A MEMO ON THE SECRET FEATURES OF 6309

by Hirotsugu Kakugawa

(kakugawa@csl.hiroshima-u.ac.jp)

Computer Systems Lab., Information Engineering Course,

Graduate School of Engineering, Hiroshima Univ., Japan

1. \*\* INTRODUCTION \*\*

The CPU 6309 by HITACHI has secret features which is not written in its manual. The purpose of this memo is to introduce them. The features was originally reported in a magazine, Oh!FM (April 1988), which was written in Japanese. I did not tried all of the features reported in the article, but I report the features as far as I know.

HITACHI says in the manual of 6309 that 6309 is compatible with 6809, but some OS-9 hackers found that it has secret features.

It has following features:

1. More registers (additional two 8 bit accumulators, 8 bit

register, and a 16 bit register),

2. Two modes (6809 emulation mode and native mode),

3. Reduced execution cycles in native mode,

4. More instructions (16 bit x 16 bit multiplication, 32 bit / 16 bit

division, inter-registers operation, block transfer, bit

manipulating operation which is compatible with 6801 has, etc)

5. Error trap by illegal instruction, zero division.

I substituted 6309 for 6809 in my personal computer, and I changed OS9/6809 Level II such that the 6309 executes in native mode. I had to change the interrupt handling routine in the kernel. I implemented illegal instruction trap; I was really happy because most bugs are caught by trap handler.

In section 2, new registers are explained. In section 3, two modes of 6309 are explained. In section 4, trapping features of the 6309 is described. In section 5, new instructions are explained. In section 6, the instruction tables of the 6309 are shown.

2. \*\* NEW REGISTERS \*\*

The 6309 has some additional registers that 6809 does not.

1. The E register, the F register

These are 8 bit accumulators. Like the D register is a pair of

the A register and the B register, these two registers can be

used as a 16 bit accumulator. The pair of the E and the F

registers is called the W register.

In addition to that, pair of two 16 bit registers, the D register

and the W register, can be used as a 32 bit accumulator called the

Q register.

2. The V register

This a 16 bit register can be used only by TFR, inter-register

operation, etc. But even if the chip is reseted, contents of

this register does not change. Some people may use this

register to keep constant value (V for value).

3. The MD register

This is a 8 bit register to keep the mode and status of the chip.

The meaning of each bit is as follow.

Read value

bit 7 --- 1 is set if zero division happen.

bit 6 --- 1 is set if illegal instruction is fetched.

Write value

bit 1 --- The mode for FIRQ interrupt.

0 -> the the action for FIRQ is the same as that

of 6809.

1 -> the the action for FIRQ is the same as IRQ.

bit 0 --- The execution mode of 6309.

0 -> the emulation mode.

1 -> the native mode.

(When the chip is reseted, all bits are 0.)

3. \*\* TWO MODES OF THE 6309 \*\*

The 6309 has two modes, emulation mode and native mode, as described in the previous section. When the chip is reseted, the initial mode of 6309 is the emulation mode.

When the 6309 is in the emulation mode, the chip emulates the action of 6809. But we can use extended registers and extended operations in this mode. The 6309 executes instructions in the same cycles as 6809 does.

When the 6309 is in the native mode, it executes instructions in less cycles. And when the chip is interrupted (IRQ, for example), it pushes extended registers (PC, U, Y, X, DP, W, D, CC, in this order). If you want to use the 6309, you must rewrite interrupt handling routine (for example, the entry of system call of OS9).

4. \*\* TRAPPING \*\*

If the following two events happen, the trap is caused.

1. A illegal instruction is fetched.

2. A number is divided by zero.

The action of the 6309 when a trap is caused is:

1. Pushs the registers on the system stack.

(In the emulation mode, PC, U, Y, X, DP, B, A, CC, in this order

and in the the native mode, PC, U, Y, X, DP, W, B, A, CC in this

order)

2. Reads the trap vector address ($FFF0) and jumps to the vector.

Note that $FFF0 was reserved by 6809.

