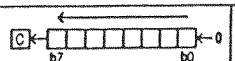
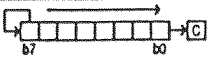
INSTRUCTIONS, ADDRESSING MODES, AND EXECUTION TIMES

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
ABA	Add Accumulators	A + B → A	INH	1B		1	2	— — Δ — Δ Δ Δ Δ
ABX	Add B to X	IX + 00:B → IX	INH	3A		1	3	— — — — — — — —
ABY	Add B to Y	IY + 00:B → IY	INH	18 3A		2	4	— — — — — — — —
ADCA (opr)	Add with Carry to A	A + M + C → A	A IMM	89	ii	2	2	— — Δ — Δ Δ Δ Δ
			A DIR	99	dd	2	3	
			A EXT	B9	hh ll	3	4	
			A IND,X	A9	ff	2	4	
			A IND,Y	18 A9	ff	3	5	
ADCB (opr)	Add with Carry to B	B + M + C → B	B IMM	C9	ii	2	2	— — Δ — Δ Δ Δ Δ
			B DIR	D9	dd	2	3	
			B EXT	F9	hh ll	3	4	
			B IND,X	E9	ff	2	4	
			B IND,Y	18 E9	ff	3	5	
ADDA (opr)	Add Memory to A	A + M → A	A IMM	8B	ii	2	2	— — Δ — Δ Δ Δ Δ
			A DIR	9B	dd	2	3	
			A EXT	BB	hh ll	3	4	
			A IND,X	AB	ff	2	4	
			A IND,Y	18 AB	ff	3	5	

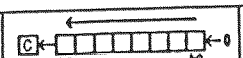
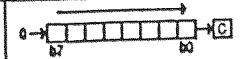
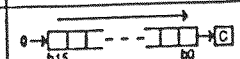
Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
ADDB (opr)	Add Memory to B	B + M → B	B IMM	CB	ii	2	2	— — Δ — Δ Δ Δ Δ
			B DIR	DB	dd	2	3	
			B EXT	FB	hh ll	3	4	
			B IND,X	EB	ff	2	4	
			B IND,Y	18 EB	ff	3	5	
ADDD (opr)	Add 16-Bit to D	D + M:M + 1 → D	IMM	C3	jj kk	3	4	— — — — Δ Δ Δ Δ
			DIR	D3	dd	2	5	
			EXT	F3	hh ll	3	6	
			IND,X	E3	ff	2	6	
			IND,Y	18 E3	ff	3	7	
ANDA (opr)	AND A with Memory	A • M → A	A IMM	84	ii	2	2	— — — — Δ Δ 0 —
			A DIR	94	dd	2	3	
			A EXT	B4	hh ll	3	4	
			A IND,X	A4	ff	2	4	
			A IND,Y	18 A4	ff	3	5	
ANDB (opr)	AND B with Memory	B • M → B	B IMM	C4	ii	2	2	— — — — Δ Δ 0 —
			B DIR	D4	dd	2	3	
			B EXT	F4	hh ll	3	4	
			B IND,X	E4	ff	2	4	
			B IND,Y	18 E4	ff	3	5	

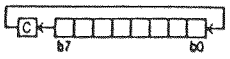
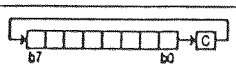
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	68	ff	2	6	
			IND,Y	18 68	ff	3	7	
			A INH	48		1	2	
			B INH	58		1	2	
ASLD	Arithmetic Shift Left Double		INH	05		1	3	— — — — Δ Δ Δ Δ
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	67	ff	2	6	
			IND,Y	18 67	ff	3	7	
			A INH	47		1	2	
			B INH	57		1	2	
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	2	3	— — — — — — — —
BCLR (opr) (msk)	Clear Bit(s)	M • (mm) → M	DIR	15	dd mm	3	6	— — — — Δ Δ 0 —
			IND,X	1D	ff mm	3	7	
			IND,Y	18 1D	ff mm	4	8	
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	2	3	— — — — — — — —
BEQ (rel)	Branch if = Zero	? Z = 1	REL	27	rr	2	3	— — — — — — — —
BGE (rel)	Branch if ≥ Zero	? N ⊕ V = 0	REL	2C	rr	2	3	— — — — — — — —
BGT (rel)	Branch if > Zero	? Z + (N ⊕ V) = 0	REL	2E	rr	2	3	— — — — — — — —
BHI (rel)	Branch if Higher	? C + Z = 0	REL	22	rr	2	3	— — — — — — — —
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	2	3	— — — — — — — —

