

# Architecture de l'ordinateur et systèmes d'exploitation

J. Landré - jerome.landre@univ-reims.fr

I.U.T. Troyes – 2009-2010



# Table des matières

# Table des matières

- 1 Généralités – Historique
- 1 Architecture de l'ordinateur
- 2 Architecture PC
- 1 Systèmes d'exploitation
- 1 Introduction à Linux

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
    - différents systèmes d'exploitation,
    - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Objectifs

Connaître :

- L'architecture des ordinateurs,
  - les composants du microprocesseur,
  - les différents bus d'entrée-sortie,
  - le codage des nombres,
  - la logique et l'arithmétique binaire.
- 
- Les services des systèmes d'exploitation,
  - l'organisation de la mémoire, des processus, du matériel,
  - différents systèmes d'exploitation,
  - introduction à Linux.

# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- Calcul
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- Calcul
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

# Histoire de l'informatique

## Définition

**Informatique** : N. f. Science du traitement rationnel et automatique de l'information (d'après le dictionnaire de l'académie française).

## En anglais

*Computer Science* : science du calcul.

# Histoire de l'informatique

## Définition

**Informatique** : N. f. Science du traitement rationnel et automatique de l'information (d'après le dictionnaire de l'académie française).

## En anglais

**Computer Science** : science du calcul.

# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- **Calcul**
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

# Calcul

- Depuis qu'il sait compter, l'homme a toujours cherché le moyen de calculer rapidement et efficacement.
- Les mathématiques et les nombres sont profondément liés à l'évolution de l'humanité.

- Les sumériens ont été les premiers à utiliser les boules et jetons d'argiles pour traiter l'information vers -10 000.
- Les égyptiens améliorent le principe de boules et jetons d'argile pour l'arpentage lors des crues du Nil vers -3000.
- Le boulier apparaît en Chine vers -3000, il est encore utilisé de nos jours.
- La logique actuelle est définie par le grec **Aristote** vers -330.
- L'abaque romain est une version occidentale du boulier, il est réinventé vers l'an 100 après JC.

# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- Calcul
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

# Moyen-Age

- Travaux du mathématicien perse Al Khwarizmi vers 820.
- Le Quipu péruvien permet de compter avec un système complexe de nœud sur une ficelle vers l'an 1200.
- Le calcul et les règles d'arithmétique se développent au Moyen-Age.

# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- Calcul
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

# Renaissance à la Révolution

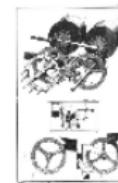
- La Renaissance permet la (re)découverte de nombres importants dans l'antiquité comme le nombre d'or.
- L'écossais **Neper** (Napier) découvre le logarithme qui porte son nom et qui permet de transformer une multiplication en une addition en 1615, il révolutionne le calcul.



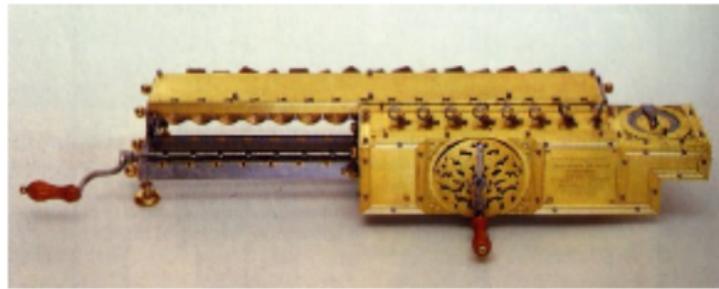
- En 1623, William Schickard invente la première machine à calculer mécanique de l'histoire, capable d'effectuer deux opérations : l'addition et la soustraction.



- Blaise Pascal construit la pascaline en 1641 pour aider son père, percepteur des impôts.

