3.A.N. MICROPROCESSEURS
Devoir surveillé du 26 octobre 2001
Durée : 2 heures Documents autorisés

REMARQUE IMPORTANTE:

Les réponses mal écrites ou non commentées seront considérées comme fausses.

PARTIE A

On considère l'instruction symbolique suivante du microprocesseur 68000, dont les spécifications sont jointes en annexe :

```
lea.l   -8(a6),a5
```

QUESTION A-1

- Indiquer les modes d'adressages de chacun de ses deux opérandes et en déduire les valeurs **numériques binaires** des champs de l'opcode de cette instruction.
- Donner la représentation numérique hexadécimale du code machine **complet** de cette instruction.

QUESTION A-2

- Rédiger et commenter sur le tableau de la figure A-1, le microcode en langage « C » décrivant le fonctionnement interne de cette instruction, en prenant comme référence le modèle « VON NEUMANN » utilisé en cours.

QUESTION A-3

- Déterminer pour chacune des phases d'exécution de l'instruction précédente, les nombres de cycles de transfert sur le bus externe de données du microprocesseur 68000 et remplir les cases correspondantes sur la figure A-1 ci-dessous :

Figure A-1 :

Phases	Microcode	Commentaires	Nbres de cycles
Φ1			
Φ2			
Φ3			
Φ4			
Φ5			

LEA

Load Effective Address
(M68000 Family)

LEA

Operation: < ea > → An

Assembler Syntax: LEA < ea > ,An

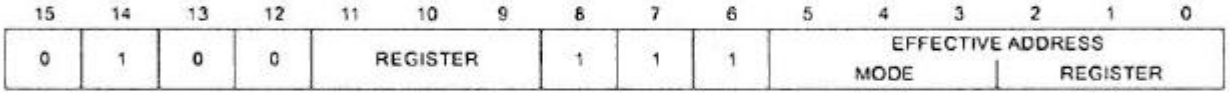
Attributes: Size = (Long)

Description: Loads the effective address into the specified address register. All 32 bits of the address register are affected by this instruction.

Condition Codes:

Not affected.

Instruction Format:



Instruction Fields:

Register field—Specifies the address register to be updated with the effective address.

Effective Address field—Specifies the address to be loaded into the address register. Only control addressing modes can be used as listed in the following tables:

Addressing Mode	Mode	Register
Dn	—	—
An	—	—
(An)	010	reg. number:An
(An) +	—	—
– (An)	—	—
(d_{16},An)	101	reg. number:An
(d_8,An,Xn)	110	reg. number:An

Addressing Mode	Mode	Register
(xxx).W	111	000
(xxx).L	111	001
#<data>	—	—
(d_{16},PC)	111	010
(d_8,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

(bd,An,Xn)	110	reg. number:An
([bd,An,Xn],od)	110	reg. number:An
([bd,An],Xn,od)	110	reg. number:An

(bd,PC,Xn)*	111	011
([bd,PC,Xn],od)	111	011
([bd,PC],Xn,od)	111	011

*Can be used with CPU32.

PARTIE B

On considère des nombres algébriques X et Y dans le mode de représentation complémentaire sur 8 bits (x7...x0 et y7...y0) tels que :

$X = -128*x_7 + 64*x_6 + 32*x_5 + 16*x_4 + 8*x_3 + 4*x_2 + 2*x_1 + x_0$

$Y = -128*y_7 + 64*y_6 + 32*y_5 + 16*y_4 + 8*y_3 + 4*y_2 + 2*y_1 + y_0$

On effectue des addition binaires au format « byte » pour différentes valeurs de ces nombres.

Pour chacune de ces opérations, on s'intéresse aux paramètres de sortie suivants:

- Le résultat sur 8 bits : S = X + Y
- Les indicateurs binaires N, Z, V et C

On rappelle qu'à l'issue d'un calcul, l'indicateur binaire de débordement algébrique V prend la valeur 1 si le résultat S de l'opération n'est pas représentable sur 8 bits dans le mode décrit plus haut.

