# Annale Simkaal

1) Dans une instruction, on trouve systématiquement :

- ☐ Une référence à l'instruction suivante
- ☒ Le code opération
- ☐ Une référence à une opérande source
- ☐ Une directive

2) Quel est un des rôles de la pile en mémoire centrale ?

- ☐ Alimenter en énergie le calculateur
- ☐ Stocker la valeur de SP (pointeur de pile)
- ☒ Stocker l'adresse de retour au programme principal lors d'un CALL
- ☐ Stocker l'adresse du sous-programme pointé par le CALL

3) Quelle est la directive de langage assembleur qui permet de réserver plusieurs mots (accessibles en lecture / écriture) en mémoire centrale ?

- ☐ tab: **CST** 10,5,3
- ☐ **ORG** 10
- ☒ tab: **VAR** 5?
- ☐ tab **EQU** 5

4) Le registre R1 contient la valeur 0x6A. Quelles opérations doit-on effectuer pour obtenir les indicateurs C = 1 et Z = 1 ?

- ☐ LDR R0,0xC9 puis ADD R1,R0R1
- ☐ LDR R0,0xC5 puis AND R1,R1R0
- ☒ LDR R0,0x96 puis ADD R1,R0R1
- ☐ Aucune des trois opérations précédentes

**C = 1 => Il y a une retenue**
**Z = 1 => Le résultat vaut 0**

5) A propos des BUS importants dans le calculateur :

- ☐ Il n'y a qu'un seul bus : le bus d'adresses
- ☒ Il y a 3 bus : adresses, données, contrôle
- ☐ Il n'y a qu'un seul bus : le bus de données
- ☐ Il n'y a pas de bus de contrôle

**Flou dans le cours entre b et d..**
**Mais BUS de contrôle dans les tp**

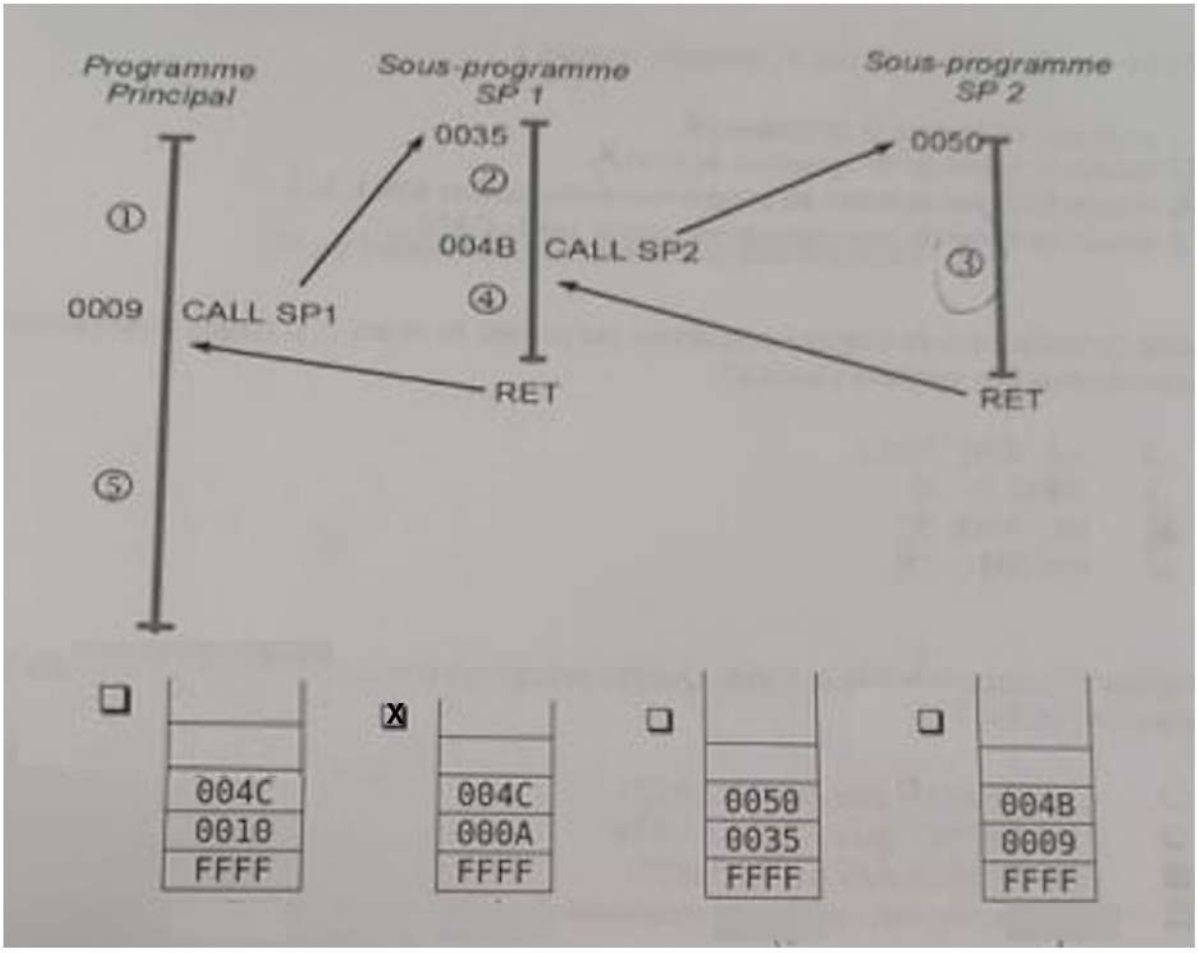
6) Dans le mode d'adressage immédiat :

- [X] L'opérande de l'instruction est une constante
- [ ] Il n'y a pas d'opérande
- [ ] L'opérande de l'instruction est une adresse
- [ ] L'opérande est un registre

7) Quelle est la marche à suivre, à partir du programme source, pour exécuter celui-ci en mémoire centrale ?

