Architecture des ordinateurs

Architecture de Von Neumann

Ordinateur = 3 aptitudes : calculer, mémoriser, communiquer.

Évolution : être humain (cerveau + 5 sens + l'écrit) → mécanique → électronique → optique.

1821 **Charles Babbage** : « machine à différence » tables de logarithmes ; projet de calculateur avec **programme** enregistré.

1939 **Alan Turing** : la « Bombe ».

1943 **John von Neumann** : modèle « anthropomorphique » constitué d'un **processeur**, une **mémoire** et de **périphériques**.

Introduit le modèle de l'exécution séquentielle ainsi que la notion de **programme** dans une **mémoire** : données & instructions.

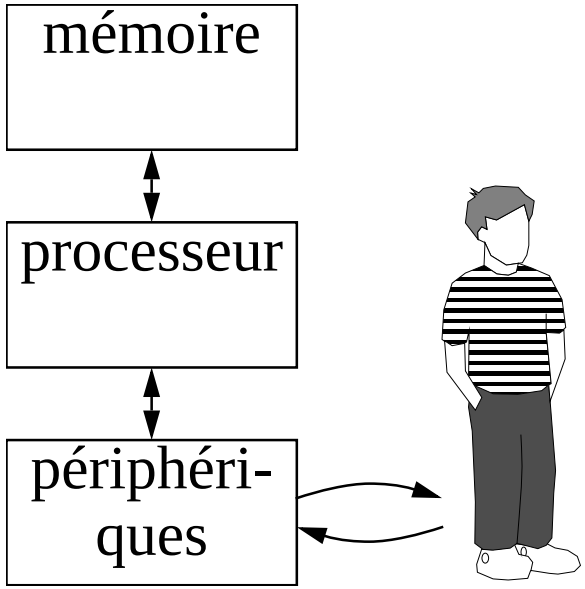
Analogique ≠ numérique. Décimal ≠ binaire.

Programme = séquence d'opérations complexes qui doivent être enregistrées.

`http://www.turing.org.uk/`

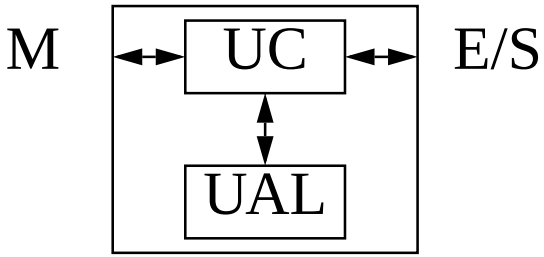
Processeur

- lit séquentiellement des instructions de la mémoire et les exécute ;
- lit ou écrit des données dans cette mémoire ;
- lit ou écrit des données dans des périphériques permettant de communiquer avec le monde extérieur.



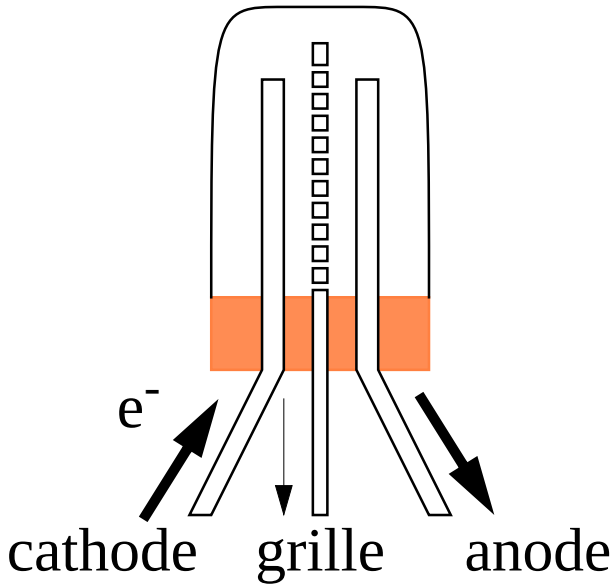
processeur

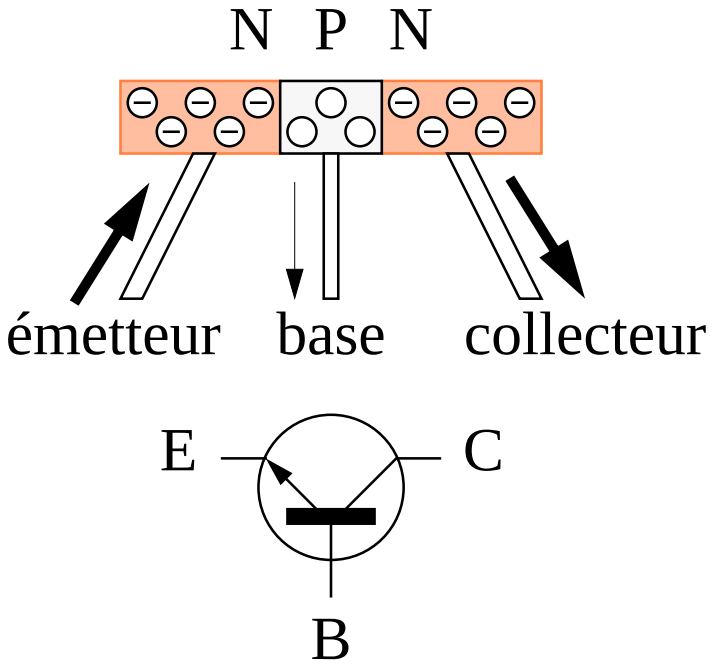
=



Transistor

Triode : vanne électronique. Le courant passe « ***ou bien*** » ne passe pas.