Le tableau de la figure B-1 représente 8 opérations d'additions pour différentes valeurs des nombres algébriques X et Y.

QUESTION B-1

- Compléter le tableau de la figure B-1 en indiquant pour chacune des 8 opérations, la représentation hexadécimale du résultat S de l'addition.

QUESTION B-2

- Indiquer dans les cases correspondantes de la figure B-1, les valeurs des indicateurs binaires N, Z, V et C à l'issue du calcul.
- Commenter de manière détaillée, les cas de débordement.

FIGURE B-1:

0x40 (X) + 0x30 (Y) (S) C = Z = V = N =	0x40 (X) + 0x70 (Y) (S) C = Z = V = N =	0x40 (X) + 0xA0 (Y) (S) C = Z = V = N =	0x40 (X) + 0xC0 (Y) (S) C = Z = V = N =
0xC0 (X) + 0xD0 (Y) (S) C = Z = V = N =	0xC0 (X) + 0xA0 (Y) (S) C = Z = V = N =	0xC0 (X) + 0x70 (Y) (S) C = Z = V = N =	0xC0 (X) + 0x30 (Y) (S) C = Z = V = N =

PARTIE C

Le listing de la figure C-1 représente un programme d'essai pour le microcontrôleur PIC 16F877, dont l'exécution commence par l'instruction « movlw 0xC5 ».

Les références « Start » et « Invb » sont les représentations littérales des adresses des instructions qu'elles désignent (respectivement « movlw 0xC5 ».et « rlf 0x20,1 »).

La figure C-2 représente sous forme littérale, les valeurs initiales de l'indicateur « c » (carry) et des bits contenus dans les registres N° 0x20 et 0x21, avant l'exécution de l'instruction « call Invb » (ligne N°10).

QUESTION C-1

- Décrire la structure et le fonctionnement général du programme de la figure C-1.

QUESTION C-2

- Compléter le tableau de la figure C-3 en indiquant les nouvelles positions des bits initiaux de la figure C-2, après l'exécution des instructions des lignes N°14 à N°38.
- Représenter dans les cases correspondantes des figures C-2 et C-3, les valeurs numériques hexadécimales des contenus des registres N° 0x20 et N°0x21, respectivement avant et après l'exécution des instructions des lignes N°14 à N°38.

QUESTION C-3

- Décrire clairement et succintement les effets produits par ce programme d'essai.

Figure C-2: (avant)

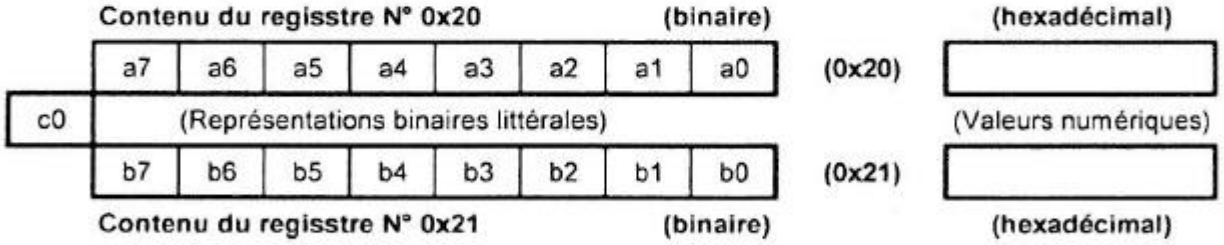


Figure C-3: (après)

