Grp **Nom**
❑A
❑B
❑C

Contrôle court n°2

Calculatrice et documents interdits - Durée 1 heure - Répondre sur la feuille

I. Architecture

1. Dans un ordinateur, quelle est la différence entre un registre et une mémoire ?
(comparez la nature, la taille, la vitesse et le moyen d'accès aux information)

Un registre est une petite mémoire (quelques octets).
Il fait partie du processeur processeur,
qui le connaît directement (adressage implicite).
Les accès sont très rapides.
Une mémoire est un bloc fonctionnel hors du processeur
et généralement hors du chip.
Elle présente une plus grande capacité
(au minimum quelques ko).
On y accède par l'intermédiaire d'un bus
en précisant une adresse (donc c'est lent).

2. Listez les registres d'un processeur 8 bits à accumulateur pouvant adresser 64 ko; précisez leur taille.

Compteur ordinal ou pointeur d'instruction, 16 bits
Registre d'instruction, 8 bits
Accumulateur, 8 bits
Registre tampon de l'Accumulateur, 8 bits
Registre tampon d'adresse, 16 bits

3. Détaillez les opérations élémentaires* (ou micro-instructions) effectuées par le processeur 8086 (processeur 16 bits) lors de l'exécution de l'instruction `ADD AX,[0000]` (code instruction `03 06 00 00`).
(*micro-instructions du type transfert de registre à bus, calculs ou passage dans un blocs de décodage)

Chargement code opération dans RI à partir du bus de données,
décodage de l'instruction, avancée de IP (deux)
Dépose du contenu de IP sur le bus d'adresse, décodage @mem,
chargement de l'adresse dans RTA à partir du bus de données
Dépose du contenu de RTA sur le bus d'adresse, décodage @mem,
chargement de la donnée dans RTAX à partir du bus de données,
avancée de IP (deux)
Ajout de RTAX à AX,
Dépose du contenu de IP sur le bus d'adresse, décodage @mem

4. La mémoire présente les octets suivants :

`A1 00 00 48 2D 01 00 3B-06 02 00 75 F6 14 77 90`

En fait il s'agit d'un programme. Les différentes instructions sont soulignées. Leur sens est donné ci-dessous.
Complétez le tableau ci-dessous après l'exécution sur un 8086 du programme précédent (attention, ce n'est pas le même processeur que dans la question précédente).

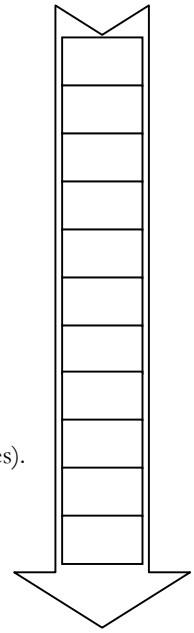
Instruction	RI	RTA	RTUAL	ACC	Flags	IP	[0000] [0001]	[0002] [0003]
Etat initial	?	?	?	?	?	0000	01 00	02 00
MOV AX,[....]	A1	0000	"	0001	"	0003	"	"
DEC AX	48	"	"	0000	ZF=1	0004	"	"
SUB AX,....	2D	"	0001	FFFF	ZF=0, CF=1	0007	"	"
CMP AX,[....]	3B06	0002	"	"		000B	"	"
JNE	75	"	"	"	"	0003	"	"

II. Assembleur

1. Complétez la déclaration assembleur pour les données suivantes : le caractère espace, le caractère retour à la ligne (code ASCII 13), un masque pour ne regarder que le bit 5 des caractères, une chaîne de caractères "Stop!", un entier X, un tableau T de 32 entiers comme X, un entier N donnant la taille du tableau et un autre I pour stocker l'indice dans le tableau (initialisé à 0).

```
Donnees         SEGMENT
Espace    DB    ' '
NewLine   DB    13
Masque    DB    00100000b
Chaine    'Stop!'
X         DW    ?
T         DW    32 dup(?)
N         DB    32
I         DB    0

Donnees         ENDS
```



2. Sur la représentation ci-contre de la mémoire, faites figurer les données (nom et valeurs éventuelles). Précisez les adresses mémoire correspondant aux différentes variables que vous avez déclaré.

```
Espace    →    [0000]
NewLine   →    [0001]
Masque    →    [0002]
Chaine    →    [0003]
X         →    [0008]
T         →    [000A]
N         →    [002A]
I         →    [002B]
```

3. Ecrire un programme qui range dans X le nombre de valeurs non nulles dans le tableau T.
Utilisez les variables I et N déclarées précédemment. Attention, l'instruction CMP @,@ n'existe pas !
Seuls deux registres sont utilisés : AL et BX.

```
     MOV BX,OFFSET T
     MOV X,0
bcl: MOV AL,I
     CMP AL,N
     JGE fini
     MOV AL,[BX]
     CMP AL,0
     JNE next
     INC X
next:INC BX
     INC I
     JMP bcl
fini:
```