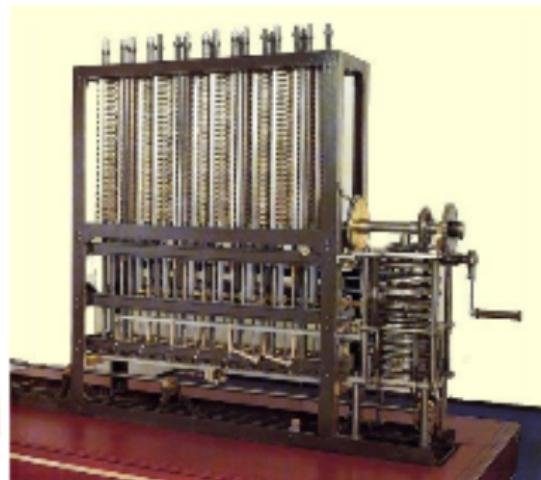


- Gottfried Wilhelm Von Leibnitz ajoute la multiplication et la division à la pascaline en 1673.



- En 1728, Falcon met au point un métier à tisser programmable grâce à des plaquettes de bois perforées.
- En 1806, Joseph-Marie Jacquard équipe les ateliers lyonnais des canuts avec un métier à tisser programmé à l'aide de cartes perforées. Il y aura jusqu'à 10 000 exemplaires en service en 1812.

- En 1820, Charles Babbage imagine deux versions de sa machine à différences dont la complexité est impressionnante (25000 pièces mécaniques).
- Aucune de ses deux versions ne sera terminée, pourtant des chercheurs britanniques ont démontré son fonctionnement en 1991.

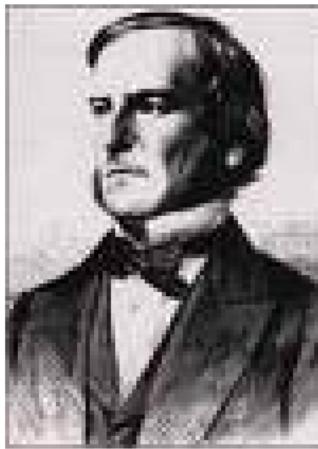


- En 1822, **Jean-Baptiste Joseph Fourier** invente le développement de fonctions en série qui porte son nom. Cette théorie est toujours utilisée aujourd'hui.
- C'est la base du traitement numérique du signal.



- En 1833, Charles Babbage et Ada Lovelace inventent la machine analytique composée d'un moulin (unité de calcul), d'un magasin (mémoire) et d'un dispositif de contrôle.
- Cet ensemble était utilisé avec des cartes opérations, des cartes variables et des cartes nombres.
- C'est le modèle exact du fonctionnement d'un ordinateur moderne.
- L'histoire retiendra que Ada Lovelace (qui deviendra lady Augusta Ada Byron) fut l'inventeur de la programmation.
- En l'honneur du mathématicien Al Khowarizmi, elle appelle algorithme le processus logique permettant l'exécution d'un programme.

- En 1854, Georges Boole introduit le calcul binaire qui structure la logique avec deux états 1/0, oui/non, vrai/faux.
- Ces travaux furent la conséquence de la redécouverte des mathématiques chinoises (yin/yang) et de la logique de Leibnitz.



# Table des matières

## 1 Généralités – Historique

- Histoire de l'informatique
- Calcul
- Moyen-Age
- 16ème à 19ème siècle
- Temps modernes

## XXème siècle

- En 1934, **Johannes Von Neumann** définit l'architecture qui porte son nom et qui est à l'origine de l'ordinateur actuel.
- En 1936, **Alan Turing** définit la notion formelle d'algorithme et de complexité.
- En 1940, le circuit imprimé permet de réunir des composants électroniques sur de petites cartes.
- En 1941, Konrad Zuse invente un ordinateur qui fonctionne grâce à des relais électro-mécaniques. C'est le Z1 puis le Z3. Ce dernier comporte alors 2600 relais, sa mémoire peut contenir 64 nombres de 22 chiffres.
- L'ABC est créé par J. Atanasoff et C. Berry la même année. Sa fréquence était de 60Hz, il réalisait 1 multiplication à la seconde. Sa mémoire était de 60 mots de 50 bits. C'est le premier ordinateur à utiliser le binaire.

