- **AS12 Assembler Directives**
- **A Summary of 9S12 instructions**
- **Disassembly of 9S12 op codes**
- **Huang Section 1.8, Chapter 2**
- **MC9S12 V1.5 Core User Guide Version 1.2, Section 12**
  - A labels is a name assigned the address of the location counter where ithe label is defined
  - Use of {\tt dc} and {\tt ds} directives
  - A summary of 9S12 instruction
  - How to tell which branch instruction to use

HC12 Instructions

1. Data Transfer and Manipulation Instructions — instructions which move and manipulate data (HCS12 Core Users Guide, Sections 4.3.1, 4.3.2, and 4.3.3).

• Load and Store—load copy of memory contents into a register; store copy of register contents into memory.

```
LDAA $2000          ; Copy contents of address $2000 into A
STD 0,X             ; Copy contents of D to address X and X+1
```

• Transfer — copy contents of one register to another.

```
TBA                 ; Copy B to A
TFR X,Y             ; Copy X to Y
```

• Exhange — exchange contents of two registers.

```
XGDX                ; Exchange contents of D and X
EXG A,B             ; Exchange contents of A and B
```

• Move — copy contents of one memory location to another.

```
MOVB $2000,$20A0    ; Copy byte at $2000 to $20A0
MOVW 2,X+,2,Y+      ; Copy two bytes from address held
                    ; in X to address held in Y
                    ; Add 2 to X and Y
```

2. Arithmetic Instructions — addition, subtraction, multiplication, divison (HCS12 Core Users Guide, Sections 4.3.4, 4.3.6 and 4.3.10).

```
ABA                 ; Add B to A; results in A
SUBD $20A1          ; Subtract contents of $20A1 from D
INX                 ; Increment X by 1
MUL                 ; Multiply A by B; results in D
```

3. Logic and Bit Instructions — perform logical operations (HCS12 Core Users Guide, Sections 4.3.8, 4.3.9, 4.3.11 and 4.3.12).

• Logic Instructions

```
ANDA $2000          ; Logical AND of A with contents of $2000
NEG -2,X            ; Negate (2's comp) contents of address (X-2)
LSLA                ; Logical shift left A by 1
```

• Bit manipulation and test instructions—work with one bit of a register or memory.

```
BITA #$08           ; Check to see if Bit 3 of A is set
BSET $0002,#$18     ; Set bits 3 and 4 of address $002
```

4. Data test instructions — test contents of a register or memory (to see if zero, negative, etc.), or compare contents of a register to memory (to see if bigger than, etc.) (HCS12 Core Users Guide, Section 4.3.7).

```
TSTA                ; (A)-0 -- set flags accordingly
CPX #$8000          ; (X) - $8000 -- set flags accordingly
```

5. Jump and Branch Instructions — Change flow of program (e.g., goto, it-then-else, switch-case) (HCS12 Core Users Guide, Sections 4.3.17 and 4.3.18).

```
JMP L1               ; Start executing code at address label L1
BEQ L2               ; If Z bit set, go to label L2
DBNE X, L3           ; Decrement X; if X not 0 then goto L3
BRCLR $1A,#$80,L4    ; If bit 7 of addr $1A clear, go to label L4
```

6. Function Call and Interrupt Instructions — initiate or terminate a subroutine; initiate or terminate and interrupt call (HCS12 Core Users Guide, Sections 4.3.18, 4.3.19).

• Subroutine instructions:

```
JSR sub1            ; Jump to subroutine sub1
RTS                 ; Return from subroutine
```

• Interrupt instructions

```
SWI                 ; Initiate software interrupt
RTI                 ; Return from interrupt
```

7. Load Effective Address Instructions — Put effective address into X, Y or SP (HCS12 Core Users Guide, Section 4.3.22).

```
LEAX 5,Y            ; Put address (Y) + 5 into X
```

8. Condition Code Instructions — change bits in Condition Code Register (HCS12 Core Users Guide, Section 4.3.23).

```
ANDCC #$f0        ; Clear N, Z, C and V bits of CCR
SEV               ; Set V bit of CCR
```

9. Stacking Instructions—push data onto and pull data off of stack (HCS12 Core Users Guide, Section 4.3.21).

```
PSHA              ; Push contents of A onto stack
PULX              ; Pull two top bytes of stack, put into X
```

10. Stop and Wait Instructions — put HC12 into low power mode (HCS12 Core Users Guide, Section 4.3.24).

```
STOP              ; Put into lowest power mode
WAI               ; Put into low power mode until next interrupt
```

11. Instructions we won't discuss or use — BCD arithmetic, fuzzy logic, minimum and maximum, multiply-accumulate, table interpolation (HCS12 Core Users Guide, Sections 4.3.5, 4.3.13, 4.3.14, 4.3.15, 4.3.16).

Disassembly of an HC12 Program

• It is sometimes useful to be able to convert *HC12 op codes* into *mnemonics*.

For example, consider the hex code:

```
ADDR DATA
---------------------------------------------------------
1000 C6 05 CE 20 00 E6 01 18 06 04 35 EE 3F
```

• To determine the instructions, use Table A-2 of the HCS12 Core Users Guide.

– If the first byte of the instruction is anything other than **$18**, use Sheet 1 of 2. From this table, determine the number of bytes of the instruction and the addressing mode. For example, **$C6** is a two-byte instruction, the mnemonic is **LDAB**, and it uses the **IMM** addressing mode. Thus, the two bytes **C6 05** is the *op code* for the instruction **LDAB #$05**.