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
BITA (opr)	Bit(s) Test A with Memory	A • M	A IMM	85	ii	2	2	— — — — Δ Δ 0 —
			A DIR	95	dd	2	3	
			A EXT	B5	hh ll	3	4	
			A IND,X	A5	ff	2	4	
			A IND,Y	18 A5	ff	3	5	
BITB (opr)	Bit(s) Test B with Memory	B • M	B IMM	C5	ii	2	2	— — — — Δ Δ 0 —
			B DIR	D5	dd	2	3	
			B EXT	F5	hh ll	3	4	
			B IND,X	E5	ff	2	4	
			B IND,Y	18 E5	ff	3	5	
BLE (rel)	Branch if ≤ Zero	? Z + (N ⊕ V) = 1	REL	2F	rr	2	3	— — — — — — — —
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	2	3	— — — — — — — —
BLS (rel)	Branch if Lower or Same	? C + Z = 1	REL	23	rr	2	3	— — — — — — — —
BLT (rel)	Branch if < Zero	? N ⊕ V = 1	REL	2D	rr	2	3	— — — — — — — —
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	2	3	— — — — — — — —
BNE (rel)	Branch if Not = Zero	? Z = 0	REL	26	rr	2	3	— — — — — — — —
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	2	3	— — — — — — — —
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	2	3	— — — — — — — —
BRCLR (opr) (msk) (rel)	Branch if Bit(s) Clear	? M • mm = 0	DIR	13	dd mm rr	4	6	— — — — — — — —
			IND,X	1F	ff mm rr	4	7	
			IND,Y	18 1F	ff mm rr	5	8	
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	2	3	— — — — — — — —
BRSET (opr) (msk) (rel)	Branch if Bit(s) Set	? (M) • mm = 0	DIR	12	dd mm rr	4	6	— — — — — — — —
			IND,X	1E	ff mm rr	4	7	
			IND,Y	18 1E	ff mm rr	5	8	
BSET (opr) (msk)	Set Bit(s)	M + mm → M	DIR	14	dd mm	3	6	— — — — Δ Δ 0 —
			IND,X	1C	ff mm	3	7	
			IND,Y	18 1C	ff mm	4	8	
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	— — — — — — — —
BVC (rel)	Branch if Overflow Clear	? V = 0	REL	28	rr	2	3	— — — — — — — —
BVS (rel)	Branch if Overflow Set	? V = 1	REL	29	rr	2	3	— — — — — — — —
CBA	Compare A to B	A – B	INH	11		1	2	— — — — Δ Δ Δ Δ
CLC	Clear Carry Bit	0 → C	INH	0C		1	2	— — — — — — — 0
CLI	Clear Interrupt Mask	0 → I	INH	0E		1	2	— — — 0 — — — —
CLR (opr)	Clear Memory Byte	0 → M	EXT	7F	hh ll	3	6	— — — — 0 1 0 0
			IND,X	6F	ff	2	6	
			IND,Y	18 6F	ff	3	7	
CLRA	Clear Accumulator A	0 → A	A INH	4F		1	2	— — — — 0 1 0 0

