7.1 Modèle d'architecture

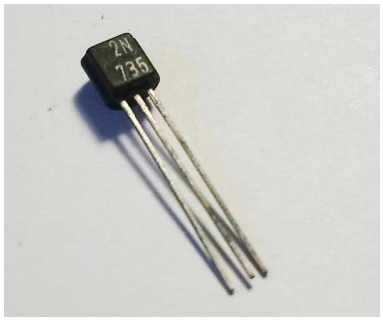
L'architecture des ordinateurs constitue un vaste sujet en constante évolution. En 20 ans, la vitesse des processeurs a été multipliée par 1 000 tandis que le coût par unité de performance a été divisé par 100 000.
L'objet de ce chapitre n'est pas d'étudier une architecture particulière, elle serait rapidement obsolète, mais plutôt un modèle de fonctionnement de l'ordinateur.

a. Représentation physique de l'information

L'information est représentée au sein des ordinateurs sous forme de différents états de la matière.

- Orientation nord ou sud dans un maériel magnétique ;
- « trou » ou « pas trou » (Cd ou DVD) ;
- lumière ou absence de lumière dans un dispositif optique ;
- courant électrique ou pas.

Quasiment tous les ordinateurs sont constitués de cirtuits électroniques. Ces circuits électroniques sont réalisés au moyen de transitors.

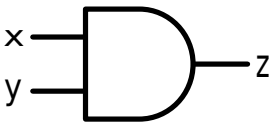


Au sein de ces transitors « tout ou rien », soit le courant passe, soit le courant ne passe pas. En combinant plusieurs transitors, on peut effectuer des calculs et des tâches complexes.

b. Les circuits logiques

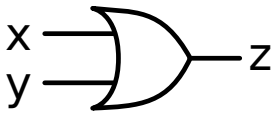
Un circuit logique est une entité qui accepte en entrée des valeurs booléennes, « 0 » ou « 1 » et réalise une opération booléenne en sortie.
Il est à noter qu'il existe deux types de circuits logiques : les circuits combinatoires et les circuits séquentiels. La différence entre ces deux types de circuit logique est que pour les combinatoires la sortie ne dépend que des valeurs en entrée, alors que pour les séquentiels, la valeur de sortie peut dépendre aussi du temps et de l'état passé du circuit logique. Dans ce chapitre on ne s'intéressera qu'aux circuits combinatoires et plus particulièrement aux plus simples d'entre eux qu'on appelle « portes logiques ».

Porte-et : la porte-et peut avoir un nombre d'entrées quelconque. Sa sortie vaut 1 si et seulement si toutes ses entrées valent 1.



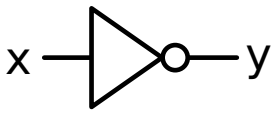
Entrées		Sortie
x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

Porte-ou : la porte-ou peut avoir un nombre d'entrées quelconque. Sa sortie vaut 1 si et seulement si au moins une de ses entrées vaut 1.



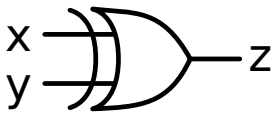
Entrées		Sortie
x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

Porte-non : la porte-non ne peut avoir qu'une seule entrée. Sa sortie vaut 1 si et seulement si son entrée vaut 0.



Entrée	Sortie
x	y
0	1
1	0

Porte-xor ou-exclusif : la porte-xor peut avoir un nombre d'entrées quelconque. Sa sortie vaut 1 si et seulement exactement une entrée vaut 1.

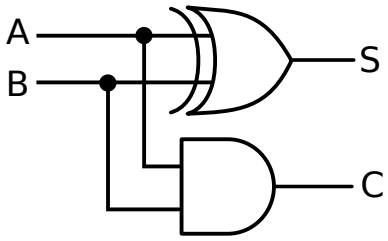


Entrées		Sortie
x	y	z
0	0	0
0	1	1
1	0	1
1	1	0

Un circuit combinatoire est une combinaison de portes logiques de base.

▶ Exercice 1 Le demi-additionneur

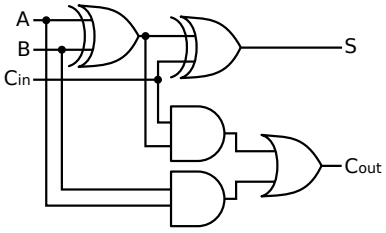
On considère le circuit logique ci-dessous constitué d'une porte ou exclusif « xor » et d'une porte ou. Compléter la table de vérité de ce circuit.



