Introduction

Pour fabriquer un ordinateur,
il suffit d 'un fer à souder (ou presque…).

Dans ce chapitre, nous voyons de quoi sont faits les ordinateurs à une échelle plus proche de la nôtre et comment architecture et langages sont liés.

Nous décrivons la manière dont sont assemblés le processeur de calcul, l'organisation de la mémoire et les bus permettant la circulation des données.

Nous voyons comment programmer le processeur au moyen d'un langage machine simple et expliquons comment dérouler une séquence d'instructions.

Nous adjoignons enfin les périphériques pour obtenir un ordinateur.

Dans les années 1940, à l'Université de Pennsylvanie, John Von Neumann (1903-1957) a conçu, avec Presper Eckert et John Mauchly, deux des premiers ordinateurs : l'ENIAC, puis l'EDVAC. Ces ordinateurs étaient organisés selon l'architecture de Von Neumann, utilisée dans la quasi-totalité des ordinateurs conçus depuis : séparation du processeur et de la mémoire, reliés par un bus de communication. L'ENIAC pesait vingt-sept tonnes.

Introduction

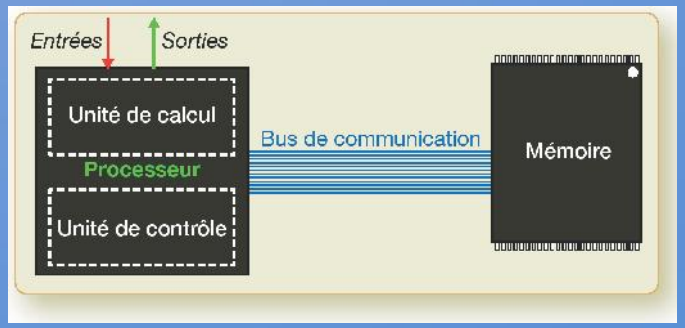
Après avoir décrit le fonctionnement d'un ordinateur à l'échelle du **transistor** puis de la **porte booléenne**, nous abordons, dans ce chapitre, une troisième échelle de description, qui est celle qui va nous permettre de véritablement en comprendre les principes d'organisation.

Un ordinateur est principalement composé de deux grands circuits : le **processeur** et la **mémoire**.

Ces deux circuits sont reliés entre eux par des fils qui constituent un ou plusieurs **bus de communication**, parmi lesquels un **bus de données** et un **bus d 'adresses**.

Le processeur est composé de deux unités. **L'unité de contrôle** lit en mémoire un programme et donne à **l'unité de calcul** la séquence des instructions à effectuer. Le processeur dispose par ailleurs de bus d 'entrées et de sorties permettant d'accéder aux autres parties de l'ordinateur, que l'on nomme les **périphériques**.

Cette organisation générale, **l'architecture de Von Neumann**, est étonnamment stable depuis les années quarante.



L'organisation d'un ordinateur

Introduction

La mémoire est composée de plusieurs milliards de circuits mémoires un bit.

Ces circuits sont organisés en agrégats de huit, seize, trente-deux, soixante-quatre bits, et parfois davantage, que l'on appelle des cases mémoires et qui peuvent donc mémoriser des mots de huit, seize, trente- deux, soixante-quatre bits, etc.

Le nombre de ces cases définit la taille de la mémoire de l'ordinateur.

Comme il faut distinguer ces cases les unes des autres, on donne à chacune un numéro : son **adresse**.

La mémoire contient les données sur lesquelles on calcule et le programme qui décrit le calcul effectué, donné sous la forme d'une séquence d'instructions.

Le processeur, de son côté, n'a qu'un très petit nombre de cases mémoires, que l'on appelle des **registres**. On peut imaginer, par exemple, qu'il ne contient que deux registres, appelés A et B.

Les registres peuvent contenir des données, mais aussi des adresses de cases mémoires.

Introduction

Taille de la mémoire

En général, on indique la taille de la mémoire en précisant le nombre d'octets, c'est-à-dire de mots de huit bits, qui peuvent être mémorisés. Ainsi, une mémoire de 4 gigaoctets (binaires), contient $4 \times 2^{30} \times 8 =$ 34 359 738 368 circuits mémoires un bit.
Si la mémoire est organisée en mots de soixante-quatre bits, ces circuits sont répartis en 536 870 912 cases permettant de mémoriser un mot chacune.

Les processeurs 32 et 64 bits

Lorsque l'on parle de processeurs 32 bits ou 64 bits, on fait référence à la taille des registres du processeur.

Trois instructions

Pour échanger des données avec la mémoire, le processeur utilise deux procédés qui permettent :

- l'un de transférer l'état d'un registre dans une case mémoire
- et l'autre de transférer l'état d'une case mémoire dans un registre.