To check the reason of the trap, BITMD instruction is provided. This instruction is explained in a later section.

5. \*\* NEW INSTRUCTIONS \*\*

5.1 The Register Addressing Mode

To specify registers in TFR and EXG, the 6809 uses bit pattern of 4 bits. New registers of the 6309 are specified by bit patterns in TFR and EXG operations. In addition to that, the bit pattern is also used in instructions of inter-register operations. We call this bit pattern used to specify register "register addressing mode".

Bit patterns for new registres are as follows:

W -> 0110,

V -> 0111,

E -> 1110,

F -> 1111.

NOTE: even if the 6309 is in a emulation mode, the action for TFR of 6309 is different from that of the 6809 if new register is specified in operand. Some hackers found this fact and they guessed that the 6309 has secret registers. At last, they found many features.

5.2 Inter-Register Operations

Operations of 6809 are operations between register and immediate value or between register and memory. Therefore, we had to store value of register on memory if operation between two registers is necessary. But the 6309 has inter-register operation. Following operations are provided:

ADDR r0,r1 (ADD of two registers),

ADCR r0,r1 (ADC of two registers),

SUBR r0,r1 (SUB of two registers),

SBCR r0,r1 (SBC of two registers),

ANDR r0,r1 (AND of two registers),

ORR r0,r1 (OR of two registers),

EORR r0,r1 (EOR of two registers),

CMPR r0,r1 (CMP of two registers).

The register addressing mode is used to specify two registers. I do not remember exactly but the result is stored in r0, the register of the first operand. Please try and find the behavior of these instructions.

5.3 Block Transfer

Block transfer instructions are provided such as Z80 has. The TFM instruction requires source address and destination address and block size as its argument. One or two 16 bit registers (X/Y/U/S) are used to specify source and destination addresses. Block size to be transferred is specified by the W register. Four style is provided:

TFR r0+,r1+ (transferred in address is increasing order),

TFR r0-,r1- (transferred in address is decreasing order),

TFR r0+,r1 (poured into the same address, I/O port for instance),

TFR r0,r1+ (read from the same address, I/O port for instance).

I tried this instructions but I do not remember exactly. Operand registers are pointers of source/destination addresses (maybe). Please try and find the behavior of these instructions.

5.4 Multiplication And Division

The 6309 has MULD instruction which performs a 16bit x 16bit multiplication. We can use various addressing modes (immediate, direct, indexed, extend) The result is stored in the Q register.

Division instructions are also provided. The 6309 has two division instructions: 16bit / 8bit, 32bit / 16bit divisions. Various addressing modes (immediate, direct, indexed, extend) can be used.

Note: I forget where its result is stored. I tried these instructions. I remember that modulo is also computed. The quotient and the modulo are stored D and W resp., maybe. I'm not sure, sorry.

5.5 Bit Manipulation / Bit Transfer

The 6309 provides AIM, OIM, EIM, TIM instructions which are compatible with instructions of the Hitachi 6301 CPU. Read the manual of the 6301 to understand these instructions.

Instructions called BAND, BOR, BEOR, BIAND, BIOR, BIEOR, LDBT, STBT are provided. Behavior of thses instructions is that a logical operation is performed for n-th bit of a data in a memory (only direct mode is allowed) and m-th bit of a register, then the result is stored in the register. The format of the object is:

$11, x, (post byte), (operand).

The say that the post byte takes strange format. I do not understand these instructions. Sorry, please try.

5.6 Misc

To change modes of the 6309, we have to set the 0th bit of the MD register. To do this, the LDMD instruction is provided:

LDMD #n (where #n is a immediate n bit data)

When trap is caused, it is necessary to examine the reason of the trap. The BITMD instruction can be used for this purpose:

BITMD #n (where #n is a immediate n bit data)

The contents of the MD register and #n is ANDed, and changes the CC register (,maybe, I do not remember exactly). Once this instruction is executed, the 6th and the 7th bit of the MD register is CLEARED. Therefore, we can't examine the MD register.

