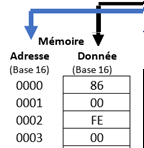
**CORRECTION des Recherches documentaires : « Architectures matérielles et assembleur »**

**Hardware**

* **La mémoire**

1. ***Quel est le rôle de la mémoire dans un système à microprocesseur ?***



La mémoire sert à **stocker** les informations nécessaires au fonctionnement du système : les **instructions** du programme (**code**) et les **données** manipulées par ces instructions. Elle constitue l'espace de travail du microprocesseur où il lit les instructions à exécuter et accède aux données à traiter.

1. **Qu’entend-on par "mot mémoire" ?**

Un mot mémoire représente **l'unité d'information élémentaire** stockée à une adresse donnée.

* **Le bus d’adresse**

1. ***A quoi sert le bus d’adresse dans un système à microprocesseur ?***

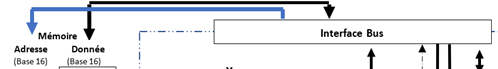
Le bus d'adresse permet au microprocesseur de **sélectionner un emplacement précis en mémoire** ou un périphérique avec lequel il souhaite communiquer. Il transporte l'adresse de l'emplacement à lire ou à écrire. C'est comme une adresse postale qui permet de cibler exactement où l'information doit être lue ou écrite.

1. ***Quelle est la direction des signaux sur le bus d’adresse : du processeur vers la mémoire ou inversement ?***

Le bus d'adresse est **unidirectionnel** : les signaux vont toujours du microprocesseur vers la mémoire ou les périphériques. C'est le processeur qui décide quelle adresse il veut accéder, jamais l'inverse. La mémoire est passive et répond uniquement aux requêtes du processeur.

* **Le bus de données**

1. ***Quelle est la fonction du bus de données ?***



Le bus de données **transporte les informations échangées** entre le microprocesseur et la mémoire ou les périphériques. Il véhicule aussi bien les instructions à exécuter que les données à traiter.

1. ***En quoi le bus de données est-il bidirectionnel ?***

Le bus de données est bidirectionnel, car les informations peuvent **circuler dans les deux sens**. Du processeur vers la mémoire lors d'une écriture et de la mémoire vers le processeur lors d'une lecture

1. ***Quelle est la différence fondamentale entre le bus d’adresse et le bus de données.***

Le bus d'adresse indique **OÙ** l'information se trouve (l'emplacement), tandis que le bus de données transporte **QUELLE** information est échangée (le contenu).

* **L’accumulateur**

1. ***Qu’est-ce qu’un accumulateur et pourquoi est-il fondamental dans un microprocesseur ?***

L'accumulateur est un **registre de travail rapide** situé au cœur du processeur. Il stocke temporairement les données pendant les opérations arithmétiques et logiques. Il est essentiel, car l'UAL (Unité Arithmétique et Logique) travaille principalement avec ce registre, permettant des calculs beaucoup plus rapides qu'avec la mémoire principale.

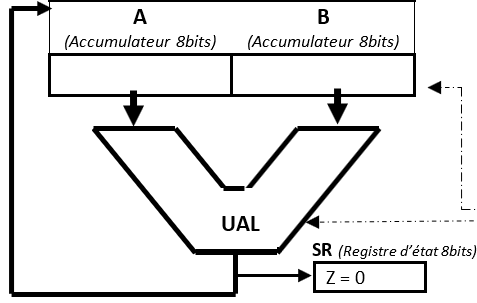
1. ***Quels types d’opérations utilise un accumulateur ?***

Les accumulateurs sont utilisés pour les **opérations arithmétiques** (addition, soustraction, incrémentation), les **opérations logiques** (ET, OU, OU exclusif, complément), les opérations de décalage et rotation, les comparaisons et tests, ainsi que pour les transferts de données entre mémoire et processeur. Pratiquement toute manipulation de donnée passe par un accumulateur.

1. ***Peut-on effectuer des opérations entre deux emplacements mémoire, ou l’accumulateur est-il obligatoire ?***

**En principe non, l'accumulateur est obligatoire**. On ne peut pas effectuer d'opérations directement entre deux "emplacements mémoire". Il faut d'abord charger une valeur dans un registre (accumulateur), effectuer l'opération, puis stocker le résultat en mémoire si nécessaire. C'est une architecture dite "load-store" ou à accumulateur.