Pour transférer le contenu du registre A dans la case mémoire d'adresse n, le processeur met les différents fils qui composent le **bus d'adresses** dans un état qui correspond à l'expression en base deux du nombre n et il met les différents fils qui composent **le bus de données** dans un état qui correspond au contenu du registre.

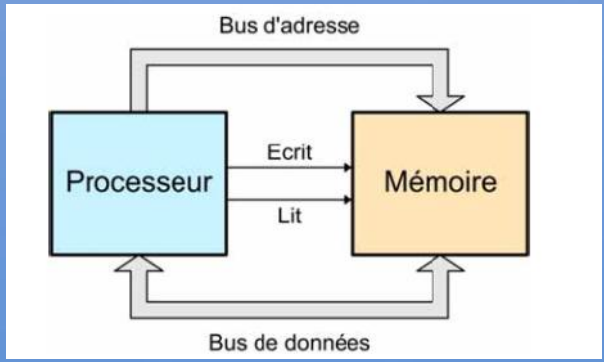
Au signal d'horloge, chaque case de la mémoire compare son propre numéro au numéro arrivé sur le bus d'adresse ; seule la case numéro n se reconnaît et elle met alors ses différentes entrées S (dans l'état 1, de manière à enregistrer le mot arrivant sur le bus de données.

Trois instructions

Un procédé symétrique permet au processeur de récupérer une valeur précédemment enregistrée : les informations circulent toujours du processeur vers la mémoire sur le bus d'adresses, mais elles circulent dans l'autre sens sur le bus de données : c'est la case n qui connecte sa sortie au bus de données et c'est le registre qui met ses entrées S à 1 de manière à enregistrer le mot qui arrive sur le bus de données.

Ces deux opérations s'appellent **le stockage (STA)** et le **chargement (LDA)** du contenu d'une case mémoire dans le registre *A* (**ST** pour ***STore***, **LD** pour ***LoaD***).

Il y a bien entendu des opérations similaires pour le registre *B* (STB et LDB).



L'organisation d'un ordinateur

Trois instructions

Une autre opération que peut exécuter le processeur est l'addition des contenus des registres A et B.

Le résultat de l'opération peut être stocké aussi bien dans le registre A (**ADD A**) que dans le registre B (**ADD B**). De même, **DEC A** décrémente la valeur contenue dans le registre A, c'est-à-dire soustrait 1 à la valeur contenue dans le registre A et stocke la valeur ainsi obtenue dans le registre A et DEC B réalise le même calcul sur la valeur contenue dans le registre B.

Trois instructions

```
LDA 7
LDB 8
ADD A
LDB 9
ADD A
LDB 10
ADD A
STA 11
```

Si, par exemple, le processeur effectue successivement les opérations ci-contre et si, dans l'état initial, la case 7 de la mémoire contient le nombre 42, la case 8 le nombre 68, la case 9 le nombre 47 et la case 10 le nombre 33, l'exécution des huit opérations a comme effet de :

- LDA 7 charger le contenu de la case 7, soit 42, dans le registre A,
- LDB 8 charger le contenu de la case 8, soit 68, dans le registre B,
- ADD A additionner les contenus des registres A et B et mettre le résultat, 110, dans le registre A,
- LDB 9 charger le contenu de la case 9, soit 47, dans le registre B,
- ADD A additionner les contenus des registres A et B et mettre le résultat, 157, dans le registre A,
- LDB 10 charger le contenu de la case 9, soit 33, dans le registre B,
- ADD A additionner les contenus des registres A et B et mettre le résultat, 190, dans le registre A,
- STA 11 stocker le contenu du registre A, soit 190, dans la case 11.

Au bout du compte, cette séquence d'opérations additionne les quatre nombres stockés dans les cases 7, 8, 9 et 10 de la mémoire et stocke le résultat dans la case 11.

Le langage machine

Un ordinateur doit être capable d'exécuter un programme.

Il faut donc un moyen d'indiquer au processeur la séquence des instructions qu'il doit exécuter ; par exemple, la séquence LDA 7, LDB 8, ADD A, LDB 9, ADD A, LDB 10, ADD A, STA 11.

Dans les premières machines, des cartes perforées ou un ruban perforé situé à l'extérieur de la machine indiquaient les opérations à effectuer, comme les cartes d'un orgue de Barbarie indiquent les notes à jouer l'une après l'autre.

Puis cette idée a été abandonnée au profit d'une autre : celle d'enregistrer le programme dans la mémoire avec les données.