Semi-conducteur → ***transistor*** : 1947 Bell Laboratories ; P, N.

Un semi-conducteur est un matériau cristallisé où « quelques » électrons sont mobiles. En y introduisant des impuretés, dans une opération appelée dopage, on accroît sa conductivité. On distingue les semi-conducteurs de type N où il y a un excédent d'électrons mobiles, et les semi-conducteurs de type P où il y a au contraire un déficit en électrons mobiles.

E. 56 : Un semi-conducteur est-il électriquement chargé ?

Lorsque l'on applique une tension entre 2 types de semi-conducteurs, les électrons vont tout naturellement accepter d'aller du type N vers le type P. Au bout d'un certain temps, le semi-conducteur de type P chargé négativement va finir par repousser ces électrons créant une tension annulant exactement la tension initiale appliquée. Si on évacue ces électrons du semi-conducteur de type P, alors le courant va passer. Le conducteur qui évacue ces électrons dans le montage du transistor est la base.

Les semi-conducteurs utilisés ont successivement été le Germanium (Ge), le Silicium (Si), et l'Arséniure de Gallium (GaAs).

Porte

À partir des années 1960 des techniques permettent de miniaturiser les transistors : croissance d'oxides métalliques sur des substrats semi-conducteurs.

MOS = Metal Oxide Semiconductors.

Ces transistors que l'on « élève » avec d'énormes précautions permettent de construire les briques élémentaires qui sont à la base des processeurs : les **portes**.

Ces portes réalisent les opérations de logique de base sur des informations qui sont binaires (le courant passe « ou bien » ne passe pas).

En groupant ces portes on construit des « circuits intégrés » : LSI, VLSI.

Une bactérie compromet la gravure d'un VLSI où les détails sont en dessous du µm (10^{-6}m).

La vitesse de commutation d'un élément électronique est le temps qu'il met pour passer de l'état passant à non-passant. Les vitesses de commutation atteintes aujourd'hui sont de l'ordre de quelques 10ps (10^{-11}s). Lorsque l'on cherche à accroître celle-ci, les composants électroniques chauffent plus. On y remédie en abaissant leur tension de fonctionnement et en équipant les circuits de radiateurs.

Il existe plusieurs familles de mise en œuvre des semi-conducteurs dans des circuits intégrés :

TTL = Transistor Transistor Logic, la plus ancienne, à base de transistors & de diodes ;

ECL = Emitter Coupled Logic, plus rapide, mais plus consommatrice ;

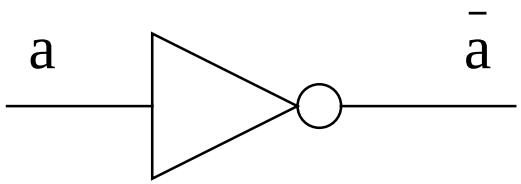
MOS = Metal Oxyde Semiconductor, faible consommation, immunité au bruit, mais très haute sensibilité à l'électricité statique ;

CMOS = Complementary MOS, la plus utilisée de nos jours.

Algèbre de Boole

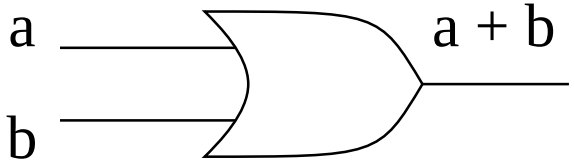
George Boole étudie les opérateurs logiques & les lois qui permettent de les composer.

Non : $a \rightarrow \bar{a}$



a	ā
0	1
1	0

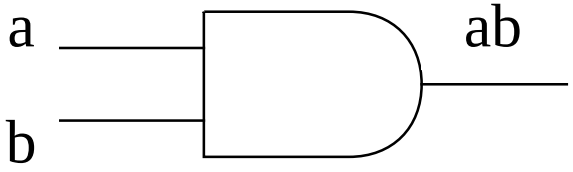
Ou : $(a, b) \rightarrow a + b$



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

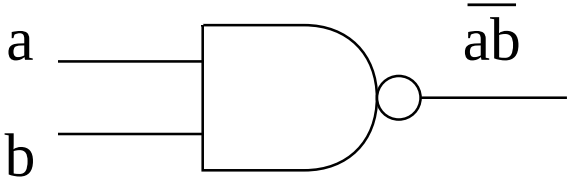
Algèbre de Boole

Et : $(a, b) \rightarrow ab$



a	b	ab
0	0	0
0	1	0
1	0	0
1	1	1

NAND : $(a, b) \rightarrow \overline{ab}$



a	b	$\overline{ab}$
0	0	1
0	1	1
1	0	1
1	1	0

Toutes les opérations arithmétiques & logiques se ramènent à des assemblages de portes **NAND**.

La table de vérité de toute fonction logique la définit complètement.

E. 57 : Faire la table de verité de $a \Rightarrow b$.

E. 58 : Comment réaliser $\bar{a}$ au moyen de la fonction NAND ?

Algèbre de Boole

Propriétés :

- commutativité ;
- associativité ;
- distributivité ;
- $a1 = a$; $a+1 = 1$;
- $a0 = 0$; $a+0 = a$.

