Algorithmes et Pensée Computationnelle

Architecture des ordinateurs - Exercices de base

Le but de cette séance est de comprendre le fonctionnement d'un ordinateur. La série d'exercices sera axée sur les conversions en base binaire, décimale ou hexadécimale, ainsi que des opérations de base en suivant le modèle Von Neumann.

1 Conversions

Question 1: (*5 minutes*) **Conversion** $Base_{10}$ **-** $Base_2$

1. Convertir le nombre $10_{(10)}$ en base 2.
2. Convertir le nombre $45_{(10)}$ en base 2.
3. Convertir le nombre $173_{(10)}$ en base 2.

Conseil

Vous pouvez utiliser la table des puissances de 2 pour vous aider.

puissance	2^5	2^4	2^3	2^2	2^1	2^0
valeur	32	16	8	4	2	1

Question 2: (*15 minutes*) **Conversion** $Base_{10}$ **-** $Base_3$**,** $Base_8$**,** $Base_{16}$

1. Convertir le nombre $40_{(10)}$ en base 8.
2. Convertir le nombre $52_{(10)}$ en base 3.
3. Convertir le nombre $254_{(10)}$ en base 16.

Conseil

S'inspirer de la solution de la question précédente.

N'oubliez pas qu'en hexadécimal, A vaut 10, B vaut 11, C vaut 12, D vaut 13, E vaut 14 et F vaut 15 !

Question 3: (*15 minutes*) **Conversion** $Base_2$**,** $Base_5$**,** $Base_{16}$ **en** $Base_{10}$

1. Convertir le nombre $10110_{(2)}$ en base 10.
2. Convertir le nombre $4321_{(5)}$ en base 10.
3. Convertir le nombre $ABC_{(16)}$ en base 10.

Conseil

N'oubliez pas qu'en hexadécimal, A vaut 10, B vaut 11, C vaut 12, D vaut 13, E vaut 14 et F vaut 15 !

2 Arithmétique

Question 4: (*15 minutes*) **Addition de nombres binaires**

1. Additionner $01010101_{(2)}$ et $10101010_{(2)}$
2. Additionner $01011111_{(2)}$ et $10000001_{(2)}$
3. Additionner $01110100_{(2)}$ et $00011010_{(2)}$

Conseil

Table d'addition binaire :

retenue	1	0	0	0	0
a	0	1	1	0	0
b	0	1	0	1	0
a+b	1	0	1	1	0

Question 5: (*15 minutes*) **Soustraction de nombres binaires**

Effectuer les opérations suivantes :

1. $01111111_{(2)}$ - $01000000_{(2)}$
2. $10000000_{(2)}$ - $00000001_{(2)}$
3. $10101010_{(2)}$ - $01010101_{(2)}$

Conseil

Table de soustraction binaire :

retenue	0	1	0	0
a	1	1	0	0
b	1	0	1	0
a-b	0	0	1	0

Exemple :

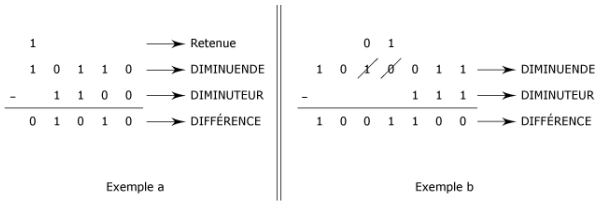

FIGURE 1 – Exemple de soustraction de nombres binaires

3 Modèle de Von Neumann

Dans cette section, nous allons simuler une opération d'addition dans le **modèle de Van Neumann**, il va vous être demandé à chaque étape (FDES) de donner la valeur des registres.

État d'origine :
A l'origine, notre `Program Counter (PC)` vaut `00100000`.

Dans la mémoire, les instructions sont les suivantes :

Adresse	Valeur
00011111	00100100
00100000	10110110
00100001	11101101

Les registres contiennent les valeurs suivantes :

Registre	Valeur
00	11100011
01	01101100
10	00100101
11	00000000

Les opérations disponibles pour l'unité de contrôle sont les suivantes :

Numéro	Valeur
00	MOV
01	XOR
10	ADD
11	SUB

Question 6: (*10 minutes*) **Fetch**
À la fin de l'opération `FETCH`, quelles sont les valeurs du `Program Counter` et de l'`Instruction Register` ?

Conseil

Pour rappel, l'**unité de contrôle (Control Unit)** commande et contrôle le fonctionnement du système. Elle est chargée du **séquençage** des opérations. Après chaque opération `FETCH`, la valeur du **Program Counter** est incrémentée (valeur initiale + 1).

Question 7: (*10 minutes*) **Decode**

1. Quelle est la valeur de l'opération à exécuter ?
2. Quelle est l'adresse du registre dans lequel le résultat doit être enregistré ?
3. Quelle est la valeur du premier nombre de l'opération ?
4. Quelle est la valeur du deuxième nombre de l'opération ?

Conseil

Pensez à décomposer la valeur de l'`Instruction Register` pour obtenir toutes les informations demandées.
Utilisez la même convention que celle présentée dans les diapositives du cours (Computer Architecture - Page 15)
Les données issues de la décomposition de l'`Instruction Register` ne sont pas des valeurs brutes, mais des références. Trouvez les tables concordantes pour y récupérer les valeurs.

Question 8: (*10 minutes*) **Execute**
Quel est résultat de l'opération ?

Conseil

Toutes les informations permettant d'effectuer l'opération se trouvent dans les données de l'`Instruction Register`.