UNIVERSITI SAINS MALAYSIA

Peperiksaan Semester Kedua
Sidang Akademik 1992/93

April 1993

IQK 205/3 - TEKNOLOGI MIKROPEMPROSES

Masa : [3 jam]

Sila pastikan bahawa kertas soalan ini mengandungi TUJUH BELAS (17) mukasurat (termasuk lampiran) yang bercetak sebelum anda memulakan peperiksaan ini.

Jawab LIMA (5) soalan. Semua soalan mesti dijawab di dalam Bahasa Malaysia.

Nota: Appendiks mengandungi

(i) Perincian set-set suruhan

(ii) Perincian bagi pendaftar-pendaftar khusus dan vektor-vektor sampuk.

1. (a) Model programan bagi suatu mikropemproses mengandungi pembilang program (PC) 20-bit, empat daftar indeks 20-bit (W, X, Y, Z), dua penumpuk (accumulator) 16-bit (A dan B).

(i) Berapa bitkah yang dikandungi oleh alamat (address bus)?

(ii) Berapa bitkah yang dikandungi oleh bas data?

(iii) Berapakah lokasi ingatan nyata yang boleh dialamatkan oleh pemproses ini?

(iv) Apakah julat alamat di dalam hex?

(v) Apakah julat (single Precision) bagi nombor-nombor binari tak bertanda bagi mesin ini? (Sila beri jawapan di dalam desimal).

(vi) Apakah julat (single Precision) bagi nombor-nombor BCD tak bertanda bagi mesin ini? (Sila beri jawapan di dalam desimal).

(vii) Apakah julat (single Precision) bagi nombor-nombor binari pelengkap 2 (two's complement) bagi mesin ini? (Sila beri jawapan di dalam desimal).

(50 markah)

1. (b) Secara ringkas jelaskan mod-mod alamat berikut, beri satu contoh suruhan di dalam bahasa penghimpun dan bahasa mesin bagi setiap satu.

(i) Pengalamatan terus (Direct addressing)

(ii) Pengalamatan terpanjang (Extended addressing)

(iii) Pengalamatan indeks dengan daftar Y.

(50 markah)

2. Tulis satu program di dalam bahasa penghimpun MC 68HC11 untuk menukar tempat (swap) dua jadual data (dengan panjang yang sama) yang disimpan di dalam ingatan. Anda boleh memilih sendiri alamat-alamat permulaan bagi jadual dan panjang jadual tersebut. Tukarkan program anda ke bahasa mesin (Guna Appendiks).

NOTA: Anda dikehendaki melukis carta alir (flow-chart).

(100 markah)

3. (a) Tulis satu suruhan atau satu siri suruhan di dalam bahasa penghimpun untuk mencapai tiap-tiap yang berikut. Buat anggapan-anggapan yang perlu mengenai lokasi-lokasi ingatan untuk memudahkan penulisan aturcara bahasa mesin.

(i) Bersihkan (clear) bit-bit 2, 5, 7 di dalam daftar B. Jangan ubah bit-bit yang lain.

3. (a) (ii) Lengkapkan bit-bit (complement bits) 2, 3, 5 pada ACC A. Jangan ubahkan bit-bit yang lain.

(iii) Load A dengan $00 jika B mengandungi nombor positif atau dengan $FF jika B mengandungi nombor negatif.

(iv) Bahagikan nombor di dalam ACC B dengan 2 dan tempatkan bakinya di dalam bit carry.

(50 markah)

(b) Tentukan kandungan hex bagi daftar-daftar yang dinyatakan selepas pelaksanaan setiap suruhan-suruhan berikut. Guna keadaan-keadaan yang sama yang diberi sebelum menjawab setiap bahagian.

Nilai-Nilai Daftar Pemproses (Semua nilai-nilai di dalam HEX)

A	=	A7
B	=	65
X	=	1000
Y	=	1000
PC	=	0200
CC	=	OF
SP	=	01FF

Lokasi-Lokasi Ingatan

1023	=	B5
1024	=	EF
1025	=	35
1026	=	51
1027	=	24
1028	=	83

<u>Suruhan-suruhan</u>

(i) ASRA

(ii) LSRA

(iii) PSHB

(iv) BRSET $23, X $34 $12

(v) BSET $28, Y $15

(50 markah)

4. (a) Terangkan mod output 'simple strobe' yang digunakan di dalam 68HC11 bagi operasi-operasi output.