Le langage machine

Ainsi, on peut exprimer le programme précédent en binaire en décidant par exemple que A s'écrit 0, B s'écrit 1 et les instructions LDA, LDB, STA, STB, ADD et DEC s'écrivent respectivement 0, 1, 2, 3, 4 et 5. Le programme de notre exemple s'écrit alors 0, 7, 1, 8, 4, 0, 1, 9, 4, 0, 1, 10, 4, 0, 2, 11, ce qui commence à devenir assez difficile à lire, même s'il est facile de passer d'une représentation à l'autre.

On peut ensuite stocker ce programme dans la mémoire, en commençant, par exemple, à la case 100 :

adresse	..	..	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	..	..
valeur	..	..	0	7	1	8	4	0	1	9	4	0	1	10	4	0	2	11	..	..

Le langage machine

Il suffit maintenant d'ajouter au processeur un nouveau registre qui débute à 100, le **compteur de programme ou PC (program counter)**, et à chaque étape, le processeur :

- charge le contenu des cases mémoires d'adresses PC et PC + 1,
- décode le premier de ces nombres en une instruction (0 devient LDA, 1 LDB, etc.),
- exécute l'instruction en question,
- et ajoute 2 au registre PC.

adresse	..	..	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	..	..
valeur	..	..	0	7	1	8	4	0	1	9	4	0	1	10	4	0	2	11	..	..

Le langage machine

Enregistrer les programmes en mémoire permet de faire très simplement des **boucles et des tests**.

On ajoute aux instructions précédentes une instruction **JMP (jump)** telle que **JMP n** charge simplement le nombre n, ou plutôt le nombre n - 2 qui sera augmenté de 2 immédiatement après l'exécution du JMP, dans le registre PC pour détourner le programme de sa route et le forcer à continuer son exécution à l'adresse n.

De même, l'instruction **JMPZ (jump if zero)**, qui effectue un saut si le contenu du registre A est 0, permet de faire des tests.

On ajoute enfin l'instruction **END**, qui termine le programme.

En langage machine, on suppose que JMP, JMPZ et END s'écrivent respectivement 6, 7 et 8 et que END prend 0 comme argument puisqu'il en faut un. Mais cet argument n'est pas utilisé.

Le langage machine

SAVOIR-FAIRE Savoir dérouler l'exécution d'une séquence d'instructions

Le principe est de suivre, instruction par instruction, l'évolution du programme en observant les effets sur les valeurs contenues dans les registres, y compris le compteur de programme PC, et les valeurs contenues dans la mémoire, un peu comme on le ferait pour l'état de l'exécution d'un programme écrit en Java.

Exercice 1

Écrire une séquence d'instructions qui multiplie par 5 le nombre contenu dans la case mémoire d'adresse 10 et stocke le résultat dans la case mémoire d'adresse 11.

Le langage machine

Exercice 2

Écrire un programme qui lit deux valeurs *x* et *y* contenues respectivement dans les cases mémoires 11 et 12, calcule la différence *y* - *x* et stocke le résultat à l'adresse 13.

On suppose que ces deux valeurs sont des nombres entiers positifs.

Compléter ce programme pour qu'il stocke la valeur 0 à l'adresse 15 si *x* est égal à *y*, ou la valeur *x* sinon.

Les périphériques

Outre un processeur, une mémoire, une horloge et des bus reliant le processeur à la mémoire, un ordinateur est également constitué de périphériques : écrans, claviers, souris, disques, haut-parleurs, imprimantes, scanners, cartes réseau, clés de mémoire flash, etc. qui permettent au processeur d'échanger des informations avec l'extérieur : des êtres humains, à travers par exemple l'écran et le clavier, d'autres ordinateurs, à travers la carte réseau, et des outils de stockage, par exemple des disques.

On peut grossièrement classer les périphériques en :

- **périphériques d 'entrée**, qui permettent au processeur de recevoir des informations de l'extérieur,
- et **périphériques de sortie**, qui lui permettent d'émettre des informations vers l'extérieur.

Les périphériques

Toutefois, beaucoup de périphériques sont à la fois des périphériques d'entrée et de sortie. Ainsi, un écran est a priori un périphérique de sortie, mais un écran tactile est aussi un périphérique d'entrée.

Pour échanger des informations avec les périphériques, le processeur procède d'une manière très similaire à celle qu'il utilise pour échanger des informations avec la mémoire.
Par exemple, on peut donner une adresse à chaque pixel d'un écran noir et blanc, exactement comme on donne une adresse à chaque case de la mémoire ; stocker une valeur comprise entre 0 et 255 à cette adresse, avec l'instruction STA par exemple, aura pour effet d'allumer ce pixel à l'écran avec le niveau de gris correspondant.