- En 1943, Howard Aiken met au point un ordinateur programmable mesurant 17 m de long et 2,5 m de hauteur, pesant 5 tonnes, c'est le Mark I d'IBM. Il contient 3300 engrenages et 1400 commutateurs reliés par 800 km de fil électrique.



- Le 9 septembre 1945 à 15h45 a lieu le premier bug de l'histoire informatique.
- Il s'agit d'une mite (*bug*, insecte en anglais) bloquée dans un relais sur le calculateur électromécanique Mark I de l'Université de Harvard (Massachusetts, USA),
- En français, on doit dire bogue (masculin).

9/9

0800  
1000

Auton started

stopped - auton ✓  
13.06 (033) MP-MC

{ 1.2700 9.037 847 025  
9.037 846 995 correct  
1.307 767 15 (-2) 4.615 925 059 (-2)

033 PRO 2 2.130476415

correct 2.130676515

Relays 6-2 in 033 failed special speed test  
in relay 11.00 test.

Relays changed

1100 Started Cosine Tape (Sine check)  
1525 Started Multi Adder Test.

Relay 2145  
Relay 3371

1545



Relay #70 Panel F  
(moth) in relay.

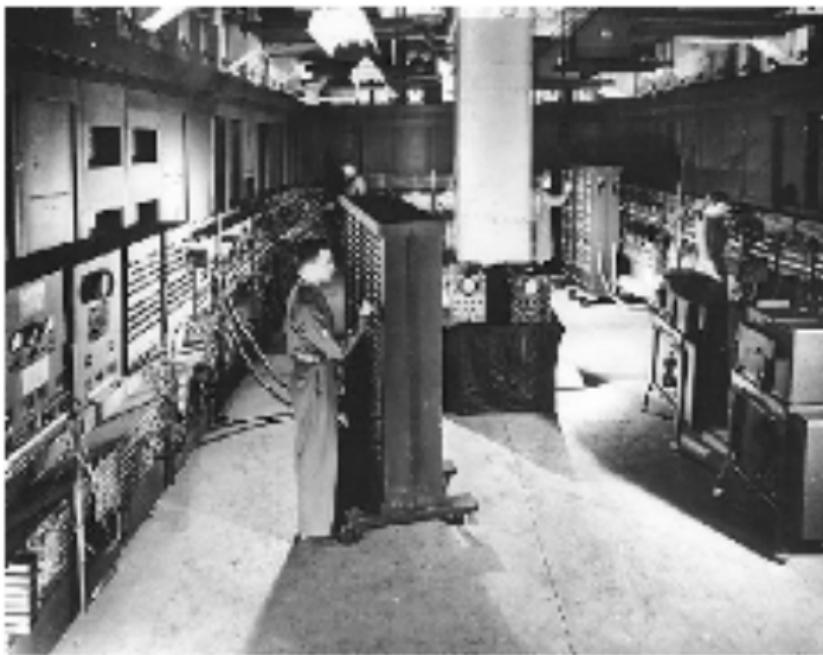
1600

First actual case of bug being found.

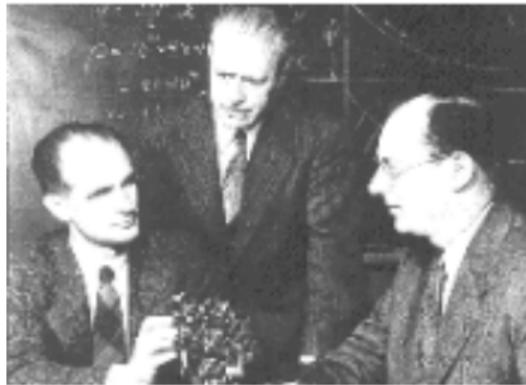
auton started.

1700 closed down.