– If the first byte is **$18**, use Sheet 2 of 2, and do the same thing. For example, **18 06** is a two byte instruction, the mnemonic is **ABA**, and it uses the **INH** addressing mode, so there is no operand. Thus, the two bytes **18 06** is the op code for the instruction **ABA**.

– Indexed addressing mode is fairly complicated to disassemble. You need to use Table A-3 to determine the operand. For example, the op code **$E6** indicates **LDAB indexed**, and may use two to four bytes (one to three bytes in addition to the op code). The postbyte **01** indicates that the operand is 1,X, which is **5-bit constant offset**, which takes only one additional byte. All 5-bit constant offset, pre and post increment and decrement, and register offset instructions use one additional byte. All **9-bit constant offset** instructions use two additional bytes, with the second byte holding 8 bits of the 9 bit offset. (**The 9th bit is a direction bit**, which is held in the first postbyte.) All 16-bit constant offset instructions use three postbytes, with the 2nd and 3rd holding the 16-bit unsigned offset.

– Transfer (**TFR**) and exchange (**EXG**) instructions all have the *op code* **$B7**. Use Table A-5 to determine whether it is **TFR** or an **EXG**, and to determine which registers are being used. If the most significant bit of the postbyte is **0, the instruction is a transfer instruction**.

– Loop instructions (Decrement and Branch, Increment and Branch, and Test and Branch) all have the op code **$04**. To determine which instruction the *op code* **$04** implies, and whether the branch is <u>positive</u> (forward) or <u>negative</u> (backward), use Table A-6. For example, in the sequence **04 35 EE**, the 04 indicates a loop instruction. The 35 indicates it is a **DBNE X** instruction (decrement register X and branch if result is not equal to zero), and the direction is backward (negative). The **EE** indicates a branch of -18 bytes.

Table A-2. CPU12 Opcode Map (Sheet 1 of 2)

