Architecture des machines et des systèmes informatiques

Alain Cazes
Ancien maître de conférences en informatique
au Conservatoire national des Arts et Métiers

Joëlle Delacroix
Maître de conférences en informatique
au Conservatoire national des Arts et Métiers

6e édition

DUNOD

Illustration de couverture : Fotolia

11 rue Paul Bert, 92240 Malakoff
www.dunod.com

ISBN 978-2-10-077947-5

Table des matières

PARTIE 2 • STRUCTURE DE L'ORDINATEUR

Chapitre **1**

Structure générale et fonctionnement d'un ordinateur

Dans cette partie introductive nous rappelons quelques éléments fondamentaux concernant la programmation et l'algorithmique afin de présenter le vocabulaire utilisé. Il ne s'agit en aucune manière de se substituer à un cours d'algorithmique mais uniquement de replacer du vocabulaire du point de vue de la structure générale d'un ordinateur, l'objectif étant de mettre en évidence les différentes phases qui interviennent dans la résolution d'un problème avec un ordinateur. Après cette partie introductive nous présentons les principaux modules constituant l'architecture d'un ordinateur type. Nous faisons un tour d'horizon des fonctionnalités de chacun de ces modules et de leurs relations fonctionnelles. Il s'agit ici uniquement de présenter de manière globale le fonctionnement de l'ordinateur.

1.1 INTRODUCTION

Le rôle de l'informatique est de résoudre des problèmes à l'aide d'un ordinateur. Un problème s'exprime sous la forme d'un énoncé qui spécifie les fonctions que l'on souhaite réaliser. Par exemple définir toutes les fonctions d'un traitement de texte. Pour résoudre un problème les informaticiens utilisent la notion d'*algorithme*.

Pour illustrer cette notion, prenons l'exemple du problème suivant : c*onfectionner* une omelette avec 6 *œufs*.

Trouver une solution à ce problème repose sur l'existence d'un *processeur* sachant exécuter une *instruction* (*confectionner*). En général un adulte saura exécuter l'instruction confectionner, c'est-à-dire connaîtra le sens du mot, et sera capable de faire toutes les actions nécessaires permettant de résoudre le problème. On dira alors que l'adulte est un *bon processeur* au sens où il saura *exécuter* l'instruction *confectionner* portant sur la *donnée* œufs. Par contre un enfant pourra ne pas connaître le mot confectionner : il ne saura pas faire les opérations nécessaires et ne pourra donc pas résoudre le problème posé (faire une omelette). L'enfant connaît d'autres instructions, sait exécuter d'autres actions que confectionner, et pour qu'il puisse résoudre le problème il faudra l'exprimer autrement, sur la base des actions, *instructions*, qu'il est *capable* d'exécuter. Pour que l'enfant puisse résoudre le problème on pourra l'exprimer sous la forme d'une séquence d'instructions appartenant au *langage* de l'enfant. Par exemple on pourra exprimer le problème, la solution, sous la forme de la séquence des instructions suivantes :

1. *casser* 6 *œufs* dans un bol ;
2. *battre* les *œufs* avec un fouet ;
3. *saler*, *poivrer* ;
4. *placer* la *poêle* sur le *gaz* ;
5. *allumer* le *gaz* ;
6. *cuisiner* les *œufs* ;
7. *éteindre* le *gaz*.

Dans cet exemple, le processeur enfant sait exécuter des instructions (casser, battre, saler, poivrer, allumer, cuisiner…). De plus il connaît les objets à manipuler (œufs, gaz, poêle…). On dit alors que le processeur enfant est un bon processeur pour exécuter l'algorithme représenté par la séquence précédente puisque l'enfant est capable d'exécuter cette séquence d'instructions. Cette séquence d'instructions exécutables par le processeur enfant est une solution du problème posé pour ce processeur.

Un algorithme peut donc se définir comme une séquence d'instructions exécutables par un processeur déterminé. Cette séquence d'instructions, cet algorithme, est une solution au problème posé.

De ce petit exemple nous pouvons tirer quelques conclusions :

– un algorithme est une solution à un problème posé ;
– un algorithme est une séquence d'instructions exécutables par un processeur ;
– un algorithme est un programme exécutable par un processeur déterminé ;
– un algorithme n'a de sens que pour un processeur déterminé.

Les instructions expriment les actions que peut exécuter un processeur, elles sont codées à partir d'un alphabet (dans notre cas l'alphabet habituel). Les instructions manipulent des données (œufs, sel, poivre, gaz, poêle…). Un processeur (ou machine virtuelle) est une entité capable d'exécuter des instructions portant sur des données. L'ensemble des instructions que le processeur (la machine virtuelle) peut manipuler, constitue son langage de programmation. Ainsi le langage de programmation du processeur définit complètement ce processeur. Il y a équivalence totale entre la

machine virtuelle et son langage de programmation. Aussi, *connaître le langage de programmation d'une machine virtuelle équivaut à connaître les capacités d'exécution de cette machine*.

En résumé résoudre un problème *avec une machine virtuelle* consiste à construire *une séquence d'instructions pour cette machine* (à partir de son langage de programmation) telle que l'exécution de *cette séquence* soit une solution à *ce* problème. En informatique la machine cible, celle avec laquelle nous devons résoudre les problèmes, est l'ordinateur. Nous devons donc connaître les caractéristiques de cette machine, tout particulièrement son langage de programmation (les instructions qu'elle est capable d'exécuter), l'alphabet permettant de coder les instructions ainsi que les données et les outils permettant d'exécuter ces instructions.

Les instructions d'un ordinateur sont *les instructions machines,* elles constituent le langage de programmation de l'ordinateur : *le langage machine.* Résoudre un problème avec un ordinateur consiste donc à exprimer ce problème sous la forme d'une séquence d'instructions machines que nous devrons soumettre aux outils permettant l'exécution de cette séquence. Cette séquence d'instructions machine exécutables par l'ordinateur s'appelle *le programme machine.*

1.2 STRUCTURE ET FONCTIONNEMENT D'UN ORDINATEUR

Après ce bref rappel sur la manière algorithmique de résoudre un problème nous allons nous intéresser à la résolution d'un problème avec comme machine cible un ordinateur. Pour cela nous donnons tout d'abord une présentation de la structure matérielle d'un ordinateur, de son fonctionnement, pour ainsi en déduire comment on peut à l'aide d'un ordinateur, résoudre un problème. L'ordinateur cible nous servant de support descriptif est un ordinateur de type Von Neumann qui caractérise bien la quasi-totalité des ordinateurs actuels. Il est composé des éléments suivants :

- une *mémoire centrale* pour le stockage des informations (programme et données) ;
- un *microprocesseur* ou processeur central pour le traitement des informations logées dans la mémoire centrale ;
- des *unités de contrôle* des périphériques et des périphériques ;
- un *bus de communication* entre ces différents modules.

