UNIVERSITI SAINS MALAYSIA

Peperiksaan Semester Kedua
Sidang Akademik 2002/2003

Februari – Mac 2003

ZAT 281/4 - Pengantar Mikropemproses

Masa : 3 jam

Sila pastikan bahawa kertas peperiksaan ini mengandungi **TUJUH BELAS** muka surat yang bercetak sebelum anda memulakan peperiksaan ini.

Jawab kesemua LIMA soalan. Kesemuanya wajib dijawab dalam Bahasa Malaysia. Kesemua jawapan mestilah ditulis diruang yang disediakan dalam kertas soalan ini. Ringkasan arahan dan masa pelaksanaan beberapa arahan mikropemproses 68000 disediakan di Lampiran A sebagai rujukan.

1. Rajah 1 merupakan suatu aturcara bahasa penghimpunan sistem mikropemproses 68000 yang dilaksanakan kepada sistem mikropemproses di makmal fizik gunaan USM, dengan pot input merupakan 8-bit suis dan pot output pula merupakan 8-bit LED.

```
PBDDR   EQU     $A00007                 ;arahan 1
PCDDR   EQU     $A00009                 ;arahan 2
PBDR    EQU     $A00013                 ;arahan 3
PCDR    EQU     $A00019                 ;arahan 4
DATA1   EQU     $03                     ;arahan 5
DATA2   EQU     $004004F0               ;arahan 6
        ORG     $400400                 ;arahan 7

MULA    MOVE.B  #$FF,PBDDR              ;arahan 8
        MOVE.B  #0,PCDDR                ;arahan 9
ULANG   MOVE.B  PCDR,D0                 ;arahan A
        MOVE.B  D0,PBDR                 ;arahan B
        MOVE.B  D0,D1                   ;arahan C
        ANDI.B  #DATA1,D0               ;arahan D
        BNE     ULANG                   ;arahan E
        MOVE.B  D1,DATA2                ;arahan F
        TRAP    #11                     ;arahan 10
        DC.W    0                       ;arahan 11
        END
```

Rajah 1

...2/-

a) Secara ringkas terangkan apakah yang dilaksanakan oleh aturcara tersebut?

.....
.....
.....
.....
.....
.....
.....
.....

(20/100)

b) Nyatakan lokasi permulaan ingatan RAM di mana aturcara tersebut ditulis.

.....
.....
.....

(10/100)

c) Nyatakan lokasi alamat suis input.

.....
.....
.....

(10/100)

d) Nyatakan lokasi alamat LED output

.....
.....
.....

(10/100)

e) Apakah yang akan dilaksanakan oleh aturcara sekiranya arahan E ditukar kepada

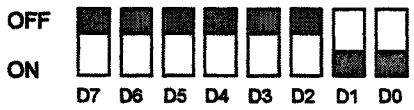
```
BNE     MULA                    ;arahan E?
```

.....
.....
.....
.....
.....
.....
.....

(10/100)

f) Sekiranya keadaan suis output pada keadaan yang ditunjukkan dalam Rajah 2 nyatakan kandungan di alamat DATA 2 apabila aturcara tersebut dilaksanakan,
(kawasan berlorek (■) menandakan keadaan suis berkenaan)

.....
.....
.....

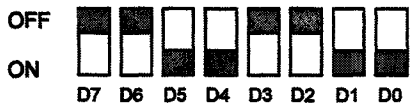


Rajah 2

(10/100)

g) Sekiranya keadaan suis output pada keadaan yang ditunjukkan dalam Rajah 3 nyatakan kandungan di alamat DATA 2 apabila aturcara tersebut dilaksanakan,
(kawasan berlorek (■) menandakan keadaan suis berkenaan)

.....
.....
.....



Rajah 3

(10/100)

h) Ubahsuai aturcara dalam Rajah 1, supaya semua LED dinyalakan apabila suis D7 pada keadaan 'ON'. Tuliskan aturcara lengkap untuk perlaksanaan keadaan tersebut.

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

(20/100)

2. Rajah 4 menunjukkan litar sistem mikropemproses dengan ingatan RAM, EPROM, SUIS, dan LED yang di antaramukakan dengan mikropemproses pada suatu lokasi alamat yang dinyahkodkan oleh penyahkod 74LS138. Berdasarkan Rajah 4 tersebut selesaikan persoalan berikut:

a) Saiz ingatan yang boleh diberikan oleh mikropemproses 68000 ialah

.....
.....

(10/100)

b) Bilangan blok ingatan yang dinyahkodkan oleh penyahkod 74LS138 ialah

.....
.....

(10/100)

c) Saiz setiap blok ingatan ialah

.....
.....

(10/100)

d) Jumlah keseluruhan ingatan EPROM mikropemproses tersebut ialah

.....
.....
.....

(10/100)

e) Ingatan EPROM tersebut dikatakan terpantul. Bilangan pantulannya ialah

.....
.....