Règles de simplification :

- $a + \bar{a} = 1$; $a\bar{a} = 0$;
- $a(a + b) = a$; $a + ab = a$.

Lois de De Morgan :

- $\overline{a + b} = \bar{a}\bar{b}$; $\overline{ab} = \bar{a} + \bar{b}$.

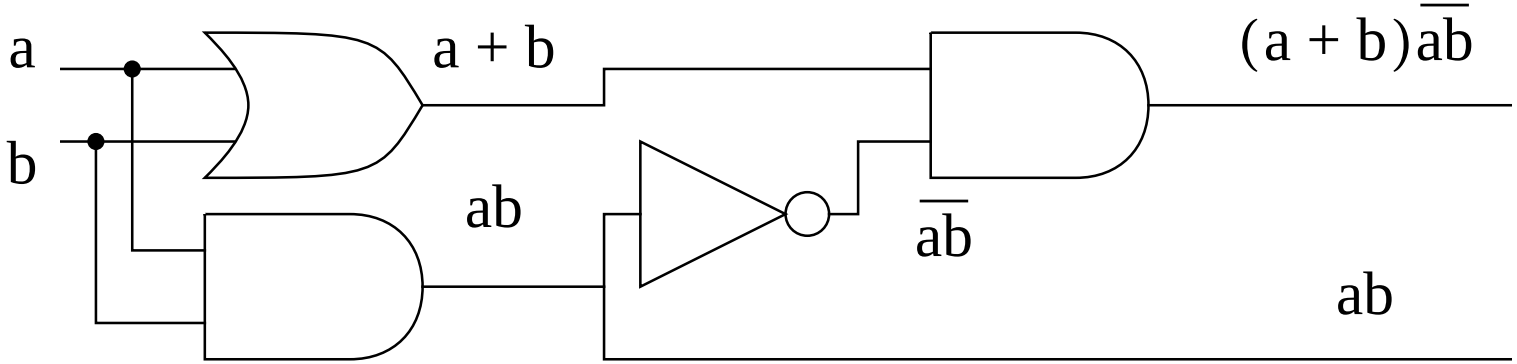
Toutes les propriétés fondamentales de l'algèbre de Boole se démontrent en construisant leur table de vérité.

E. 59 : Construire les tables de vérité des règles de simplification.

Pour réaliser toute fonction logique, on commence par en définir la table de vérité, puis au moyen des règles de simplifications, on la ramène à la forme la plus simple possible. À partir de cette version simplifiée, on peut passer à la mise en œuvre sous forme de portes.

Semi-additionneur

Add : $(a, b) \rightarrow ((a + b)\overline{ab}, ab)$



a	b	a + b	ab	$\overline{ab}$	$(a + b)\overline{ab}$
0	0	0	0	1	
0	1	1	0	1	
1	0	1	0	1	
1	1	1	1	0	

E. 60 : Compléter la table de vérité du semi-additionneur.

Un semi-additionneur permet de construire un additionneur en tenant compte en entrée de la retenue venant de la paire d'entrées précédentes.

Processeur

À partir de tels circuits logiques on construit assez simplement toutes les opérations arithmétiques & logiques portant sur des données binaires.

La partie du processeur qui s'occupe de ces opérations est l'UAL.

L'autre, qui s'occupe de supervision de ces opérations est l'UC.

L'UC, à partir d'un code d'instruction donné va contrôler quels circuits de l'UAL doivent être « actifs ».

Un élément d'information représenté par 2 états différents : le courant passe « ou bien » ne passe pas, et que l'on représente respectivement par les chiffres 1 et 0 est un **bit**.

Représentation « naturelle » interne à un ordinateur : **binaire**.

Mémoire

Besoin de conserver de l'information : données, résultats intermédiaires d'une opération, séquence des opérations à effectuer.

Bascule : en couplant 2 NAND, on construit un circuit qui « mémorise » de quel côté est passé le courant pour la dernière fois.

Une bascule permet donc de mémoriser « un bit » d'information.

En groupant ces bascules en « circuits intégrés », on arrive à mémoriser des quantités énormes d'information : 16 Mbits (= 16 x 10^6 bits).

On distingue plusieurs types de mémoire :

RAM, ROM, EEPROM, Flash.

RAM = Random Access Memory, mémoire à accès direct ; on distingue les SRAM & les DRAM ;

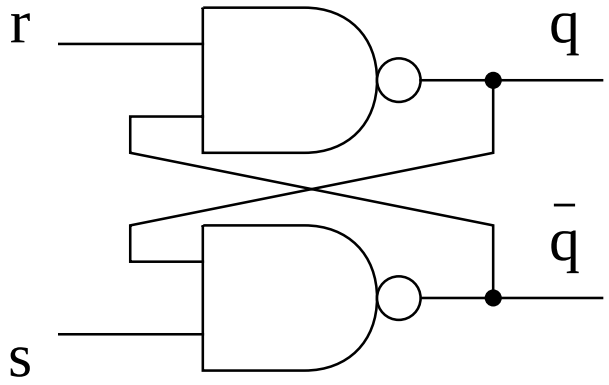
ROM = Read Only Memory, mémoire fixe ;

PROM = Programmable ROM, mémoire fixe programmable ;

EPROM = Erasable PROM ; aujourd'hui supplantée par l'EEPROM = Electrically Erasable PROM.