Pushing and popping the W registers on/from stack:

PSHSW (Push the W register on the system stack),

PULSW (Pop the W register from the system stack),

PSHUW (Push the W register on the user stack),

PULUW (Pop the W register from the user stack).

6. \*\* INSTRUCTION TABLES \*\*

In this section, only additional instructions of the 6309 are shown.

How to read the following table :

The first column : + ... New instruction of 6309

(blank) ... a instruction of 6089/6309,

--Op-- : Operational code,

--Mnem-- : Mnemonic,

--Mode-- : Addressing mode,

--Cyc-- : Execution Cycles (Parenthesized value is the value

in the native mode),

--Len-- : Length of the instruction,

6.1 Instructions without pre-byte

--Op-- --Mnem-- --Mode-- --Cyc-- --Len --

$00 NEG DIRECT 6 (5) 2

+ $01 OIM DIRECT 6 3

+ $02 AIM DIRECT 6 3

$03 COM DIRECT 6 (5) 2

$04 LSR DIRECT 6 (5) 2

+ $05 EIM DIRECT 6 3

$06 ROR DIRECT 6 (5) 2

$07 ASR DIRECT 6 (5) 2

$08 ASL/LSL DIRECT 6 (5) 2

$09 ROL DIRECT 6 (5) 2

$0A DEC DIRECT 6 (5) 2

+ $0B TIM DIRECT 6 3

$0C INC DIRECT 6 (5) 2

$0D TST DIRECT 6 (4) 2

$0E JMP DIRECT 3 (2) 2

$0F CLR DIRECT 6 (5) 2

$10 (PREBYTE)

$11 (PREBYTE)