00 †5 BGND IH 1	10 1 ANDCC IM 2	20 3 BRA RL 2	30 3 PULX IH 1	40 1 NEGA IH 1	50 1 NEGB IH 1	60 3-6 NEG ID 2-4	70 4 NEG EX 3	80 1 SUBA IM 2	90 3 SUBA DI 2	A0 3-6 SUBA ID 2-4	B0 3 SUBA EX 3	C0 1 SUBB IM 2	D0 3 SUBB DI 2	E0 3-6 SUBB ID 2-4	F0 3 SUBB EX 3
01 5 MEM IH 1	11 11 EDIV IH 1	21 1 BRN RL 2	31 3 PULY IH 1	41 1 COMA IH 1	51 1 COMB IH 1	61 3-6 COM ID 2-4	71 4 COM EX 3	81 1 CMPA IM 2	91 3 CMPA DI 2	A1 3-6 CMPA ID 2-4	B1 3 CMPA EX 3	C1 1 CMPB IM 2	D1 3 CMPB DI 2	E1 3-6 CMPB ID 2-4	F1 3 CMPB EX 3
02 1 INY IH 1	12 ‡1 MUL IH 1	22 3/1 BHI RL 2	32 3 PULA IH 1	42 1 INCA IH 1	52 1 INCB IH 1	62 3-6 INC ID 2-4	72 4 INC EX 3	82 1 SBCA IM 2	92 3 SBCA DI 2	A2 3-6 SBCA ID 2-4	B2 3 SBCA EX 3	C2 1 SBCB IM 2	D2 3 SBCB DI 2	E2 3-6 SBCB ID 2-4	F2 3 SBCB EX 3
03 1 DEY IH 1	13 3 EMUL IH 1	23 3/1 BLS RL 2	33 3 PULB IH 1	43 1 DECA IH 1	53 1 DECB IH 1	63 3-6 DEC ID 2-4	73 4 DEC EX 3	83 2 SUBD IM 3	93 3 SUBD DI 2	A3 3-6 SUBD ID 2-4	B3 3 SUBD EX 3	C3 2 ADDD IM 3	D3 3 ADDD DI 2	E3 3-6 ADDD ID 2-4	F3 3 ADDD EX 3
04 3 loop* RL 3	14 1 ORCC IM 2	24 3/1 BCC RL 2	34 2 PSHX IH 1	44 1 LSRA IH 1	54 1 LSRB IH 1	64 3-6 LSR ID 2-4	74 4 LSR EX 3	84 1 ANDA IM 2	94 3 ANDA DI 2	A4 3-6 ANDA ID 2-4	B4 3 ANDA EX 3	C4 1 ANDB IM 2	D4 3 ANDB DI 2	E4 3-6 ANDB ID 2-4	F4 3 ANDB EX 3
05 3-6 JMP ID 2-4	15 4-7 JSR ID 2-4	25 3/1 BCS RL 2	35 2 PSHY IH 1	45 1 ROLA IH 1	55 1 ROLB IH 1	65 3-6 ROL ID 2-4	75 4 ROL EX 3	85 1 BITA IM 2	95 3 BITA DI 2	A5 3-6 BITA ID 2-4	B5 3 BITA EX 3	C5 1 BITB IM 2	D5 3 BITB DI 2	E5 3-6 BITB ID 2-4	F5 3 BITB EX 3
06 3 JMP EX 3	16 4 JSR EX 3	26 3/1 BNE RL 2	36 2 PSHA IH 1	46 1 RORA IH 1	56 1 RORB IH 1	66 3-6 ROR ID 2-4	76 4 ROR EX 3	86 1 LDAA IM 2	96 3 LDAA DI 2	A6 3-6 LDAA ID 2-4	B6 3 LDAA EX 3	C6 1 LDAB IM 2	D6 3 LDAB DI 2	E6 3-6 LDAB ID 2-4	F6 3 LDAB EX 3
07 4 BSR RL 2	17 4 JSR DI 2	27 3/1 BEQ RL 2	37 2 PSHB IH 1	47 1 ASRA IH 1	57 1 ASRB IH 1	67 3-6 ASR ID 2-4	77 4 ASR EX 3	87 1 CLRA IH 1	97 1 TSTA IH 1	A7 1 NOP IH 1	B7 1 TFR/EXG IH 2	C7 1 CLRB IH 1	D7 1 TSTB IH 1	E7 3-6 TST ID 2-4	F7 3 TST EX 3
08 1 INX IH 1	18 - Page 2 - -	28 3/1 BVC RL 2	38 3 PULC IH 1	48 1 ASLA IH 1	58 1 ASLB IH 1	68 3-6 ASL ID 2-4	78 4 ASL EX 3	88 1 EORA IM 2	98 3 EORA DI 2	A8 3-6 EORA ID 2-4	B8 3 EORA EX 3	C8 1 EORB IM 2	D8 3 EORB DI 2	E8 3-6 EORB ID 2-4	F8 3 EORB EX 3
09 1 DEX IH 1	19 2 LEAY ID 2-4	29 3/1 BVS RL 2	39 2 PSHC IH 1	49 1 LSRD IH 1	59 1 ASLD IH 1	69 ‡2-4 CLR ID 2-4	79 3 CLR EX 3	89 1 ADCA IM 2	99 3 ADCA DI 2	A9 3-6 ADCA ID 2-4	B9 3 ADCA EX 3	C9 1 ADCB IM 2	D9 3 ADCB DI 2	E9 3-6 ADCB ID 2-4	F9 3 ADCB EX 3
0A ‡7 RTC IH 1	1A 2 LEAX ID 2-4	2A 3/1 BPL RL 2	3A 3 PULD IH 1	4A ‡7 CALL EX 4	5A 2 STAA DI 2	6A ‡2-4 STAA ID 2-4	7A 3 STAA EX 3	8A 1 ORAA IM 2	9A 3 ORAA DI 2	AA 3-6 ORAA ID 2-4	BA 3 ORAA EX 3	CA 1 ORAB IM 2	DA 3 ORAB DI 2	EA 3-6 ORAB ID 2-4	FA 3 ORAB EX 3
0B †8 RTI IH 1	1B 2 LEAS ID 2-4	2B 3/1 BMI RL 2	3B 2 PSHD IH 1	4B ‡7-10 CALL ID 2-5	5B 2 STAB DI 2	6B ‡2-4 STAB ID 2-4	7B 3 STAB EX 3	8B 1 ADDA IM 2	9B 3 ADDA DI 2	AB 3-6 ADDA ID 2-4	BB 3 ADDA EX 3	CB 1 ADDB IM 2	DB 3 ADDB DI 2	EB 3-6 ADDB ID 2-4	FB 3 ADDB EX 3
0C 4-6 BSET ID 3-5	1C 4 BSET EX 4	2C 3/1 BGE RL 2	3C ‡+5 wavr SP 1	4C 4 BSET DI 3	5C 2 STD DI 2	6C ‡2-4 STD ID 2-4	7C 3 STD EX 3	8C 2 CPD IM 3	9C 3 CPD DI 2	AC 3-6 CPD ID 2-4	BC 3 CPD EX 3	CC 2 LDD IM 3	DC 3 LDD DI 2	EC 3-6 LDD ID 2-4	FC 3 LDD EX 3
0D 4-6 BCLR ID 3-5	1D 4 BCLR EX 4	2D 3/1 BLT RL 2	3D 5 RTS IH 1	4D 4 BCLR DI 3	5D 2 STY DI 2	6D ‡2-4 STY ID 2-4	7D 3 STY EX 3	8D 2 CPY IM 3	9D 3 CPY DI 2	AD 3-6 CPY ID 2-4	BD 3 CPY EX 3	CD 2 LDY IM 3	DD 3 LDY DI 2	ED 3-6 LDY ID 2-4	FD 3 LDY EX 3
0E ‡4-6 BRSET ID 4-6	1E 5 BRSET EX 5	2E 3/1 BGT RL 2	3E ‡†7 WAI IH 1	4E 4 BRSET DI 4	5E 2 STX DI 2	6E ‡2-4 STX ID 2-4	7E 3 STX EX 3	8E 2 CPX IM 3	9E 3 CPX DI 2	AE 3-6 CPX ID 2-4	BE 3 CPX EX 3	CE 2 LDX IM 3	DE 3 LDX DI 2	EE 3-6 LDX ID 2-4	FE 3 LDX EX 3
0F ‡4-6 BRCLR ID 4-6	1F 5 BRCLR EX 5	2F 3/1 BLE RL 2	3F 9 SWI IH 1	4F 4 BRCLR DI 4	5F 2 STS DI 2	6F ‡2-4 STS ID 2-4	7F 3 STS EX 3	8F 2 CPS IM 3	9F 3 CPS DI 2	AF 3-6 CPS ID 2-4	BF 3 CPS EX 3	CF 2 LDS IM 3	DF 3 LDS DI 2	EF 3-6 LDS ID 2-4	FF 3 LDS EX 3