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
CLRB	Clear Accumulator B	0 → B	B INH	5F		1	2	— — — — 0 1 0 0
CLV	Clear Overflow Flag	0 → V	INH	0A		1	2	— — — — — — 0 —
CMPA (opr)	Compare A to Memory	A – M	A IMM	81	ii	2	2	— — — — Δ Δ Δ Δ
			A DIR	91	dd	2	3	
			A EXT	B1	hh ll	3	4	
			A IND,X	A1	ff	2	4	
			A IND,Y	18 A1	ff	3	5	
CMPB (opr)	Compare B to Memory	B – M	B IMM	C1	ii	2	2	— — — — Δ Δ Δ Δ
			B DIR	D1	dd	2	3	
			B EXT	F1	hh ll	3	4	
			B IND,X	E1	ff	2	4	
			B IND,Y	18 E1	ff	3	5	
COM (opr)	1's Complement Memory Byte	$FF – M → M	EXT	73	hh ll	3	6	— — — — Δ Δ 0 1
			IND,X	63	ff	2	6	
			IND,Y	18 63	ff	3	7	
COMA	1's Complement A	$FF – A → A	A INH	43		1	2	— — — — Δ Δ 0 1
COMB	1's Complement B	$FF – B → B	B INH	53		1	2	— — — — Δ Δ 0 1
CPD (opr)	Compare D to Memory 16-Bit	D – M:M + 1	IMM	1A 83	jj kk	4	5	— — — — Δ Δ Δ Δ
			DIR	1A 93	dd	3	6	
			EXT	1A B3	hh ll	4	7	
			IND,X	1A A3	ff	3	7	
			IND,Y	CD A3	ff	3	7	
CPX (opr)	Compare X to Memory 16-Bit	IX – M:M + 1	IMM	8C	jj kk	3	4	— — — — Δ Δ Δ Δ
			DIR	9C	dd	2	5	
			EXT	BC	hh ll	3	6	
			IND,X	AC	ff	2	6	
			IND,Y	CD AC	ff	3	7	
CPY (opr)	Compare Y to Memory 16-Bit	IY – M:M + 1	IMM	18 8C	jj kk	4	5	— — — — Δ Δ Δ Δ
			DIR	18 9C	dd	3	6	
			EXT	18 BC	hh ll	4	7	
			IND,X	1A AC	ff	3	7	
			IND,Y	18 AC	ff	3	7	
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	— — — — Δ Δ Δ Δ
DEC (opr)	Decrement Memory Byte	M – 1 → M	EXT	7A	hh ll	3	6	— — — — Δ Δ Δ —
			IND,X	6A	ff	2	6	
			IND,Y	18 6A	ff	3	7	
DECA	Decrement Accumulator A	A – 1 → A	A INH	4A		1	2	— — — — Δ Δ Δ —
DECB	Decrement Accumulator B	B – 1 → B	B INH	5A		1	2	— — — — Δ Δ Δ —
DES	Decrement Stack Pointer	SP – 1 → SP	INH	34		1	3	— — — — — — — —
DEX	Decrement Index Register X	IX – 1 → IX	INH	09		1	3	— — — — — Δ — —

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
EORA (opr)	Exclusive OR A with Memory	A ⊕ M → A	A IMM	88	ii	2	2	— — — — Δ Δ 0 —
			A DIR	98	dd	2	3	
			A EXT	B8	hh ll	3	4	
			A IND,X	A8	ff	2	4	
			A IND,Y	18 A8	ff	3	5	
EORB (opr)	Exclusive OR B with Memory	B ⊕ M → B	B IMM	C8	ii	2	2	— — — — Δ Δ 0 —
			B DIR	D8	dd	2	3	
			B EXT	F8	hh ll	3	4	
			B IND,X	E8	ff	2	4	
			B IND,Y	18 E8	ff	3	5	
FDIV	Fractional Divide 16 by 16	D/X → IX; r → D	INH	03		1	41	— — — — — Δ Δ Δ
IDIV	Integer Divide 16 by 16	D/X → IX; r → D	INH	02		1	41	— — — — — Δ 0 Δ
INC (opr)	Increment Memory Byte	M + 1 → M	EXT	7C	hh ll	3	6	— — — — Δ Δ Δ —
			IND,X	6C	ff	2	6	
			IND,Y	18 6C	ff	3	7	
INCA	Increment Accumulator A	A + 1 → A	A INH	4C		1	2	— — — — Δ Δ Δ —
INCB	Increment Accumulator B	B + 1 → B	B INH	5C		1	2	— — — — Δ Δ Δ —
INS	Increment Stack Pointer	SP + 1 → SP	INH	31		1	3	— — — — — — — —
INX	Increment Index Register X	IX + 1 → IX	INH	08		1	3	— — — — — — Δ — —
INY	Increment Index Register Y	IY + 1 → IY	INH	18 08		2	4	— — — — — — Δ — —
JMP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	— — — — — — — —
			IND,X	6E	ff	2	3	
			IND,Y	18 6E	ff	3	4	
JSR (opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	— — — — — — — —
			EXT	BD	hh ll	3	6	
			IND,X	AD	ff	2	6	
			IND,Y	18 AD	ff	3	7	
LDAA (opr)	Load Accumulator A	M → A	A IMM	86	ii	2	2	— — — — Δ Δ 0 —
			A DIR	96	dd	2	3	
			A EXT	B6	hh ll	3	4	
			A IND,X	A6	ff	2	4	
			A IND,Y	18 A6	ff	3	5	
LDAB (opr)	Load Accumulator B	M → B	B IMM	C6	ii	2	2	— — — — Δ Δ 0 —
			B DIR	D6	dd	2	3	
			B EXT	F6	hh ll	3	4	
			B IND,X	E6	ff	2	4	
			B IND,Y	18 E6	ff	3	5	