$12 NOP IMP 2 (1) 1

$13 SYNC IMP 2 (1) 1

+ $14 SEXW IMP 4 1

$16 LBRA REL 5 (4) 3

$17 LBSR REL 9 (7) 3

$19 DAA IMP 2 (1) 1

$1A ORCC IMMED 3 (2) 2

$1C ANDCC IMMED 3 2

$1D SEX IMP 2 (1) 1

$1E EXG REGIST 8 (5) 2

$1F TFR REGIST 6 (4) 2

$20 BRA REL 3 2

$21 BRN REL 3 2

$22 BHI REL 3 2

$23 BLS REL 3 2

$24 BHS/BCC REL 3 2

$25 BLO/BCS REL 3 2

$26 BNE REL 3 2

$27 BEQ REL 3 2

$28 BVC REL 3 2

$29 BVS REL 3 2

$2A BPL REL 3 2

$2B BMI REL 3 2

$2C BGE REL 3 2

$2D BLT REL 3 2

$2E BGT REL 3 2

$2F BLE REL 3 2

$30 LEAX REL 4+ 2+

$31 LEAY REL 4+ 2+

$32 LEAS REL 4+ 2+

$33 LEAU REL 4+ 2+

$34 PSHS REGIST 5+ (4+) 2

$35 PULS REGIST 5+ (4+) 2

$36 PSHU REGIST 5+ (4+) 2

$37 PULU REGIST 5+ (4+) 2

$39 RTS 5 (4) 1

$3A ABX IMP 3 (1) 1

$3B RTI IMP 6/15 (17) 1

$3C CWAI IMP 22 (20) 2

$3D MUL IMP 11 (10) 1

$3F SWI IMP 19 (21) 1

$40 NEGA IMP 2 (1) 1

$43 COMA IMP 2 (1) 1

$44 LSRA IMP 2 (1) 1

$46 RORA IMP 2 (1) 1

$47 ASRA IMP 2 (1) 1

$48 ASLA/LSLA IMP 2 (1) 1

$49 ROLA IMP 2 (1) 1

$4A DECA IMP 2 (1) 1

$4C INCA IMP 2 (1) 1

$4D TSTA IMP 2 (1) 1

$4F CLRA IMP 2 (1) 1

$50 NEGB IM P 2 (1) 1

$53 COMB IMP 2 (1) 1

$54 LSRB IMP 2 (1) 1

$56 RORB IMP 2 (1) 1

$57 ASRB IMP 2 (1) 1

$58 ASLB/LSLB IMP 2 (1) 1

$59 ROLB IMP 2 (1) 1

$5A ECB IMP 2 (1) 1

$5C NCB IMP 2 (1) 1

$5D STB IMP 2 (1) 1

$5F LRB IMP 2 (1) 1

$60 NEG INDEXD 6+ 2+

+ $61 OIM INDEXD 7+ 3+

+ $62 AIM INDEXD 7 3+

$63 COM INDEXD 6+ 2+

$64 LSR INDEXD 6+ 2+

+ $65 EIM INDEXD 7+ 3+

$66 ROR INDEXD 6+ 2+

$67 ASR INDEXD 6+ 2+

$68 ASL/LSL INDEXD 6+ 2+

$69 ROL INDEXD 6+ 2+

$6A DEC INDEXD 6+ 2+

+ $6B TIM INDEXD 7+ 3+

$6C INC INDEXD 6+ 2+

$6D TST INDEXD 6+ (5+) 2+

$6E JMP INDEXD 3+ 2+

$6F CLR INDEXD 6+ 2+

$70 NEG EXTEND 7 (6) 3

+ $71 OIM EXTEND 7 4

+ $72 AIM EXTEND 7 4

$73 COM EXTEND 7 (6) 3

$74 LSR EXTEND 7 (6) 3

+ $75 EIM EXTEND 7 4

$76 ROR EXTEND 7 (6) 3

$77 ASR EXTEND 7 (6) 3

$78 ASL/LSL EXTEND 7 (6) 3

$79 ROL EXTEND 7 (6) 3

$7A DEC EXTEND 7 (6) 3

+ $7B TIM EXTEND 5 4

$7C INC EXTEND 7 (6) 3

$7D TST EXTEND 7 (5) 3

