Nom, prénom: Grp: A B C	**ARCHITECTURE** **Contrôle Long n°1 (durée 3 heures)** **Sans documents ni calculatrice - Répondre sur l'énoncé.**

Lisez attentivement tout le texte d'un exercice avant de commencer à le traiter, prenez votre temps avant de répondre. Le correcteur tiendra compte de la clarté de la présentation, de la justification des réponses ainsi que de l'orthographe.

❖ Partie A : Questions de cours.

COURS 1. : OPERATIONS ARITHMETIQUES ET LOGIQUES

1.1. Dans un registre 8 bits, on effectue des opérations sur des nombres signés.
Donner le résultat des opérations suivantes et positionner les indicateurs d'état.

1 0 1 1 0 0 0 0	1 1 1 1 0 0 0 0	0 1 0 1 0 0 0 0
+ 1 0 1 1 1 1 0 0	+ 0 0 0 1 0 0 0 0	+ 0 1 1 0 0 0 0 0
= 0 1 1 0 1 1 0 0	= 0 0 0 0 0 0 0 0	= 1 0 1 1 0 0 0 0
SF = 0 CF = 1	SF = 0 CF = 1	SF = 1 CF = 0
ZF = 0 OF = 1	ZF = 1 OF = 0	ZF = 0 OF = 1

COURS 2. : LANGAGE DE PROGRAMMATION

2.1. Quel est l'opération permettant de transcrire un programme du code symbolique (ou code source) en code machine (ou code objet).

L'opération s'appelle :l'assemblage

2.2. Dans l'extrait de programme suivant, précisez pour chacune des instructions le mode d'adressage.

Instruction	Mode d'adressage
`MOV AL, [000B]`	***Adressage direct***
`ADD AL, C4`	***Adressage immédiat***
`INC AX`	***Adressage implicite***
`MOV [BX],00`	***Adressage indirect***
`JNE 010A`	***Adressage relatif***

2.3. Parmi les instructions suivantes, indiquer celles qui sont incorrectes et corrigez-les.

Instruction	OK ?	Proposition de correction
`PUSH AL`	***Non***	***PUSH AX***
`MOV AX, [1]`	***Oui***	
`ROL AX, 2`	***Non***	***ROL AX, 1***
`CMP [1000], 2`	***Non***	***CMP [BX], 2***
`MOV AX, Toto`	***Oui***	
`MOV AX, BL`	***Non***	***MOV AX, BX***

COURS 3. : PROCESSEUR 8086

3.1. Donner la définition du registre d'état et citer 4 indicateurs d'état en précisant leur fonction.

Le registre d'état est une registre qui regroupe les indicateurs d'état.
Citons 4 indicateurs d'état :
ZF : indique si le résultat de la dernière opération est nul.
CF : indique une éventuelle retenue
SF : indique le signe du résultat d'une opération
OF: indique un éventuel dépassement.

3.2. Sachant que la taille du bus d'adresse d'un processeur est de 20 bits, combien de segments peut-il gérer ? Quelle est la taille de ces segments ? Justifier votre réponse.

Le registre de segment est sur 16 bits donc il y a 216 segments, soit 65536 segments.
A chaque segment correspond une adresse d'offset sur 16 bits donc il y a 65536 adresses par segment.

Combien de segments indépendants peut-il gérer (sans recouvrement) ?

On dénombre 16 segments indépendants. (0000h, 1000h, 2000h, 3000h, ... , F000h).
En effet, une adresse sur 20 bits est composée d'un segment sur 16 bits décalée de 4 bits vers la gauche auquel on ajoute un offset sur 16 bits.

3.3. Indiquer les symboles et le nom des 3 registres qui permettent de gérer la pile.

SS : Segment de pile *SP : Pointeur de Pile*
BP: Pointeur de base

Il s'agit d'une pile LIFO. Que signifie ce terme ?

LIFO : Last In, First Out : Dernier entré, Premier sorti.

