

Addressing Mode Description

Addressing Mode	Source Format	Abbrev.	Description
Inherent	INST (no operands)	INH	Operands (if any) are in CPU registers
Immediate	INST #opr8i or INST #opr16i	IMM	Operand is included in instruction stream. 8- or 16-bit size implied by context
Direct	INST opr8a	DIR	Operand is the lower 8 bits of an address in the range \$0000-\$00FF
Extended (Direct)	INST opr16a	EXT	Operand is a 16-bit address
Relative	INST rel8 or INST rel16	REL	An 8-bit or 16-bit relative offset from the current PC is supplied in the instruction
Indexed (5-bit offset)	INST oprx5,xysp	IDX	5-bit signed constant offset from x,y,sp, or pc
Indexed (9-bit offset)	INST oprx9,xysp	IDX1	9-bit signed constant offset from x, y, sp, or pc (lower 8-bits of offset in one extension byte)
Indexed (16-bit offset)	INST oprx16,xysp	IDX2	16-bit constant offset from x, y, sp, or pc (16-bit offset in two extension bytes)
Indexed (accumulator offset)	INST abd,xysp	IDX	Indexed with 8-bit (A or B) or 16-bit (D) accumulator offset from x, y, sp, or pc
Indexed-Indirect (16-bit offset)	INST [oprx16,xysp]	[IDX2]	Pointer to operand is found at 16-bit constant offset from x, y, sp, or pc
Indexed-Indirect (D acc. offset)	INST [D,xysp]	[D,IDX]	Pointer to operand is found at x, y, sp, or pc plus the value in D
Indexed (pre-decrement)	INST oprx3,-xys	IDX	Auto pre-decrement x, y, or sp by 1 ~ 8
Indexed (pre-increment)	INST oprx3,+xys	IDX	Auto pre-increment x, y, or sp by 1 ~ 8
Indexed (post-decrement)	INST oprx3,xys-	IDX	Auto post-decrement x, y, or sp by 1 ~ 8
Indexed (post-increment)	INST oprx3,xys+	IDX	Auto post-increment x, y, or sp by 1 ~ 8

Load and Store Instructions

Mnemonic	Function	Operation
LDAA	Load A	$(M) \Rightarrow A$
LDAB	Load B	$(M) \Rightarrow B$
LDD	Load D	$(M:M+1) \Rightarrow (A:B)$
LDS	Load SP	$(M:M+1) \Rightarrow SP$
LDX	Load index register X	$(M:M+1) \Rightarrow X$
LDY	Load index register Y	$(M:M+1) \Rightarrow X$
LEAS	Load effective address into SP	Effective address \Rightarrow SP
LEAX	Load effective address into X	Effective address \Rightarrow X
LEAY	Load effective address into Y	Effective address \Rightarrow Y
STAA	Store A	$(A) \Rightarrow M$
STAB	Store B	$(B) \Rightarrow M$
STD	Store D	$(A) \Rightarrow M, (B) \Rightarrow M+1$
STS	Store SP	$(SP) \Rightarrow M, M+1$
STX	Store X	$(X) \Rightarrow M:M+1$
STY	Store Y	$(Y) \Rightarrow M:M+1$

Transfer Instructions

Mnemonic	Function	Operation
TAB	Transfer A to B	$(A) \Rightarrow B$
TAP	Transfer A to CCR	$(A) \Rightarrow \text{CCR}$
TBA	Transfer B to A	$(B) \Rightarrow A$
TFR	Transfer register to register	$(A, B, \text{CCR}, D, X, Y, \text{ or } SP) \Rightarrow A, B, \text{CCR}, D, X, Y, \text{ or } SP$
TPA	Transfer CCR to A	$(\text{CCR}) \Rightarrow A$
TSX	Transfer SP to X	$(SP) \Rightarrow X$
TSY	Transfer SP to Y	$(SP) \Rightarrow Y$
TXS	Transfer X to SP	$(X) \Rightarrow SP$
TYS	Transfer Y to SP	$(Y) \Rightarrow SP$