$7E JMP EXTEND 4 (3) 3

$7F CLR EXTEND 7 (6) 3

$80 SUBA IMMED 2 2

$81 CMPA IMMED 2 2

$82 SBCA IMMED 2 2

$83 SUBD IMMED 4 (3) 3

$84 ANDA IMMED 2 2

$85 BITA IMMED 2 2

$86 LDA IMMED 2 2

$88 EORA IMMED 2 2

$89 ADCA IMMED 2 2

$8A ORA IMMED 2 2

$8B ADDA IMMED 2 2

$8C CMPX IMMED 4 (3) 3

$8D BSR IMMED 7 (6) 2

$8E LDX IMMED 3 3

$90 SUBA DIRECT 4 (3) 2

$91 CMPA DIRECT 4 (3) 2

$92 SBCA DIRECT 4 (3) 2

$93 SUBD DIRECT 6 (4) 3

$94 ANDA DIRECT 4 (3) 2

$95 BITA DIRECT 4 (3) 2

$96 LDA DIRECT 4 (3) 2

$97 STA DIRECT 4 (3) 2

$98 EORA DIRECT 4 (3) 2

$99 ADCA DIRECT 4 (3) 2

$9A ORA DIRECT 4 (3) 2

$9B ADDA DIRECT 4 (3) 2

$9C CMPX DIRECT 6 (4) 2

$9D JSR DIRECT 7 (6) 2

$9E LDX DIRECT 5 (4) 2

$9F STX DIRECT 5 (4) 2

$A0 SUBA INDEXD 4+ 2+

$A1 CMPA INDEXD 4+ 2+

$A2 SBCA INDEXD 4+ 2+

$A3 SUBD INDEXD 6+ (5+) 2+

$A4 ANDA INDEXD 4+ 2+

$A5 BITA INDEXD 4+ 2+

$A6 LDA INDEXD 4+ 2+

$A7 STA INDEXD 4+ 2+

$A8 EORA INDEXD 4+ 2+

$A9 ADCA INDEXD 4+ 2+

$AA ORA INDEXD 4+ 2+

$AB ADDA INDEXD 4+ 2+

$AC CMPX INDEXD 6+ (5+) 2+

$AD JSR INDEXD 7+ (6+) 2+

$AE LDX INDEXD 5+ 2+

$AF STX INDEXD 5+ 2+

$B0 SUBA EXTEND 5 (4) 3

$B1 CMPA EXTEND 5 (4) 3

$B2 SBCA EXTEND 5 (4) 3

$B3 SUBD EXTEND 7 (5) 3

$B4 ANDA EXTEND 5 (4) 3

$B5 BITA EXTEND 5 (4) 3

$B6 LDA EXTEND 5 (4) 3

$B7 STA EXTEND 5 (4) 3

$B8 EORA EXTEND 5 (4) 3

$B9 ADCA EXTEND 5 (4) 3

$BA ORA EXTEND 5 (4) 3

$BB ADDA EXTEND 5 (4) 3

$BC CMPX EXTEND 7 (5) 3

$BD JSR EXTEND 8 (7) 3

$BE LDX EXTEND 6 (5) 3

$BF STX EXTEND 6 (5) 3

$C0 SUBB IMMED 2 2

$C1 CMPB IMMED 2 2

$C2 SBCB IMMED 2 2

$C3 ADDD IMMED 4 (3) 3

$C4 ANDB IMMED 2 2

$C5 BITB IMMED 2 2

$C6 LDB IMMED 2 2

$C8 EORB IMMED 2 2

$C9 ADCB IMMED 2 2

$CA ORB IMMED 2 2

$CB ADDB IMMED 2 2

$CC LDD IMMED 3 3

+ $CD LDQ IMMED 5 5

$CE LDU IMMED 3 3

$D0 SUBB DIRECT 4 (3) 2

$D1 CMPB DIRECT 4 (3) 2

$D2 SBCB DIRECT 4 (3) 2

$D3 ADDD DIRECT 6 (4) 3

$D4 ANDB DIRECT 4 (3) 2

$D5 BITB DIRECT 4 (3) 2

$D6 LDB DIRECT 4 (3) 2

$D7 STB DIRECT 4 (3) 2

$D8 EORB DIRECT 4 (3) 2

$D9 ADCB DIRECT 4 (3) 2