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
LDD (opr)	Load Double Accumulator D	M → A, M + 1 → B	IMM	CC	jj kk	3	3	— — — — Δ Δ 0 —
			DIR	DC	dd	2	4	
			EXT	FC	hh ll	3	5	
			IND,X	EC	ff	2	5	
			IND,Y	18 EC	ff	3	6	
LDS (opr)	Load Stack Pointer	M:M + 1 → SP	IMM	8E	jj kk	3	3	— — — — Δ Δ 0 —
			DIR	9E	dd	2	4	
			EXT	BE	hh ll	3	5	
			IND,X	AE	ff	2	5	
			IND,Y	18 AE	ff	3	6	
LDX (opr)	Load Index Register X	M:M + 1 → IX	IMM	CE	jj kk	3	3	— — — — Δ Δ 0 —
			DIR	DE	dd	2	4	
			EXT	FE	hh ll	3	5	
			IND,X	EE	ff	2	5	
			IND,Y	CD EE	ff	3	6	
LDY (opr)	Load Index Register Y	M:M + 1 → IY	IMM	18 CE	jj kk	4	4	— — — — Δ Δ 0 —
			DIR	18 DE	dd	3	5	
			EXT	18 FE	hh ll	4	6	
			IND,X	1A EE	ff	3	6	
			IND,Y	18 EE	ff	3	6	
LSL (opr)	Logical Shift Left		EXT	78	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	68	ff	2	6	
			IND,Y	18 68	ff	3	7	
			A INH	48		1	2	
			B INH	58		1	2	
LSLD	Logical Shift Left Double		INH	05		1	3	— — — — Δ Δ Δ Δ
LSR (opr)	Logical Shift Right		EXT	74	hh ll	3	6	— — — — 0 Δ Δ Δ
			IND,X	64	ff	2	6	
			IND,Y	18 64	ff	3	7	
			A INH	44		1	2	
			B INH	54		1	2	
LSRD	Logical Shift Right Double		INH	04		1	3	— — — — 0 Δ Δ Δ
MUL	Multiply B by B	A x B → D	INH	3D		1	10	— — — — — — — Δ
NEG (opr)	2's Complement Memory Byte	0 − M → M	EXT	70	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	60	ff	2	6	
			IND,Y	18 60	ff	3	7	
NEGA	2's Complement A	0 − A → A	A INH	40		1	2	— — — — Δ Δ Δ Δ
NEGB	2's Complement B	0 − B → B	B INH	50		1	2	— — — — Δ Δ Δ Δ
NOP	No Operation	No Operation	INH	01		1	2	— — — — — — — —