(25 markah)

(b) Tulis satu program untuk memaparkan kandungan lima lokasi ingatan berturutan pada LED yang disambung ke Port B bagi tempoh minimum 1 ms. Anggapkan jam-E beroperasi pada 1 MHz.

(75 markah)

5. (a) Senaraikan 5 jenis sampuk (interrupt) yang terdapat pada MC68HC11 MPU dan beri keterangan yang ringkas mengenainya. Sampuk manakah yang tidak diperdulikan (ignored) oleh MPU jika bit topeng (mask bit) disetkan?

(50 markah)

(b) Senaraikan turutan-turutan kejadian yang berlaku apabila 'interrupt request IRQ' terjadi. (Guna gambarajah alir).

(50 markah)

6. (a) Secara ringkas terangkan sifat-sifat Perangkapan Input (Input Capture) dan Perbandingan Output (Output Compare) yang terdapat pada Modul Timer MC68HC11.

(40 markah)

(b) Terangkan fungsi setiap langkah di dalam program berikut. Apakah yang dicapai oleh program ini. Rujuk kepada Appendiks bagi keterangan yang diperlukan.

```
        LDX  #$1000
        BSET $21, X, $20
        BCLR $21, X, $10
        LDA #4
        STAA $23, X
L1      BRCLR $23, X 4 L1
        LDY $10, X
        STAA $23, X
L2      BRCLR $23, X 4 L2
        LDD $10, X
        STY TEMP
        SUBD TEMP
```

(60 markah)

oooooooooooooooooooooooooooo

Table 3–2. Instruction Set (1 of 5)

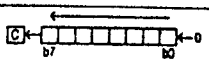
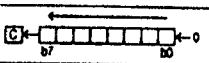
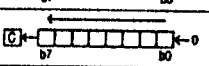
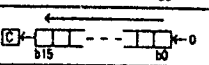
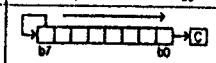
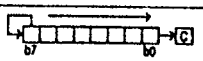
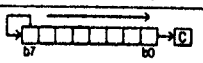
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
ABA	Add Accumulators	A + B ⇒ A	INH	1B	—	2	—	—	Δ	—	Δ	Δ	Δ	Δ
ABX	Add B to X	IX + (00 : B) ⇒ IX	INH	3A	—	3	—	—	—	—	—	—	—	—
ABY	Add B to Y	IY + (00 : B) ⇒ IY	INH	18 3A	—	4	—	—	—	—	—	—	—	—
ADCA (opr)	Add with Carry to A	A + M + C ⇒ A	A IMM	89	ii	2	—	—	Δ	—	Δ	Δ	Δ	Δ
			A DIR	99	dd	3								
			A EXT	B9	hh ll	4								
			A IND,X	A9	ff	4								
			A IND,Y	18 A9	ff	5								
ADCB (opr)	Add with Carry to B	B + M + C ⇒ B	B IMM	C9	ii	2	—	—	Δ	—	Δ	Δ	Δ	Δ
			B DIR	D9	dd	3								
			B EXT	F9	hh ll	4								
			B IND,X	E9	ff	4								
			B IND,Y	18 E9	ff	5								
ADDA (opr)	Add Memory to A	A + M ⇒ A	A IMM	8B	ii	2	—	—	Δ	—	Δ	Δ	Δ	Δ
			A DIR	9B	dd	3								
			A EXT	BB	hh ll	4								
			A IND,X	AB	ff	4								
			A IND,Y	18 AB	ff	5								
ADDB (opr)	Add Memory to B	B + M ⇒ B	B IMM	CB	ii	2	—	—	Δ	—	Δ	Δ	Δ	Δ
			B DIR	DB	dd	3								
			B EXT	FB	hh ll	4								
			B IND,X	EB	ff	4								
			B IND,Y	18 EB	ff	5								
ADDD (opr)	Add 16-Bit to D	D + (M : M + 1) ⇒ D	IMM	C3	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			DIR	D3	dd	5								
			EXT	F3	hh ll	6								
			IND,X	E3	ff	6								
			IND,Y	18 E3	ff	7								
ANDA (opr)	AND A with Memory	A • M ⇒ A	A IMM	84	ii	2	—	—	—	—	Δ	Δ	0	—
			A DIR	94	dd	3								
			A EXT	B4	hh ll	4								
			A IND,X	A4	ff	4								
			A IND,Y	18 A4	ff	5								
ANDB (opr)	AND B with Memory	B • M ⇒ B	B IMM	C4	ii	2	—	—	—	—	Δ	Δ	0	—
			B DIR	D4	dd	3								
			B EXT	F4	hh ll	4								
			B IND,X	E4	ff	4								
			B IND,Y	18 E4	ff	5								
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND,X	68	ff	6								
			IND,Y	18 68	ff	7								
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
ASR	Arithmetic Shift Right		EXT	77	hh ll	6	—	—	—	—	Δ	Δ	Δ	Δ
			IND,X	67	ff	6								
			IND,Y	18 67	ff	7								
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 • ($\overline{mm}$) ⇒ M	DIR	15	dd mm	6	—	—	—	—	Δ	Δ	0	—
			IND,X	1D	ff mm	7								
			IND,Y	18 1D	ff mm	8								
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 ≥ Zero	? N ⊕ V = 0	REL	2C	rr	3	—	—	—	—	—	—	—	—
BGT (rel)	Branch if > Zero	? Z + (N ⊕ 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	85	ii	2	—	—	—	—	Δ	Δ	0	—
			A DIR	95	dd	3								
			A EXT	B5	hh ll	4								
			A IND,X	A5	ff	4								
			A IND,Y	18 A5	ff	5								
BITB (opr)	Bit(s) Test B with Memory	B • M	B IMM	C5	ii	2	—	—	—	—	Δ	Δ	0	—
			B DIR	D5	dd	3								
			B EXT	F5	hh ll	4								
			B IND,X	E5	ff	4								
			B IND,Y	18 E5	ff	5								