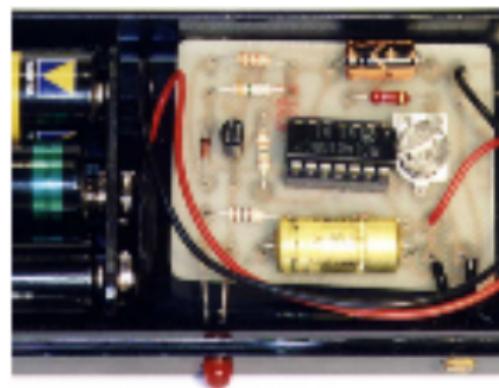
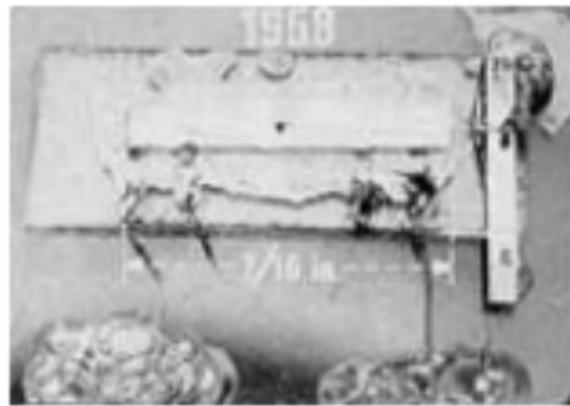
- En 1946, le premier ordinateur sans pièce mécanique est créé : l'ENIAC. Il comporte 18 000 lampes à vide, occupe 23 m<sup>2</sup>, pèse 30 tonnes, consomme 200 kW, possède 70 000 résistances, 10 000 condensateurs, 1 500 relais et 6 000 commutateurs manuels.



- En 1948, le transistor est inventé par John Bardeen, Walter Brattain et William Shockley. Le prix Nobel leur est décerné en 1956. Cette invention permet de réduire considérablement la taille et la consommation des ordinateurs.
- Egalement en 1948, **Claude Shannon** définit la théorie mathématique de l'information.



- En 1951 apparaît l'UNIVAC équipé d'un lecteur de bande magnétique, il comporte 5 000 tubes, sa mémoire est de 1 000 mots de 12 bits, il peut réaliser 8333 additions et 555 multiplications à la seconde. Sa superficie au sol est de 25 m<sup>2</sup>.
- En 1958, le circuit intégré, inventé par Texas Instrument, permet de réunir des composants électroniques dans des boîtiers très petits.



# Temps modernes

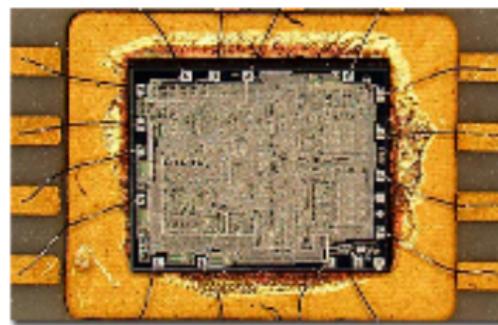
- En 1960, l'IBM 7000 est le premier ordinateur à base de transistors.
- En 1964, l'IBM 360 est commercialisé. Il possède cinq versions différentes, **compatibles entre elles**. Il est l'ancêtre de l'IBM AS/400 encore utilisé dans de nombreuses entreprises.

# Internet

- En 1969, le projet ARPANET du département de la défense américain permet de relier 4 ordinateurs entre eux à travers les Etats-Unis.
- Ce projet tombe rapidement dans le domaine universitaire, **Internet** est né.

# Microprocesseurs

- En 1971, le premier microprocesseur apparaît : l'Intel 4004. Il permet d'effectuer des opérations sur 4 bits. Sa cadence est de 108 kHz, il adresse 640 octets de mémoire et effectue 60000 opérations par seconde.
- En 1972, son successeur le 8008 travaille sur 8 bits, cadencé à 200 kHz et adressant 16 ko de mémoire. La disquette apparaît la même année.



# Micro-ordinateurs 8 bits

- En 1973, le premier micro-ordinateur de l'histoire est français, c'est le Bull Micral basé sur un 8080.
- En 1974, le disque dur de 1Mo est lancé.
- En 1976, le premier Apple est mis au point dans un garage.
- En 1977, L'apple II d'Apple Computer est un succès immédiat.