Bascule

Bascule RS : $(r, s) \rightarrow (q, \bar{q})$



r	s	q_n	$\overline{q_n}$
0	0	-	-
0	1	1	0
1	0	0	1
1	1	q_{n-1}	$\overline{q_{n-1}}$

L'élément de mémoire de un **bit** est réalisé par une bascule D, constituée de 5 portes NAND.

En groupant des bascules on constitue des registres.

Bascule Reset-Set. L'une des 2 entrées doit être à 1. Lorsque l'une des 2 entrées seulement est à 1, la bascule définit les 2 sorties. Si r est à 1, q est mise à 0 (reset), inversement, si s est à 1, q est mise à 1 (set).

Si les 2 entrées sont à 1, les sorties q & $\bar{q}$ donnent leur valeur précédente. On parle de valeur précédente car le temps de traversée d'une porte n'est pas du tout négligeable.

Utilisation de la mémoire

Données & instructions.

Instructions : code de l'opération que doit réaliser l'UAL, lire une information, faire une addition, …. Ces instructions sont codées sous la forme de séquences de bits groupés de façon a pouvoir être manipulés plus vite : mot.

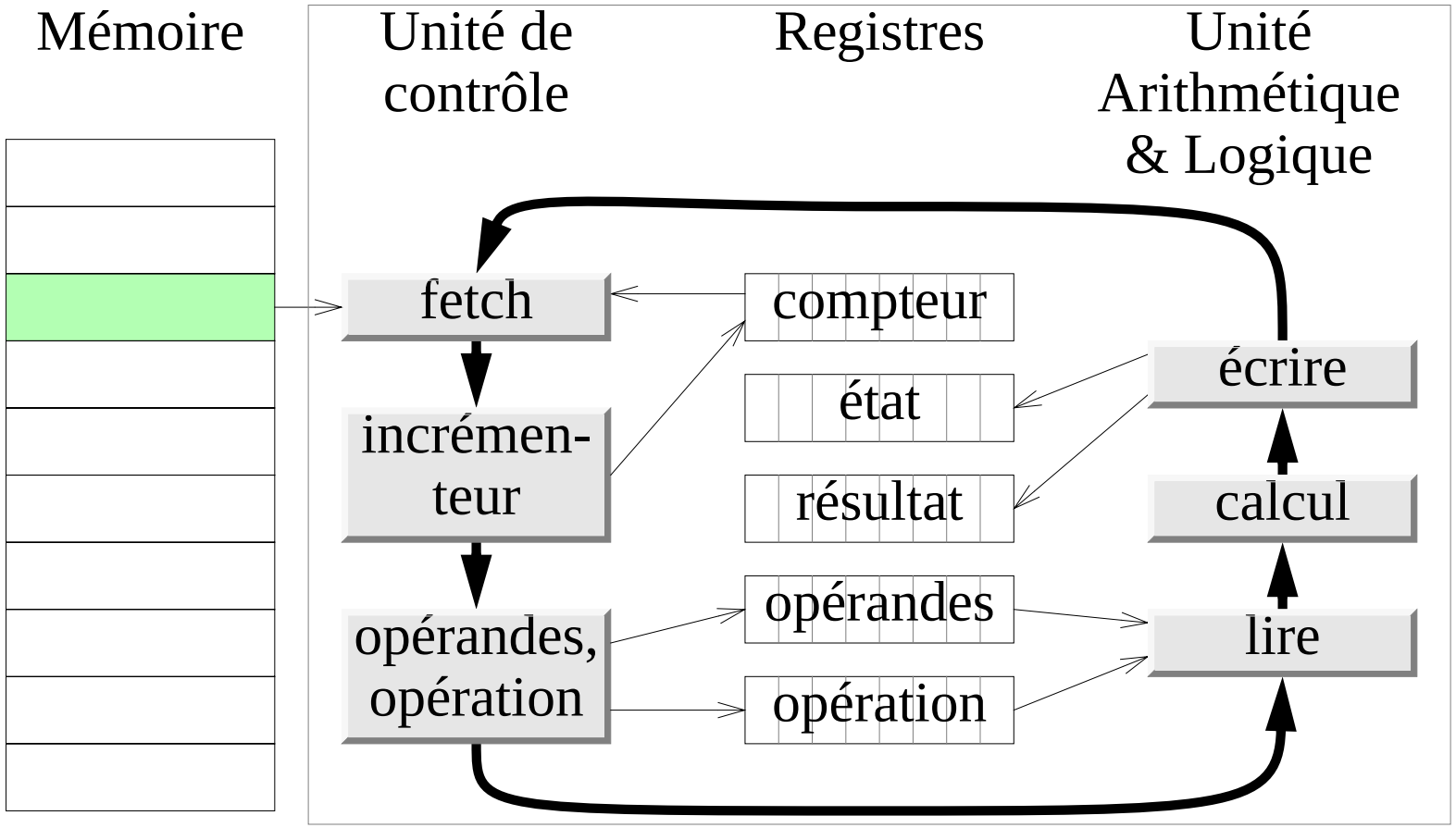
La taille de ces mots évolue : 8 → 16 → 32 → 64 → ….