(10/100)

f) Jumlah keseluruhan ingatan RAM mikropemproses tersebut ialah

.....
.....

(10/100)

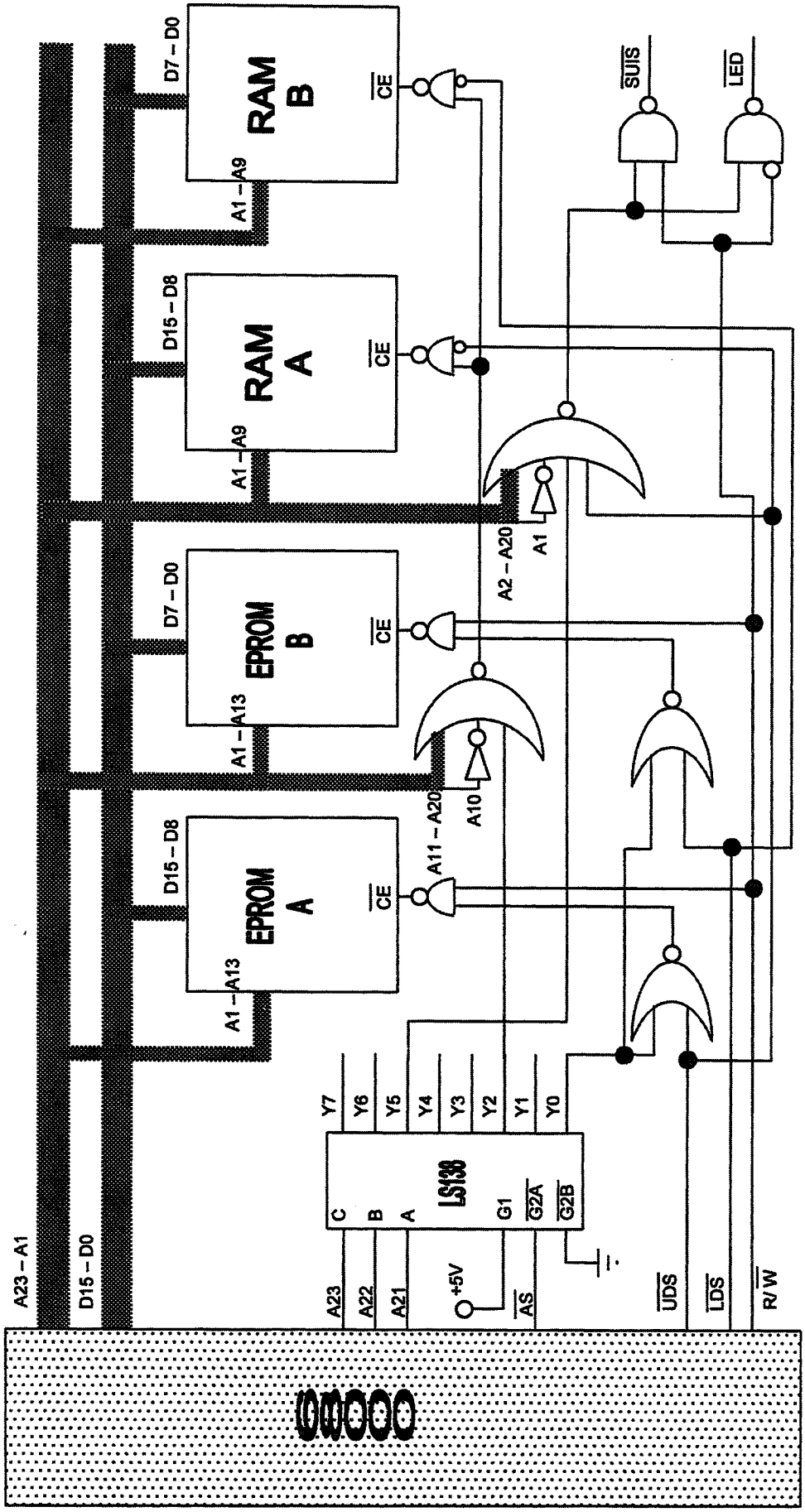
g) Lokasi alamat ingatan RAM ialah

.....
.....

(10/100)

h) Lokasi alamat SUIS dan LED ialah

.....
.....
.....


RAM B
RAM A
EPROM B
EPROM A
D7 – D0
D15 – D8
A1 – A9
A1 – A13
CE
A2 – A20
A1
A11 – A20
A10
SUIS
LED
LS138
C
B
A
G1
G2A
G2B
Y7
Y6
Y5
Y4
Y3
Y2
Y1
Y0
A23
A22
A21
+5V
AS
UDS
LDS
R/W
A23 – A1
D15 – D0

Rajah 4

i) Tunjukkan bahawa ingatan EPROM boleh dicapai pada ingatan $00000000 hingga $00003FFF

.....