* **L’unité arithmétique et logique (UAL)**



1. ***Quel est le rôle de l’unité arithmétique et logique ?***

L'UAL est le composant qui **effectue tous les calculs et opérations logiques** du microprocesseur.

1. ***Quelles sont les principales opérations logiques (booléennes) effectuées par l’UAL ?***

Les opérations logiques incluent le **ET logique**, le **OU logique**, le **OU exclusif**, le **complément**, ainsi que les décalages (shift) et rotations à gauche ou à droite.

1. ***Quelles sont les principales opérations arithmétiques effectuées par l’UAL ?***

**Addition, soustraction, incrémentation, décrémentation**, comparaisons, négations et opérations sur les nombres en complément à deux. Elle gère également les retenues pour les opérations multioctets et peut effectuer des additions avec retenue et soustractions avec retenue, etc.

* **Le registre d’état**

1. ***Qu’appelle-t-on registre d’état et que contient-il ?***

Le registre d'état (**SR : State Register**), aussi appelé CCR (Condition Code Register), est un registre dont chaque bit individuel (appelé drapeau ou flag) indique une information particulière sur le résultat de la dernière opération effectuée par l'UAL. Il permet au processeur de "se souvenir" des caractéristiques du dernier calcul.

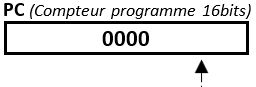
1. ***Quels sont les principaux indicateurs (flags) présents dans un registre d’état ? Donnez leur rôle.***

**I** (Interrupt Mask, masque d'interruption), **N** (Negative, résultat négatif), **Z** (Zero, résultat nul), **V** (Overflow, dépassement de capacité en arithmétique signée), et **C** (Carry, retenue ou emprunt).

* **Le compteur programme**

1. ***Qu’elle est la fonction du compteur programme ?***

Le compteur programme (PC, Program Counter) est un **registre qui contient l'adresse de la prochaine instruction à exécuter**. Il agit comme un pointeur qui suit l'avancement du programme, permettant au processeur de savoir où il en est dans l'exécution du code. C'est essentiellement le "marque-page" du processeur dans le programme.

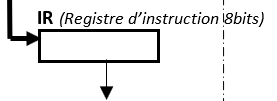


1. ***Comment le compteur programme évolue-t-il lors de l'exécution d'un programme ?***

Après le chargement (fetch) de chaque instruction, le **PC est automatiquement modifié pour pointer vers l'instruction suivante**.

* **Le registre d’instructions**

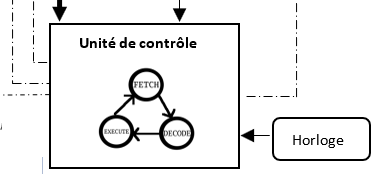
1. ***À quoi sert le registre d'instructions ?***



Le registre d'instructions (IR, Instruction Register) **conserve temporairement l'instruction en cours d'exécution** après qu'elle ait été lue depuis la mémoire. Il maintient le code de l'instruction pendant tout le temps nécessaire à son décodage et à son exécution, permettant à l'unité de contrôle d'analyser ce qu'elle doit faire.

* **L’unité de contrôle**

1. ***Quelle est la fonction principale de l'unité de contrôle ?***



L'unité de contrôle est le **chef d'orchestre du microprocesseur**. Elle coordonne et synchronise toutes les opérations en générant les signaux de contrôle appropriés pour les différents composants. Elle **décode les instructions**, séquence les étapes d'exécution et gère le timing de toutes les opérations selon le rythme imposé par l'horloge.

* **L’horloge**

1. ***Pourquoi un microprocesseur a-t-il besoin d'une horloge ?***

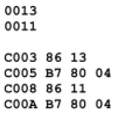
L'horloge fournit un signal périodique qui **synchronise toutes les opérations du microprocesseur**. Sans horloge, les différents composants ne pourraient pas coordonner leurs actions dans le temps. Elle garantit que chaque étape d'une instruction se déroule dans l'ordre correct, que les données sont stables avant d'être lues, et que les composants ne travaillent pas les uns contre les autres.

1. ***Comment la fréquence de l'horloge influence-t-elle les performances du système ?***

La fréquence d'horloge, mesurée en **Hertz (Hz), détermine la vitesse d'exécution du processeur**. Plus la fréquence est élevée, plus le processeur peut exécuter d'instructions par seconde, améliorant ainsi les performances globales. Cependant, augmenter la fréquence a des **limites physiques** et énergétiques : **dissipation thermique accrue**, **consommation électrique plus importante** et contraintes technologiques des composants.