- [ ] Éditer le source, l'assembler, le charger en mémoire centrale, l'exécuter
- [X] Éditer le source, l'assembler, le charger en mémoire centrale, mettre IP à 0, exécuter
- [ ] Éditer le source, exécuter le source après RESET
- [ ] Éditer le source, l'assembler, l'exécuter après un RESET

8) Quel est le contenu de la pile dans la phase 3



9) Suite à l'appel du sous-programme suivant :

```
PUSH FL
PUSH R2
LDR R3, R0
PUSH R0
ADD R0, R0R3
POP R0
POP FL
RET
```

**Push -> Envoie la valeur en haut de la pile**
**Pop -> Enlève la valeur du haut de la pile**

**Ici on push 3 valeurs, on en pop que 2, le programme va considérer que l'adresse de retour est la première valeurs qu'on a push, soit FL ici (les flags)**

- ☐ Le registre R0 sera modifié
- ☐ Les registres R0 et FL seront inchangés
- ☒ L'adresse de retour vers le programme principal sera fausse
- ☐ Le Stack Pointer (SP) revient sur le fond de pile

10) Que contient la table des vecteurs d'IT ?

- ☐ L'adresse de la pile
- ☐ L'adresse de retour dans le programme principal après exécution du sous-programme
- ☒ L'adresse des sous-programmes d'interruption
- ☐ L'adresse du pointeur d'instruction

11) Un sous programme « d'interruption » ?

- ☐ Est lancé par un CALL et terminé par un RET
- ☐ N'a pas d'adresse dans la mémoire centrale car on ne le lance pas par un CALL
- ☒ Est accessible via son adresse en table des vecteurs, suite à une interruption
- ☐ Est lancé par un CALL et terminé par un iRET

12) Dans le cadre d'une opération de sortie par « test du mot d'état » :

- ☐ Le CPU est informé automatiquement par le contrôleur d'E/S de la disponibilité du périphérique
- ☐ Le CPU décide de l'état du périphérique en écrivant la valeur qu'il souhaite dans le mot d'état
- ☒ Le CPU doit s'informer de la disponibilité du périphérique
- ☐ Le périphérique est plus rapide

13) Quel élément de l'unité centrale peut effectuer des calculs ?

☒ L'unité Arithmétique et Logique
☐ Le Décodeur
☐ Le Séquenceur
☐ Le Registre Instruction

14) Quelle est la zone mémoire dont l'emplacement ne peut pas être modifié par le programmeur ?

☐ La table des vecteurs d'interruption
☒ La pile
☐ La zone de données
☐ La zone de code

15) Conversion binaire, hexadécimale et décimale, compléter le tableau suivant

| Représentation : Décimale | Hexadécimale | Binaire |
|---|---|---|
| **68** | 44 | **1000100** |
| 77 | **4D** | **1001101** |
| **173** | **AD** | 10101101 |

16) Quel est l'extrait (juste) du dialogue se déroulant entre le contrôleur d'interruption (PIC) et le calculateur ?

☐ Le PIC envoie au Calculateur un EOI (End Of Interrupt)
☐ Le PIC envoie un CALL pour se dérouter dans le sous programme d'interruption
☐ Le calculateur envoie le niveau d'interruption au PIC
☒ Le PIC envoie au calculateur le niveau d'interruption

17) Pendant l'exécution d'un sous-programme (ou procédure) d'interruption :

☒ On ne peut jamais prendre un compte une nouvelle interruption
☐ On prend en compte par défaut toute nouvelle interruption
☐ On peut prendre en compte une nouvelle interruption à l'aide d'instruction(s) appropriée(s)
☐ Les flags ne sont pas sauvegardés

**Assez flou entre a et c, théoriquement c'est possible en changeant l'indice de i (réponse c) mais ça fou le bordel dans le programme à cause des piles etc si un autre programme était en cours**

18) Après l'exécution des quatre instructions suivantes :

```
LDR R0,0x2D
PUSH R0
LDR R0,0x14
LDR R1,R0
LDR R0,0x27
AND R0,R0R1
```

Quel sera le contenu de R0 (en hexadécimal) ?

- [X] 0x04
- [ ] 0x06
- [ ] 0x17
- [ ] 0x27

**ET logique entre R0 qui est 0x27 et R1 qui est 0x14**
**Donc ça donne 0x04**

19) Le fait de faire un RESET sur le simulateur SimKaal :

- [ ] Remet à zéro IP (le pointeur d'instruction)
- [ ] Remet à zéro IP (le pointeur d'instruction) et SP (pointeur de pile)
- [X] Remet à zéro IP (le pointeur d'instruction) et le flags
- [ ] Remet à zéro IP (le pointeur d'instruction) et les registres généraux (R0 à R4)

20) Pour démasquer le bit b2 de ce mot d'état stocké dans R1 il faut faire l'opération suivante :

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|

- [ ] LDR R0,2 puis ADD R0,R1R0
- [ ] LDR R0,4 puis ADD R0,R1R0
- [ ] LDR R0,2 puis AND R0,R1R0
- [X] LDR R0,4 puis AND R0,R1R0

**4 en binaire c'est 100**
**Donc ET logique entre 100 et la chaine B ne va ressortir que b2**
**Si b2 = 1 => 1 ET 1 = 1**
**Si b2 = 0 => 1 ET 0 = 0**