1.2.1 Structure générale d'un ordinateur

La figure 1.1 présente l'organisation générale d'un ordinateur. On y trouve deux parties principales :

- le processeur comprenant les modules mémoire centrale, processeur central (microprocesseur), les unités d'échange et le bus de communication entre ces différents modules ;

– les périphériques avec lesquels dialogue le processeur au travers des unités d'échange (ou contrôleurs). On distingue en général :
  – les périphériques d'entrée tels que le clavier ou la souris ;
  – les périphériques de sortie tels que les imprimantes et les écrans de visualisation ;
  – les périphériques d'entrée et de sortie tels que les disques magnétiques ou les modems pour accéder aux réseaux de communication.

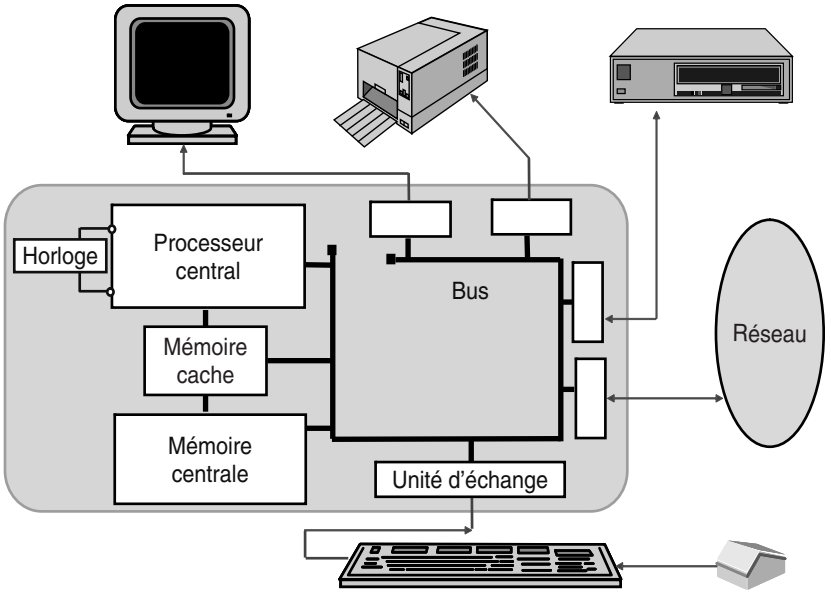


Figure 1.1 Structure matérielle générale.

Globalement le processeur permet l'exécution d'un programme. Chaque processeur dispose d'un langage de programmation (les instructions machine) spécifique. Ainsi résoudre un problème avec un processeur consiste à exprimer ce problème comme une suite de ses instructions machine. La solution à un problème est donc spécifique de chaque processeur. Le programme machine et les données qui sont manipulées par les instructions machine sont placés dans la mémoire centrale.

Examinons à présent plus en détail la composition et les fonctions de chacun des modules composant le processeur.

1.2.2 La mémoire centrale

La mémoire centrale assure la fonction de stockage de l'information qui peut être manipulée par le microprocesseur (processeur central), c'est-à-dire le programme machine accompagné de ses données. En effet, le microprocesseur n'est capable d'exécuter une instruction que si elle est placée dans la mémoire centrale.

Cette mémoire est constituée de circuits élémentaires nommés bits (*binary digit*). Il s'agit de circuits électroniques qui présentent deux états stables codés sous la forme d'un 0 ou d'un 1. De par sa structure la mémoire centrale permet donc de coder les informations sur la base d'un alphabet binaire et toute information stockée en mémoire centrale est représentée sous la forme d'une suite de digits binaires. La figure 1.2 présente l'organisation générale d'une mémoire centrale.

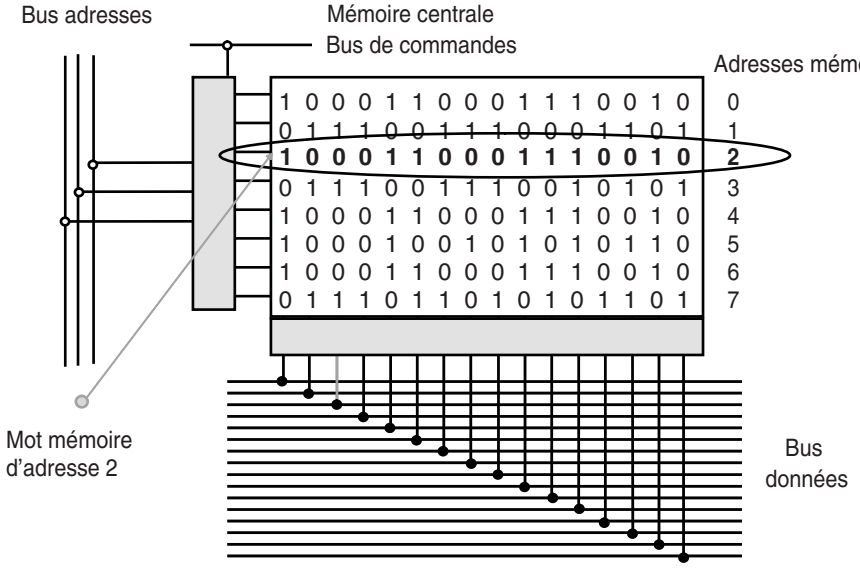


Figure 1.2 La mémoire centrale.

Pour stocker l'information la mémoire est découpée en cellules mémoires : *les mots mémoires*. Chaque mot est constitué par un certain nombre de bits qui définissent sa taille. On peut ainsi trouver des mots de 1 bit, 4 bits (*quartet*) ou encore 8 bits (*octet* ou *byte*), 16 bits voire 32 ou 64 bits. Chaque mot est repéré dans la mémoire par une *adresse*, un numéro qui identifie le mot mémoire. Ainsi un mot est un contenant accessible par son adresse et la suite de digits binaires composant le mot représente le contenu ou valeur de l'information.

La mémoire centrale est un module de stockage de l'information dont la valeur est codée sur des mots. L'information est accessible par mot. La capacité de stockage de la mémoire est définie comme étant le nombre de mots constituant celle-ci. Dans l'exemple de la figure 1.2, notre mémoire a une capacité de 8 mots de 16 bits chacun. On exprime également cette capacité en nombre d'octets ou de bits. Notre mémoire a donc une capacité de 16 octets ou de 128 bits. L'information que l'on trouve en mémoire centrale est donc codée sur un alphabet binaire. La figure 1.3 rappelle le nombre de combinaisons que l'on peut réaliser à partir d'une suite d'éléments binaires. Coder l'information en mémoire centrale c'est donc associer à chaque suite de bits un sens particulier.