Table 3–2. Instruction Set (2 of 5)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
BLE (rel)	Branch if ≤ Zero	? Z + (N ⊕ 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 ⊕ 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	? (M) • 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 ⇒ 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	—	—	—	—	—	—	—	—
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	—
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	—	—

Table 3–2. Instruction Set (3 of 5)

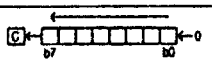
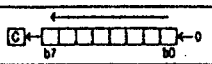
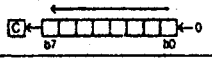
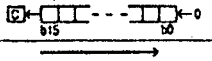
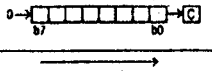
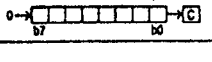
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
EORA (opr)	Exclusive OR A with Memory	A ⊕ M ⇒ A	A IMM	88	ii	2	—	—	—	—	Δ	Δ	0	—
			A DIR	98	dd	3								
			A EXT	B8	hh ll	4								
			A IND,X	A8	ff	4								
			A IND,Y	18 A8	ff	5								
EORB (opr)	Exclusive OR B with Memory	B ⊕ M ⇒ B	B IMM	C8	ii	2	—	—	—	—	Δ	Δ	0	—
			B DIR	D8	dd	3								
			B EXT	F8	hh ll	4								
			B IND,X	E8	ff	4								
			B IND,Y	18 E8	ff	5								
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	7C	hh ll	6	—	—	—	—	Δ	Δ	Δ	—
			IND,X	6C	ff	6								
			IND,Y	18 6C	ff	7								
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	7E	hh ll	3	—	—	—	—	—	—	—	—
			IND,X	6E	ff	3								
			IND,Y	18 6E	ff	4								
JSR (opr)	Jump to Subroutine	See Figure 3–2	DIR	9D	dd	5	—	—	—	—	—	—	—	—
			EXT	BD	hh ll	6								
			IND,X	AD	ff	6								
			IND,Y	18 AD	ff	7								
LDAA (opr)	Load Accumulator A	M ⇒ A	A IMM	86	ii	2	—	—	—	—	Δ	Δ	0	—
			A DIR	96	dd	3								
			A EXT	B6	hh ll	4								
			A IND,X	A6	ff	4								
			A IND,Y	18 A6	ff	5								
LDAB (opr)	Load Accumulator B	M ⇒ B	B IMM	C6	ii	2	—	—	—	—	Δ	Δ	0	—
			B DIR	D6	dd	3								
			B EXT	F6	hh ll	4								
			B IND,X	E6	ff	4								
			B IND,Y	18 E6	ff	5								
LDD (opr)	Load Double Accumulator D	M ⇒ A, M + 1 ⇒ B	IMM	CC	jj kk	3	—	—	—	—	Δ	Δ	0	—
			DIR	DC	dd	4								
			EXT	FC	hh ll	5								
			IND,X	EC	ff	5								
			IND,Y	18 EC	ff	6								
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