Exchange Instructions

Mnemonic	Function	Operation
EXG	Exchange register to register	$(A, B, \text{CCR}, D, X, Y, \text{ or } SP) \Leftrightarrow A, B, \text{CCR}, D, X, Y, \text{ or } SP$
XGDX	Exchange D with X	$(D) \Leftrightarrow (X)$
XGDY	Exchange D with Y	$(D) \Leftrightarrow (Y)$

Sign Extension Instruction

Mnemonic	Function	Operation
SEX	Sign extend 8-bit operand	$(A, B, \text{ or } \text{CCR}) \Rightarrow D, X, Y, \text{ or } SP$

Add and Subtract Instructions

Mnemonic	Function	Operation
ABA	Add B to A	$(A) + (B) \Rightarrow A$
ABX	Add B to X	$(B) + (X) \Rightarrow X$
ABY	Add B to Y	$(B) + (Y) \Rightarrow Y$
ADCA	Add with carry to A	$(A) + (M) + C \Rightarrow A$
ADCB	Add with carry to B	$(B) + (M) + C \Rightarrow B$
ADDA	Add without carry to A	$(A) + (M) \Rightarrow A$
ADDB	Add without carry to B	$(B) + (M) \Rightarrow B$
ADDD	Add to D	$(A:B) + (M : M + 1) \Rightarrow A : B$
SBA	Subtract B from A	$(A) - (B) \Rightarrow A$
SBCA	Subtract with borrow from A	$(A) - (M) - C \Rightarrow A$
SBCB	Subtract with borrow from B	$(B) - (M) - C \Rightarrow B$
SUBA	Subtract memory from A	$(A) - (M) \Rightarrow A$
SUBB	Subtract memory from B	$(B) - (M) \Rightarrow B$
SUBD	Subtract memory from D	$(A:B) (D) - (M : M + 1) \Rightarrow D$

Boolean Logic Instructions

Mnemonic	Function	Operation
ANDA	AND A with memory	$(A) \cdot (M) \Rightarrow A$
ANDB	AND B with memory	$(B) \cdot (M) \Rightarrow B$
ANDCC	AND CCR with memory (clear CCR bits)	$(CCR) \cdot (M) \Rightarrow CCR$
EORA	Exclusive OR A with memory	$(A) \square (M) \Rightarrow A$
EORB	Exclusive OR B with memory	$(B) \square (M) \Rightarrow B$
ORAA	OR A with memory	$(A) + (M) \Rightarrow A$
ORAB	OR B with memory	$(B) + (M) \Rightarrow B$
ORCC	OR CCR with memory (set CCR bits)	$(CCR) + (M) \Rightarrow CCR$

BCD Add and Subtract Instructions

Mnemonic	Function	Operation
ABA	Add B to A	$(A) + (B) \Rightarrow A$
ADCA	Add with carry to A	$(A) + (M) + C \Rightarrow A$
ADCB(1)	Add with carry to B	$(B) + (M) + C \Rightarrow B$
ADDA	Add memory to A	$(A) + (M) \Rightarrow A$
ADDB(1)	Add memory to B	$(B) + (M) \Rightarrow B$
DAA	Decimal adjust A	$(A) \Rightarrow \text{bcd}(A)$

1. These instructions are not normally used for BCD operations because, although they affect H correctly, they do not leave the result in the correct accumulator (A) to be used with the DAA instruction. Thus additional steps would be needed to adjust the result to correct BCD form.

Decrement and Increment Instructions

Mnemonic	Function	Operation
DEC	Decrement memory	$(M) - \$01 \Rightarrow M$
DECA	Decrement A	$(A) - \$01 \Rightarrow A$
DECB	Decrement B	$(B) - \$01 \Rightarrow B$
DES	Decrement SP	$(SP) - \$0001 \Rightarrow SP$
DEX	Decrement X	$(X) - \$0001 \Rightarrow X$
DEY	Decrement Y	$(Y) - \$0001 \Rightarrow Y$
INC	Increment memory	$(M) + \$01 \Rightarrow M$
INCA	Increment A	$(A) + \$01 \Rightarrow A$
INCB	Increment B	$(B) + \$01 \Rightarrow B$
INS	Increment SP	$(SP) + \$0001 \Rightarrow SP$
INX	Increment X	$(X) + \$0001 \Rightarrow X$
INY	Increment Y	$(Y) + \$0001 \Rightarrow Y$

