

# Évaluation Intermédiaire 2

## Mémoire & Architecture CPU

**Chapitres couverts :** 03, 04

**Durée :** 1h00

**Documents autorisés :** Aucun

**Barème :** 20 points

## **Partie A : QCM (5 points)**

**Consignes :** Une seule réponse correcte par question. +1 point par bonne réponse, 0 si faux.

## Question 1

Quel composant permet de **mémoriser** un bit ?

- ☐ A. Une porte AND
- ☐ B. Un multiplexeur
- ☐ C. Une bascule D (DFF)
- ☐ D. Un décodeur

## Question 2

Combien d'adresses peut gérer une mémoire avec 10 bits d'adresse ?

- ☐ A. 10
- ☐ B. 100
- ☐ C. 1024
- ☐ D. 10240

### Question 3

Dans l'architecture A32, quelle instruction charge une valeur de la mémoire vers un registre ?

- ☐ A. MOV
- ☐ B. ADD
- ☐ C. LDR
- ☐ D. STR

## Question 4

Quel registre contient l'adresse de la prochaine instruction à exécuter ?

- ☐ A. R0
- ☐ B. SP (Stack Pointer)
- ☐ C. LR (Link Register)
- ☐ D. PC (Program Counter)

## Question 5

Dans une architecture Load/Store, les opérations arithmétiques :

- ☐ A. Peuvent accéder directement à la mémoire
- ☐ B. Travaillent uniquement sur les registres
- ☐ C. Utilisent le bus de données
- ☐ D. Modifient toujours le PC

## **Partie B : Mémoire (7 points)**



### **Exercice 1 : DFF et Registres (2 points)**

- a) Décrivez le comportement d'une bascule D (DFF) sur front montant.
- b) Combien de DFF faut-il pour construire un registre 32 bits ?

## Exercice 2 : Adressage RAM (3 points)

Une mémoire RAM a les caractéristiques suivantes :

- Bus d'adresse : 8 bits
- Bus de données : 32 bits

a) Combien de mots de 32 bits peut contenir cette RAM ?

b) Quelle est la capacité totale en octets ?

c) Si on veut lire l'adresse 0x2A, que faut-il faire avec le signal `load` ?

### **Exercice 3 : Program Counter (2 points)**

Le PC (Program Counter) est un registre spécial.

- a) Quelle est sa fonction principale ?
- b) De combien le PC est-il incrémenté après chaque instruction A32 ? Pourquoi ?

## **Partie C : Architecture & ISA (8 points)**

## Exercice 4 : Format d'instruction (2 points)

Identifiez les champs de cette instruction A32 (32 bits) :

```
1110 000 0100 0 0010 0001 0000 0000 0011
```

Champ	Bits	Valeur
cond	31-28	?
class	27-25	?
op	24-21	?
Rn	19-16	?
Rd	15-12	?
Rm	3-0	?

Quelle instruction cela représente-t-il ?

## Exercice 5 : Trace d'exécution (4 points)

Tracez l'exécution du programme suivant. Donnez la valeur des registres après chaque instruction.

```
MOV R0, #5      ; Instruction 1
MOV R1, #3      ; Instruction 2
ADD R2, R0, R1  ; Instruction 3
SUB R3, R0, R1  ; Instruction 4
```

Instruction	R0	R1	R2	R3	PC
Initial	0	0	0	0	0x0000
Après 1					
Après 2					
Après 3					
Après 4					

## Exercice 6 : Branchement conditionnel (2 points)

Analysez ce code :

```
MOV R0, #10
MOV R1, #10
CMP R0, R1
B.EQ equal
MOV R2, #0
B end
equal:
MOV R2, #1
end:
HALT
```

- a) Que vaut R2 à la fin de l'exécution ?
- b) Quels flags sont modifiés par l'instruction CMP ? Quelle est leur valeur ?

## Barème Détaillé

Partie	Points
A. QCM (5 questions × 1pt)	5
B. Ex1 DFF et Registres	2
B. Ex2 Adressage RAM	3
B. Ex3 Program Counter	2
C. Ex4 Format instruction	2
C. Ex5 Trace d'exécution	4
C. Ex6 Branchement	2
<b>Total</b>	<b>20</b>



# Notes pour la Correction

## Partie A - Réponses QCM

1. **C** (DFF)
2. **C** ( $1024 = 2^{10}$ )
3. **C** (LDR = Load Register)
4. **D** (PC)
5. **B** (registres uniquement)

## Partie B - Solutions

### Ex1 :

- a) Sur front montant du clock, la sortie Q prend la valeur de l'entrée D et la conserve jusqu'au prochain front montant.
- b) 32 DFF (un par bit)

**Ex2 :**

- a)  $2^8 = 256$  mots de 32 bits
- b)  $256 \times 4 = 1024$  octets = 1 Ko
- c) `load = 1` pour activer la lecture

**Ex3 :**

- a) Contient l'adresse de la prochaine instruction à exécuter
- b) +4 octets (car chaque instruction fait 32 bits = 4 octets)

## Partie C - Solutions

Ex4 :

Champ	Bits	Valeur	Signification
cond	31-28	1110	AL (toujours)
class	27-25	000	ALU registre
op	24-21	0100	ADD
Rn	19-16	0010	R2
Rd	15-12	0001	R1
Rm	3-0	0011	R3

**Instruction :** ADD R1, R2, R3

**Ex5 :**

Instruction	R0	R1	R2	R3	PC
Initial	0	0	0	0	0x0000
Après MOV R0, #5	5	0	0	0	0x0004
Après MOV R1, #3	5	3	0	0	0x0008
Après ADD R2, R0, R1	5	3	8	0	0x000C
Après SUB R3, R0, R1	5	3	8	2	0x0010

### Ex6 :

- a) **R2 = 1** (car  $R0 == R1$ , donc B.EQ est pris)
- b) CMP modifie N, Z, C, V. Ici :
  - **Z = 1** ( $R0 - R1 = 0$ )
  - **N = 0** (résultat non négatif)
  - **C = 1** (pas d'emprunt)
  - **V = 0** (pas d'overflow)

*Fin de l'évaluation*