Département d'Informatique
Kénitra

Durée : 1h30mn

Examen d'Architecture des ordinateurs et Programmation Assembleur
IF et IA
S3

Nom :
Prénom :
N° Apogée :

Question de cours

1. Citer et classer les registres du micro-processeur 80X86 ?

2. A quoi sert la segmentation de la mémoire ?

3. Qu'est ce qu'un registre ?

4. Donner quatre modes d'adressage et expliquer leurs principes par l'exemple

5. Comment calculer l'adresse physique d'un octet ?

6. Donner l'équivalent de l'adresse 0000:0042 dans le segment n°3 et dans le segment n°4.

Exercice 1

Que fait le programme suivant :

```
        Mov bx ,[0300]
        Mov ax,10
Repet : SHL bx
        Dec ax
        Jne repet
```

```
Mov [0140],bx
Mov ah,4ch
Int 21h
```

Exercice 2

Ecrire un programme assembleur qui permet de faire les trois taches suivantes :

1/lire des chaînes de caractères autant de fois qu'on veut. Utiliser une procédure « lire_chaine » qui permet la lecture d'une chaîne de caractères.

2/ Pour chaque chaine lue, calculer la longueur et déposer sa valeur dans le registre AX. Utiliser une procédure « longueur_chaine qui calcule la longueur d'une chaine de caractères.

3/ Afficher un par ligne les caractères de chaque chaine lue.

Indication : ***Rédiger vos réponses en double colonnes sur l'espace suivant***

Exercice 3

Ecrire un programme assembleur qui permet de voir si un caractère minuscule figure dans une chaine de caractères minuscules ou non et si le caractère existe, il le convertit en majuscule dans la chaîne.
Afficher la chaine modifiée une fois le parcours terminé.
La chaîne et le caractère doivent être initialement lus.

Examen Architecture des Ordinateurs

2024-2025 NOR.

Site: https://spaceuit.com/2025/Fall/architecture/introduction

Question de cours:

1. Citer et classer les registres du microprocesseur 80X86 : (5 lignes)

Registres généraux: AX, BX, CX, DX

Registres segments: CS, DS, SS et ES

Registres d'offset: IP, BP, SP

Registre d'état: Les indicateurs OF, CF, ZF, NF, ...

2. A quoi sert la segmentation de la mémoire : (2 lignes):

- La segmentation de la mémoire est une technique de gestion de la mémoire pour diviser la mémoire en segments: CX, CX, SS et ES.

3. Qu'est ce qu'un registre ? (2 lignes):

- Un registre est un mémoire de petite taille, utilisée pour conserver temporairement les données, instructions ou adresses.

4. Donner quatre modes d'adressage et expliquer leurs principes par l'exemple: (5 lignes):

- ~~Impli~~ Immédiat: Le code opérand est une donnée (Valeur): MOV AX, 5
- Implicite: L'instruction contient seulement le code d'opération: INC AX
- Relatif: Le code d'opérand contient l'adresse qui séra ajouté au IP: JMP 0110
- Direct: Le code d'opérand est l'adresse de donnée: MOV AX, [0110h]

5. Comment calculer l'adresse physique d'un octet (2 lignes):

L'adresse physique effective d'un octet est calculer par la formule:

Adresse physique = Segment x 16 + offset

Exercice 1:

Que fait le programme suivant:

```
    MOV bx, [0300]
    MOV ax, 10
Repet: SHL bx
    DEC ax
    Jne Repet
    MOV [0140], bx
    MOV ah, 4ch
    Int 21h
```

(4 lignes):

Le programme charge le registre bx par la valeur initiale de l'adresse mémoire 0300h, la multiplie par 2^{10} via 10 décalages à gauche, puis stocke le résultat dans l'adresse 0400h. Enfin, il termine son exécution par retour en dos.

Exercice 2:

Ecrire un programme assembleur qui permet de faire les trois taches suivantes:

1°/ lire des chaines de caractères autant de fois qu'on veut.
Utiliser une procédure «lire_chaine» qui permet la lecture d'une chaine.

2°/ Pour chaque chaine lue, calculer la longueur et déposer sa valeur dans le registre AX.
Utiliser une procédure «longueur_chaine» qui calcule la longueur de la chaine.

3°/ Afficher un par ligne les caractères de chaque chaine lue.

```
data SEGMENT
msg_cont db "Veuillez vous continuer? O/N: ",'$'
msg_entrer db "Entrer une chaine: ",'$'
chaine db 20 dup('?'),'$'
data ENDS
code SEGMENT
    ASSUME DS:data, CS:code
    debut:
        MOV AX, data
        MOV DS, AX
    boucle:
        MOV DX, offset msg_cont
        MOV AH, 09h
        INT 21h
        MOV AH, 01h
        INT 21h
        CMP AL, 'O'
        JNE fin
        MOV DL, 0Ah
        MOV AH, 02h
        INT 21h
        MOV DX, offset msg_entrer
        MOV AH, 09h
        INT 21h
        CALL lire_chaine
        CALL longueur_chaine
    JMP boucle
    fin:
        MOV AH, 4Ch
        INT 21h
    lire_chaine PROC NEAR
        MOV DX, offset chaine
        MOV AH, 0Ah
        INT 21h
        RET
    lire_chaine ENDP
    longueur_chaine PROC NEAR
        MOV BX, offset chaine
        MOV CX, 0
```

```
boucle_proc:
    MOV AL, [BX]
    CMP AL, '$'
    JNE fin_proc
    MOV DL, 0AH
    MOV AH, 02h
    INT 21h
    MOV DL, [BX]
    MOV AH, 02h
    INT 21h
    INC CX
    INC BX
JMP boucle_proc
```

```
fin_proc:
    MOV DL, 0Ah
    MOV AH, 02h
    INT 21h
    RET
Longueur_chaine ENDP
code ENDS
END debut
```

totale : 60 lignes

Exercice 3:

Ecrire un programme assembleur qui permet de voir si un caractère minuscule se figure dans une chaine de caractères minuscules ou non, et si le caractère existe, il le convertit en majuscule dans la chaîne. Afficher la chaine modifiée une fois le parcours terminé. La chaine et le caractère doivent être initialement lus.

```
data SEGMENT
  msg_entrer db "Entrer une chaine: ",'$'
  msg_char db "Entrer un caractere min: ",'$'
  msg_apres db "Apres conversion: ",'$'
  chaine db 20 dup('?'), '$'
data ENDS
code SEGMENT
  ASSUME DS:data, CS:code
  debut:
    MOV AX, data
    MOV DS, AX
    MOV DX, offset msg_entrer
    MOV AH, 09h
    INT 21h
    MOV DL, 0AH
    MOV AH, 02h
    INT 21h
    MOV DX, offset msg_char
    MOV AH, 09h
    INT 21h
    MOV AH, 01h
    INT 21h
    MOV CL, AL
    MOV BX, offset chaine
  boucle:
    MOV AL, [BX]
    CMP AL, '$'
    JE fin
```

```
    CMP AL, CL
    JE convertit
    INC BX
JMP boucle
convertit:
    AND AL,11011111b
    MOV [BX], AL
    INC BX
JMP boucle
fin:
    MOV DL, 0Ah
    MOV AH, 02h
    INT 21h
    MOV DX, offset msg-apres
    MOV AH, 09h
    INT 21h
    MOV DX, offset chaine
    MOV AH, 09h
    INT 21h
    MOV AH, 4Ch
    INT 21h
code ENDS
END debut
```

totale : 48 lignes