Key to Table A-2

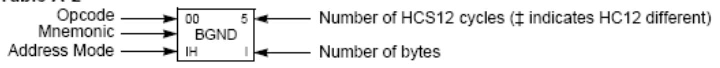


Table A-2. CPU12 Opcode Map (Sheet 2 of 2)

00 4 MOVW IM-ID 5	10 12 IDIV IH 2	20 4 LBRA RL 4	30 10 TRAP IH 2	40 10 TRAP IH 2	50 10 TRAP IH 2	60 10 TRAP IH 2	70 10 TRAP IH 2	80 10 TRAP IH 2	90 10 TRAP IH 2	A0 10 TRAP IH 2	B0 10 TRAP IH 2	C0 10 TRAP IH 2	D0 10 TRAP IH 2	E0 10 TRAP IH 2	F0 10 TRAP IH 2
01 5 MOVW EX-ID 5	11 12 FDIV IH 2	21 3 LBRN RL 4	31 10 TRAP IH 2	41 10 TRAP IH 2	51 10 TRAP IH 2	61 10 TRAP IH 2	71 10 TRAP IH 2	81 10 TRAP IH 2	91 10 TRAP IH 2	A1 10 TRAP IH 2	B1 10 TRAP IH 2	C1 10 TRAP IH 2	D1 10 TRAP IH 2	E1 10 TRAP IH 2	F1 10 TRAP IH 2
02 5 MOVW ID-ID 4	12 13 EMACS SP 4	22 4/3 LBHI RL 4	32 10 TRAP IH 2	42 10 TRAP IH 2	52 10 TRAP IH 2	62 10 TRAP IH 2	72 10 TRAP IH 2	82 10 TRAP IH 2	92 10 TRAP IH 2	A2 10 TRAP IH 2	B2 10 TRAP IH 2	C2 10 TRAP IH 2	D2 10 TRAP IH 2	E2 10 TRAP IH 2	F2 10 TRAP IH 2
03 5 MOVW IM-EX 6	13 3 EMULS IH 2	23 4/3 LBLS RL 4	33 10 TRAP IH 2	43 10 TRAP IH 2	53 10 TRAP IH 2	63 10 TRAP IH 2	73 10 TRAP IH 2	83 10 TRAP IH 2	93 10 TRAP IH 2	A3 10 TRAP IH 2	B3 10 TRAP IH 2	C3 10 TRAP IH 2	D3 10 TRAP IH 2	E3 10 TRAP IH 2	F3 10 TRAP IH 2
04 6 MOVW EX-EX 6	14 12 EDIVS IH 2	24 4/3 LBCC RL 4	34 10 TRAP IH 2	44 10 TRAP IH 2	54 10 TRAP IH 2	64 10 TRAP IH 2	74 10 TRAP IH 2	84 10 TRAP IH 2	94 10 TRAP IH 2	A4 10 TRAP IH 2	B4 10 TRAP IH 2	C4 10 TRAP IH 2	D4 10 TRAP IH 2	E4 10 TRAP IH 2	F4 10 TRAP IH 2
05 5 MOVW ID-EX 5	15 12 IDIVS IH 2	25 4/3 LBCS RL 4	35 10 TRAP IH 2	45 10 TRAP IH 2	55 10 TRAP IH 2	65 10 TRAP IH 2	75 10 TRAP IH 2	85 10 TRAP IH 2	95 10 TRAP IH 2	A5 10 TRAP IH 2	B5 10 TRAP IH 2	C5 10 TRAP IH 2	D5 10 TRAP IH 2	E5 10 TRAP IH 2	F5 10 TRAP IH 2
06 2 ABA IH 2	16 2 SBA IH 2	26 4/3 LBNE RL 4	36 10 TRAP IH 2	46 10 TRAP IH 2	56 10 TRAP IH 2	66 10 TRAP IH 2	76 10 TRAP IH 2	86 10 TRAP IH 2	96 10 TRAP IH 2	A6 10 TRAP IH 2	B6 10 TRAP IH 2	C6 10 TRAP IH 2	D6 10 TRAP IH 2	E6 10 TRAP IH 2	F6 10 TRAP IH 2
07 3 DAA IH 2	17 2 CBA IH 2	27 4/3 LBEQ RL 4	37 10 TRAP IH 2	47 10 TRAP IH 2	57 10 TRAP IH 2	67 10 TRAP IH 2	77 10 TRAP IH 2	87 10 TRAP IH 2	97 10 TRAP IH 2	A7 10 TRAP IH 2	B7 10 TRAP IH 2	C7 10 TRAP IH 2	D7 10 TRAP IH 2	E7 10 TRAP IH 2	F7 10 TRAP IH 2