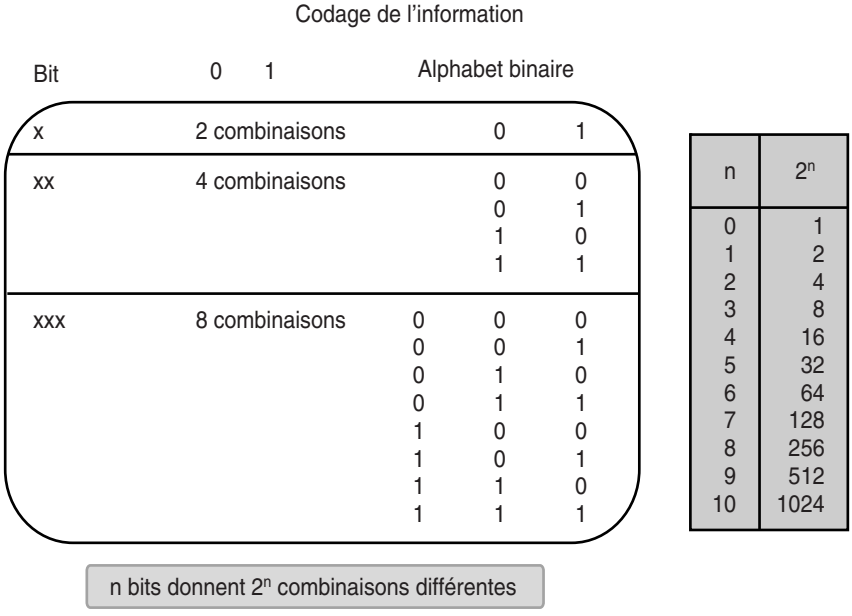


Figure 1.3 Codage binaire.

La figure 1.4 présente succinctement les différentes informations que l'on trouve dans la mémoire centrale : instructions machines et données manipulées par les instructions.

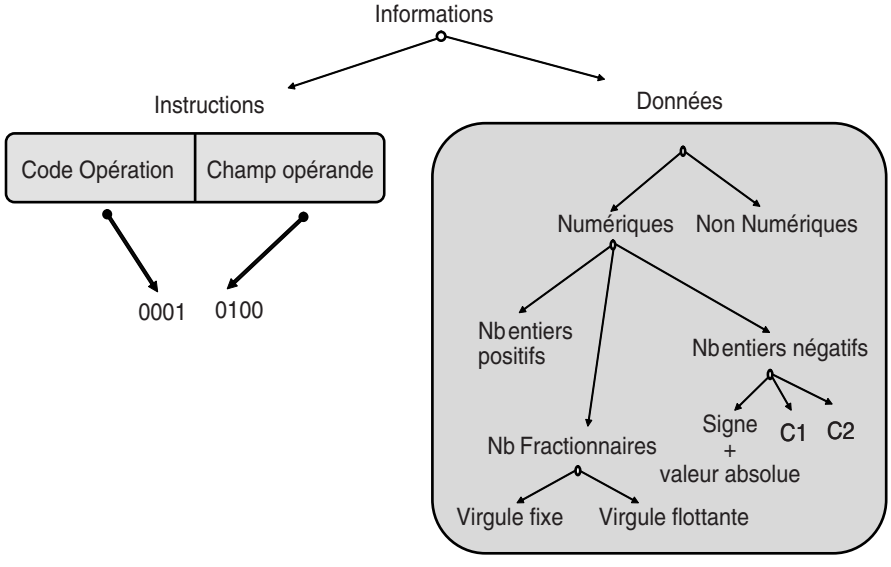


Figure 1.4 Les différentes informations présentes en mémoire centrale.

Les instructions et les données sont codées sur des mots mémoires : elles peuvent occuper un ou plusieurs mots mémoires selon la nature de l'ordinateur. Les instructions machines sont propres à chaque microprocesseur mais sont toujours construites de la même manière : un code opération qui définit l'opération à exécuter, le champ opérande qui définit la ou les données sur lesquelles portent l'opération :

– le *code opération* est codé sur un nombre de digits binaires qui caractérise un microprocesseur. Ce nombre de bits définit en fait le nombre d'opérations possibles avec cet ordinateur : un code opération sur 3 bits admet 8 combinaisons permettant la définition de 8 opérations différentes (instructions machine) possibles, sur 4 bits 16 instructions possibles etc. La taille du code opération est donc un facteur déterminant qui caractérise complètement le nombre d'instructions qu'est capable d'exécuter un processeur;
– le *champ opérande* est une suite de bits qui permet de caractériser l'adresse de la ou des donnée(s) que manipule(nt) l'instruction machine définie par le code opération. Il existe plusieurs types d'instructions machines qui peuvent manipuler une ou plusieurs données selon la « puissance » du langage machine du microprocesseur utilisé. Il existe également plusieurs manières de définir, à partir du champ opérande, l'adresse d'une donnée : cela repose sur le mécanisme d'adressage d'un microprocesseur qui définit les différentes manières de calculer une adresse de données. On parle également de *modes d'adressages* du microprocesseur.

Les données sont les objets que manipulent les instructions, elles sont codées sur un ou plusieurs mots machines et sont donc adressables (repérables) dans la mémoire centrale. L'adresse de la donnée est déterminée par le type d'adressage utilisé par l'instruction machine. Le codage d'une donnée en mémoire dépend de son type : la figure 1.4 donne les différents types de données que manipulent les instructions machines. Pour chaque type il existe des règles de codage. Par exemple pour coder les caractères alphanumériques on utilise un dictionnaire (table ASCII, table EBCDIC, codage Unicode) tandis que pour coder un nombre entier non signé on utilise une règle traditionnelle de codage d'un nombre sur un alphabet binaire.