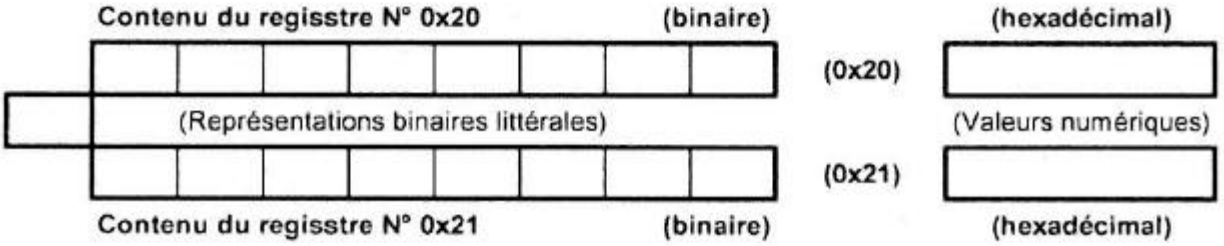


Figure C-1:

```
1 ;*****************************************************************
2 ;*       PROGRAMME D'ESSAI
3 ;*       POUR LE MICROCONTROLEUR PIC 16F877
4 ;*****************************************************************
5 Start   movlw   0xC5    ; 1ere instruction du programme
6         movwf   0x20
7         movlw   0xF0
8         movwf   0x21
9
10        call    Invb
11
12 Stop   goto    Stop
13 ;-----------------------------------------------------------------
14 Invb   rlf     0x20,1
15        rrf     0x21,1
16
17        rlf     0x20,1
18        rrf     0x21,1
19
20        rlf     0x20,1
21        rrf     0x21,1
22
23        rlf     0x20,1
24        rrf     0x21,1
25
26        rlf     0x20,1
27        rrf     0x21,1
28
29        rlf     0x20,1
30        rrf     0x21,1
31
32        rlf     0x20,1
33        rrf     0x21,1
34
35        rlf     0x20,1
36        rrf     0x21,1
37
38        rlf     0x20,1
39
40        return
41 ;*****************************************************************
42        end
```

PARTIE D

On se propose d'étudier un sous-programme « addbcd » pour le microprocesseur 68000, qui effectue l'addition de 2 variables X et Y **codées en B.C.D.** sur 6 octets chacune (soit l'équivalent de 12 chiffres décimaux par variable) selon la formule suivante :

$Y_{(BCD)} = X_{(BCD)} + Y_{(BCD)}$

- Avant le calcul, les variables X et Y sont stockées dans une pile dont le sommet est pointé par le registre A6, selon la figure D-1-A.
- Après le calcul, seule subsiste la variable Y dans la pile dont le sommet est pointé par le registre A6, selon la figure D-1-B.
- Il est conseillé de lire attentivement les spécifications de l'instruction « abcd » dans le cours, ainsi que celles jointes en annexe.

QUESTION D-1

❖ Décrire la méthode que vous envisagez d'utiliser pour effectuer le calcul demandé.

QUESTION D-2

❖ Rédiger et commenter le code source du sous-programme « addbcd » correspondant dans les cases du tableau de la figure D-2.

Figure D-1-A:

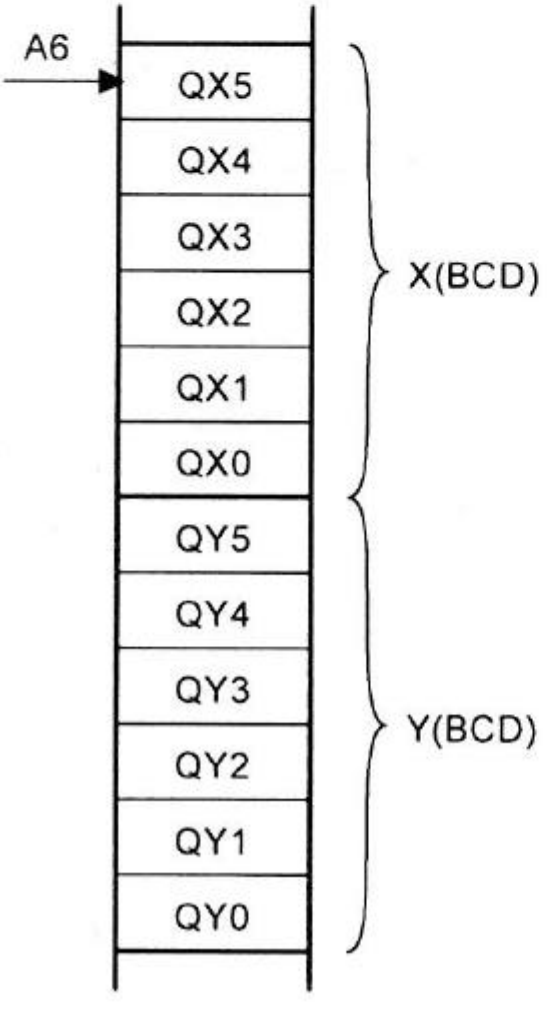


Figure D-1-B:

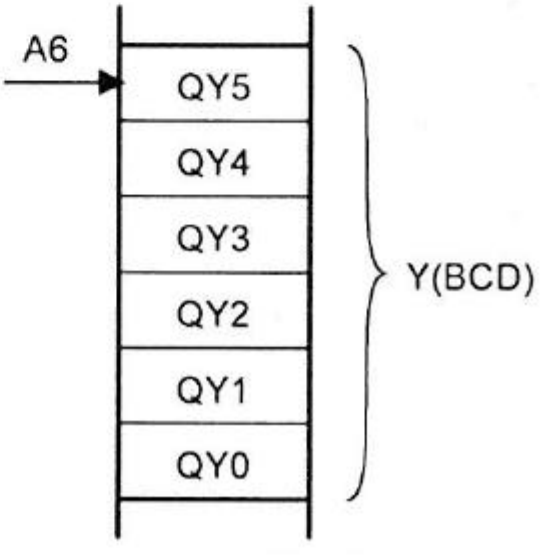


Figure D-2: Fiche de programmation

	Référence	Opcode	Opérande(s)	Commentaires
1	addbcd			
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				

ABCD — Add Decimal with Extend (M68000 Family) — ABCD

Operation: Source10 + Destination10 + X → Destination

Assembler Syntax:
ABCD Dy,Dx
ABCD – (Ay), – (Ax)

Attributes: Size = (Byte)

Description: Adds the source operand to the destination operand along with the extend bit, and stores the result in the destination location. The addition is performed using binary-coded decimal arithmetic. The operands, which are packed binary-coded decimal numbers, can be addressed in two different ways:

1. Data Register to Data Register: The operands are contained in the data registers specified in the instruction.
2. Memory to Memory: The operands are addressed with the predecrement addressing mode using the address registers specified in the instruction.

This operation is a byte operation only.

Condition Codes:

X	N	Z	V	C
•	U	•	U	•

X — Set the same as the carry bit.
N — Undefined.
Z — Cleared if the result is nonzero; unchanged otherwise.
V — Undefined.
C — Set if a decimal carry was generated; cleared otherwise.

NOTE

Normally, the Z condition code bit is set via programming before the start of an operation. This allows successful tests for zero results upon completion of multiple-precision operations.

ABCD

Add Decimal with Extend (M68000 Family)

ABCD

Instruction Format:

15	14	13	12	11 10 9	8	7	6	5	4	3	2 1 0
1	1	0	0	REGISTER Rx	1	0	0	0	0	R/M	REGISTER Ry

Instruction Fields:

Register Rx field—Specifies the destination register.
If R/M = 0, specifies a data register.
If R/M = 1, specifies an address register for the predecrement addressing mode.

R/M field—Specifies the operand addressing mode.
0 — The operation is data register to data register.
1 — The operation is memory to memory.

Register Ry field—Specifies the source register.
If R/M = 0, specifies a data register.
If R/M = 1, specifies an address register for the predecrement addressing mode.