.....

.....

.....

.....

.....

.....

(20/100)

3. Aturcara dalam Rajah 5 merupakan 'fail listing' suatu aturcara untuk menguji samada sistem mikropemproses boleh mencapai RAM dan LED di pot output sistem mikropemproses tersebut. Pin FC0-FC2, A1-A23, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{DTACK}$, dan D0-D15 sistem mikropemproses tersebut disambungkan kepada penganalisis logik bagi menganalisis pelaksanaan arahan ujian tersebut.

```
1                       *
2  00A00007             PBDDR   EQU     $A00007         ;lokasi penentu arah pot B
3  00A00013             PBDR    EQU     $A00013         ;lokasi pot B
4  00400400                     ORG     $400400         ;alamat aturcara disimpan
5  00400400 13FC00FF            MOVE.B  #$FF,PBDDR       ;set pot B sebagai pot output
   00400404 00A00007
6  00400408 103C00BA            MOVE.B  #$BA,D0         ;muatkan byte $BA ke D0
7  0040040C 13C000A0    ULANG   MOVE.B  D0,PBDR         ;hantar byte $BA ke pot output
   00400410 0013
8  00400412 4EF90040            JMP     ULANG           ;ulang selama-lamanya
   00400416 040C
9  00400418                     END
```

Rajah 5

a) Terangkan keadaan demi keadaan kitaran baca sistem mikropemproses 68000.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

(30/100)

b) Kirakan bilangan kitaran jam yang diperlukan untuk melaksanakan arahan 7 dan arahan 8.

.....

.....

(10/100)

c) Sekiranya frekuensi jam sistem mikropemproses ialah 10 MHz, kirakan masa yang diperlukan untuk melaksanakan arahan 7 dan arahan 8.

.....

.....

(10/100)

d) Terangkan bagaimana satu kitaran isyarat logik pelaksanaan arahan 7 dan arahan 8 yang pegun dihasilkan di skrin penganalisis logik.

.....

.....

.....

.....

(10/100)

e) Nyatakan keadaan LED output ketika aturcara dilaksanakan.

.....

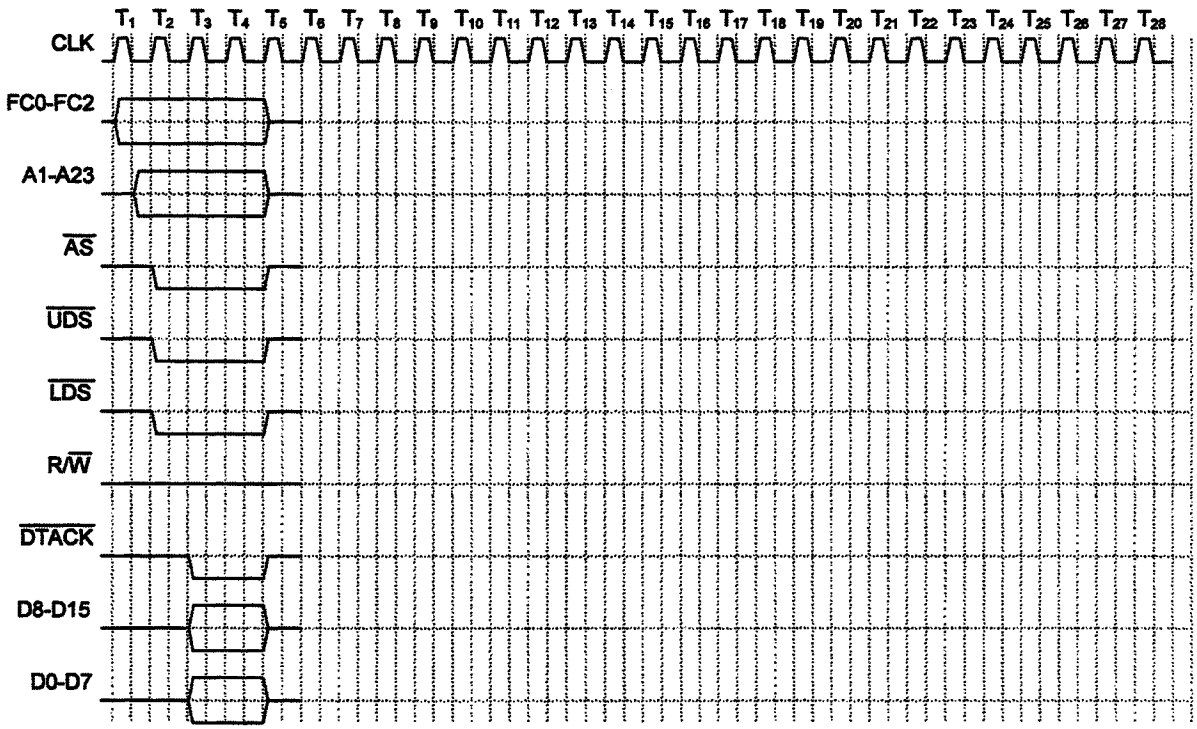
.....