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
ORAA (opr)	OR Accumulator A (Inclusive)	A + M → A	A IMM	8A	ii	2	2	— — — — Δ Δ 0 —
			A DIR	9A	dd	2	3	
			A EXT	BA	hh ll	3	4	
			A IND,X	AA	ff	2	4	
			A IND,Y	18 AA	ff	3	5	
ORAB (opr)	OR Accumulator B (Inclusive)	B + M → B	B IMM	CA	ii	2	2	— — — — Δ Δ 0 —
			B DIR	DA	dd	2	3	
			B EXT	FA	hh ll	3	4	
			B IND,X	EA	ff	2	4	
			B IND,Y	18 EA	ff	3	5	
PSHA	Push A onto Stack	A ↑ Stk, SP = SP – 1	A INH	36		1	3	— — — — — — — —
PSHB	Push B onto Stack	B ↑ Stk, SP = SP – 1	B INH	37		1	3	— — — — — — — —
PSHX	Push X onto Stack (Lo First)	IX ↑ Stk, SP = SP – 2	INH	3C		1	4	— — — — — — — —
PSHY	Push Y onto Stack (Lo First)	IY ↑Stk, SP = SP – 2	INH	18 3C		2	5	— — — — — — — —
PULA	Pull A from Stack	SP = SP + 1, A ↓ Stk	A INH	32		1	4	— — — — — — — —
PULB	Pull B from Stack	SP = SP + 1, B ↓ Stk	B INH	33		1	4	— — — — — — — —
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX ↓ Stk	INH	38		1	5	— — — — — — — —
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY ↓ Stk	INH	18 38		2	6	— — — — — — — —
ROL (opr)	Rotate Left		EXT	79	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	69	ff	2	6	
			IND,Y	18 69	ff	3	7	
			A INH	49		1	2	
			B INH	59		1	2	
ROR (opr)	Rotate Right		EXT	76	hh ll	3	6	— — — — Δ Δ Δ Δ
			IND,X	66	ff	2	6	
			IND,Y	18 66	ff	3	7	
			A INH	46		1	2	
			B INH	56		1	2	
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	Δ Δ Δ Δ Δ Δ Δ Δ
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	— — — — — — — —
SBA	Subtract B from A	A – B → A	INH	10		1	2	— — — — Δ Δ Δ Δ
SBCA (opr)	Subtract with Carry from A	A – M – C → A	A IMM	82	ii	2	2	— — — — Δ Δ Δ Δ
			A DIR	92	dd	2	3	
			A EXT	B2	hh ll	3	4	
			A IND,X	A2	ff	2	4	
			A IND,Y	18 A2	ff	3	5	
SBCB (opr)	Subtract with Carry from B	B – M – C → B	B IMM	C2	ii	2	2	— — — — Δ Δ Δ Δ
			B DIR	D2	dd	2	3	
			B EXT	F2	hh ll	3	4	
			B IND,X	E2	ff	2	4	
			B IND,Y	18 E2	ff	3	5	