Entrées		Sorties	
A	B	S	C
0	0		
0	1		
1	0		
1	1		

Ce circuit calcule le bit de somme des bits A et B en S et une éventuelle retenue en C (carry en anglais).
Pour concevoir un circuit logique capable d'additionner des nombres binaires, il faut pouvoir tenir compte de la propagation d'une eventuelle retenue. Il s'agit d'additionner A et B et une retenue C_{in} (valant 0 ou 1). On obtiendra alors un résultat en sortie S ainsi qu'une retenue C_{out} (valant 0 ou 1).

Voici le circuit combinatoire dit « additionneur » :



Entrées			Sorties	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

En enchaînant n additionneurs, on peut calculer la somme de nombres binaires de longueur n. On additionne bit à bit et on propage la retenue de la droite vers la gauche comme lorsqu'on pose une addition.

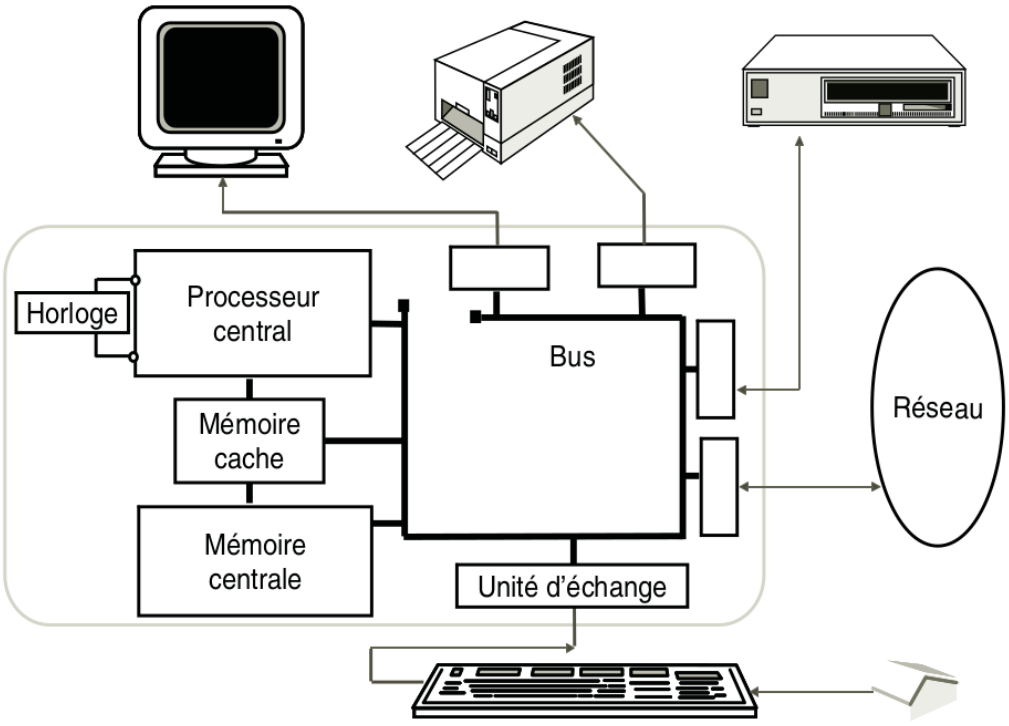
$$\begin{array}{cccccc} & 1 & 0 & 1 & 0 \\ + & 0 & 0 & 1 & 1 \\ \hline & 1 & 1 & 0 & 1 \end{array}$$

Ces circuits combinatoires peuvent être construits physiquement à l'aide de transistors, ce qui permet de concevoir tout type de circuits électroniques capables de traiter n'importe quelles fonctions booléennes. Les circuits logiques sont les briques élémentaires permettant de construire un édifice complexe qu'est l'ordinateur.

c. Structure d'un ordinateur

Le modèle d'architecture servant de support est un ordinateur dit de type Von Neumann qui caractérise la quasi-totalité des ordinateurs actuels. Il est composé des éléments suivants :

- *une mémoire centrale* pour le stockage des informations (programmes et données) ;
- *une unité de traitement* ou processeur central CPU pour le traitement des informations stockées en mémoire centrale ;
- des *unités de contrôle des périphériques et périphériques* ;
- *un bus de communication* entre ces différents modules.



La mémoire centrale

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

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 (1 ou 0).

Le microprocesseur (unité centrale) a pour objet d'exécuter les instructions machines placées en mémoire centrale. Il est lui-même constitué d'une unité arithmétique et logique (UAL), de registres (zones de mémorisation internes du processeur) et d'une unité de commandes (qui execute les instructions).