Multiplication, Multiplication and Addition and Division Instructions

Mnemonic	Function	Operation
MUL	8 by 8 multiply (unsigned)	$(A) \times (B) \Rightarrow A : B$
EMUL	16 by 16 multiply (unsigned)	$(D) \times (Y) \Rightarrow Y : D$
EMULS	16 by 16 multiply (signed)	$(D) \times (Y) \Rightarrow Y : D$
EMACS	Multiply and accumulate (signed) 16 bit by 16 bit \Rightarrow 32 bit	$((M(X):M(X+1)) \times (M(Y):M(Y+1))) +$ $(M \sim M + 3) \Rightarrow M \sim M + 3$
EDIV	32 by 16 divide (unsigned)	$(Y : D) \div (X) \Rightarrow Y, \text{ Remainder} \Rightarrow D$
EDIVS	32 by 16 divide (signed)	$(Y : D) \div (X) \Rightarrow Y, \text{ Remainder} \Rightarrow D$
FDIV	16 by 16 fractional divide	$(D) \div (X) \Rightarrow X, \text{ Remainder} \Rightarrow D$
IDIV	16 by 16 integer divide (unsigned)	$(D) \div (X) \Rightarrow X, \text{ Remainder} \Rightarrow D$
IDIVS	16 by 16 integer divide (signed)	$(D) \div (X) \Rightarrow X, \text{ Remainder} \Rightarrow D$

Compare Instructions and Test Instructions

Mnemonic	Function	Operation
CBA	Compare A to B	$(A) - (B)$
CMPA	Compare A to memory	$(A) - (M)$
CMPB	Compare B to memory	$(B) - (M)$
CPD	Compare D to memory (16-bit)	$(A : B) - (M : M + 1)$
CPS	Compare SP to memory (16-bit)	$(SP) - (M : M + 1)$
CPX	Compare X to memory (16-bit)	$(X) - (M : M + 1)$
CPY	Compare Y to memory (16-bit)	$(Y) - (M : M + 1)$
TST	Test memory for zero or minus	$(M) - \$00$
TSTA	Test A for zero or minus	$(A) - \$00$
TSTB	Test B for zero or minus	$(B) - \$00$

Unary and Simple Branch Instructions

Mnemonic	Function	Equation or Operation
	Unary Branches	
BRA	Branch always	$1 = 1$
BRN	Branch never	$1 = 0$
	Simple Branches	
BCC	Branch if carry clear	$C = 0$
BCS	Branch if carry set	$C = 1$
BEQ	Branch if equal	$Z = 1$
BMI	Branch if minus	$N = 1$
BNE	Branch if not equal	$Z = 0$
BPL	Branch if plus	$N = 0$
BVC	Branch if overflow clear	$V = 0$
BVS	Branch if overflow set	$V = 1$

Unsigned and Signed Branch Instructions

Mnemonic	Function	Relation	Equation or Operation
Unsigned Branches			
BHI	Branch if higher	$R > M$	$C + Z = 0$
BHS	Branch if higher or same	$R \geq M$	$C = 0$
BLO	Branch if lower	$R < M$	$C = 1$
BLS	Branch if lower or same	$R \leq M$	$C + Z = 1$
Signed Branches			
BGE	Branch if greater than or equal	$R \geq M$	$N \oplus V = 0$
BGT	Branch if greater than	$R > M$	$Z + (N \oplus V) = 0$
BLE	Branch if less than or equal	$R \leq M$	$Z + (N \oplus V) = 1$
BLT	Branch if less than	$R < M$	$N \oplus V = 1$

Bit Condition Branch Instructions

Mnemonic	Function	Equation or Operation
BRCLR	Branch if selected bits clear	$(M) \cdot (mm) = 0$
BRSET	Branch if selected bits set	$(\sim M) \cdot (mm) = 0$