- Puis viennent le PET de Commodore, le TRS-80 de Tandy, le ZX81, le Commodore 64, l'Oric 1.
- En 1981, IBM invente l'IBM PC dont tous les PC actuels sont les descendants.
- En 1983, Thomson sort le MO5 suivi du TO7.
- En 1985, Amstrad lance les CPC 464 et CPC 6128.

# Micro-ordinateurs 16 bits

- En 1985, Apple commercialise le Macintosh.
- En 1986, L'abandon des machines 8 bits au profit des machines 16 bits est déclenché par l'arrivée de l'Atari 520ST et de l'Amiga 500.



- Internet se développe aux Etats-Unis, puis dans le monde entier à une vitesse exceptionnelle.
- Microsoft lance Windows 3.11, Windows 95, 98, Me, NT, 2000 et XP.  
L'ordinateur devient un produit grand public.
- Les microprocesseurs suivent la loi de Moore : "le nombre de transistor est multiplié par quatre tous les trois ans". Ainsi en 2010, on devrait atteindre le milliard de transistors.

- Intel sortira à la suite du 8080 le 8086, le 80286 (16 bits), le 80386, le 80486, le pentium pour arriver au pentium II, pentium III et pentium IV (32 bits) ; au Core 2 Duo et à l'Itanium 1 et 2 (64 bits),
- Les cartes vidéos actuelles possèdent des possibilités de calculs étonnantes,
- Les processeurs à quatres coeurs sont disponibles pour le grand public.



# Aujourd'hui

- L'Intel Quad Core Xeon est cadencé à 3 GHz, avec 16 Mo de cache interne, il effectue environ quatre milliards d'opérations à la seconde. La mémoire standard est de 2 Go. Le disque dur de base stocke 300 Go de données,
- La plupart des micro-processeurs sont entièrement 64 bits, la génération 128 bits est en cours de préparation dans les laboratoires,
- Intel et AMD prévoient des octo-cœurs en 2009, 16 et 32 cœurs éventuellement en 2010.

# Architecture de l'ordinateur et systèmes d'exploitation

J. Landré - jerome.landre@univ-reims.fr

I.U.T. Troyes – 2009-2010



# Table des matières

- Introduction

# Table des matières

- 1 Généralités – Historique
- 1 Architecture de l'ordinateur
- 2 Architecture PC
- 1 Systèmes d'exploitation
- 1 Introduction à Linux