Par souci de simplicité et de généralité, nous n'entrons pas dans les détails de fonctionnement, il est cependant important de s'intéresser à l'exécution d'une séquence d'instructions en langage machine. Le processeur execute uniquement des programmes en langage machine (suite de 0 et de 1), il nous est impossible de programmer directement en langage machine.

On utilisera le langage assembleur.

7.2 Langage assembleur

Le langage assembleur est le langage de plus bas niveau qui représente le langage machine sous une forme lisible par un humain. Il est spécifique à chaque type de processeur.

MIPS Microprocessor without interlocked pipeline stages est une architecture de microprocesseur de type jeu d'instruction réduit RISC -Reduced Instruction-Set Computer. Elle fut développée par la compagnie MIPS Computer Systems Inc. Les processeurs fabriqués selon cette architecture sont surtout utilisés dans les systèmes SGI http ://www.sgi.fr/index.shtml, dans les systèmes embarqués, comme les ordinateurs de poche, les routeurs Cisco et les consoles de jeux vidéo (Nintendo 64 et Sony PlayStation, PlayStation 2 et PSP). Vers la fin des années 1990, les premières implémentations de l'architecture MIPS étaient de 32 bits (registres et chemins de données), puis passèrent à 64 bits. Le jeu d'instructions est basique et relativement simple à manipuler. Pour illustrer les concepts développés ici, nous utiliserons un simulateur multi-plateforme Java nommé **Mars** (Mips architecture and runtine simulator) en téléchargement sur `http://courses.missouristate.edu/KenVollmar/MARS/`

a. Syntaxe des instructions MIPS

Une instruction en langage assembleur MIPS peut être de trois différentes formes :

- *NomInstruction* Opérande 1, Opérande 2, Opérande 3
- *NomInstruction* Opérande 1
- *NomInstruction*

NomInstruction représente le nom symbolique de l'instruction.
tandis que Opérande représente une donnée ou le résultat de l'instruction.
Les opérandes utilisent les modes d'adressage pour savoir comment **obtenir la valeur de la donnée recherchée**, ou pour savoir où ranger le résultat de l'instruction.