Dans l'exemple de la figure 1.5, on suppose un nombre codé sur un octet (8 bits) dont la position de chaque bit est numérotée de 0 à 7, en partant du bit de poids

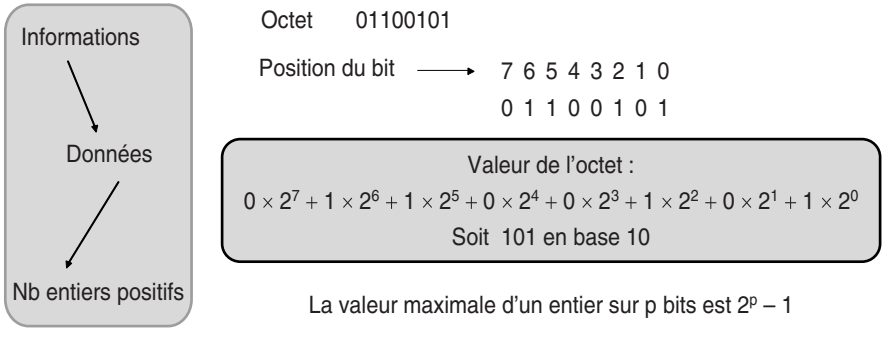


Figure 1.5 Un exemple de codage de l'information.

faible. La valeur de l'entier est alors la somme des produits de chaque bit par le nombre de symboles possibles dans la base d'expression du nombre (ici 2) élevé à la puissance du rang du bit. Cette règle est générale et permet de déterminer la valeur d'un entier codé sur n'importe quel alphabet. Dans le cas d'un alphabet binaire la valeur maximale que peut prendre un entier codé sur p bits est 2^p. Les principales normes de codages existantes à l'heure actuelle ainsi que la structure des instructions machine et les modes d'adressages courants sont détaillés au chapitre 4.

La mémoire centrale a pour objet le stockage des instructions et des données que peut manipuler le microprocesseur. Les opérations possibles sur la mémoire sont la lecture (acquisition par le microprocesseur) d'un mot et l'écriture (le microprocesseur place un nouveau contenu) dans un mot mémoire. Une opération de lecture d'un mot consiste à définir l'adresse du mot et à déclencher une commande de lecture qui amène le contenu du mot de la mémoire vers le microprocesseur. Une opération d'écriture consiste à définir l'adresse du mot dont on veut changer le contenu puis à déclencher une opération d'écriture qui transfère l'information du processeur vers le mot mémoire dont l'adresse est spécifiée.

Enfin d'autres éléments importants complètent la caractérisation d'une mémoire centrale :

– le temps d'accès à la mémoire qui mesure le temps nécessaire pour obtenir une information logée en mémoire ;
– les technologies qui président à la construction de ces mémoires ;
– le coût de réalisation de ces mémoires.

Nous reviendrons en détail sur l'ensemble de ces points dans le chapitre sur la mémorisation (chapitre 8).

1.2.3 Le bus de communication

Le *bus de communication* peut se représenter comme une nappe de fils transportant des signaux et permettant l'échange des informations entre les différents modules du processeur. Chaque fil transporte ou non un signal : il est présent ou absent. On représente par 1 un signal présent et par 0 un signal absent. Le nombre de fils du bus détermine sa largeur et définit ainsi le nombre d'informations différentes que peut véhiculer le bus. Ainsi un bus de 3 fils permet une combinaison de 8 signaux différents et donc représente 8 informations possibles différentes.

Le bus est construit comme un ensemble de trois bus :

– *le bus d'adresses* transporte des combinaisons de signaux qui sont interprétées comme des nombres entiers représentant l'adresse d'un mot mémoire. Par exemple, figure 1.6, le bus d'adresses a une largeur de 3 fils et est donc capable de coder des adresses allant de 0 à 7. Pour adresser un mot mémoire on fait appel à un circuit de sélection (décodeur) qui en entrée reçoit n signaux (3 dans notre exemple) et fournit 2^n signaux de sortie (8 dans notre exemple). Parmi les signaux de sortie un seul est positionné à 1 tous les autres valant 0. Dans notre exemple, les sorties sont numérotées de 0 à 7 et de haut en bas. Ainsi si la valeur d'entrée est 000, seule la

sortie 0 vaut 1 et toutes les autres valent 0. Pour adresser le mot mémoire 2 le microprocesseur place sur le bus d'adresses la chaîne 010 (qui a pour valeur 2 en base 10), la sortie 1 du décodeur vaut alors 1 et les autres valent 0. Ce circuit permet donc de sélectionner un mot mémoire dans la mémoire centrale. La largeur du bus d'adresses définit la *capacité d'adressage du microprocesseur* et il ne faut pas confondre capacité d'adressage et taille physique de la mémoire ;

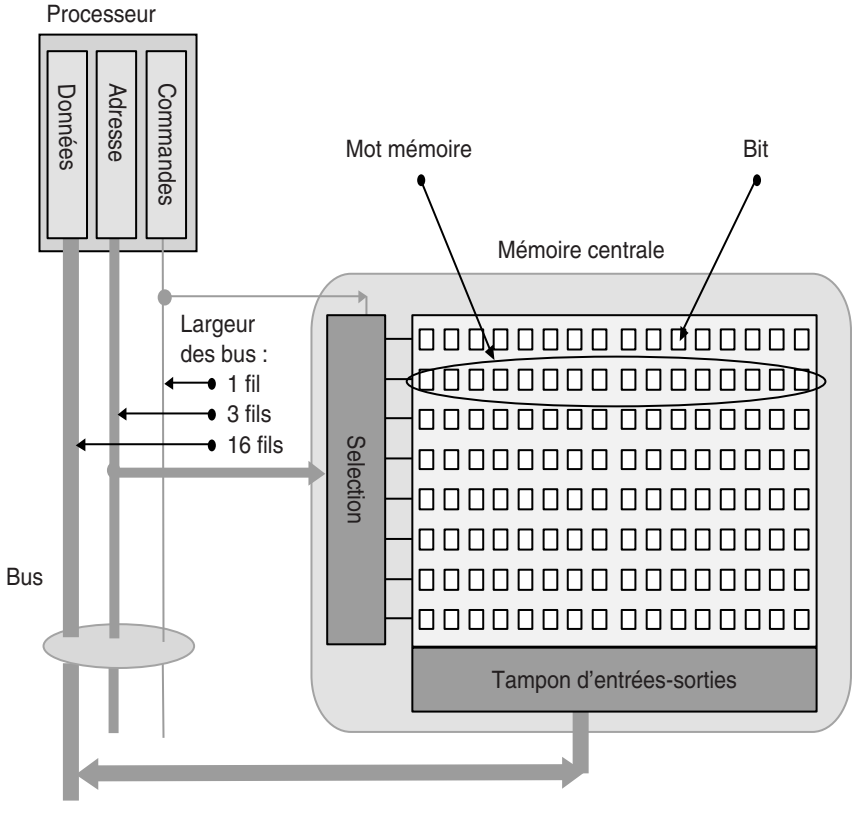


Figure 1.6 Bus de communication.