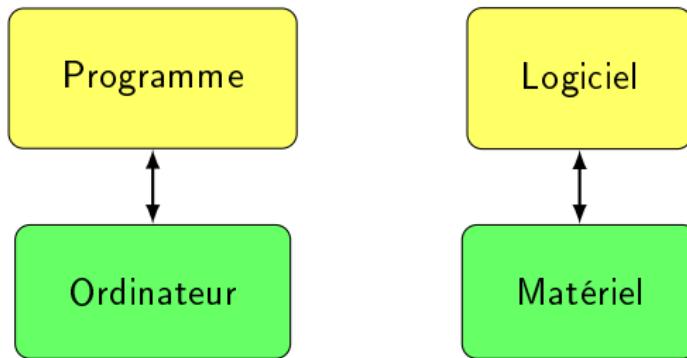
# Table des matières

- Introduction

# Définitions

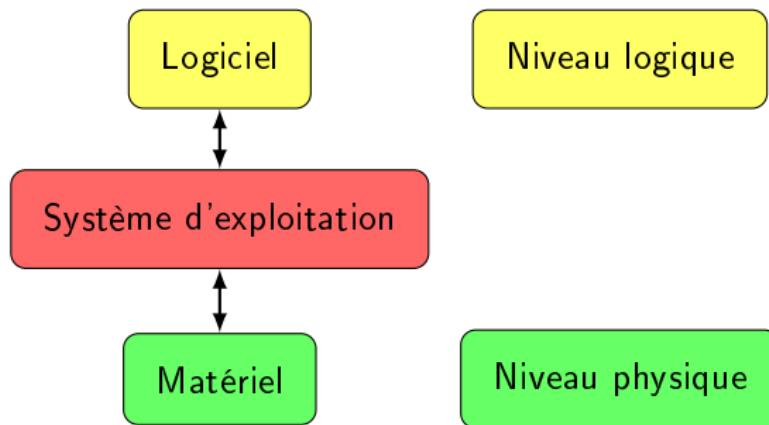
- Ordinateur : Machine capable de résoudre des problèmes en appliquant des instructions préalablement définies.
- Programme : Suite des instructions décrivant la façon dont l'ordinateur doit effectuer un certain travail.

# Architecture la plus simple



# Système d'exploitation

Le système d'exploitation réalise l'interface entre le matériel (*hardware*) et les logiciels (*software*).



## Niveau physique

Tout ce qui est réellement dans l'ordinateur :

- Microprocesseur,
- Mémoire,
- Disque dur...

## Niveau logique

Ce qui est montré à l'utilisateur

- Liste des processus,
- Organisation du disque dur,
- Périphériques...

Ordinateur → Machine électronique → Binaire

```
100110110010101101010110010001010111010100101  
000001010011010101110101111010110101010101001  
011001010101010101010101010101010101010101101  
001101010101010010010001010010101010101010101  
010011010110100110000110101000101010101010101  
010010100100101011010101010100101010101010010  
101010101010100101010101011101010110101010101  
00110101011001010101010010101010101001010101  
10100001001101011101011111010101010100110010
```

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Langage machine

- L'ordinateur est une machine électronique dont les circuits ne peuvent exécuter qu'un nombre réduit d'instructions simples : additionner deux nombres, voir si un nombre est égal à zéro, lire et écrire des nombres en mémoire, ...
- **Langage machine** : ensemble des instructions compréhensibles directement par l'ordinateur.
- Problème : L'ordinateur étant une machine électronique, son langage machine L1 est en binaire, chaque instruction représente une opération électronique au niveau des registres (mémoire interne) du micro-processeur, le code est donc très difficile à lire.
- Exemple :  
00000001 10000101 // charger registre A avec adresse 133  
00000010 00000000 // incrémenter valeur registre A  
...

# Machines virtuelles

## Solution :

- On construit un langage L2 plus facile à comprendre : on associe chaque instruction de L2 avec un ensemble d'instructions de L1.
- Exemple :

LOAD A,(133) // charger registre A avec adresse 133  
INC A // incrémenter valeur registre A

...

On a donc créé une machine virtuelle L2 capable de comprendre le langage L2 et de le transformer en langage L1.

Problème : Le langage L2 reste difficile à comprendre.

## Solution :

- On construit un langage L3 plus facile à comprendre : on associe chaque instruction de L3 avec un ensemble d'instructions de L2.
- Exemple :

```
int a ;
```

```
a=mem[133] ; // charger registre A avec adresse 133
```

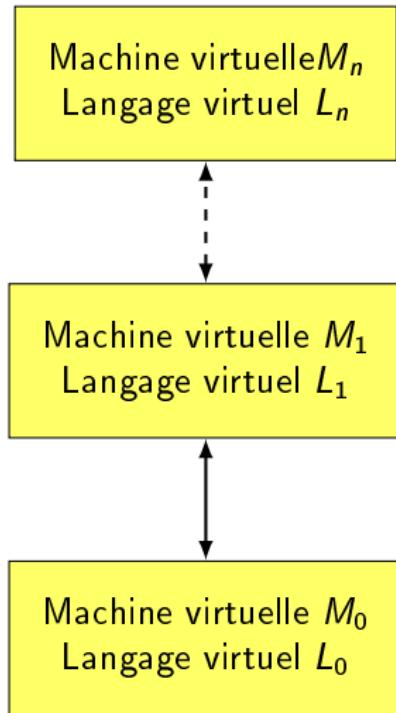
```
a++ ; // incrémenter valeur registre A
```

```
...
```

On a donc créé une machine virtuelle L3 capable de comprendre le langage L3 et de le transformer en langage L2.