08 4 MOVB IM-ID 4	18 4-7 MAXA ID 3-5	28 4/3 LBVC RL 4	38 10 TRAP IH 2	48 10 TRAP IH 2	58 10 TRAP IH 2	68 10 TRAP IH 2	78 10 TRAP IH 2	88 10 TRAP IH 2	98 10 TRAP IH 2	A8 10 TRAP IH 2	B8 10 TRAP IH 2	C8 10 TRAP IH 2	D8 10 TRAP IH 2	E8 10 TRAP IH 2	F8 10 TRAP IH 2
09 5 MOVB EX-ID 5	19 4-7 MINA ID 3-5	29 4/3 LBVS RL 4	39 10 TRAP IH 2	49 10 TRAP IH 2	59 10 TRAP IH 2	69 10 TRAP IH 2	79 10 TRAP IH 2	89 10 TRAP IH 2	99 10 TRAP IH 2	A9 10 TRAP IH 2	B9 10 TRAP IH 2	C9 10 TRAP IH 2	D9 10 TRAP IH 2	E9 10 TRAP IH 2	F9 10 TRAP IH 2
0A 5 MOVB ID-ID 4	1A 4-7 EMAXD ID 3-5	2A 4/3 LBPL RL 4	3A †3n REV SP 2	4A 10 TRAP IH 2	5A 10 TRAP IH 2	6A 10 TRAP IH 2	7A 10 TRAP IH 2	8A 10 TRAP IH 2	9A 10 TRAP IH 2	AA 10 TRAP IH 2	BA 10 TRAP IH 2	CA 10 TRAP IH 2	DA 10 TRAP IH 2	EA 10 TRAP IH 2	FA 10 TRAP IH 2
0B 4 MOVB IM-EX 5	1B 4-7 EMIND ID 3-5	2B 4/3 LBMI RL 4	3B †5n/3n REVW SP 2	4B 10 TRAP IH 2	5B 10 TRAP IH 2	6B 10 TRAP IH 2	7B 10 TRAP IH 2	8B 10 TRAP IH 2	9B 10 TRAP IH 2	AB 10 TRAP IH 2	BB 10 TRAP IH 2	CB 10 TRAP IH 2	DB 10 TRAP IH 2	EB 10 TRAP IH 2	FB 10 TRAP IH 2
0C 6 MOVB EX-EX 6	1C 4-7 MAXM ID 3-5	2C 4/3 LBGE RL 4	3C ‡†7B WAV SP 2	4C 10 TRAP IH 2	5C 10 TRAP IH 2	6C 10 TRAP IH 2	7C 10 TRAP IH 2	8C 10 TRAP IH 2	9C 10 TRAP IH 2	AC 10 TRAP IH 2	BC 10 TRAP IH 2	CC 10 TRAP IH 2	DC 10 TRAP IH 2	EC 10 TRAP IH 2	FC 10 TRAP IH 2
0D 5 MOVB ID-EX 5	1D D4-7 MINM ID 3-5	2D 4/3 LBLT RL 4	3D ‡6 TBL ID 3	4D 10 TRAP IH 2	5D 10 TRAP IH 2	6D 10 TRAP IH 2	7D 10 TRAP IH 2	8D 10 TRAP IH 2	9D 10 TRAP IH 2	AD 10 TRAP IH 2	BD 10 TRAP IH 2	CD 10 TRAP IH 2	DD 10 TRAP IH 2	ED 10 TRAP IH 2	FD 10 TRAP IH 2
0E 2 TAB IH 2	1E 4-7 EMAXM ID 3-5	2E 4/3 LBGT RL 4	3E ‡8 STOP IH 2	4E 10 TRAP IH 2	5E 10 TRAP IH 2	6E 10 TRAP IH 2	7E 10 TRAP IH 2	8E 10 TRAP IH 2	9E 10 TRAP IH 2	AE 10 TRAP IH 2	BE 10 TRAP IH 2	CE 10 TRAP IH 2	DE 10 TRAP IH 2	EE 10 TRAP IH 2	FE 10 TRAP IH 2
0F 2 TBA IH 2	1F 4-7 EMINM ID 3-5	2F 4/3 LBLE RL 4	3F 10 ETBL ID 3	4F 10 TRAP IH 2	5F 10 TRAP IH 2	6F 10 TRAP IH 2	7F 10 TRAP IH 2	8F 10 TRAP IH 2	9F 10 TRAP IH 2	AF 10 TRAP IH 2	BF 10 TRAP IH 2	CF 10 TRAP IH 2	DF 10 TRAP IH 2	EF 10 TRAP IH 2	FF 10 TRAP IH 2