Table 3–2. Instruction Set (4 of 5)

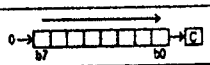
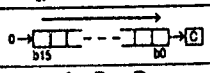
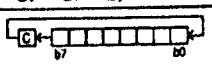
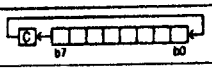
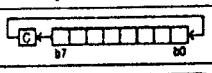
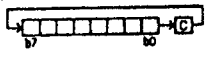
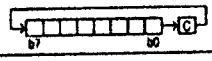
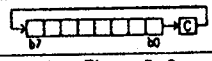
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
LSRB	Logical Shift Right B		B INH	54	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRD	Logical Shift Right Double		INH	04	—	3	—	—	—	—	0	Δ	Δ	Δ
MUL	Multiply 8 by 8	A • B ⇒ D	INH	3D	—	10	—	—	—	—	—	—	—	Δ
NEG (opr)	Twos Complement Memory Byte	0 – M ⇒ M	EXT IND,X IND,Y	70 60 18 60	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
NEGA	Twos Complement A	0 – A ⇒ A	A INH	40	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Twos 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 A DIR A EXT A IND,X A IND,Y	8A 9A BA AA 18 AA	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
ORAB (opr)	OR Accumulator B (Inclusive)	B + M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	CA DA FA EA 18 EA	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
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 IMM A DIR A EXT A IND,X A 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 IMM B DIR B EXT B IND,X B 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	OD	—	2	—	—	—	—	—	—	—	1
SEI	Set Interrupt Mask	1 ⇒ I	INH	OF	—	2	—	—	—	1	—	—	—	—
SEV	Set Overflow Flag	1 ⇒ V	INH	O8	—	2	—	—	—	—	—	—	1	—
STAA (opr)	Store Accumulator A	A ⇒ M	A DIR A EXT A IND,X A 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 DIR B EXT B IND,X B IND,Y	D7 F7 E7 18 E7	dd hh ll ff ff	3 4 4 5	—	—	—	—	Δ	Δ	0	—

Table 3–2. Instruction Set (5 of 5)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
STD (opr)	Store Accumulator D	A ⇒ M, B ⇒ M + 1	DIR	DD	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	FD	hh ll	5								
			IND,X	ED	ff	5								
			IND,Y	18 ED	ff	6								
STOP	Stop Internal Clocks	—	INH	CF	—	2	—	—	—	—	—	—	—	—
STS (opr)	Store Stack Pointer	SP ⇒ M : M + 1	DIR	9F	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	BF	hh ll	5								
			IND,X	AF	ff	5								
			IND,Y	18 AF	ff	6								
STX (opr)	Store Index Register X	IX ⇒ M : M + 1	DIR	DF	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	FF	hh ll	5								
			IND,X	EF	ff	5								
			IND,Y	CD EF	ff	6								
STY (opr)	Store Index Register Y	IY ⇒ M : M + 1	DIR	18 DF	dd	5	—	—	—	—	Δ	Δ	0	—
			EXT	18 FF	hh ll	6								
			IND,X	1A EF	ff	6								
			IND,Y	18 EF	ff	6								
SUBA (opr)	Subtract Memory from A	A – M ⇒ A	A IMM	80	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			A DIR	90	dd	3								
			A EXT	B0	hh ll	4								
			A IND,X	A0	ff	4								
			A IND,Y	18 A0	ff	5								
SUBB (opr)	Subtract Memory from B	B – M ⇒ B	A IMM	C0	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			A DIR	D0	dd	3								
			A EXT	F0	hh ll	4								
			A IND,X	E0	ff	4								
			A IND,Y	18 E0	ff	5								
SUBD (opr)	Subtract Memory from D	D – M : M + 1 ⇒ D	IMM	83	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			DIR	93	dd	5								
			EXT	B3	hh ll	6								
			IND,X	A3	ff	6								
			IND,Y	18 A3	ff	7								
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	7D	hh ll	6	—	—	—	—	Δ	Δ	0	0
			IND,X	6D	ff	6								
			IND,Y	18 6D	ff	7								
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	—	—	—	—	—	—	—	—
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 (Twos 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