Pour accéder à un élément donné de la mémoire, un « mot », on utilise son adresse.

Les adresses sont codées en binaire. Avec 8 bits, on peut donc former $2^8 = 256$ adresses.

nombre de bits d'adresse	nombre de mots adressables
16	65 536
32	4 294 967 296
64	$1{,}84\ 10^{19}$

UAL, UC


Mémoire
Unité de contrôle
Registres
Unité Arithmétique & Logique
fetch
incrémen-teur
opérandes, opération
compteur
état
résultat
opérandes
opération
écrire
calcul
lire

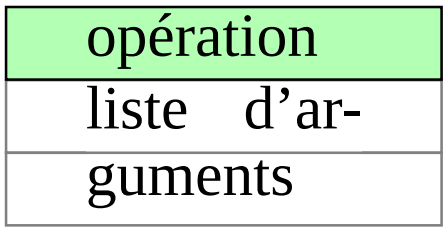
UAL = Unité Arithmétique & Logique ; UC = Unité de Contrôle.

Langage machine

Les instructions se divisent en :

- déplacements de données ;
- opérations sur des données ;
- contrôle.

Elles sont décomposées en 2 champs :




```
load @M, Ri
store Ri, @M
add Ri,Rj,Rk
branch @M
jump @M
```

On distingue selon leur jeu d'instructions les CISC, RISC.

CISC : Complex Instruction Set Computer ;

RISC : Reduced Instruction Set Computer.

Dans un ordinateur d'architecture CISC, une multiplication entière sera mise en œuvre comme une seule instruction, alors que dans un ordinateur d'architecture RISC, elle sera mise en œuvre comme une séquence de décalages & additions.

E. 61 : Que fait la séquence d'instructions suivante :

```
debut   0       load 0, R0
        1       load 5, R1
        2       load 0, R2
suite   3       add R2, R0, R2
        4       add R1, 111111, R1
        5       branch @fin
        6       add R0, 1, R0
        7       jump @suite
fin     8       store R2, @result
```

Représentation des données

Le langage du processeur est binaire.

Toutes les informations peuvent être codées « assez bien » en binaire.

Pour les nombres : systèmes de numération, ….

Pour le texte : systèmes de codage des caractères.

Pour les sons : échantillonage (CAD).

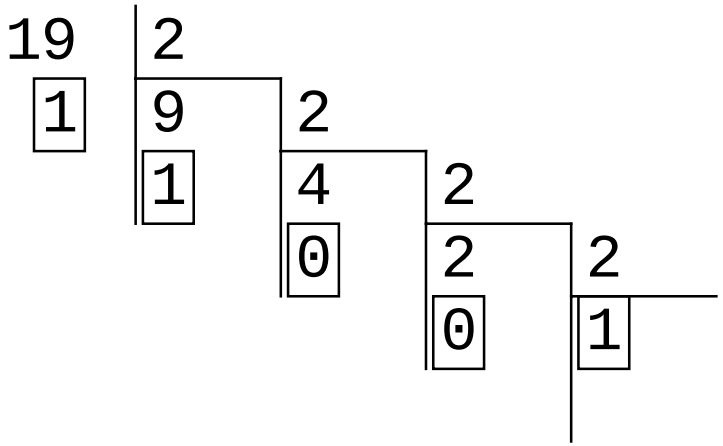
Pour les images : « pixels », numérisation, ….

Grâce à divers systèmes de codages, on ramène tout au codage des entiers en binaire.

Nombres entiers

Système de numération binaire :

$$19_{10} = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 10011_2$$



0
1
10
11
100
...

Arithmétique élémentaire.

Tables d'addition & de multiplication les + simples possibles.

Addition entière

+	0	1
0	0	1
1	1	10

			2^5	2^4	2^3	2^2	2^1	2^0
	19		0	1	0	0	1	1
+	9		0	0	1	0	0	1
	28	0	0	1	1	1	0	0

Comme en base « dix », une addition se fait en additionnant les chiffres en partant de ceux de poids le plus faible et report éventuel d'une retenue.

On construit la multiplication par une suite de décalages du multiplicande et addition à chaque 1 du multiplicateur.

Multiplication entière

		2^5	2^4	2^3	2^2	2^1	2^0	
	5	0	0	0	1	0	1	
×	6	0	0	0	1	1	0	
		0	0	0	0	0	0	0
		0	0	1	0	1	0	1
		0	1	0	1	0	0	1
		0	0	0	0	0	0	0
		0	0	0	0	0	0	0
		0	0	0	0	0	0	0
	30	0	1	1	1	1	0	

¥	0	1
0	0	0
1	0	1

Nombres entiers négatifs

Complément à 2 de x = (complément bit à bit de x) + 1.

Entiers négatifs → bit de signe, complément à 2.

19	0	1	0	0	1	1	
	1	0	1	1	0	0	inverseur
							incrémenteur
-19	1	0	1	1	0	1	

La soustraction se ramène à une addition :

19		0	1	0	0	1	1
+ -9		1	1	0	1	1	1
10							

On construit la division par une suite de décalages et soustraction.