Loop Primitive Instructions

Mnemonic	Function	Equation or Operation
DBEQ	Decrement counter and branch if = 0, (counter = A, B, D, X, Y, or SP)	$(\text{counter}) - 1 \Rightarrow \text{counter}$, If (counter) = 0, then branch; else continue to next instruction
DBNE	Decrement counter and branch if \neq 0, (counter = A, B, D, X, Y, or SP)	$(\text{counter}) - 1 \Rightarrow \text{counter}$, If (counter) not = 0, then branch; else continue to next instruction
IBEQ	Increment counter and branch if = 0, (counter = A, B, D, X, Y, or SP)	$(\text{counter}) + 1 \Rightarrow \text{counter}$, If (counter) = 0, then branch; else continue to next instruction
IBNE	Increment counter and branch if \neq 0, (counter = A, B, D, X, Y, or SP)	$(\text{counter}) + 1 \Rightarrow \text{counter}$, If (counter) not = 0, then branch; else continue to next instruction
TBEQ	Test counter and branch if = 0, (counter = A, B, D, X, Y, or SP)	If (counter) = 0, then branch; else continue to next instruction
TBNE	Test counter and branch if \neq 0, (counter = A, B, D, X, Y, or SP)	If (counter) not = 0, then branch; else continue to next instruction

Shift and Rotate Instructions

Mnemonic	Function	Operation
	Logical Shifts	
LSL LSLA LSLB	Logic shift left memory Logic shift left A Logic shift left B	
LSLD	Logic shift left D	
LSR LSRA LSRB	Logic shift right memory Logic shift right A Logic shift right B	
LSRD	Logic shift right D	
	Arithmetic Shifts	
ASL ASLA ASLB	Arithmetic shift left memory Arithmetic shift left A Arithmetic shift left B	
ASLD	Arithmetic shift left D	
ASR ASRA ASRB	Arithmetic shift right memory Arithmetic shift right A Arithmetic shift right B	
	Rotates	
ROL ROLA ROLB	Rotate left memory through carry Rotate left A through carry Rotate left B through carry	
ROR RORA RORB	Rotate right memory through carry Rotate right A through carry Rotate right B through carry	

Bit Test and Manipulation Instructions

Mnemonic	Function	Operation
BCLR	Clear bits in memory	$(M) \cdot (\text{not}(mm)) \Rightarrow M$
BITA	Bit test A	$(A) \cdot (M)$
BITB	Bit test B	$(B) \cdot (M)$
BSET	Set bits in memory	$(M) + (mm) \Rightarrow M$

Stack Operation Instructions

Mnemonic	Function	Operation
PSHA	Push A	$(SP) - 1 \Rightarrow SP; (A) \Rightarrow M(SP)$
PSHB	Push B	$(SP) - 1 \Rightarrow SP; (B) \Rightarrow M(SP)$
PSHC	Push CCR	$(SP) - 1 \Rightarrow SP; (A) \Rightarrow M(SP)$
PSHD	Push D	$(SP) - 2 \Rightarrow SP; (A : B) \Rightarrow M(SP) : M(SP+1)$
PSHX	Push X	$(SP) - 2 \Rightarrow SP; (X) \Rightarrow M(SP) : M(SP+1)$
PSHY	Push Y	$(SP) - 2 \Rightarrow SP; (Y) \Rightarrow M(SP) : M(SP+1)$
PULA	Pull A	$(M(SP)) \Rightarrow A; (SP) + 1 \Rightarrow SP$
PULB	Pull B	$(M(SP)) \Rightarrow B; (SP) + 1 \Rightarrow SP$
PULC	Pull CCR	$(M(SP)) \Rightarrow CCR; (SP) + 1 \Rightarrow SP$
PULD	Pull D	$(M(SP) : M(SP+1)) \Rightarrow A : B; (SP) + 2 \Rightarrow SP$
PULX	Pull X	$(M(SP) : M(SP+1)) \Rightarrow X; (SP) + 2 \Rightarrow SP$
PULY	Pull Y	$(M(SP) : M(SP+1)) \Rightarrow Y; (SP) + 2 \Rightarrow SP$