$DA ORB DIRECT 4 (3) 2

$DB ADDB DIRECT 4 (3) 2

$DC LDD DIRECT 5 (4) 2

$DD STD DIRECT 5 (4) 2

$DE LDU DIRECT 5 (4) 2

$DF STU DIRECT 5 (4) 2

$E0 SUBB INDEXD 4+ 2+

$E1 CMPB INDEXD 4+ 2+

$E2 SBCB INDEXD 4+ 2+

$E3 ADDD INDEXD 6+ (5+) 2+

$E4 ANDB INDEXD 4+ 2+

$E5 BITB INDEXD 4+ 2+

$E6 LDB INDEXD 4+ 2+

$E7 STB INDEXD 4+ 2+

$E8 EORB INDEXD 4+ 2+

$E9 ADCB INDEXD 4+ 2+

$EA ORB INDEXD 4+ 2+

$EB ADDB INDEXD 4+ 2+

$EC LDD INDEXD 5+ 2+

$ED STD INDEXD 5+ 2+

$EE LDU INDEXD 5+ 2+

$EF STU INDEXD 5+ 2+

$F0 SUBB EXTEND 5 (4) 3

$F1 CMPB EXTEND 5 (4) 3

$F2 SBCB EXTEND 5 (4) 3

$F3 ADDD EXTEND 7 (5) 3

$F4 ANDB EXTEND 5 (4) 3

$F5 BITB EXTEND 5 (4) 3

$F6 LDB EXTEND 5 (4) 3

$F7 STB EXTEND 5 (4) 3

$F8 EORB EXTEND 5 (4) 3

$F9 ADCB EXTEND 5 (4) 3

$FA ORB EXTEND 5 (4) 3

$FB ADDB EXTEND 5 (4) 3

$FC LDD EXTEND 6 (5) 3

$FD STD EXTEND 6 (5) 3

$FE LDU EXTEND 6 (5) 3

$FF STU EXTEND 6 (5) 3

6.2 Instructions whose pre-byte is $10

--Op-- --Mnem-- --Mode-- --Cyc-- --Len --

$21 LBRN REL 5 4

$22 LBHI REL 5/6 (5) 4

$23 LBLS REL 5/6 (5) 4

$24 LBHS/LBCC REL 5/6 (5) 4

$25 LBLO/LBCS REL 5/6 (5) 4

$26 LBNE REL 5/6 (5) 4

$27 LBEQ REL 5/6 (5) 4

$28 LBVC REL 5/6 (5) 4

$29 LBVS REL 5/6 (5) 4

$2A LBPL REL 5/6 (5) 4

$2B LBMI REL 5/6 (5) 4

$2C LBGE REL 5/6 (5) 4

$2D LBLT REL 5/6 (5) 4

$2E LBGT REL 5/6 (5) 4

$2F LBLE REL 5/6 (5) 4

+ $30 ADDR REGIST 4 3

+ $31 ADCR REGIST 4 3

+ $32 SUBR REGIST 4 3

+ $33 SBCR REGIST 4 3

+ $34 ANDR REGIST 4 3

+ $35 ORR REGIST 4 3

+ $36 EORR REGIST 4 3

+ $37 CMPR REGIST 4 3

+ $38 PSHSW IMP 6 2

+ $39 PULSW IMP 6 2

+ $3A PSHUW IMP 6 2

+ $3B PULUW IMP 6 2

$3F SWI2 IMP 20 (22) 2

+ $40 NEGD IMP 3 (2) 2

+ $43 COMD IMP 3 (2) 2

+ $44 LSRD IMP 3 (2) 2

+ $46 RORD IMP 3 (2) 2

+ $47 ASRD IMP 3 (2) 2

+ $48 ASLD IMP 3 (2) 2

+ $49 ROLD IMP 3 (2) 2

+ $4A DECD IMP 3 (2) 2

+ $4C INCD IMP 3 (2) 2

+ $4D TSTD IMP 3 (2) 2

+ $4F CLRD IMP 3 (2) 2

+ $53 COMW IMP 3 (2) 2

+ $54 LSRW IMP 3 (2) 2

+ $56 RORW IMP 3 (2) 2

+ $59 ROLW IMP 3 (2) 2

+ $5A DECW IMP 3 (2) 2

