1A	SUB	$3A	:	$5A	Z	$7A	z
$1B	ESCAPE	$3B	;	$5B	[	$7B	{
$1C	FS	$3C	<	$5C	\	$7C	|
$1D	GS	$3D	=	$5D	]	$7D	}
$1E	RS	$3E	>	$5E	^	$7E	~
$1F	US	$3F	?	$5F	_ *under*	$7F	DEL *delete*

Hexadecimal to Decimal Conversion

To convert a hexadecimal number (up to four hexadecimal digits) to decimal, look up the decimal equivalent of each hexadecimal digit. The decimal equivalent of the original hexadecimal number is the sum of the weights found in the table for all hexadecimal digits.

To convert a decimal number (up to $65{,}535_{10}$) to hexadecimal, find the largest decimal number that is less than or equal to the number you are converting. The corresponding hexadecimal digit is the most significant hexadecimal digit of the result. Subtract the decimal number found from the original decimal number to get the *remaining decimal value*. Repeat the procedure using the remaining decimal value for each subsequent hexadecimal digit.

15	**Bit**	**8**		**7**	**Bit**	**0**	
15	**12**	**11**	**8**	**7**	**4**	**3**	**0**
4th Hex Digit		**3rd Hex Digit**		**2nd Hex Digit**		**1st Hex Digit**	
Hex	**Decimal**	**Hex**	**Decimal**	**Hex**	**Decimal**	**Hex**	**Decimal**
0	0	0	0	0	0	0	0
1	4,096	1	256	1	16	1	1
2	8,192	2	512	2	32	2	2
3	12,288	3	768	3	48	3	3
4	16,384	4	1,024	4	64	4	4
5	20,480	5	1,280	5	80	5	5
6	24,576	6	1,536	6	96	6	6
7	28,672	7	1,792	7	112	7	7
8	32,768	8	2,048	8	128	8	8
9	36,864	9	2,304	9	144	9	9
A	40,960	A	2,560	A	160	A	10
B	45,056	B	2,816	B	176	B	11
C	49,152	C	3,072	C	192	C	12
D	53,248	D	3,328	D	208	D	13
E	57,344	E	3,484	E	224	E	14
F	61,440	F	3,840	F	240	F	15

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

M68HC11ERG
Rev. 2.1, 07/2005