.....

.....

(10/100)

f) Lakarkan kitaran isyarat logik yang mungkin diperhatikan di skrin penganalisis logik dengan melengkapkan kitaran logik dalam Rajah 6.



Rajah 6

(30/100)

4. Aturcara dalam Rajah 7, melibatkan pangilan subrutin dan arahan LINK. Berdasarkan aturcara tersebut selesaikan persoalan yang berikut.

a) Nyatakan kandungan D1 dan D2 sebaik sahaja sebelum aturcara mencabang ke subrutin.

Kandungan D1 ialah .....

dan kandungan D2 ialah .....

(10/100)

b) Apabila aturcara mencabang ke subrutin, kandungan pembilang program (PC) akan di simpan di alamat stak. Nyatakan kandungan PC dan alamat di mana ia disimpan.

Kandungan PC ialah .....

dan ia disimpan di alamat .....

(10/100)

c) Kandungan D7 selepas perlaksanaan arahan MOVE.W 8(A5) ialah

.....

(10/100)

d) Nyatakan kandungan D1, D2, D3, dan D7 sebaik sahaja sebelum arahan UNLK A5 dilaksanakan.

Kandungan D1 ialah .....

Kandungan D2 ialah .....

Kandungan D3 ialah .....

Kandungan D7 ialah .....

(20/100)

e) Nyatakan kandungan PC dan SP (penunjuk stak) selepas arahan RTS dilaksanakan.

Kandungan PC ialah .....

Kandungan SP ialah .....

(10/100)

f) Nyatakan kandungan D1, D2, D3, dan D7 selepas arahan MOVE.W D1,DATALOC dilaksanakan.

Kandungan D1 ialah .....

Kandungan D2 ialah .....

Kandungan D3 ialah .....

Kandungan D7 ialah .....

(20/100)

g) Nyatakan dua kegunaan utama arahan LINK.

.....

.....

.....

(20/100)

```
 1                          *
 2  004004DE                SPLOC    EQU     $004004DE       ;alamat penunjuk stak
 3  00400490                DATALOC  EQU     $00400490       ;lokasi jumlah disimpan
 4  00000002                DATA1    EQU     $02             ;data bernilai 2
 5  00400400                         ORG     $400400         ;lokasi aturcara ditulis
 6  00400400 2E7C0040       MULA     MOVEA.L #SPLOC,A7       ;penunjuk stak menunjuk lokasi $4004DE
    00400404 04DE
 7  00400406 123C0002               MOVE.B  #DATA1,D1       ;nombor 2 dimuatkan dalam D1
 8  0040040A D23C0011               ADD.B   #$11,D1         ;$11 dicampur ke dalam D1
 9  0040040E B502                   EOR.B   D2,D2           ;reset kandungan D1
10  00400410 3F02                   MOVE.W  D2,-(SP)        ;kandungan D2 disimpan di stak
11  00400412 3F01                   MOVE.W  D1,-(SP)        ;kandungan D1 disimpan di stak
12  00400414 6100009A               BSR     SBRTN           ;mencabang ke subrutin
13  00400418 33C10040               MOVE    D1,DATALOC      ;kandungan D1 diselamatkan
    0040041C 0490
14  0040041E 4E4B                   TRAP    #11             ;ke vektor 11
15  00400420 0000                   DC.W    0               ;
16                         *
17  004004B0                        ORG     $004004B0       ;lokasi subrutin disimpan
18  004004B0 4E55FFF4       SBRTN   LINK    A5,#-12         ;subrutin penyambung
19  004004B4 3E2D0008               MOVE.W  8(A5),D7        ;data D2 dimuatkan ke dalam D7
20  004004B8 362DFFFE               MOVE.W  -2(A5),D3       ;perkataan pertama storan dimuat ke D3
21  004004BC 322DFFFC               MOVE.W  -4(A5),D1       ;pekataan ke dua storan dimuat ke D1
22  004004C0 D243                   ADD.W   D3,D1           ;kandungan D3 dicampukan ke dalam D1
23  004004C2 4E5D                   UNLK    A5
24  004004C4 4E75                   RTS
25                         *
26  004004C6                        ORG     $004004C6
27  004004C6 00010203               DC.B    $00,$01,$02,$03,$04,$05,$06,$07,$08,$09,$0A,$0B,$0C
    004004CA 04050607
    004004CE 08090A0B
    004004D2 0C
28  004004D3 3F065B4F               DC.B    $3F,$06,$5B,$4F,$66,$6D,$7D,$07,$7F,$67,$5F,$7C,$39
    004004D7 666D7D07
    004004DB 7F675F7C
    004004DF 39
29  004004E0 5E797176               DC.B    $5E,$79,$71,$76,$38,$73,$3E
    004004E4 38733E
30  004004E7 11                     DC.B    $11
31  004004E8                        END
```