– le *bus de données* permet l'échange des informations (les contenus) entre les différents modules. Dans notre exemple le bus de données a une largeur de 16 fils et donc la taille des mots mémoires auxquels on peut accéder ou dont on peut modifier le contenu est de 16 bits ;
– le *bus de commandes* : c'est par ce bus que le microprocesseur indique la nature des opérations qu'il veut effectuer. Dans notre exemple il a une largeur d'un fil et donc le microprocesseur ne peut passer que deux commandes (la lecture et l'écriture).

La figure 1.7 résume les différents points abordés concernant la mémoire et les bus permettant la communication avec la mémoire. Sur cette figure 1.7, le bus

d'adresses a une largeur de m bits, le bus de données une largeur de p bits ce qui détermine la capacité de stockage en bits de cette mémoire, soit 2^m mots de p bits. Le bus de commandes permet de déterminer le type d'opération (lecture ou écriture) que l'on souhaite réaliser.

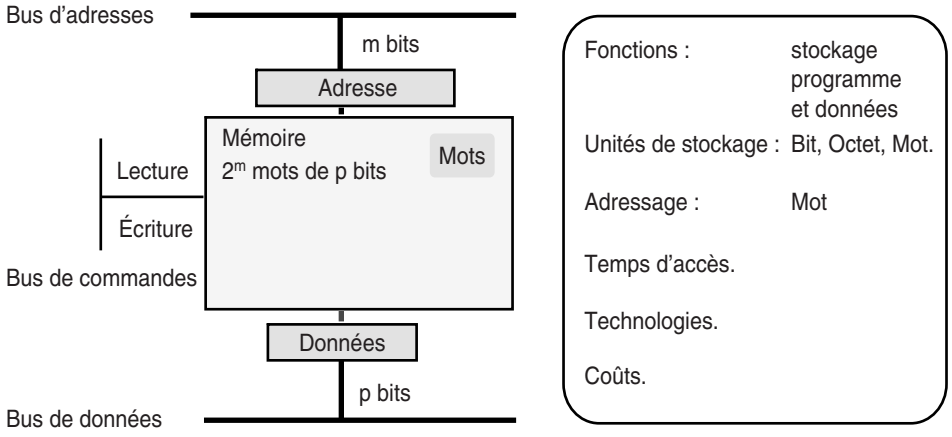


Figure 1.7 Bus de communication et mémoire centrale.

1.2.4 Le processeur central ou microprocesseur

Le *microprocesseur* (unité centrale) a pour objet d'exécuter les instructions machines placées en mémoire centrale. La figure 1.8 présente son architecture générale.

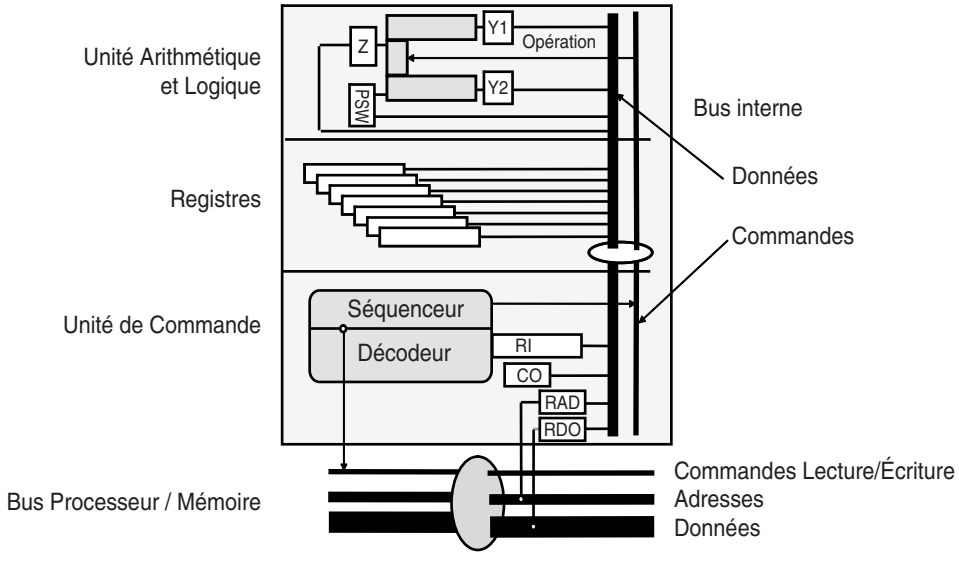


Figure 1.8 Le microprocesseur.

Il est constitué de quatre parties : l'unité arithmétique et logique (UAL), les registres, l'unité commande et le bus de communication interne permettant l'échange des données et des commandes entre les différentes parties du microprocesseur.

Les registres

Ce sont des zones de mémorisation de l'information internes au microprocesseur. Ils sont de faible capacité et de temps d'accès très faible. Leur nombre et leur taille sont variables en fonction du type de microprocesseur. Ils peuvent être de type adresse (ils contiennent alors une adresse de mot mémoire) ou données (ils contiennent alors le contenu d'un mot mémoire). Ils peuvent être spécifiques et avoir une fonction très précise (par exemple le registre pointeur de pile) ou généraux et servir essentiellement aux calculs intermédiaires, par exemple, de l'unité arithmétique et logique.

L'unité arithmétique et logique (UAL)

Ce module est chargé de l'exécution de tous les calculs que peut réaliser le microprocesseur. Cette unité est constituée de l'ensemble des circuits arithmétiques et logiques permettant au processeur d'effectuer les opérations élémentaires nécessaires à l'exécution des instructions machine. Elle inclut donc les circuits d'addition, de soustraction, de multiplication, de comparaison, etc. Dans ce module se trouvent également des registres dont l'objet est de contenir les données sur lesquelles vont porter les opérations à effectuer. Dans notre exemple, l'UAL possède deux registres d'entrée (E1 et E2) et un registre de sortie (S).

Pour faire une addition :

– la première donnée est placée dans E1 via le bus interne de données ;
– la seconde donnée est placée dans E2 via le bus interne de données ;
– la commande d'addition est délivrée au circuit d'addition via le bus interne de commandes ;
– le résultat est placé dans le registre S.

Sur notre machine on note également un registre particulier, le PSW (*Program Status Word*), qui joue un rôle fondamental de contrôle de l'exécution d'un programme et qui à tout instant donne des informations importantes sur l'état de notre microprocesseur. Par exemple puisque nous travaillons sur des mots de longueur finie la valeur d'un entier codé sur un mot ne peut dépasser la valeur maximale représentable sur ce mot. Lorsque nous faisons l'addition de deux entiers le résultat peut avoir une valeur qui n'est pas représentable sur un mot mémoire : il y a alors *dépassement de capacité*. Ce dépassement de capacité doit être signalé et noté pour ne pas perturber le fonctionnement de l'ordinateur. Ce type d'information est stocké dans le PSW.