Nombres entiers représentables

Entiers représentables : n bits $\rightarrow [-2^{n-1}, 2^{n-1} - 1]$

nombre de bits / mot	nombres entiers représentables
16	-32 768 → 32 767
32	-2 147 483 648 → 2 147 483 647
64	$-9{,}22\ 10^{18} \rightarrow$

Les opérations arithmétiques ne sont pas des lois de composition internes : certains résultats ne sont pas représentables.

Ces cas sont signalés à l'UC comme une erreur de l'UAL : 1 bit du CSR est mis à 1.

Ex. : sur notre machine à 6 bits, 19 + 20 = -25 !

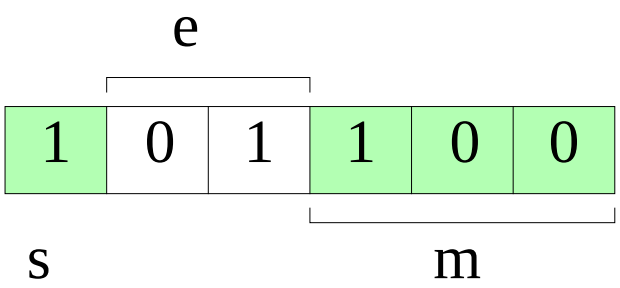
Nombres réels

Notation décimale consomme trop de place pour les 0 non significatifs.

Notation scientifique : $2{,}001 \times 10^{3}$, incomplète : $1 \div 3 \rightarrow ?$

Représentation en virgule flottante : $(-s) \times (1{,}m) \times 2^{e - biais}$

$$-1{,}5_{10} = (-1) \times 1{,}1_{2} \times 2^{0}$$

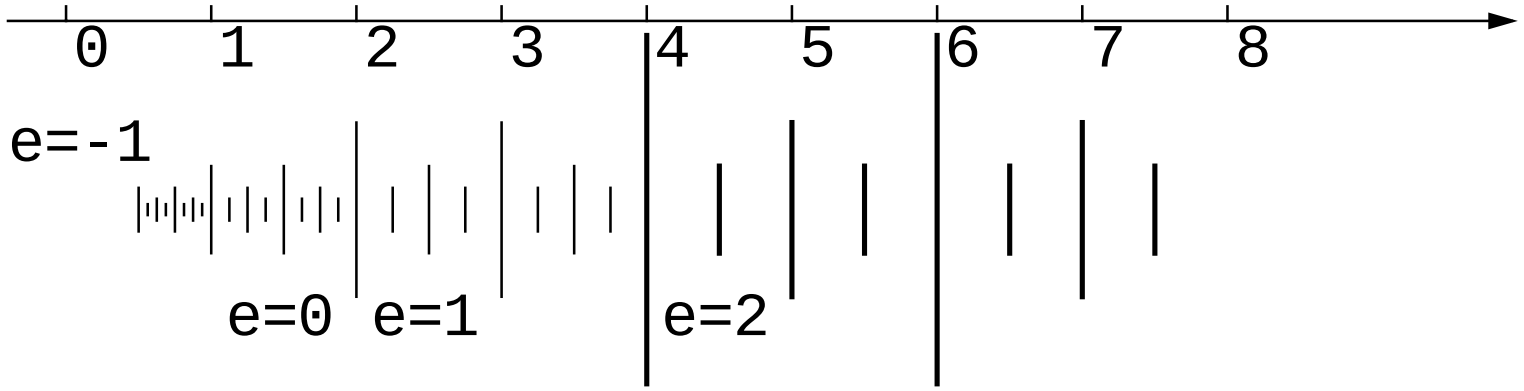


L'exposant est représenté en notation par excés de : biais $= 2^{|e|-1} - 1$, donc : $[-2^{|e|-1} + 1, 2^{|e|-1}] \rightarrow [0, 2^{|e|} - 1]$.

|e| = taille de l'exposant, ici 2 bits. Donc, biais = 1.

Représentation en virgule flottante

e \ m	000	001	010	011	100	101	110	111
	8/8	9/8	10/8	11/8	12/8	13/8	14/8	15/8
	1	1,125	1,25	1,375	1,5	1,625	1,75	1,875
00	-1	1/2	9/16	5/8	11/16	3/4	13/16	7/8	15/16
01	0	1	9/8	5/4	11/8	3/2		7/4	15/8
10	1	2	9/4	5/2	11/4	3	13/4	7/2	
11	2		9/2	5	11/2				



Limitations : les nombres réels sont représentés sur un intervalle fini avec une erreur relative asez stable (inférieure à $2^{-|m|}$) sauf au voisinage de zéro où elle tend vers l'infini.

Les nombres réels représentés en virgule flottante sont l'objets de 3 types d'erreurs. D'une part en entrée, tout ne sont pas correctement représentables. En interne, lors de toute opération il y a perte d'information comme nous allons le voir sur l'exemple le plus simple de l'addition. Enfin en externe, pour représenter les résultats à l'utilisateur humain, il est bien pratique d'utiliser le système de numération décimale, et lors de cette opération de conversion il y a encore des pertes d'informations.

Comparaisons : les exposants sont représentés par des valeurs positives afin de faciliter le tri sur celui-ci dans un premier temps.