Problème : Le langage L3 reste difficile à comprendre.

# Machines multi-couches



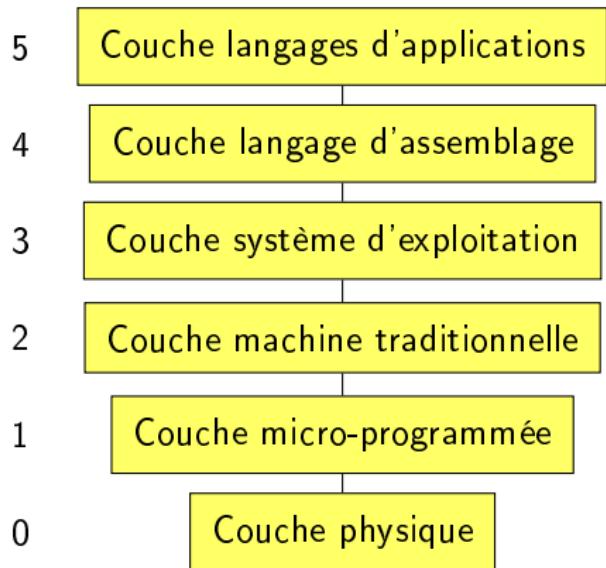
Un ordinateur est donc une machine réelle électronique comprenant un langage machine à laquelle on a ajouté des couches de machines virtuelles comprenant chacune un langage virtuel.

# Compilation/interprétation

Au niveau k, il y a deux façons de voir le langage de niveau k+1 :

- Dans sa globalité : le programme virtuel écrit dans le langage L(k+1) est transformé entièrement en langage Lk : c'est la **compilation**, réalisée par un **compilateur**,
- Ligne par ligne : le programme virtuel écrit dans le langage (k+1) est lu ligne par ligne en effectuant à chaque fois les instructions correspondantes du langage Lk : c'est l'**interprétation** réalisée grâce à un **interpréteur**.

# Architecture à six niveaux



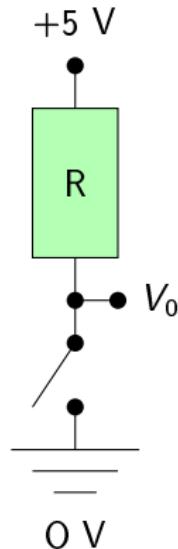
- Niveau 5 : langages de programmation évolués (C, C++, BASIC, ADA, LISP, ...)
- Niveau 4 : langage assembleur
- Niveau 3 : services du système d'exploitation
- Niveau 2 : micro-programme évolué
- Niveau 1 : micro-programme physique
- Niveau 0 : portes logiques numériques

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- **Niveau physique**
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Système binaire



Le matériel qui compose l'ordinateur est électronique, il ne comprend que deux états : 0 ou 1.

- Circuit ouvert :  $V_0 = 5V$
- Circuit fermé :  $V_0 = 0V$

Notations :

- Numérique : 1 ou 0 (bit : binary digit)
- Logique : vrai ou faux (oui ou non)
- Électronique : on ou off (haut ou bas)

# Nombre d'états

Un conducteur électrique ne peut représenter que deux états : 0 ou 1.

Pour représenter plus d'états, il suffit d'associer des conducteurs en parallèle.

- 1 bit :  $2^1 = 2$  états : 0 et 1,
- 2 bits :  $2^2 = 4$  états : 00, 01, 10 et 11,
- 3 bits :  $2^3 = 8$  états : 000, 001, 010, 011, 100, 101, 110 et 111,
- ...
- 8 bits :  $2^8 = 256$  états : 00000000, 00000001, 00000010, ..., 11111110 et 11111111.
- ...