L'unité de commande

Elle exécute les instructions machines et pour cela utilise les registres et l'UAL du microprocesseur. On y trouve deux registres pour la manipulation des instructions (le compteur ordinal CO, le registre d'instruction RI), le décodeur, le séquenceur et deux registres (le registre d'adresses RAD et le registre de données RDO) permettant

la communication avec les autres modules via le bus. Enfin, via le bus de commandes, elle commande la lecture et/ou l'écriture dans la mémoire centrale.

➤ Le compteur ordinal CO

C'est un registre d'adresses. À chaque instant il contient l'adresse de la prochaine instruction à exécuter. Lors de l'exécution d'une instruction il est prévu, au cours de cette exécution, la modification du contenu du CO. Ainsi en fin d'exécution de l'instruction courante le compteur ordinal pointe sur la prochaine instruction à exécuter et le programme machine peut continuer à se dérouler.

➤ Le registre d'instruction RI

C'est un registre de données. Il contient l'instruction à exécuter.

➤ Le décodeur

Il s'agit d'un ensemble de circuits dont la fonction est d'identifier l'instruction à exécuter qui se trouve dans le registre RI, puis d'indiquer au séquenceur la nature de cette instruction afin que ce dernier puisse déterminer la séquence des actions à réaliser.

➤ Le séquenceur

Il s'agit d'un ensemble de circuits permettant l'exécution effective de l'instruction placée dans le registre RI. Le séquenceur exécute, rythmé par l'horloge du microprocesseur, une séquence de *microcommandes* (*micro-instructions*) réalisant le travail associé à cette instruction machine. Pour son fonctionnement le séquenceur utilise les registres et l'UAL. Ainsi l'exécution effective d'une instruction machine se traduit par l'exécution d'une séquence de micro-instructions exécutables par les circuits de base du microprocesseur. Nous reviendrons plus en détail sur cet aspect des choses dans le chapitre 7, consacré à l'exécution des instructions machines.

➤ Le registre RAD

C'est un registre d'adresses. Il est connecté au bus d'adresses et permet la sélection d'un mot mémoire via le circuit de sélection. L'adresse contenue dans le registre RAD est placée sur le bus d'adresses et devient la valeur d'entrée du circuit de sélection de la mémoire centrale qui va à partir de cette entrée sélectionner le mot mémoire correspondant.

➤ Le registre RDO

C'est un registre de données. Il permet l'échange d'informations (contenu d'un mot mémoire) entre la mémoire centrale et le processeur (registre).

Ainsi lorsque le processeur doit exécuter une instruction il :

– place le contenu du registre CO dans le registre RAD via le bus d'adresses et le circuit de sélection ;
– déclenche une commande de lecture mémoire via le bus de commandes ;
– reçoit dans le registre de données RDO, via le bus de données, l'instruction ;

– place le contenu du registre de données RDO dans le registre instruction RI via le bus interne du microprocesseur.

Pour lire une donnée le processeur :

– place l'adresse de la donnée dans le registre d'adresses RAD ;
– déclenche une commande de lecture mémoire ;
– reçoit la donnée dans le registre de données RDO ;
– place le contenu de RDO dans un des registres du microprocesseur (registres généraux ou registres d'entrée de l'UAL).

On dit que pour transférer une information d'un module à l'autre le microprocesseur établit un *chemin de données* permettant l'échange d'informations. Par exemple pour acquérir une instruction depuis la mémoire centrale, le chemin de données est du type : CO, RAD, commande de lecture puis RDO, RI.

1.3 FONCTIONNEMENT : RELATION MICROPROCESSEUR / MÉMOIRE CENTRALE

L'objet de cette partie est de présenter succinctement comment s'exécute un programme machine sur le matériel que nous venons de définir. Pour être exécutable une instruction doit nécessairement être présente en mémoire centrale. La mémoire centrale contient donc des instructions et des données. De plus toutes les informations en mémoire sont codées sur un alphabet binaire. Les informations sont alors, quelle que soit leur nature, des suites de 0 et de 1. Il faut donc pouvoir différencier instructions et données afin que le registre instruction RI contienne bien des instructions et non des données (et réciproquement). Dans le cas contraire le décodeur ne pourrait interpréter la nature du travail à faire (RI ne contenant pas une instruction mais une

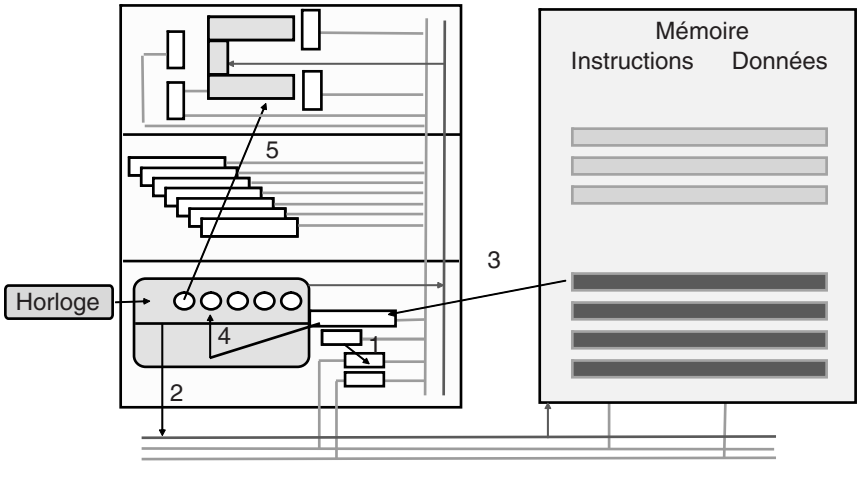


Figure 1.9 Exécution d'une instruction.

donnée). En général lors du placement du programme machine et des données dans la mémoire centrale les instructions et les données sont séparées et occupent des espaces mémoires différents (voir figure 1.9). À la fin du chargement du programme machine et des données en mémoire le compteur ordinal CO reçoit l'adresse de la première instruction du programme à exécuter.

L'exécution peut alors commencer. Le principe général d'exécution est illustré dans la figure 1.9.