Rajah 7

D0 = $1234 5678	A0 = $0000 2000	$0000 2000 = $FACE
D1 = $9ABC DEF0	A1 = $0000 2001	$0000 2002 = $0AFE
D2 = $1122 3344	A2 = $0000 2002	$0002 2004 = $4A63
D3 = $5566 7788	A3 = $0000 2004	$0000 2006 = $0002
D4 = $0012 0034	A4 = $0001 1006	$0000 2008 = $FFFF
D5 = $CO1D FF32	A5 = $0000 200A	$0000 200A = $2006
D6 = $AABB CCDD	A6 = $0001 1004	$0000 200C = $1010
D7 = $0001 0004	A7 = $0000 4000	$0000 200E = $ABCD

Rajah 8

5. Rajah 8 menunjukkan data yang terkandung di dalam alatdaftar alamat dan alatdaftar data serta kandungan di beberapa lokasi alamat tertentu. Gunakan data tersebut di mana yang perlu bagi menyelesaikan persoalan berikut:

a) Nyatakan kandungan setiap alatdaftar destinasi selepas perlaksanaan setiap arahan berikut;

		Kandungan alatdaftar destinasi
i.	MOVE.L D2,D0	.....
ii.	MOVE.W (A0),D3	.....
iii.	MOVE.B D6,D1	.....
iv.	MOVEA (A5),A2	.....
v.	MOVE.B $3(A3),D4	.....
vi.	MOVEA.L A1,A5	.....
vii.	MOVEA #$CE45,A6	.....
viii.	MULU D2,D7	.....
ix.	DIVU 5(A1),D6	.....
x.	EOR.W D1,D1	.....

(20/100)

b) Nyatakan kandungan alatdaftar data selepas perlaksanaan arahan MOVEM.L (A2)+,D0-D5. Nyatakan juga kandungan A2 selepas perlaksanaan arahan tersebut.

A2 = ...

D0 = ...

D1 = ...

D2 = ...

D3 = ...

D4 = ...

D5 = ...

(14/100)

c) Terangkan samaada arahan-arahan berikut menyebabkan bendera limpahan diset atau tidak apabila ia dilaksanakan.

i. ADD D0,D3

.....

.....

ii. DIVU D2,D5

.....

.....

iii. MULU D5,D6

.....

.....

iv. ADD.L D6,D5

.....

.....

v. DIVU D7,D6

.....

.....

vi. MULU D7,D4

.....

.....

(36/100)

d) Tuliskan suatu segmen aturcara yang ringkas bagi memindahkan 53 data perkataan yang berada di lokasi alamat permulaan $0000 2002, ke lokasi alamat permulaan $0001 1006. Kedua bank ingatan tersebut, meningkat dari segi lokasi ingatannya daripada lokasi permulaan. Gunakan arahan BDcc dengan D4 sebagai alatdaftar pembilang bagi jawapan anda.

.....

.....

.....

.....

.....

.....

(30/100)

LAMPIRAN A

Mnemonic	Assembler Syntax	Operand Size	Allowable Addressing Modes		Condition Codes
			Source	Destination	X N Z V C
ABCD	ABCD Dy,Dx ABCD –(Ay),–(Ax)	8 8	Dn –(An)	Dn –(An)	* U * U * * U * U *
ADD	ADD <ea>,Dn ADD Dn,<ea>	8, 16, 32 8, 16, 32	All (1) Dn	Dn Alterable	* * * * * * * * * *
ADDA	ADD <ea>,An	16, 32	All	An	- - - - -
ADDI	ADDI #d,<ea>	8, 16, 32	#d	Data Alterable	* * * * *
ADDQ	ADDQ #d,<ea>	8, 16, 32	#d (2)	Alterable (1)	* * * * *
ADDX	ADDX Dy,Dx ADDX –(Ay),–(Ax)	8, 16, 32 8, 16, 32	Dn –(An)	Dn –(An)	* * * * * * * * * *
AND	AND <ea>,Dn AND Dn,<ea>	8, 16, 32 8, 16, 32	Data Dn	Dn Alterable	- * * 0 0 - * * 0 0
ANDI	ANDI #d,<ea> ANDI #d,SR (3)	8, 16, 32 8, 16	#d #d	Data Alterable SR	- * * 0 0 * * * * *
ASL	ASL Dx,Dy ASL #d,Dn ASL <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* * * * * * * * * * * * * * *
ASR	ASR Dx,Dy ASR #d,Dn ASR <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* * * * * * * * * * * * * * *
Bcc	Bcc <label>	8, 16	If cc, then PC + d → PC		- - - - -
BCHG	BCHG Dn,<ea> BCHG #d,<ea>	8, 32 8, 32	Dn #d	Data Alterable Data Alterable	- - * - - - - * - -
BCLR	BCLR Dn,<ea> BCLR #d,<ea>	8, 32 8, 32	Dn #d	Data Alterable Data Alterable	- - * - - - - * - -
BRA	BRA <label>	8, 16	PC + d → PC		- - - - -
BSET	BSET Dn,<ea> BSET #d,<ea>	8, 32 8, 32	Dn #d	Data Alterable Data Alterable	- - * - - - - * - -
BSR	BSR <label>	8, 16	PC → –(SP); PC + d → PC		- - - - -
BTST	BTST Dn,<ea> BTST #d,<ea>	8, 32 8, 32	Dn #d	Data, Except Immediate Data, Except Immediate	- - * - - - - * - -
CHK	CHK <ea>,Dn	16	If Dn < 0 or Dn > (ea), then TRAP	Data	- * U U U
CLR	CLR <ea>	8, 16, 32	Data Alterable		- 0 1 0 0
CMP	CMP <ea>,Dn	8, 16, 32	All (1)	Dn	- * * * *
CMPA	CMPA <ea>,An	16, 32	All	An	- * * * *
CMPI	CMPI #d,<ea>	8, 16, 32	#d	Data Alterable	- * * * *
CMPM	CMPM (Ay)+,(Ax)+	8, 16, 32	(An)+	(An)+	- * * * *