Jump and Subroutine Instructions

Mnemonic	Function	Operation
BSR	Branch to subroutine	$SP - 2 \Rightarrow SP$ $RTNH : RTNL \Rightarrow M(SP) : M(SP+1)$ Subroutine address $\Rightarrow PC$
CALL	Call subroutine in expanded memory	$SP - 2 \Rightarrow SP$ $RTNH:RTNL \Rightarrow M(SP) : M(SP+1)$ $SP - 1 \Rightarrow SP$ $(PPAGE) \Rightarrow M(SP)$ Page $\Rightarrow PPAGE$ Subroutine address $\Rightarrow PC$
JMP	Jump Address	$\Rightarrow PC$
JSR	Jump to subroutine	$SP - 2 \Rightarrow SP$ $RTNH : RTNL \Rightarrow M(SP) : M(SP+1)$ Subroutine address $\Rightarrow PC$
RTC	Return from call	$M(SP) \Rightarrow PPAGE$ $SP + 1 \Rightarrow SP$ $M(SP) : M(SP+1) \Rightarrow PCH : PCL$ $SP + 2 \Rightarrow SP$
RTS	Return from subroutine	$M(SP) : M(SP+1) \Rightarrow PCH : PCL$ $SP + 2 \Rightarrow SP$

9S12DP256 Memory Map

Address	Module	Size (Bytes)
\$0000 - \$0017	CORE (Ports A, B, E, Modes, Inits, Test)	24
\$0018 - \$0019	Reserved	2
\$001A - \$001B	Device ID register (PARTID)	2
\$001C - \$001F	CORE (MEMSIZ, IRQ, HPRIO)	4
\$0020 - \$0027	Reserved	8
\$0028 - \$002F	CORE (Background Debug Mode)	8
\$0030 - \$0033	CORE (PPAGE, Port K)	4
\$0034 - \$003F	Clock and Reset Generator (PLL, RTI, COP)	12
\$0040 - \$007F	Enhanced Capture Timer 16-bit 8 channels	64
\$0080 - \$009F	Analog to Digital Converter 10-bit 8 channels (ATD0)	32
\$00A0 - \$00C7	Pulse Width Modulator 8-bit 8 channels (PWM)	40
\$00C8 - \$00CF	Serial Communications Interface 0 (SCI0)	8
\$00D0 - \$00D7	Serial Communications Interface 0 (SCI1)	8
\$00D8 - \$00DF	Serial Peripheral Interface (SPI0)	8
\$00E0 - \$00E7	Inter IC Bus	8
\$00E8 - \$00EF	Byte Data Link Controller (BDLC)	8
\$00F0 - \$00F7	Serial Peripheral Interface (SPI1)	8
\$00F8 - \$00FF	Serial Peripheral Interface (SPI2)	8
\$0100 - \$010F	Flash Control Register	16
\$0110 - \$011B	EEPROM Control Register	12
\$011C - \$011F	Reserved	4
\$0120 - \$013F	Analog to Digital Converter 10-bit 8 channels (ATD1)	32
\$0140 - \$017F	Motorola Scalable Can (CAN0)	64
\$0180 - \$01BF	Motorola Scalable Can (CAN1)	64
\$01C0 - \$01FF	Motorola Scalable Can (CAN2)	64
\$0200 - \$023F	Motorola Scalable Can (CAN3)	64
\$0240 - \$027F	Port Integration Module (PIM)	64
\$0280 - \$02BF	Motorola Scalable Can (CAN4)	64
\$02C0 - \$03FF	Reserved	320
\$0000 - \$0FFF	EEPROM array	4096
\$1000 - \$3FFF	RAM array	12288
\$4000 - \$7FFF	Fixed Flash EEPROM	16384
\$8000 - \$BFFF	Flash EEPROM Page Window	16384
\$C000 - \$FFFF	Fixed Flash EEPROM	16384