Les périphériques

De même l'instruction LDA permet de recevoir des informations d'un périphérique de sortie, par exemple la position de la souris. Selon les processeurs, ce sont les instructions STA et LDA elles-mêmes qui sont utilisées, ou des instructions voisines, spécialisées pour la communication avec les périphériques.

Le système d'exploitation

La description des ordinateurs que l'on a donnée dans ce chapitre diffère quelque peu de l'expérience pratique qu'ont tous ceux qui ont déjà utilisé un ordinateur.

Tout d'abord, on s'aperçoit que si on bouge la souris, le curseur de souris bouge sur l'écran, il semble donc y avoir un programme qui interroge en permanence la souris pour connaître sa position et dessine un curseur de souris à l'endroit correspondant de l'écran.

De même, il est possible d'utiliser simultanément plusieurs programmes, alors que dans la description que nous avons donnée, le processeur n'en exécute qu'un seul à la fois.

Le système d'exploitation

Cela est dû au fait que dès qu'on allume un ordinateur, un programme spécial est lancé : le **système d 'exploitation**.

Ce programme, souvent gigantesque, a plusieurs fonctions :

- Il permet l'exécution simultanée de plusieurs programmes, selon le système du temps partagé : le système d'exploitation exécute chacun des programmes à tour de rôle et pendant une courte durée, garantissant à chacun que ses données ne seront pas modifiées par les autres.

Le système d'exploitation

- Il gère les périphériques ; ainsi, pour imprimer un caractère sur l'écran, il n'est pas nécessaire d'allumer chaque pixel l'un après l'autre, mais on peut demander au système d'exploitation d'afficher un « A » et celui-ci traduira cette instruction en une suite d'instructions qui afficheront un « A » pixel par pixel.

La partie du système d'exploitation qui gère un périphérique s'appelle un ***pilote***.

- En particulier, il gère le disque, son découpage en fichiers, l'attribution d'un nom à chaque fichier et leur organisation arborescente.

• Il gère aussi l'écran, c'est-à-dire son découpage en fenêtres, l'ouverture et la fermeture des fenêtres.

• Dans certains systèmes utilisés par plusieurs personnes, il gère l'authentification de chaque utilisateur et les droits, en particulier de lecture et d'écriture des fichiers, associés à chacun d'eux.

Le système d'exploitation

ALLER PLUS LOIN : Plusieurs systèmes d'exploitation

Il existe plusieurs systèmes d'exploitation : **Unix**, **Linux** ou **GNU/Linux**, **Windows**, **Mac OS**, etc.

Toutefois, le développement d'un système d'exploitation est une tâche si coûteuse en temps, qu'il n'existe guère plus d'une dizaine de systèmes d'exploitation réellement utilisés.

Le système d'exploitation

ALLER PLUS LOIN : Les ordinateurs parallèles

Une manière de fabriquer des ordinateurs plus rapides est de mettre dans un ordinateur plusieurs processeurs qui calculent *en parallèle*, c'est-à-dire en même temps. Les ordinateurs qui ont plusieurs processeurs parallèles sont appelés ***ordinateurs parallèles*** ou ***multicœurs***.

Certains algorithmes sont très faciles à paralléliser : par exemple pour augmenter la luminosité d'une image en niveaux de gris, il faut ajouter une constante à chaque pixel. Chaque pixel peut être traité indépendamment des autres et utiliser deux processeurs au lieu d'un divise le temps de calcul par deux.

Le système d'exploitation

ALLER PLUS LOIN : Les ordinateurs parallèles

Toutefois, d'autres algorithmes sont plus difficiles à paralléliser.

La programmation des processeurs parallèles est plus difficile car, en plus d'écrire des instructions, il faut pré- voir sur quel processeur elles vont s'exécuter.

Les processeurs peuvent se partager une mémoire ou plusieurs. Si deux processeurs partagent la même mémoire et doivent se communiquer des données, il faut s'assurer que le premier ait bien fini de les calculer et de les stocker en mémoire avant que le second ne les lise.

On dit que les deux processeurs doivent se ***synchroniser***. Ils peuvent aussi avoir des mémoires différentes et communiquer par un bus.

Le système d'exploitation

ALLER PLUS LOIN : Les ordinateurs parallèles

Les processeurs peuvent fonctionner sur la même horloge ; on dit alors qu'ils sont synchrones.

Ou bien chaque processeur peut avoir sa propre horloge ; on dit qu'ils sont asynchrones.

Le parallélisme existe aussi à l'intérieur des processeurs, entre les instructions du langage machine. Cette forme de parallélisme s'appelle le parallélisme d'instructions.