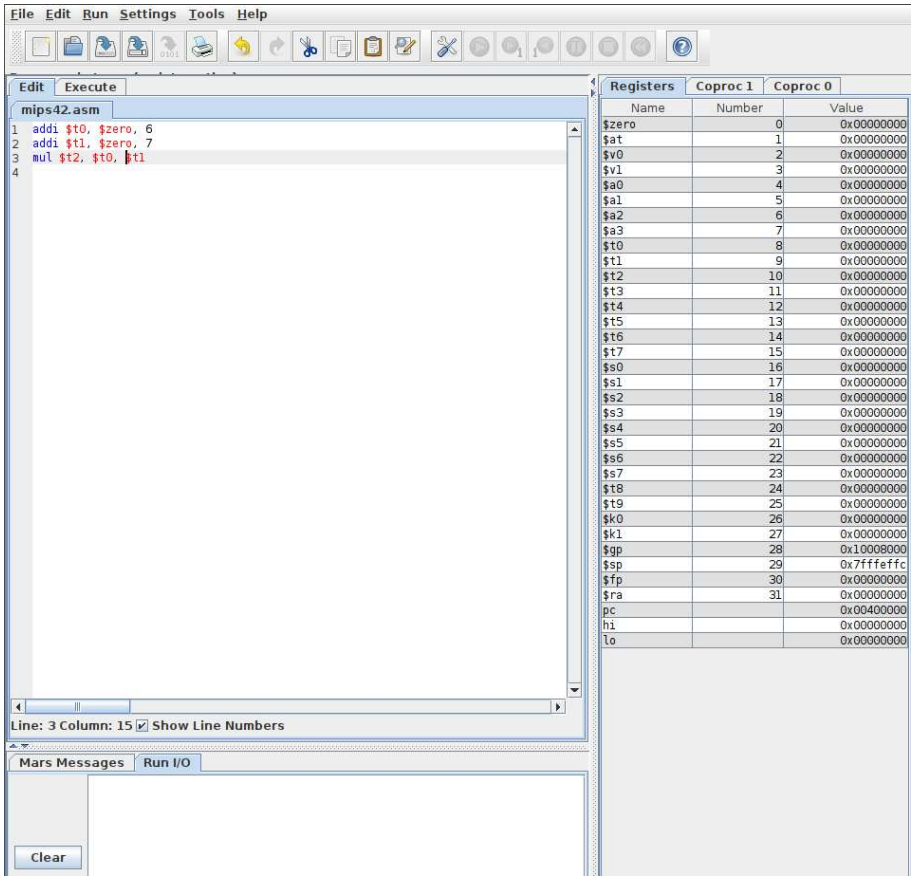
Exemple :

```
add $t0 , $t1 , $t2
```

Additionne les valeurs stockées aux adresses mémoires temporaires t1 et t2 et stocke le résulat en mémoire temporaire en t0. Cette commande va être assemblée pour constituer un nombre binaire de 32 bits. Pour cet exemple cela donne en hexadécimal `0x01095020` soit `0000 0001 0000 0009 0005 0000 0002 0000`
Téléchargez l'archive java `Mars4_5.jar` le simulateur *MIPS* à l'adresse
`https://courses.missouristate.edu/KenVollmar/mars/download.htm`

▶ Exercice 2

Lancer le simulateur MIPS (`java -jar Mars4_5.jar`)



Puis, dans la fenêtre edit, écrire ce premier programme en assembleur MIPS :

```
addi $t0, $zero, 6
addi $t1, $zero, 7
mul $t2, $t0, $t1
```

Assembler (*Run* > *Assemble*) ce programme puis noter les trois nombres binaires de 32 bits correspondant à chacune des trois lignes de ce programme en langage machine. On trouve ces nombres dans la colonne `code` de l'onglet `execute` Executer pas à pas le programme et bien vérifier que la valeur 42 en hexadecimal est contenue dans le registre en `$t2`.

▶ Exercice 3 Suite de Fibonacci

Voici un exemple de programme plus complexe en assembleur, `https://pastebin.com/jCbHEWaK`, télécharger et l'enregistrer avec l'extension `.asm` puis l'ouvrir avec `Mars`. L'objectif est de calculer les élements de la suite de Fibonacci qui suivent ce schéma :
1 1 2 3 5 8 13 21 34 ... chaque nombre étant la somme des deux précédents
Exécuter le programme pas à pas et observer l'évolution des registres mémoires ainsi que l'exécution de chaque instruction.

7.3 Évolutions architecturales

a. Historique

1946 : Ordinateur ENIAC Architecture à base de lampes et tubes à vide : 30 tonnes, 170 m2 au sol, 5000 additions par seconde

1947 : Invention du transistor

1958 : Invention du circuit intégré sur silicium, Multiples transistors agencés sur le même substrat

1971 : Processeur Intel 4004
2300 transistors dans un unique circuit intégré Fréquence de 740 kHz,

2011 : Processeur Intel Core i7 2600K Plus de 1,4 milliards de transistors Fréquence de 3,4 GHz 4 coeurs.

b. Évolution

Les calculs des ordinateurs sont cadencés par une horloge. Plus la fréquence de l'horloge est élevée, et plus l'ordinateur pourra effectuer d'opérations par seconde (s'il n'est pas ralenti par autre chose...). On mesure cette fréquence en Hertz. Ce qui importe aux usagers, c'est le nombre d'opérations (plus généralement, « d'instructions ») qu'un ordinateur est capable d'effectuer par seconde

On pense souvent que la puissance d'un ordinateur dépend de sa fréquence de fonctionnement mais c'est loin d'être toujours vrai.

c. Les limites

Plus on a de transistors par unité de surface, plus on a d'énergie à évacuer. La dissipation thermique évolue de façon proportionnelle au carré de la tension éléectique. La tension de fonctionnement des circuits a été abaissée de 5V pour les premières générations à 0,9V maintenant. Il n'est plus vraiment possible de la diminuer avec les technologies actuelles.

À surface constante, le nombre de transistors double tous les 2 ans. La diminution continuelle de la taille de gravage des transistors et circuits sur les puces de silicium atteint bientôt les limites atomiques.
Il ne sera bientôt plus possible d'intégrer plus, cependant on veut toujours plus de puissance de calcul.
Les concepteurs et constructeurs de puces ont décidé de changer de stratégie et d'interger plusieurs coeurs à leurs microprocesseurs.
On a vu apparaître des processeurs bi-coeurs, quadri-coeurs, octo-coeurs, jusqu'à 128 coeurs. Cependant pour tirer profit des architechtures multicoeurs, il faut les programmer efficacement. C'est une tâche délicate comme celle de faire jouer ensemble un orchestre symphonique.