* The opcode $04 (on sheet 1 of 2) corresponds to one of the loop primitive instructions DBEQ, DBNE, IBEQ, IBNE, TBEQ, or TBNE.

† Refer to instruction summary for more information.

‡ Refer to instruction summary for different HC12 cycle count.

Page 2: When the CPU encounters a page 2 opcode ($18 on page 1 of the opcode map), it treats the next byte of object code as a page 2 instruction opcode.

Table A-3. Indexed Addressing Mode Postbyte Encoding (xb)

00 0,X 5b const	10 −16,X 5b const	20 1,+X pre-inc	30 1,X+ post-inc	40 0,Y 5b const	50 −16,Y 5b const	60 1,+Y pre-inc	70 1,Y+ post-inc	80 0,SP 5b const	90 −16,SP 5b const	A0 1,+SP pre-inc	B0 1,SP+ post-inc	C0 0,PC 5b const	D0 −16,PC 5b const	E0 n,X 9b const	F0 n,SP 9b const
01 1,X 5b const	11 −15,X 5b const	21 2,+X pre-inc	31 2,X+ post-inc	41 1,Y 5b const	51 −15,Y 5b const	61 2,+Y pre-inc	71 2,Y+ post-inc	81 1,SP 5b const	91 −15,SP 5b const	A1 2,+SP pre-inc	B1 2,SP+ post-inc	C1 1,PC 5b const	D1 −15,PC 5b const	E1 −n,X 9b const	F1 −n,SP 9b const
02 2,X 5b const	12 −14,X 5b const	22 3,+X pre-inc	32 3,X+ post-inc	42 2,Y 5b const	52 −14,Y 5b const	62 3,+Y pre-inc	72 3,Y+ post-inc	82 2,SP 5b const	92 −14,SP 5b const	A2 3,+SP pre-inc	B2 3,SP+ post-inc	C2 2,PC 5b const	D2 −14,PC 5b const	E2 n,X 16b const	F2 n,SP 16b const
03 3,X 5b const	13 −13,X 5b const	23 4,+X pre-inc	33 4,X+ post-inc	43 3,Y 5b const	53 −13,Y 5b const	63 4,+Y pre-inc	73 4,Y+ post-inc	83 3,SP 5b const	93 −13,SP 5b const	A3 4,+SP pre-inc	B3 4,SP+ post-inc	C3 3,PC 5b const	D3 −13,PC 5b const	E3 [n,X] 16b indr	F3 [n,SP] 16b indr
04 4,X 5b const	14 −12,X 5b const	24 5,+X pre-inc	34 5,X+ post-inc	44 4,Y 5b const	54 −12,Y 5b const	64 5,+Y pre-inc	74 5,Y+ post-inc	84 4,SP 5b const	94 −12,SP 5b const	A4 5,+SP pre-inc	B4 5,SP+ post-inc	C4 4,PC 5b const	D4 −12,PC 5b const	E4 A,X A offset	F4 A,SP A offset
05 5,X 5b const	15 −11,X 5b const	25 6,+X pre-inc	35 6,X+ post-inc	45 5,Y 5b const	55 −11,Y 5b const	65 6,+Y pre-inc	75 6,Y+ post-inc	85 5,SP 5b const	95 −11,SP 5b const	A5 6,+SP pre-inc	B5 6,SP+ post-inc	C5 5,PC 5b const	D5 −11,PC 5b const	E5 B,X B offset	F5 B,SP B offset
06 6,X 5b const	16 −10,X 5b const	26 7,+X pre-inc	36 7,X+ post-inc	46 6,Y 5b const	56 −10,Y 5b const	66 7,+Y pre-inc	76 7,Y+ post-inc	86 6,SP 5b const	96 −10,SP 5b const	A6 7,+SP pre-inc	B6 7,SP+ post-inc	C6 6,PC 5b const	D6 −10,PC 5b const	E6 D,X D offset	F6 D,SP D offset
07 7,X 5b const	17 −9,X 5b const	27 8,+X pre-inc	37 8,X+ post-inc	47 7,Y 5b const	57 −9,Y 5b const	67 8,+Y pre-inc	77 8,Y+ post-inc	87 7,SP 5b const	97 −9,SP 5b const	A7 8,+SP pre-inc	B7 8,SP+ post-inc	C7 7,PC 5b const	D7 −9,PC 5b const	E7 [D,X] D indirect	F7 [D,SP] D indirect
08 8,X 5b const	18 −8,X 5b const	28 8,−X pre-dec	38 8,X− post-dec	48 8,Y 5b const	58 −8,Y 5b const	68 8,−Y pre-dec	78 8,Y− post-dec	88 8,SP 5b const	98 −8,SP 5b const	A8 8,−SP pre-dec	B8 8,SP− post-dec	C8 8,PC 5b const	D8 −8,PC 5b const	E8 n,Y 9b const	F8 n,PC 9b const
09 9,X 5b const	19 −7,X 5b const	29 7,−X pre-dec	39 7,X− post-dec	49 9,Y 5b const	59 −7,Y 5b const	69 7,−Y pre-dec	79 7,Y− post-dec	89 9,SP 5b const	99 −7,SP 5b const	A9 7,−SP pre-dec	B9 7,SP− post-dec	C9 9,PC 5b const	D9 −7,PC 5b const	E9 −n,Y 9b const	F9 −n,PC 9b const
0A 10,X 5b const	1A −6,X 5b const	2A 6,−X pre-dec	3A 6,X− post-dec	4A 10,Y 5b const	5A −6,Y 5b const	6A 6,−Y pre-dec	7A 6,Y− post-dec	8A 10,SP 5b const	9A −6,SP 5b const	AA 6,−SP pre-dec	BA 6,SP− post-dec	CA 10,PC 5b const	DA −6,PC 5b const	EA n,Y 16b const	FA n,PC 16b const
0B 11,X 5b const	1B −5,X 5b const	2B 5,−X pre-dec	3B 5,X− post-dec	4B 11,Y 5b const	5B −5,Y 5b const	6B 5,−Y pre-dec	7B 5,Y− post-dec	8B 11,SP 5b const	9B −5,SP 5b const	AB 5,−SP pre-dec	BB 5,SP− post-dec	CB 11,PC 5b const	DB −5,PC 5b const	EB [n,Y] 16b indr	FB [n,PC] 16b indr
0C 12,X 5b const	1C −4,X 5b const	2C 4,−X pre-dec	3C 4,X− post-dec	4C 12,Y 5b const	5C −4,Y 5b const	6C 4,−Y pre-dec	7C 4,Y− post-dec	8C 12,SP 5b const	9C −4,SP 5b const	AC 4,−SP pre-dec	BC 4,SP− post-dec	CC 12,PC 5b const	DC −4,PC 5b const	EC A,Y A offset	FC A,PC A offset
0D 13,X 5b const	1D −3,X 5b const	2D 3,−X pre-dec	3D 3,X− post-dec	4D 13,Y 5b const	5D −3,Y 5b const	6D 3,−Y pre-dec	7D 3,Y− post-dec	8D 13,SP 5b const	9D −3,SP 5b const	AD 3,−SP pre-dec	BD 3,SP− post-dec	CD 13,PC 5b const	DD −3,PC 5b const	ED B,Y B offset	FD B,PC B offset
0E 14,X 5b const	1E −2,X 5b const	2E 2,−X pre-dec	3E 2,X− post-dec	4E 14,Y 5b const	5E −2,Y 5b const	6E 2,−Y pre-dec	7E 2,Y− post-dec	8E 14,SP 5b const	9E −2,SP 5b const	AE 2,−SP pre-dec	BE 2,SP− post-dec	CE 14,PC 5b const	DE −2,PC 5b const	EE D,Y D offset	FE D,PC D offset
0F 15,X 5b const	1F −1,X 5b const	2F 1,−X pre-dec	3F 1,X− post-dec	4F 15,Y 5b const	5F −1,Y 5b const	6F 1,−Y pre-dec	7F 1,Y− post-dec	8F 15,SP 5b const	9F −1,SP 5b const	AF 1,−SP pre-dec	BF 1,SP− post-dec	CF 15,PC 5b const	DF −1,PC 5b const	EF [D,Y] D indirect	FF [D,PC] D indirect