+ $5C INCW IMP 3 (2) 2

+ $5D TSTW IMP 3 (2) 2

+ $5F CLRW IMP 3 (2) 2

+ $80 SUBW IMMED 5 (4) 4

+ $81 CMPW IMMED 5 (4) 4

+ $82 SBCD IMMED 5 (4) 4

$83 CMPD IMMED 5 (4) 4

+ $84 ANDD IMMED 5 (4) 4

+ $85 BITD IMMED 5 (4) 4

+ $86 LDW IMMED 4 4

+ $88 EORD IMMED 5 (4) 4

+ $89 ADCD IMMED 5 (4) 4

+ $8A ORD IMMED 5 (4) 4

+ $8B ADDW IMMED 5 (4) 4

$8C CMPY IMMED 5 (4) 4

$8E LDY IMMED 4 4

+ $90 SUBW DIRECT 7 (5) 3

+ $91 CMPW DIRECT 7 (5) 3

+ $92 SBCD DIRECT 7 (5) 3

$93 CMPD DIRECT 7 (5) 3

+ $94 ANDD DIRECT 7 (5) 3

+ $95 BITD DIRECT 7 (5) 3

+ $96 LDW DIRECT 6 (5) 3

+ $97 STW DIRECT 6 (5) 3

+ $98 EORD DIRECT 7 (5) 3

+ $99 ADCD DIRECT 7 (5) 3

+ $9A ORD DIRECT 7 (5) 3

+ $9B ADDW DIRECT 7 (5) 3

$9C CMPY DIRECT 7 (5) 3

$9E LDY DIRECT 6 (5) 3

$9F STY DIRECT 6 (5) 3

+ $A0 SUBW INDEXD 7+ (6+) 3+

+ $A1 CMPW INDEXD 7+ (6+) 3+

+ $A2 SBCD INDEXD 7+ (6+) 3+

$A3 CMPD INDEXD 7+ (6+) 3+

+ $A4 ANDD INDEXD 7+ (6+) 3+

+ $A5 BITD INDEXD 7+ (6+) 3+

+ $A6 LDW INDEXD 6+ 3+

+ $A7 STW INDEXD 6+ 3+

+ $A8 EORD INDEXD 7+ (6+) 3+

+ $A9 ADCD INDEXD 7+ (6+) 3+

+ $AA ORD INDEXD 7+ (6+) 3+

+ $AB ADDW INDEXD 7+ (6+) 3+

$AC CMPY INDEXD 7+ (6+) 3+

$AE LDY INDEXD 6+ 3+

$AF STY INDEXD 6+ 3+

+ $B0 SUBW EXTEND 8 (6) 4

+ $B1 CMPW EXTEND 8 (6) 4

+ $B2 SBCD EXTEND 8 (6) 4

$B3 CMPD EXTEND 8 (6) 4

+ $B4 ANDD EXTEND 8 (6) 4

+ $B5 BITD EXTEND 8 (6) 4

+ $B6 LDW EXTEND 7 (6) 4

+ $B7 STW EXTEND 7 (6) 4

+ $B8 EORD EXTEND 8 (6) 4

+ $B9 ADCD EXTEND 8 (6) 4

+ $BA ORD EXTEND 8 (6) 4

+ $BB ADDW EXTEND 8 (6) 4

$BC CMPY EXTEND 8 (6) 4

$BE LDY EXTEND 7 (6) 4

$BF STY EXTEND 7 (6) 4

$CE LDS IMMED 4 4

+ $DC LDQ DIRECT 8 (7) 3

+ $DD STQ DIRECT 8 (7) 3

$DE LDS DIRECT 6 (5) 3

$DF STS DIRECT 6 (5) 3

+ $EC LDQ INDEXD 8+ 3+

+ $ED STQ INDEXD 8+ 3+

$EE LDS INDEXD 6+ 3+

$EF STS INDEXD 6+ 3+

+ $FC LDQ EXTEND 9 (8) 4

+ $FD STQ EXTEND 9 (8) 4

$FE LDS EXTEND 7 (6) 4

$FF STS EXTEND 7 (6) 4

6.3 Instructions whose pre-byte is $11

--Op-- --Mnem-- --Mode-- --Cyc-- --Len --

+ $30 BAND 7 (6) 4