...14/-

Mnemonic	Assembler Syntax	Operand Size	Allowable Addressing Modes		Condition Codes
			Source	Destination	X N Z V C
DBcc	BDcc Dn,<label>	16	If cc, then Dn − 1 → Dn; If Dn ≠ − 1, then PC + d → PC		- - - - -
DIVS	DIVS <ea>,Dn	16	Data	Dn	- * * * 0
DIVU	DIVU <ea>,Dn	16	Data	Dn	- * * * 0
EOR	EOR Dn,<ea>	8, 16, 32	Dn	Data Alterable	- * * 0 0
EORI	EORI #d,<ea> EORI #d,SR (3)	8, 16, 32 8, 16	#d #d	Data Alterable SR	- * * 0 0 * * * * *
EXG	EXG Rx,Ry	32	Dn or An	Dn or An	- - - - -
EXT	EXT Dn	16, 32	Dn		- * * 0 0
JMP	JMP <ea>		<ea> → PC	Control	- - - - -
JSR	JSR <ea>		PC → −(SP); <ea> → PC	Control	- - - - -
LEA	LEA <ea>,An	32	Control	An	- - - - -
LINK	LINK An,#d	Unsized	An		- - - - -
LSL	LSL Dx,Dy LSL #d,Dn LSL <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* * * 0 * * * * 0 * * * * 0 *
LSR	LSR Dx,Dy LSR #d,Dn LSR <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* 0 * 0 * * 0 * 0 * * 0 * 0 *

MOVE	MOVE <ea>,<ea>
	MOVE <ea>,CCR
	MOVE <ea>,SR (6)
	MOVE SR,<ea>
	MOVE USP,An (6)
	MOVE An,USP (6)
		8, 16, 32
		16
		16
		16
		32
		32
			All (1)
			Data
			Data
			SR
			USP
			An
				Data Alterable
				CCR
				SR
				Data Alterable
				An
				USP
					- * * 0 0
					* * * * *
					* * * * *
					- - - - -
					- - - - -
					- - - - -

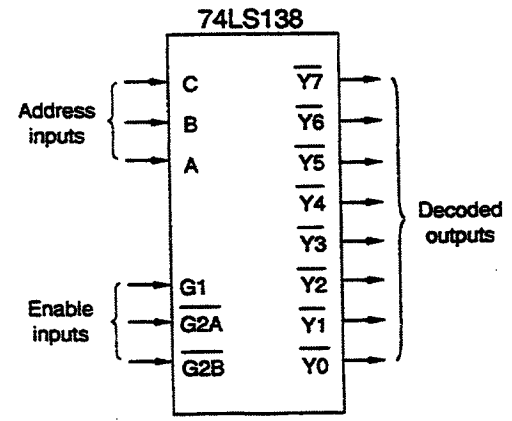
MOVEA	MOVEA <ea>,An	16, 32	All	An	- - - - -
MOVEM	MOVEM <list>,<ea> MOVEM <ea>,<list>	16, 32 16, 32	 Control or (An)+	Control Alterable or −(An)	- - - - - - - - - -
MOVEP	MOVEP Dx,d(Ay) MOVEP d(Ay),Dx	16, 32 16, 32	Dn d(An)	d(An) Dn	- - - - - - - - - -
MOVEQ	MOVEQ #d,Dn	32	#d (7)	Dn	- * * 0 0
MULS	MULS <ea>,Dn	16	Data	Dn	- * * 0 0
MULU	MULU <ea>,Dn	16	Data	Dn	- * * 0 0
NBCD	NBCD <ea>	8		Data Alterable	* U * U *
NEG	NEG <ea>	8, 16, 32	Data Alterable		* * * * *
NEGX	NEGX <ea>	8, 16, 32	Data Alterable		* * * * *
NOP	NOP		PC + 2 - PC		- - - - -
NOT	NOT <ea>	8, 16, 32		Data Alterable	- * * 0 0