Key to Table A-3

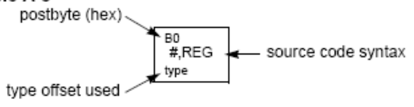


Table A-5. Transfer and Exchange Postbyte Encoding

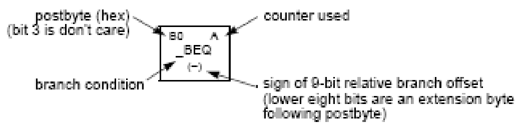
TRANSFERS									
⇓LS	MS⇒	0	1	2	3	4	5	6	7
0		A ⇒ A	B ⇒ A	CCR ⇒ A	TMP3$_L$ ⇒ A	B ⇒ A	X$_L$ ⇒ A	Y$_L$ ⇒ A	SP$_L$ ⇒ A
1		A ⇒ B	B ⇒ B	CCR ⇒ B	TMP3$_L$ ⇒ B	B ⇒ B	X$_L$ ⇒ B	Y$_L$ ⇒ B	SP$_L$ ⇒ B
2		A ⇒ CCR	B ⇒ CCR	CCR ⇒ CCR	TMP3$_L$ ⇒ CCR	B ⇒ CCR	X$_L$ ⇒ CCR	Y$_L$ ⇒ CCR	SP$_L$ ⇒ CCR
3		sex:A ⇒ TMP2	sex:B ⇒ TMP2	sex:CCR ⇒ TMP2	TMP3 ⇒ TMP2	D ⇒ TMP2	X ⇒ TMP2	Y ⇒ TMP2	SP ⇒ TMP2
4		sex:A ⇒ D SEX A,D	sex:B ⇒ D SEX B,D	sex:CCR ⇒ D SEX CCR,D	TMP3 ⇒ D	D ⇒ D	X ⇒ D	Y ⇒ D	SP ⇒ D
5		sex:A ⇒ X SEX A,X	sex:B ⇒ X SEX B,X	sex:CCR ⇒ X SEX CCR,X	TMP3 ⇒ X	D ⇒ X	X ⇒ X	Y ⇒ X	SP ⇒ X
6		sex:A ⇒ Y SEX A,Y	sex:B ⇒ Y SEX B,Y	sex:CCR ⇒ Y SEX CCR,Y	TMP3 ⇒ Y	D ⇒ Y	X ⇒ Y	Y ⇒ Y	SP ⇒ Y
7		sex:A ⇒ SP SEX A,SP	sex:B ⇒ SP SEX B,SP	sex:CCR ⇒ SP SEX CCR,SP	TMP3 ⇒ SP	D ⇒ SP	X ⇒ SP	Y ⇒ SP	SP ⇒ SP
EXCHANGES									
⇓LS	MS⇒	8	9	A	B	C	D	E	F
0		A ⇔ A	B ⇔ A	CCR ⇔ A	TMP3$_L$ ⇒ A \$00:A ⇒ TMP3	B ⇒ A A ⇒ B	X$_L$ ⇒ A \$00:A ⇒ X	Y$_L$ ⇒ A \$00:A ⇒ Y	SP$_L$ ⇒ A \$00:A ⇒ SP
1		A ⇔ B	B ⇔ B	CCR ⇔ B	TMP3$_L$ ⇒ B \$FF:B ⇒ TMP3	B ⇒ B \$FF ⇒ A	X$_L$ ⇒ B \$FF:B ⇒ X	Y$_L$ ⇒ B \$FF:B ⇒ Y	SP$_L$ ⇒ B \$FF:B ⇒ SP
2		A ⇔ CCR	B ⇔ CCR	CCR ⇔ CCR	TMP3$_L$ ⇒ CCR \$FF:CCR ⇒ TMP3	B ⇒ CCR \$FF:CCR ⇒ D	X$_L$ ⇒ CCR \$FF:CCR ⇒ X	Y$_L$ ⇒ CCR \$FF:CCR ⇒ Y	SP$_L$ ⇒ CCR \$FF:CCR ⇒ SP
3		\$00:A ⇒ TMP2 TMP2$_L$ ⇒ A	\$00:B ⇒ TMP2 TMP2$_L$ ⇒ B	\$00:CCR ⇒ TMP2 TMP2$_L$ ⇒ CCR	TMP3 ⇔ TMP2	D ⇔ TMP2	X ⇔ TMP2	Y ⇔ TMP2	SP ⇔ TMP2
4		\$00:A ⇒ D	\$00:B ⇒ D	\$00:CCR ⇒ D B ⇒ CCR	TMP3 ⇔ D	D ⇔ D	X ⇔ D	Y ⇔ D	SP ⇔ D
5		\$00:A ⇒ X X$_L$ ⇒ A	\$00:B ⇒ X X$_L$ ⇒ B	\$00:CCR ⇒ X X$_L$ ⇒ CCR	TMP3 ⇔ X	D ⇔ X	X ⇔ X	Y ⇔ X	SP ⇔ X
6		\$00:A ⇒ Y Y$_L$ ⇒ A	\$00:B ⇒ Y Y$_L$ ⇒ B	\$00:CCR ⇒ Y Y$_L$ ⇒ CCR	TMP3 ⇔ Y	D ⇔ Y	X ⇔ Y	Y ⇔ Y	SP ⇔ Y
7		\$00:A ⇒ SP SP$_L$ ⇒ A	\$00:B ⇒ SP SP$_L$ ⇒ B	\$00:CCR ⇒ SP SP$_L$ ⇒ CCR	TMP3 ⇔ SP	D ⇔ SP	X ⇔ SP	Y ⇔ SP	SP ⇔ SP