Addition en virgule flottante

$(6 + 1{,}25) - 6 = 1$

	6	0	1	1	1	0	0	$1{,}100_2 \times 2^2$	1,100
+	1,25	0	0	1	0	1	0	$1{,}010_2 \times 2^0$	0,010
=	7	0	1	1	1	1	0	$1{,}110_2 \times 2^2$	1,110
-	6	1	1	1	1	0	0	$1{,}100_2 \times 2^2$	1,100
=	1	0	0	1	0	0	0	$1{,}000_2 \times 2^0$	0,010

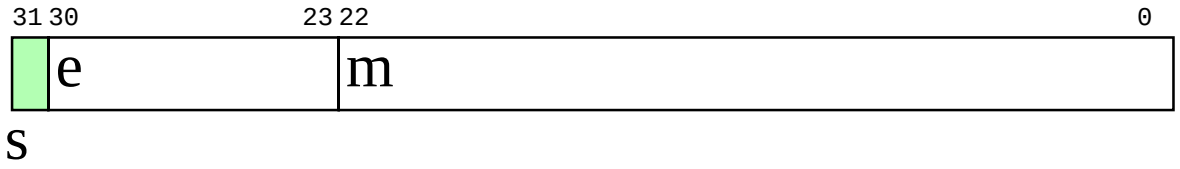
Erreur relative lors de cette addition : $\frac{\Delta x}{x} = 25\%$.

Addition : on amène le nombre le plus petit au même exposant que le plus grand. Pour ce faire, on décale à droite sa mantisse d'un nombre de positions égal à la différence entre les 2 exposants. Lors de ce décalage, des bits de poids faibles du plus petit nombre sont perdus. On additionne les 2 mantisses. Enfin on ramène éventuellement la mantisse du résultat à la forme $1,m$ en ajoutant 1 à l'exposant.

Dans cette opération on perd d'autant plus de précision que l'écart entre les exposants est grand.

L'addition n'est pas tout à fait exacte et nécessite un grand soin si l'on doit beaucoup en utiliser. Faute de quoi les erreurs cumulées peuvent dépasser les valeurs à calculer.

Représentation en virgule flottante



Dans le cas du standard IEEE754, $|e| = 8$, $|m| = 23$,

l'exposant : $e \in [-127,128] \rightarrow [0,255]$.

Les 2 valeurs extrêmes de l'exposant : $\{0, 2^{|e|} - 1\}$ sont réservées à la représentation des cas « extrêmes » $\{-\infty, 0, +\infty\}$.

Les nombres extrêmes représentables en virgule flottante sont :

$(1 + \varepsilon) \times 2^{-126} = 1{,}18 \times 10^{-38}$; $(2 - \varepsilon) \times 2^{127} = 3{,}40 \times 10^{38}$.

La représentation des nombres flottants telle que spécifiée par la norme IEEE754 est un peu plus complexe que la description qui en est faite ici. Pour en savoir plus voir :

`http://renoir.vill.edu/mnt/a/cassel/html/1200/numsys.html`

Caractères, chaînes

Communiquer avec le monde extérieur $\Rightarrow$ coder du texte.

3 codages :

- en entrée, clavier : coordonnées clavier $\rightarrow$ code interne ;
- en interne : convention ;
- en sortie, écran : code interne $\rightarrow$ matrice de points.

Code interne = jeux de caractères codés sur n bits.

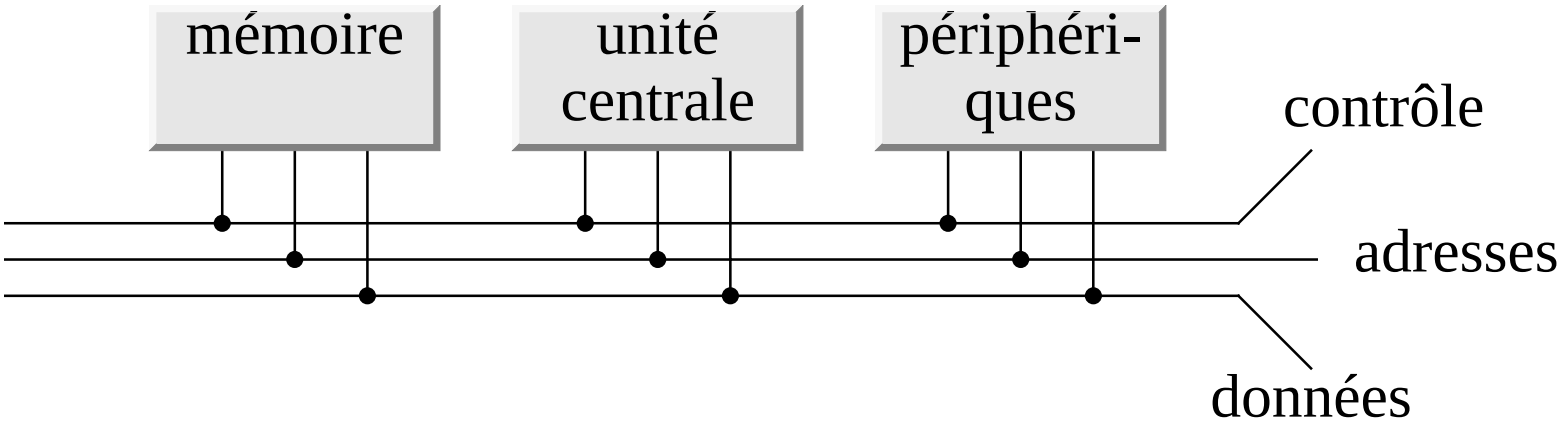
1 octet = 8 bits (unité pratique, car diviseur entier d'un mot mémoire).