Table 4–1. Register and Control Bit Assignments (1 of 2)

The register block can be remapped to any 4K boundary

	Bit 7	6	5	4	3	2	1	Bit 0	
$1000	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PORTA
$1001									Reserved
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC
$1003	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
$1004	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
$1005	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0	PORTCL
$1006									Reserved
$1007	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
$1008	0	0	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
$1009	0	0	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
$100A	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	PORTE
$100B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D
$100E	Bit 15	14	13	12	11	10	9	Bit 8	TCNT (High)
$100F	Bit 7	6	5	4	3	2	1	Bit 0	TCNT (Low)
$1010	Bit 15	14	13	12	11	10	9	Bit 8	TIC1 (High)
$1011	Bit 7	6	5	4	3	2	1	Bit 0	TIC1 (Low)
$1012	Bit 15	14	13	12	11	10	9	Bit 8	TIC2 (High)
$1013	Bit 7	6	5	4	3	2	1	Bit 0	TIC2 (Low)
$1014	Bit 15	14	13	12	11	10	9	Bit 8	TIC3 (High)
$1015	Bit 7	6	5	4	3	2	1	Bit 0	TIC3 (Low)
$1016	Bit 15	14	13	12	11	10	9	Bit 8	TOC1(High)
$1017	Bit 7	6	5	4	3	2	1	Bit 0	TOC1 (Low)
$1018	Bit 15	14	13	12	11	10	9	Bit 8	TOC2 (High)
$1019	Bit 7	6	5	4	3	2	1	Bit 0	TOC2 (Low)
$101A	Bit 15	14	13	12	11	10	9	Bit 8	TOC3 (High)
$101B	Bit 7	6	5	4	3	2	1	Bit 0	TOC3 (Low)
$101C	Bit 15	14	13	12	11	10	9	Bit 8	TOC4 (High)
$101D	Bit 7	6	5	4	3	2	1	Bit 0	TOC4 (Low)
$101E	Bit 15	14	13	12	11	10	9	Bit 8	TI4/O5 (High)
$101F	Bit 7	6	5	4	3	2	1	Bit 0	TI4/O5 (Low)
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1
$1021	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2
$1022	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I	TMSK1

Table 4–1. Register and Control Bit Assignments (2 of 2)

$1023	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F	TFLG1
$1024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2
$1025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2
$1026	DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0	PACTL
$1027	Bit 7	6	5	4	3	2	1	Bit 0	PACNT
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
$1029	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
$102A	Bit 7	6	5	4	3	2	1	Bit 0	SPDR
$102B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD
$102C	R8	T8	0	M	WAKE	0	0	0	SCCR1
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR
$102F	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0	SCDR
$1030	CCF	0	SCAN	MULT	CD	CC	CB	CA	ADCTL
$1031	Bit 7	6	5	4	3	2	1	Bit 0	ADR1
$1032	Bit 7	6	5	4	3	2	1	Bit 0	ADR2
$1033	Bit 7	6	5	4	3	2	1	Bit 0	ADR3
$1034	Bit 7	6	5	4	3	2	1	Bit 0	ADR4
$1035	0	0	0	PTCON	BPRT3	BPRT2	BPRT1	BPRT0	BPROT
$1036–8									Reserved
$1039	ADPU	CSEL	IRQE	DLY	CME	0	CR1	CR0	OPTION
$103A	Bit 7	6	5	4	3	2	1	Bit 0	COPRST
$103B	ODD	EVEN	ELAT	BYTE	ROW	ERASE	EELAT	EEPGM	PPROG
$103C	RBOOT	SMOD	MDA	IRVNE	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT
$103E	TILOP	0	OCCR	CBYP	DISR	FCM	FCOP	TCON	TEST1
$103F	0	0	0	0	NOSEC	NOCOP	ROMON	EEON	CONFIG

Hardware priority is built into RAM and I/O remapping. Registers have priority over RAM and RAM has priority over ROM. When the 64-byte register block is mapped at the same location as the RAM, a read of the dual-mapped location results in a read of the register. If RAM is relocated on ROM, RAM has priority.