TMP2 and TMP3 registers are for factory use only.

Table A-6. Loop Primitive Postbyte Encoding (lb)

00 A DBEQ (+)	10 A DBEQ (−)	20 A DBNE (+)	30 A DBNE (−)	40 A TBEQ (+)	50 A TBEQ (−)	60 A TBNE (+)	70 A TBNE (−)	80 A IBEQ (+)	90 A IBEQ (−)	A0 A IBNE (+)	B0 A IBNE (−)
01 B DBEQ (+)	11 B DBEQ (−)	21 B DBNE (+)	31 B DBNE (−)	41 B TBEQ (+)	51 B TBEQ (−)	61 B TBNE (+)	71 B TBNE (−)	81 B IBEQ (+)	91 B IBEQ (−)	A1 B IBNE (+)	B1 B IBNE (−)
02 —	12 —	22 —	32 —	42 —	52 —	62 —	72 —	82 —	92 —	A2 —	B2 —
03 —	13 —	23 —	33 —	43 —	53 —	63 —	73 —	83 —	93 —	A3 —	B3 —
04 D DBEQ (+)	14 D DBEQ (−)	24 D DBNE (+)	34 D DBNE (−)	44 D TBEQ (+)	54 D TBEQ (−)	64 D TBNE (+)	74 D TBNE (−)	84 D IBEQ (+)	94 D IBEQ (−)	A4 D IBNE (+)	B4 D IBNE (−)
05 X DBEQ (+)	15 X DBEQ (−)	25 X DBNE (+)	35 X DBNE (−)	45 X TBEQ (+)	55 X TBEQ (−)	65 X TBNE (+)	75 X TBNE (−)	85 X IBEQ (+)	95 X IBEQ (−)	A5 X IBNE (+)	B5 X IBNE (−)
06 Y DBEQ (+)	16 Y DBEQ (−)	26 Y DBNE (+)	36 Y DBNE (−)	46 Y TBEQ (+)	56 Y TBEQ (−)	66 Y TBNE (+)	76 Y TBNE (−)	86 Y IBEQ (+)	96 Y IBEQ (−)	A6 Y IBNE (+)	B6 Y IBNE (−)
07 SP DBEQ (+)	17 SP DBEQ (−)	27 SP DBNE (+)	37 SP DBNE (−)	47 SP TBEQ (+)	57 SP TBEQ (−)	67 SP TBNE (+)	77 SP TBNE (−)	87 SP IBEQ (+)	97 SP IBEQ (−)	A7 SP IBNE (+)	B7 SP IBNE (−)

Key to Table A-6



• Use up all the bytes for one instruction, then go on to the next instruction.

C6 05	**⇒ LDAA #$05**	two-byte LDAA, IMM addressing mode
CE 20 00	**⇒ LDX #$2000**	three-byte LDX, IMM addressing mode
E6 01	**⇒ LDAB 1,X**	two to four-byte LDAB, IDX addressing mode. Operand 01 => 1,X, a 5b constant offset which uses only one postbyte
18 06	**⇒ ABA**	two-byte ABA, INH addressing mode
04 35 EE	**⇒ DBNE X,(-18)**	three-byte loop instruction Postbyte 35 indicates DBNE X, negative
3F	**⇒ SWI**	one-byte SWI, INH addressing mode