Les différentes phases de l'exécution d'une instruction sont les suivantes :

1. le contenu du compteur ordinal CO est placé dans le registre d'adresses RAD : il y a sélection de l'instruction à exécuter ;
2. une commande de lecture de la mémoire centrale est déclenchée via le bus de commandes ;
3. l'instruction est transférée de la mémoire centrale vers le registre instruction RI via le bus de données et le registre de données RDO ;
4. le décodeur analyse l'instruction placée dans le registre instruction RI, reconnaît cette instruction et indique au séquenceur la nature de l'instruction ;
5. le séquenceur déclenche au rythme de l'horloge la séquence de micro-instructions nécessaires à la réalisation de l'instruction.

On peut résumer les étapes de l'exécution d'une instruction (chargement/décodage/exécution) par l'algorithme suivant :

```
début
charger l'instruction à exécuter depuis la mémoire dans le registre
➥ instruction;
modifier le compteur ordinal pour qu'il pointe sur la prochaine
➥ instruction à exécuter;
décoder l'instruction qui vient d'être chargée dans le registre
➥ d'instruction;
charger les données éventuelles dans les registres;
exécuter la séquence des micro-instructions permettant la réalisation
➥ de l'instruction;
fin
```

Les opérations charger/modifier réalisent le chargement de l'instruction dans le registre d'instruction RI. Cette phase est la *phase dite de FETCH*.

Enfin, l'exécution d'un programme machine peut être décrite par l'algorithme suivant :

```
début
exécuter la première instruction du programme
tant que ce n'est pas la dernière instruction
faire
    exécuter l'instruction;
fin faire
fin
```

1.4 UN EXEMPLE

Pour résumer ce que nous venons d'étudier, nous allons prendre un exemple de problème à résoudre avec un ordinateur. Nous définissons tout d'abord le problème et l'ordinateur cible c'est-à-dire son langage de programmation (langage machine). Puis nous construisons le programme machine exécutable par cet ordinateur. Enfin nous plaçons ce programme en mémoire centrale. Il peut alors être exécuté par le microprocesseur.

1.4.1 Le problème

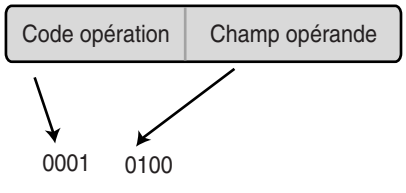
Notre problème consiste à réaliser l'addition de X qui vaut 4 avec Y qui vaut 1 et à placer le résultat dans Z. Nous souhaitons donc, à l'aide de notre ordinateur, réaliser l'opération : Z = X + Y avec X = 4 et Y = 1.

1.4.2 L'ordinateur

Cet ordinateur a la structure générale définie dans la figure 1.11. Les mots de la mémoire sont des octets (8 bits), tous les registres du microprocesseur ont une largeur de 8 bits, les instructions et les données entières sont codées sur un mot mémoire. Données : X est codé : 00000100_2 ; Y est codé : 00000001_2.

1.4.3 Le langage machine

Le code opération est codé sur 4 bits, le champ opérande sur 4 bits. Le champ opérande ne référence qu'une donnée. Ainsi pour faire l'addition de deux nombres un tel langage suppose que la première donnée est spécifiée dans l'instruction et la seconde occupe une adresse implicite. Dans de nombreuses machines cette adresse implicite est un registre appelé *registre Accumulateur* (noté A). L'addition porte alors sur la donnée spécifiée dans l'instruction et le contenu de A, le résultat étant placé dans A.



0001	0100	Charger le contenu du mot mémoire d'adresse 0100 dans le registre A
0010	0101	Additionner le contenu du registre A avec le contenu du mot mémoire d'adresse 0101 et placer le résultat dans le registre A
0011	0110	Placer le contenu du registre A dans le mot mémoire d'adresse 0110
0000		Stopper le programme

Figure 1.10 Le langage de la machine.

Les instructions du langage sont définies dans la figure 1.10. La figure 1.11 résume les différentes phases amenant à l'exécution du programme machine solution de notre problème sur cette machine :

– les données ont été chargées aux adresses 0100 (pour X), 0101 (pour Y), le résultat à l'adresse 0110 (pour Z) ;
– le programme machine est chargé à l'adresse 0111 ;
– le compteur ordinal est chargé avec l'adresse 0111.

À titre d'exercice vérifiez que l'exécution du programme machine résout bien notre problème.

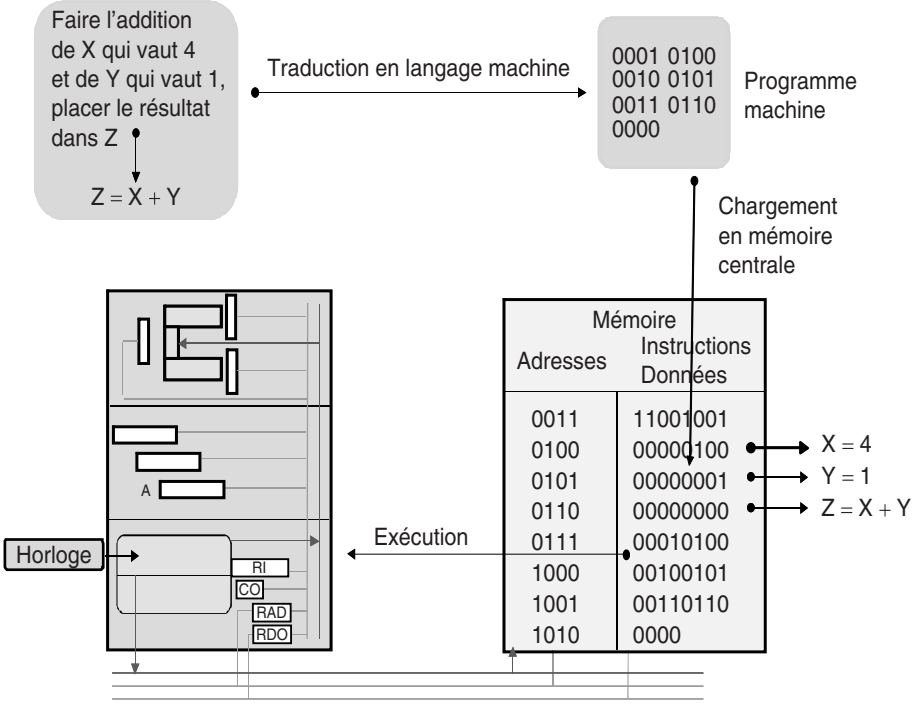


Figure 1.11 Le programme solution.

1.5 LES UNITÉS D'ÉCHANGES

Les *unités d'échanges* permettent la communication entre les modules du processeur et les périphériques. Cette fonction de communication est complexe et nous la détaillons dans le chapitre 9.