+ $31 BIAND 7 (6) 4

+ $32 BOR 7 (6) 4

+ $33 BIOR 7 (6) 4

+ $34 NEOR 7 (6) 4

+ $35 BIEOR 7 (6) 4

+ $36 LDBT 7 (6) 4

+ $37 STBT 8 (7) 4

+ $38 TFR (r1+,r2+) 6+3n 3

+ $39 TFR (r1-,r2-) 6+3n 3

+ $3A TFR (r1+,r) 6+3n 3

+ $3B TFR (r1,r2+) 6+3n 3

+ $3C BITMD IMMED 4 3

+ $3D LDMD IMMED 5 3

$3F SWI2 IMP 20 (22) 2

+ $43 COME IMP 3 (2) 2

+ $4A DECE IMP 3 (2) 2

+ $4C INCE IMP 3 (2) 2

+ $4D TSTE IMP 3 (2) 2

+ $4F CLRE IMP 3 (2) 2

+ $53 COMF IMP 3 (2) 2

+ $5A DECF IMP 3 (2) 2

+ $5C INCF IMP 3 (2) 2

+ $5D TSTF IMP 3 (2) 2

+ $5F CLRF IMP 3 (2) 2

+ $80 SUBE IMMED 3 3

+ $81 CMPE IMMED 3 3

$83 CMPU IMMED 5 (4) 4

+ $86 LDE IMMED 3 3

+ $8B ADDE IMMED 3 3

$8C CMPS IMMED 5 (4) 4

+ $8D DIVD IMMED 25 3

+ $8E DIVQ IMMED 34 4

+ $8F MULD IMMED 28 4

+ $90 SUBE DIRECT 5 (4) 3

+ $91 CMPE DIRECT 5 (4) 3

$93 CMPU DIRECT 7 (5) 3

+ $96 LDE DIRECT 5 (4) 3

+ $97 STE DIRECT 5 (4) 3

+ $9B ADDE DIRECT 5 (4) 3

$9C CMPS DIRECT 7 (5) 3

+ $9D DIVD DIRECT 27 (26) 3

+ $9E DIVQ DIRECT 36 (35) 3

+ $9F MULD DIRECT 30 (29) 3

+ $A0 SUBE INDEXD 5+ 3+

+ $A1 CMPE INDEXD 5+ 3+

$A3 CMPU INDEXD 7+ (6+) 3+

+ $A6 LDE INDEXD 5+ 3+

+ $A7 STE INDEXD 5+ 3+

+ $AB ADDE INDEXD 5+ 3+

$AC CMPS INDEXD 7+ (6+) 3+

+ $AD DIVD INDEXD 27+ 3+

+ $AE DIVQ INDEXD 36+ 3+

+ $AF MULD INDEXD 30+ 3+

+ $B0 SUBE EXTEND 6 (5) 4

+ $B1 CMPE EXTEND 6 (5) 4

$B3 CMPU EXTEND 8 (6) 4

+ $B6 LDE EXTEND 6 (5) 4

+ $B7 STE EXTEND 6 (5) 4

+ $BB ADDE EXTEND 6 (5) 4

$BC CMPS EXTEND 8 (6) 4

+ $BD DIVD EXTEND 28 (27) 4

+ $BE DIVQ EXTEND 37 (36) 4

+ $BF MULD EXTEND 31 (30) 4

+ $C0 SUBF IMMED 3 3

+ $C1 CMPF IMMED 3 3

+ $C6 LDF IMMED 3 3

+ $CB ADDF IMMED 3 3

+ $D0 SUBF DIRECT 5 (4) 3

+ $D1 CMPF DIRECT 5 (4) 3

+ $D6 LDF DIRECT 5 (4) 3

+ $D7 STF DIRECT 5 (4) 3

+ $DB ADDF DIRECT 5 (4) 3

+ $E0 SUBF INDEXD 5+ 3+

+ $E1 CMPF INDEXD 5+ 3+

+ $E6 LDF INDEXD 5+ 3+

+ $E7 STF INDEXD 5+ 3+

+ $EB ADDF INDEXD 5+ 3+

+ $F0 SUBF EXTEND 6 (5) 4

+ $F1 CMPF EXTEND 6 (5) 4

+ $F6 LDF EXTEND 6 (5) 4

+ $F7 STF EXTEND 6 (5) 4

+ $FB ADDF EXTEND 6 (5) 4