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
SEC	Set Carry	1 → C	INH	0D		1	2	— — — — — — — 1
SEI	Set Interrupt Mask	1 → I	INH	0F		1	2	— — — 1 — — — —
SEV	Set Overflow Flag	1 → V	INH	0B		1	2	— — — — — — 1 —
STAA (opr)	Store Accumulator A	A → M	A DIR	97	dd	2	3	— — — — Δ Δ 0 —
			A EXT	B7	hh ll	3	4	
			A IND,X	A7	ff	2	4	
			A IND,Y	18 A7	ff	3	5	
STAB (opr)	Store Accumulator B	B → M	B DIR	D7	dd	2	3	— — — — Δ Δ 0 —
			B EXT	F7	hh ll	3	4	
			B IND,X	E7	ff	2	4	
			B IND,Y	18 E7	ff	3	5	
STD (opr)	Store Accumulator D	A → M, B → M + 1	DIR	DD	dd	2	4	— — — — Δ Δ 0 —
			EXT	FD	hh ll	3	5	
			IND,X	ED	ff	2	5	
			IND,Y	18 ED	ff	3	6	
STOP	Stop Internal Clocks		INH	CF		1	2	— — — — — — — —
STS (opr)	Store Stack Pointer	SP → M:M + 1	DIR	9F	dd	2	4	— — — — Δ Δ 0 —
			EXT	BF	hh ll	3	5	
			IND,X	AF	ff	2	5	
			IND,Y	18 AF	ff	3	6	
STX (opr)	Store Index Register X	IX → M:M + 1	DIR	DF	dd	2	4	— — — — Δ Δ 0 —
			EXT	FF	hh ll	3	5	
			IND,X	EF	ff	2	5	
			IND,Y	CD EF	ff	3	6	
STY (opr)	Store Index Register Y	IY → M:M + 1	DIR	18 DF	dd	3	5	— — — — Δ Δ 0 —
			EXT	18 FF	hh ll	4	6	
			IND,X	1A EF	ff	3	6	
			IND,Y	18 EF	ff	3	6	
SUBA (opr)	Subtract Memory from A	A – M → A	A IMM	80	ii	2	2	— — — — Δ Δ Δ Δ
			A DIR	90	dd	2	3	
			A EXT	B0	hh ll	3	4	
			A IND,X	A0	ff	2	4	
			A IND,Y	18 A0	ff	3	5	
SUBB (opr)	Subtract Memory from B	B – M → B	B IMM	C0	ii	2	2	— — — — Δ Δ Δ Δ
			B DIR	D0	dd	2	3	
			B EXT	F0	hh ll	3	4	
			B IND,X	E0	ff	2	4	
			B IND,Y	18 E0	ff	3	5	
SUBD (opr)	Subtract Memory from D	D – M:M + 1 → D	IMM	83	jj kk	3	4	— — — — Δ Δ Δ Δ
			DIR	93	dd	2	5	
			EXT	B3	hh ll	3	6	
			IND,X	A3	ff	2	6	
			IND,Y	18 A3	ff	3	7	

Source Forms	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycles	Condition Codes S X H I N Z V C
TBA	Transfer B to A	B → A	INH	17		1	2	— — — — Δ Δ 0 —
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	*	— — — — — — — —
TPA	Transfer CC Register to A	CCR → A	INH	07		1	2	— — — — — — — —
TST (opr)	Test for Zero or Minus	M – 0	EXT IND,X IND,Y	7D 6D 18 6D	hh ll ff ff	3 2 3	6 6 7	— — — — Δ Δ 0 0
TSTA		A – 0	A INH	4D		1	2	— — — — Δ Δ 0 0
TSTB		B – 0	B INH	5D		1	2	— — — — Δ Δ 0 0
TSX	Transfer Stack Pointer to X	SP + 1 → IX	INH	30		1	3	— — — — — — — —
TSY	Transfer Stack Pointer to Y	SP + 1 → IY	INH	18 30		2	4	— — — — — — — —
TXS	Transfer X to Stack Pointer	IX – 1 → SP	INH	35		1	3	— — — — — — — —
TYS	Transfer Y to Stack Pointer	IY – 1 → SP	INH	18 35		2	4	— — — — — — — —
WAI	Wait for Interrupt	Stack Regs and WAIT	INH	3E		1	**	— — — — — — — —
XGDX	Exchange D with X	IX → D, D → IX	INH	8F		1	3	— — — — — — — —
XGDY	Exchange D with Y	IY → D, D → IY	INH	18 8F		2	4	— — — — — — — —

NOTES:

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 MCU E-clock cycles (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (total = 14 + n).

Operands:

dd = 8-bit direct address $0000–$00FF. (High byte assumed to be $00.)
ff = 8-bit positive offset $00 (0) to $FF (255) 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 the address following the machine code offset byte.

Condition Codes:

— = Bit not changed
0 = Always cleared (logic 0).
1 = Always set (logic 1).
Δ = Bit cleared or set depending on operation.
↓ = Bit may be cleared, cannot become set.

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

Simple Conditional Branches				
	True		False	
Test	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				
	True		False	
Test	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				
	True		False	
Test	Instruction	Opcode	Instruction	Opcode
r > m	BHI	22	BLS	23
r ≥ m	BHS/BCC	24	BLO/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)