**Software**

* **Langage machine**



Adresses

Une Instruction

par ligne

1. ***Qu’est-ce que le langage machine ?***

Le langage machine est le **langage de programmation de plus bas niveau**, directement **compréhensible et exécutable par le processeur** d'un ordinateur. Il est constitué uniquement de séquences de bits (0 et 1) qui représentent les instructions que le processeur peut effectuer directement, sans traduction ni interprétation.

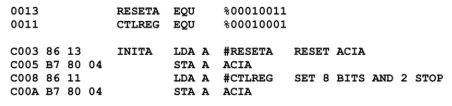
1. ***Comment s’appellent les mots du langage machine ?***

Les mots du langage machine s'appellent des **instructions-machine** ou instructions binaires. On peut aussi les appeler des mots machine.

1. ***Qu’est-ce qu’un programme du point de vue du processeur ?***

Du point de vue du processeur, **un programme est une suite d'instructions-machine** stockées en mémoire, que le processeur lit et exécute séquentiellement (ou dans un ordre modifié par des instructions de saut). C'est essentiellement une séquence de nombres binaires que le processeur interprète comme des commandes à effectuer.

* **Structure d’une instruction**



Extrait d’un programme écrit en assembleur 6802 (1980) et code machine correspondant

Opcode

Opérande

Opcode

Opérande

1. ***Définissez les mots suivants et donnez un exemple : Opcode, opérande***

* **Opcode** (code opération) : C'est la partie de l'instruction qui spécifie **l'opération à effectuer** par le processeur (addition, soustraction, chargement, etc.).

Exemple : LDA A (**8616**) pour charger une donnée dans le registre A,

STA A pour stocker une donnée de A vers la mémoire.

* **Opérande** : C'est la partie de l'instruction qui indique sur quelles **données** l'opération doit être effectuée (registres, adresses mémoire, valeurs immédiates).

Exemple : #RESETA (**1316**) est un opérande (valeur)

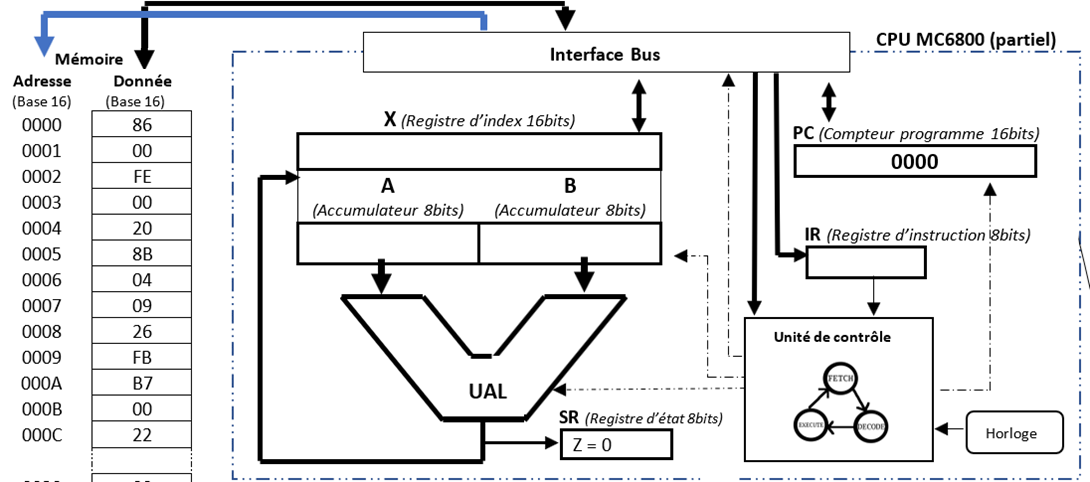
* **Assembleur**

1. ***Qu’est-ce que l’assembleur ?***

**L'assembleur a deux significations :**

* **Le langage assembleur** : un langage de programmation de bas niveau qui utilise des mnémoniques (symboles textuels lisibles) pour représenter les instructions-machine, facilitant ainsi la programmation par rapport au binaire pur.
* **L'assembleur (programme)** : l'outil logiciel qui traduit le code écrit en langage assembleur en langage machine exécutable par le processeur. C'est un traducteur qui convertit les mnémoniques en codes binaires correspondants.

**Exemple de microprocesseur : MC6800 exécutant un programme en mémoire**

****