Mnemonic	Assembler Syntax	Operand Size	Allowable Addressing Modes		Condition Codes
			Source	Destination	X N Z V C
OR	OR <ea>, Dn OR Dn,<ea>	8, 16, 32 8, 16, 32	Data Dn	Dn Alterable	- * * 0 0 - * * 0 0
ORI	ORI #d,<ea> ORI #d,SR (3)	8, 16, 32 8, 16	#d #d	Data Alterable SR	- * * 0 0 * * * * *
PEA	PEA <ea>	32	Control		- - - - -
RESET (6)	RESET				- - - - -
ROL	ROL Dx,Dy ROL #d,Dn ROL <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	- * * 0 * - * * 0 * - * * 0 *
ROR	ROR Dx,Dy ROR #d,Dn ROR <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	- * * 0 * - * * 0 * - * * 0 *
ROXL	ROXL Dx,Dy ROXL #d,Dn ROXL <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* * * 0 * * * * 0 * * * * 0 *
ROXR	ROXR Dx,Dy ROXR #d,Dn ROXR <ea>	8, 16, 32 8, 16, 32 16	Dn (4) #d (5)	Dn Dn Memory Alterable	* * * 0 * * * * 0 * * * * 0 *
RTE (6)	RTE		(SP) + → SP; (SP) + → PC		* * * * *
RTR	RTR		(SP)+ → CCR; (SP) + → PC		* * * * *
RTS	RTS		(SP) + → PC		- - - - -
SBCD	SBCD Dy,Dx SBCD -(Ay),-(Ax)	8 8	Dn -(An)	Dn -(An)	* U * U * * U * U *
Scc	Scc <ea>	8	If cc, then 1s → (ea); otherwise 0s → (ea)	Data Alterable	- - - - -
STOP (6)	STOP #d	16	#d → SR, then STOP		* * * * *
SUB	SUB <ea>,Dn SUB Dn,<ea>	8, 16, 32 8, 16, 32	All (1) Dn	Dn Alterable	* * * * * * * * * *
SUBA	SUBA <ea>,An	16, 32	All	An	- - - - -
SUBI	SUBI #d,<ea>	8, 16, 32	#d	Data Alterable	* * * * *
SUBQ	SUBQ #d,<ea>	8, 16, 32	#d (2)	Alterable (1)	* * * * *
SUBX	SUBX Dy,Dx SUBX -(Ay),-(Ax)	8, 16, 32 8, 16, 32	Dn -(An)	Dn -(An)	* * * * * * * * * *
SWAP	SWAP Dn	16	Dn		- - - - -
TAS	TAS <ea>	8	Data Alterable		- * * 0 0
TRAP	TRAP #<vector>		PC → -(SP); SR → -(SP); #<vector> → PC		- - - - -
TRAPV	TRAPV		If V = 1, then TRAP		- - - - -
TST	TST <ea>	8, 16, 32	Data Alterable		- * * 0 0
UNLK	UNLK An	Unsized		An	- - - - -

Footnotes:
(1) If the operation size is byte, the address register direct addressing mode is not allowed.
(2) Immediate operand, with a value from 1 to 8.
(3) If the operation size is word, the instruction is privileged.
(4) Source data register contains the shift count. Count = 0 to 63, where 0 produces a count of 64.
(5) The data is the shift count, 1 to 8.
(6) This operation is privileged.
(7) Eight bits of immediate data, which are sign-extended to a 32-bit long operand.

Effective Addressing Mode Categories

Addressing Mode	Addressing Categories				Assembler Syntax
	Data	Memory	Control	Alterable	
Data register direct.	X			X	Dn
Address register direct.				X	An
Register indirect.	X	X	X	X	(An)
Register indirect with postincrement.	X	X		X	(An)+
Register indirect with predecrement.	X	X		X	–(An)
Register indirect with displacement.	X	X	X	X	d(An)
Register indirect with index.	X	X	X	X	d(An,Ri)
Absolute short.	X	X	X	X	xxxx
Absolute long.	X	X	X	X	xxxxxxxx
PC relative with displacement.	X	X	X		d
PC relative with index.	X	X	X		d(Ri)
Immediate.	X	X			#xxxx



(a)

G1	$\overline{G2A}$	$\overline{G2B}$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
0	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	1	x	x	x	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0

(b)

The 74LS138 3-line to 8-line decoder. (a) symbol, (b) truth table.