ASCII = American Standard Code for Information Interchange codage sur 7 bit, le bit de poids fort = 0.

$\rightarrow$ iso-latin-1, iso-latin-15, Unicode

Pour « voir » les tables de codage iso-8859 : `http://babel.alis.com/codage/iso8859/jeuxiso.fr.htm`

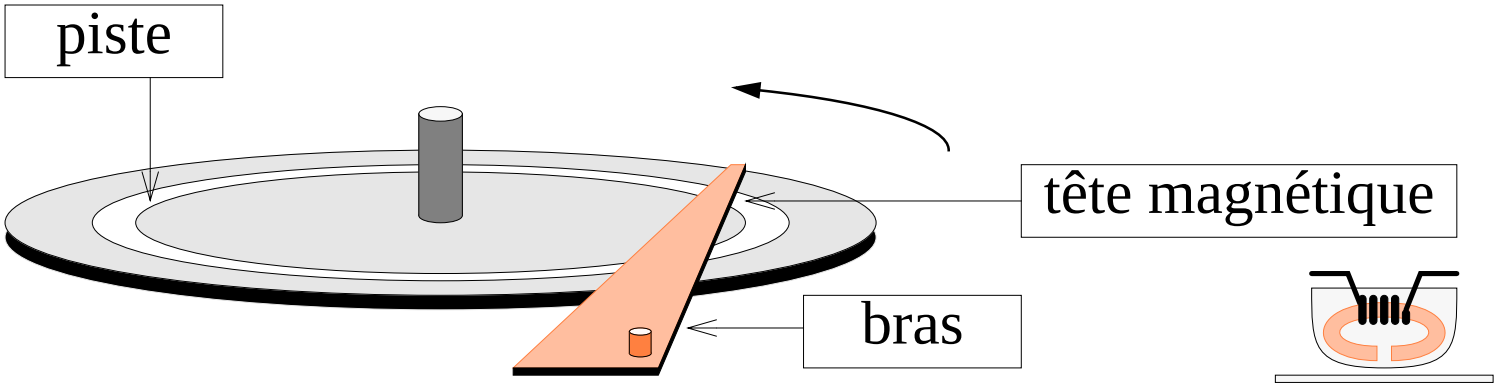
Bus & périphériques



Les différents éléments de l'ordinateur communiquent entre-eux par un certain nombre de connexions parallèles : **bus**.

- Le partie contrôle détermine qui peut à un moment donné émettre sur le bus : protocole d'arbitrage $\Rightarrow$ horloge.
- La partie adresse sélectionne le périphérique.

Disques



Périphérique de mémoire « à long terme », par enregistrement magnétique sur un support métallique recouvert d'une couche magnétisable.

Accès séquentiel, par secteur, lié à la rotation du disque : $\frac{\Delta\theta}{\Delta t} \approx 120\ s^{-1}$ et à la translation de la tête.

Le disque magnétique est utilisable en lecture & en écriture. Toute lecture ou écriture se décompose en 2 opérations :

- un positionnement précisant la tête, la piste, et le secteur ou doit commencer le transfert ;
- transfert des données commençant lorsque la tête voit passer le début du bon secteur.

Le disque magnétique est un périphérique à accès direct. Le temps d'accès aux données est assez indépendant de leur position sur le medium.

Le temps d'accès caractéristique d'un disque magnétique est le temps qu'il faut au bras pour atteindre une piste quelconque partant de sa position de repos. L'autre caractéristique d'un disque magnétique est son débit d'entrée-sortie qui se mesure usuellement en octet/s (Moctet/s).

E. 62 : Où se trouve la position de repos du bras pour minimiser le temps d'accès maximum ?

E. 63 : Quelle est la vitesse relative d'une tête magnétique par rapport à la surface du disque en km/h ?

E. 64 : Que devient cette vitesse si l'on réduit le rayon du plateau du disque ?

Les disques magnétiques actuels sont constitués de plusieurs plateaux surmontés chacun de leur bras. Chacun de ces bras étant lui-même équipé d'une voire plusieurs têtes. Ainsi c'est le débit d'entrée-sortie qui est multiplié d'autant.

Le temps d'accès caractéristique des disques magnétiques du commerce est de l'ordre de 10 ms. Leur capacité est de l'ordre du Goctet (10^9 octets). Bien qu'énorme, la durée de vie de l'information enregistrée sur disque magnétique ne dépasse pas quelques années. Par ailleurs, son taux d'erreur est actuellememt de l'ordre de 10^{-9}. Pour dépasser ces limites, il faut avoir recours à d'autres technologies. Les disques magnéto-optiques, où l'enregistrement est fait par utilisation de l'effet Kerr, permettent d'atteindre des durées de vie de l'ordre de la dizaine d'années (la trentaine prétendent certaines publicités) avec des taux d'erreur de l'ordre de 10^{-15}.

Les disques optiques (rebaptisés cédérom) ne permettent que l'accès en lecture. Leur écriture est réalisée par pressage. En utilisant des lasers plus puissants et un substrat photo-sensible, on arrive aux technologies des CD-R & CD-RW.