Préciser quelles sont les opérations effectuées lors de l'exécution de l'instruction PUSH AX.

SP ⇐ SP - 2
[SP] ⇐ AX

❖ Partie B : Exercices.

EXERCICES 4.

4.1. Le programme suivant réalise une temporisation. Pour ce faire, il décrémente la valeur dans un registre 16 bits, *i.e.* de 0100H (fixé au départ) à 0. Pour chacune des lignes du programme, on donne la durée d'exécution d'une instruction complète en micro-cycle (µc). Le processeur travaille à une fréquence de 10MHz, c'est à dire que chacune des opérations élémentaires est effectuée en 1 µc de 100 ns (rappel 1 ns = 10^{-9} s).

```
          ASSUME    CS : Code, DS : Data
Data      SEGMENT
Tempo     DW        0100H
Data      ENDS

Code      SEGMENT                        Nombre de µc
Debut :   MOV       AX , Data              10
          MOV       DS , AX                 2

          MOV       AX , Tempo             10
Boucle :  ADD       AX , -1                 4
          JNE       Boucle                 16

          MOV       AH , 4CH                4
          INT       21H                    52

Code      ENDS
          END       Debut
```

Calculez la durée d'exécution de ce programme à la nano-seconde près. Justifier votre réponse.

Durée = 10 + 2 + 10 + 256 × (4 + 16) + 4 + 52
= 5198 µc
= 5198 × 100 ns
= 519 800 ns

4.2. Modifier la valeur initiale de la variable Tempo pour que l'ensemble de la temporisation atteigne une durée de 1 ms. Posez l'expression littérale sans faire le calcul.

1 000 000 ns = [10+2+10+ Tempo×(4+16) +4+52] × 100 ns
10000 = 10+2+10+ Tempo×(4+16) +4+52
Tempo×(4+16) = 10000 – 10-2-10-4-52
Tempo = 9922 / 20
Tempo = 496 (en décimal) = 1F0 H

EXERCICES 5.

5.1. Sur une carte mère sont disposés un processeur 8 bits dont le bus d'adresse est de 24 bits, ainsi que des circuits mémoire 8 bits de capacité 512 Ko.
Si l'on suppose que la totalité de l'espace adressable est occupé par les mémoires, combien de circuits mémoire sont présents sur cette carte mère ? Justifier brièvement votre réponse.

24 bits d'adresse permettent d'adresser 224 adresses occupées chacune par 1 octet ⇐ 16 Mo soit 16 × 1024 Ko.
16 × 1024 / 512 = 32 donc la carte mère intègre 32 circuits mémoire pour gérer l'espace adressable.

EXERCICES 6.

6.1. AL contient le code ASCII d'une lettre minuscule (ex : 'c'). On veut mettre en majuscule cette lettre (ex : 'C'). Ecrire ***une*** seule instruction qui permette d'effectuer cette modification.

AND , ODFH

EXERCICES 7.

7.1. Dites ce que le programme suivant range dans AL en fin d'exécution : dites ce que cela représente et donnez la valeur.

```
        ASSUME CS:CODE, DS:DATA
DATA    SEGMENT
TAB     DB     18,11,29,7,15,34,42,89,8,76,4,61,43,12,6
NELT    DW     14
DATA    ENDS

CODE    SEGMENT
Tri:    MOV    AX,DATA
        MOV    DS,AX
        MOV    BX,offset TAB
        MOV    AL,[BX]
        INC    BX
        MOV    CX,NELT
Boucle: MOV    AH,[BX]
        CMP    AL,AH
        JB     Suite              ; Test <
        MOV    AL,AH
Suite:  INC    BX
        DEC    CX
        JNE    Boucle             ; Test ≠
Fin:    MOV    AH,4CH
        INT    21H
CODE    ENDS
        END    Tri
```

En fin de prog, AL contient le minimum du tableau.
Dans notre cas, AL contient la valeur 4.

EXERCICES 8.