Masa pelaksanaan beberapa arahan mikropemproses 68000

Table D-2. Move Byte and Move Word Instruction Execution Times.

	Destination								
Source	Dn	An	(An)	(An)+	-(An)	d16(An)	d8(An,Xn)*	xxx.W	xxx.L
Dn	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1)	12(2/1)	14(2/1)	12(2/1)	16(3/1)
An	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1)	12(2/1)	14(2/1)	12(2/1)	16(3/1)
(An)	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)
(An)+	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)
-(An)	10(2/0)	10(2/0)	14(2/1)	14(2/1)	14(2/1)	18(3/1)	20(3/1)	18(3/1)	22(4/1)
d16(An)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
d8(An,Xn)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)
xxx.W	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
xxx.L	16(4/0)	16(4/0)	20(4/1)	20(4/1)	20(4/1)	24(5/1)	26(5/1)	24(5/1)	28(6/1)
d16(PC)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)
d8(PC,Xn)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)
#data	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)

* The size of the index register (Xn) does not affect execution time.

Table D-3. Move Long Instruction Execution Times.

	Destination								
Source	Dn	An	(An)	(An)+	-(An)	d16(An)	d8(An,Xn)*	xxx.W	xxx.L
Dn	4(1/0)	4(1/0)	12(1/2)	12(1/2)	12(1/2)	16(2/2)	18(2/2)	16(2/2)	20(3/2)
An	4(1/0)	4(1/0)	12(1/2)	12(1/2)	12(1/2)	16(2/2)	18(2/2)	16(2/2)	20(3/2)
(An)	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	24(4/2)	28(5/2)
(An)+	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	24(4/2)	28(5/2)
-(An)	14(3/0)	14(3/0)	22(3/2)	22(3/2)	22(3/2)	26(4/2)	28(4/2)	26(4/2)	30(5/2)
d16(An)	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(6/2)
d8(An,Xn)*	18(4/0)	18(4/0)	26(4/2)	26(4/2)	26(4/2)	30(5/2)	32(5/2)	30(5/2)	34(6/2)
xxx.W	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(6/2)
xxx.L	20(5/0)	20(5/0)	28(5/2)	28(5/2)	28(5/2)	32(6/2)	34(6/2)	32(6/2)	36(7/2)
d16(PC)	16(4/0)	16(4/0)	24(4/2)	24(4/2)	24(4/2)	28(5/2)	30(5/2)	28(5/2)	32(5/2)
d8(PC,Xn)*	18(4/0)	18(4/0)	26(4/2)	26(4/2)	26(4/2)	30(5/2)	32(5/2)	24(4/2)	28(5/2)
#data	12(3/0)	12(3/0)	20(3/2)	20(3/2)	20(3/2)	24(4/2)	26(4/2)	30(5/2)	34(6/2)

* The size of the index register (Xn) does not affect execution time.

Table D-10. JMP, JSR, LEA, PEA, and MOVEM Instruction Execution Times.

Inst.	Size	(An)	(An)+	-(An)	d16(An)	d8(An,Xn)*	xxx.W	xxx.L	d16(PC)	d8(PC,Xn)*
JMP	-	8(2/0)	-	-	10(2/0)	14(3/0)	10(2/0)	12(3/0)	10(2/0)	14(3/0)
JSR	-	16(2/2)	-	-	18(2/2)	22(2/2)	18(2/2)	20(3/2)	18(2/2)	22(2/2)
LEA	-	4(1/0)	-	-	8(2/0)	12(2/0)	8(2/0)	12(3/0)	8(2/0)	12(2/0)
PEA	-	12(1/2)	-	-	16(2/2)	20(2/2)	16(2/2)	20(3/2)	16(2/2)	20(2/2)
MOVEM	Word	12+4n	12+4n	-	16+4n	18+4n	16+4n	20+4n	16+4n	18+4n
M→R		(3+n/0)	(3+n/0)		(4+n/0)	(4+n/0)	(4+n/0)	(5+n/0)	(4n/0)	(4+n/0)
	Long	12+8n	12+8n	-	16+8n	18+8n	16+8n	20+8n	16+8n	18+8n
		(3+2n/0)	(3+2n/0)		(4+2n/0)	(4+2n/0)	(4+2n/0)	(5+2n/0)	(4+2n/0)	(4+2n/0)
MOVEM	Word	8+4n	-	8+4n	12+4n	14+4n	12+4n	16+4n	-	-
R→M		(2/n)	-	(2/n)	(3/n)	(3/n)	(3/n)	(4/n)		
	Long	8+8n	-	8+8n	12+8n	(14+8n	12+8n	16+8n	-	-
		(2/2n)		(2/2n)	3/2n)	(3/2n)	(3/2n)	(4/2n)		

n is the number of registers to move.

* The size of the index register (Xn) does not affect the instruction's execution time.