Une unité d'échange a une double nature :

– elle est en communication, via le bus interne du processeur, avec la mémoire centrale et le microprocesseur. Des instructions machines spécifiques permettent

les échanges entre mémoire centrale et unité d'échange. On trouve des instructions d'écriture permettant au microprocesseur de placer des informations dans l'unité d'échange et des instructions de lecture permettant au microprocesseur d'acquérir des informations à partir des unités d'échanges. De plus des d'outils de synchronisation permettant d'harmoniser les activités du processeur avec celles des périphériques qu'il pilote sont nécessaires : il ne faut, par exemple, envoyer des caractères vers une imprimante que si celle-ci est prête à imprimer et attendre dans le cas contraire. Dans nos ordinateurs les unités d'échanges ne communiquent pas directement avec le bus interne du processeur mais au travers de bus, dits *bus d'extension*. Il existe une assez grande variété de ces bus d'extension (ISA, USB, *FireWire*, *Fiberchanel*, PCI…) qui satisfont des fonctionnalités diverses et plus ou moins générales. Nous reviendrons plus en détail sur cette question dans le chapitre traitant des questions de communication ;

– elle est en communication avec les périphériques et à ce titre doit être capable de les piloter.

1.6 CONCLUSION

Dans ce chapitre nous avons fait le tour de toutes les étapes importantes nécessaires à la résolution d'un problème avec un ordinateur. La figure 1.12 résume ces différentes étapes.

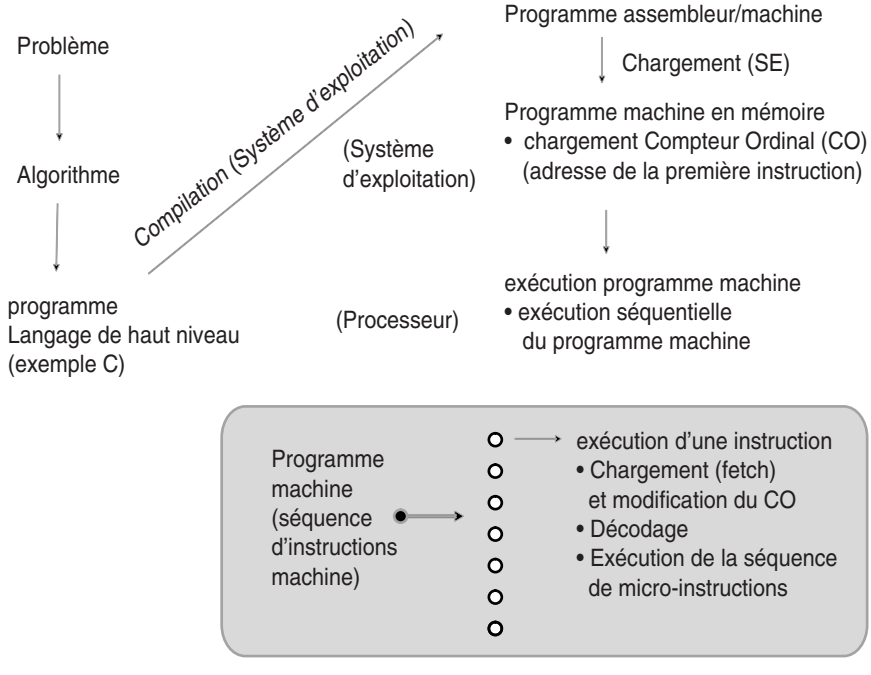


Figure 1.12 Principales étapes de la résolution d'un problème avec l'ordinateur.

Nous voyons qu'une des étapes consiste à exprimer l'algorithme avec un langage de haut niveau. En fait il existe plusieurs niveaux de langage de programmation :

- les *langages de haut niveau* (C, C++ , ADA, JAVA, COBOL, PASCAL, PYTHON…) ne sont pas directement exécutables par le processeur cible que nous utilisons : l'ordinateur. Il faut donc « traduire » les programmes exprimés dans ces langages afin d'obtenir le programme machine exécutable par le processeur cible. C'est l'objet de la *compilation* et/ou de l'*interprétation*. Les compilateurs (interpréteurs) sont des programmes (du Système d'Exploitation : SE) qui ont la connaissance de la syntaxe des langages de haut niveau et qui connaissent le langage machine de la machine cible. Ces langages sont nécessaires car on ne sait pas traduire directement le langage naturel en langage machine alors que l'on sait construire des traducteurs pour ces langages de haut niveau ;
- les *langages de bas niveaux* sont des langages directement exécutables par un microprocesseur. Ce sont les langages machines. Ils se présentent sous deux formes :
  - les langages machine codés binaires. Ce sont les seuls réellement exécutables par le processeur. Codés sous forme binaire ils sont difficiles à manipuler ;
  - les langages d'assemblage. Ce sont des langages machines symboliques au sens où les codes opération et les champs opérande d'une instruction sont exprimés à l'aide de symboles.

Par exemple dans la petite machine qui nous a servis d'exemple, 00010010 est une instruction machine demandant de charger le contenu de l'adresse mémoire 0010 dans le registre accumulateur A du microprocesseur. Il existe dans le langage d'assemblage de cette machine une instruction qui a la forme « `Load X` » où `Load` exprime à l'aide de symboles alphanumériques le code opération codé binaire 0001 et X exprime symboliquement l'adresse mémoire de la donnée X. Ce type de langage est beaucoup plus simple à utiliser que le langage machine codé binaire. Bien sûr un programme écrit en langage d'assemblage n'est pas directement exécutable par le microprocesseur. Cependant la traduction, langage d'assemblage, langage machine pur, est simple car à chaque instruction du langage machine correspond une instruction du langage d'assemblage. Le traducteur de tels langages s'appelle l'assembleur. Une programmation dans ce type de langage est possible mais de moins en moins fréquente. Elle se justifie encore lorsque l'on recherche une grande efficacité, par exemple, dans la programmation des jeux vidéos ou des interfaces graphiques.

Le traducteur (compilateur/interpréteur) place le résultat de sa traduction (programme machine) sur le disque magnétique. Le programme machine doit être chargé en mémoire centrale pour être exécuté par le microprocesseur. C'est un programme du système d'exploitation, *le chargeur*, qui réalise ce chargement en mémoire. Le chargeur connaît l'adresse de la première instruction du programme machine, il peut ainsi initialiser le compteur ordinal pour lancer l'exécution. À partir de ce moment les instructions, une à une, sont prises en charge par le microprocesseur qui les exécute.

La première partie de cet ouvrage est consacrée au passage du problème au programme machine ; elle présente en détail le fonctionnement du compilateur et des autres outils permettant de construire un programme machine. Elle approfondit