8.1. On souhaite réaliser l'affichage du contenu du registre AL à l'écran. Or pour ce faire, on converti les 2 quartets qui composent le registre AL en leur code ASCII correspondant.
Par exemple : Si AL = 5B alors en fin d'exécution de programme, la chaîne RESULT doit contenir les octets : 35H ('5'), 42H ('B') suivi du caractère de fin de chaîne '$' en vue d'un affichage.
Le programme ci-dessous réalise cette transformation.

```
        ASSUME CS : Code, DS : Data

Data    SEGMENT
RESULT  DB 3 DUP(?)
Data    ENDS
                                ;            Registres
Code    SEGMENT                 ; Adresses   AX = 4B 5B
Debut:  MOV BX,offset RESULT    ;  0000      BX = 0000 H
        MOV AH,AL               ;  0003      AX = 5B5B H

QuartH: MOV CL,4                ;  0005
        SHR AH,CL               ;  0007
        AND AH,0Fh              ;  0009      AH = 05 H
        CMP AH,0Ah              ;  000C
        JAE AlphaH              ;  000F      IP = 0011 H
        ADD AH,'0'              ;  0011
        JMP FinH                ;  0014
AlphaH: ADD AH,'A'              ;  0017
        SUB AH,0Ah              ;  001A
FinH:   MOV [BX],AH             ;  001D      AH = 35 H

QuartL: AND AL,0Fh              ;  001F      AL = 0B H
        CMP AL,0Ah              ;  0021
        JAE AlphaL              ;  0023      IP = 002A H
        ADD AL,'0'              ;  0025
        JMP FinL                ;  0027
AlphaL: ADD AL,'A'              ;  002A
        SUB AL,0Ah              ;  002C
FinL:   MOV Byte ptr[BX+1],AL   ;  002E      AL = 42 H

FinCH:  MOV Byte ptr[BX+2],'$'  ;  0031      AX = 3542 H

Code    ENDS
        END Debut
```

Complétez les valeurs des registres demandés dans le champ commentaires en fin de ligne
Attention : Les valeurs de ces registres doivent être données sachant que l'instruction de la ligne correspondant a **déjà** été exécutée (si elle l'est !).
Indications :
`SHR AX,CX` Décalage logique du registre AX vers la droite d'un nombre de bits indiqué par CX.
JAE Branchement si ≥

❖ Partie C : Problème.

On se propose de réaliser un programme en assembleur avec toutes les initialisations utiles. Ce programme fait la somme des N premiers entiers naturels ne dépassant pas une valeur MAXI positive fixée par avance dans une variable. N est inconnu au départ. Le but du programme est de trouver le nombre N d'entiers.

Exemple : Si MAXI=12 alors SOMME = 1+2+3+4 = 10
N = 4

Les variables utilisées dans ce programme seront déclarées comme suit dans le segment de données :

```
data    SEGMENT
N       DB      ?       ;Entier naturel courant
SOMME   DW      ?       ;Somme
MAXI    DW      100     ;Valeur maximale
data    ENDS
```

La difficulté de cet exercice et d'éviter le débordement lors de la somme des entiers codés sur 8 bits. Pour cela on effectue la somme sur 16 bits. Il faut convertir au préalable l'entier courant (sur 8 bits) en valeur sur 16 bits avant de l'additionner.

Proposition de correction…

```
        ASSUME CS : Code, DS : Data

Data    SEGMENT
N       DB      ?       ;Entier naturel courant
SOMME   DW      ?       ;Somme
MAXI    DW      100     ;Valeur maximale
Data    ENDS

Code    SEGMENT
DEBUT:  MOV     AX,DATA
        MOV     DS,AX
        MOV     AX,0
        MOV     BX,0
BOUCLE: INC     BL
        ADD     AX,BX
        CMP     AX,MAXI
        JB      BOUCLE
        SUB     AX,BX
        DEC     BL
        MOV     SOMME,AX
        MOV     N,BL

        MOV     AX,4C00H
        INT     21H
Code    ENDS
        END     Debut
```