Parallel I/O Control Register

The parallel handshake functions are available only in the single-chip operating mode. PIOC is a read/write register except for bit 7, which is read only. Table 6–2 shows a summary of handshake operations.

PIOC — Parallel I/O Control $1002

	Bit 7	6	5	4	3	2	1	Bit 0
	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB
RESET:	0	0	0	0	0	U	1	1

STAF — Strobe A Interrupt Status Flag

0 = No edge on strobe A

1 = Selected edge on strobe A

STAF is set when the selected edge occurs on Strobe A. This bit can be cleared by a read of PIOC with STAF set followed by a read of PORTCL (simple strobed or full input handshake mode) or a write to PORTCL (output handshake mode).

STAI — Strobe A Interrupt Enable Mask

0 = STAF does not request interrupt

1 = STAF requests interrupt

CWOM — Port C Wired-OR Mode (affects all eight port C pins)

0 = Port C outputs are normal CMOS outputs

1 = Port C outputs are open-drain outputs

HNDS — Handshake Mode

0 = Simple strobe mode

1 = Full input or output handshake mode

OIN — Output or Input Handshake Select

HNDS must be set to one for this bit to have meaning.

0 = Input handshake

1 = Output handshake

PLS — Pulsed/Interlocked Handshake Operation

HNDS must be set to one for this bit to have meaning. When interlocked handshake is selected, strobe B is active until the selected edge of strobe A is detected.

0 = Interlocked handshake

1 = Pulsed handshake (Strobe B pulses high for two E-clock cycles.)

TMSK1 — Timer Interrupt Mask 1

$1022

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I
RESET:	0	0	0	0	0	0	0	0

OC1I–OC4I — Output Compare x Interrupt Enable
If the OCxI enable bit is set when the OCxF flag bit is set, a hardware interrupt sequence is requested.

I4/O5I — Input Capture 4 or Output Compare 5 Interrupt Enable
When I4/O5 in PACTL is one, I4/O5I is the input capture 4 interrupt enable bit. When I4/O5 in PACTL is zero, I4/O5I is the output compare 5 interrupt enable bit.

IC1I–IC3I — Input Capture x Interrupt Enable
If the ICxI enable bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

TFLG1 — Timer Interrupt Flag 1

$1023

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F
RESET:	0	0	0	0	0	0	0	0

Clear flags by writing a one to the corresponding bit position(s).

OC1F–OC5F — Output Compare x Flag
Set each time the counter matches output compare x value.

I4/O5F — Input Capture 4/Output Compare 5 Flag
Set by IC4 or OC5, depending on the function enabled by I4/O5 bit in PACTL.

IC1F–IC3F — Input Capture x Flag
Set each time a selected active edge is detected on the ICx input line.

Timer Control Register 1

The bits of this register specify the action taken as a result of a successful OCx compare.

TCTL1 — Timer Control 1 $1020

	Bit 7	6	5	4	3	2	1	Bit 0
	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
RESET:	0	0	0	0	0	0	0	0

OM[2:5] — Output Mode
OL[2:5] — Output Level

These control bit pairs are encoded to specify the action taken after a successful OCx compare. OC5 functions only if the I4/O5 bit in the PACTL register is clear. Refer to the following table for the coding.

OMx	OLx	Action Taken on Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

TCTL2 — Timer Control 2 $1021

	Bit 7	6	5	4	3	2	1	Bit 0
	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A
RESET:	0	0	0	0	0	0	0	0

Table 9–2. Timer Control Configuration

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge

Table 2-3. Bootstrap Mode Interrupt Vectors

Address	Vector
00C4	SCI
00C7	SPI
00CA	Pulse Accumulator Input Edge
00CD	Pulse Accumulator Overflow
00D0	Timer Overflow
00D3	Timer Output Compare 5
00D6	Timer Output Compare 4
00D9	Timer Output Compare 3
00DC	Timer Output Compare 2
00DF	Timer Output Compare 1
00E2	Timer Input Capture 3
00E5	Timer Input Capture 2
00E8	Timer Input Capture 1
00EB	Real Time Interrupt
00EE	IRQ
00F1	XIRQ
00F4	SWI
00F7	Illegal Opcode
00FA	COP Fail
00FD	Clock Monitor
BF40 (Boot)	Reset