# Nombre binaire

nombre binaire	1	0	0	1	1	0	1	0	bit de poids fort	bit de poids faible
puissance de deux	$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$		
valeur décimale	128	64	32	16	8	4	2	1		

# Taille des données

- 8 bits, 256 valeurs possibles ( $2^8$ )
- 16 bits, 65 536 valeurs possibles ( $2^{16}$ )
- 32 bits, 4 294 967 396 valeurs possibles ( $2^{32}$ )
- 64 bits, 18 446 744 073 709 551 616 valeurs possibles ( $2^{64}$ )

# Vocabulaire

Nom des nombres binaires :

- 1 chiffre (*digit*) binaire : 1 bit (*binary digit*)
- 4 bits : un quartet (*quad*)
- 8 bits : un octet (*byte*)
- 16 bits : un mot (*word*)
- 32 bits : un double mot (*double word*)
- 64 bits : un quadruple mot (*quad word*)

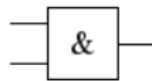
# Algèbre de Boole

En 1854, Georges Boole a défini une structure d'algèbre pour la réalisation des fonctions logiques. Elle est basée sur trois fonctions de base : ET (AND), OU (OR), NON (NOT).

Tables de vérité :

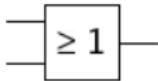
$a$	$b$	$a.b$
0	0	0
0	1	0
1	0	0
1	1	1

ET



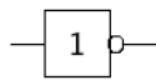
$a$	$b$	$a + b$
0	0	0
0	1	1
1	0	1
1	1	1

OU



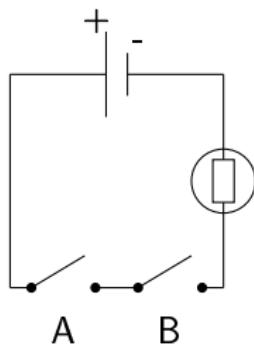
$a$	$\bar{a}$
0	1
1	0

NON

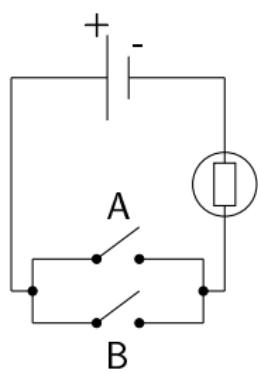


# Algèbre de Boole

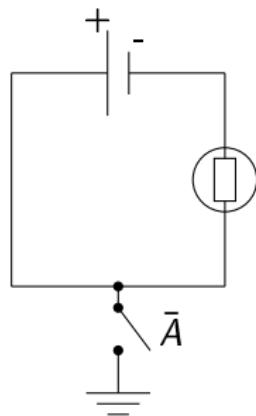
Le schéma ci-dessous donne une analogie entre les portes logiques et les circuits électriques pour bien comprendre leur fonctionnement :



A et B (noté A.B)



A ou B (noté A+B)



NON A (noté  $\bar{A}$ )

# Lois de l'algèbre de Boole

Loi	ET	OU
identité	$1 \cdot a = a$	$0 + a = a$
nullité	$0 \cdot a = 0$	$1 + a = 1$
idempotence	$a \cdot a = a$	$a + a = a$
inversion	$a \cdot \bar{a} = 0$	$a + \bar{a} = 1$
commutativité	$ab = ba$	$a + b = b + a$
associativité	$(ab)c = a(bc)$	$(a + b) + c = a + (b + c)$
distributivité	$a + bc = (a + b)(a + c)$	$a(b + c) = ab + ac$
absorption	$a(a + b) = a$	$a + ab = a$
loi de De Morgan	$\overline{ab} = \bar{a} + \bar{b}$	$\overline{a + b} = \bar{a} \cdot \bar{b}$

$$\bar{\bar{a}} = a$$

$$\bar{a} \cdot a = 0$$

$$\bar{a} + a = 1$$

$$a + (\bar{a} \cdot b) = a + b$$

$$a + (a \cdot b) = a$$

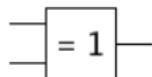
$$a \cdot (a + b) = a$$

- Une proposition logique toujours vraie est une **tautologie**

# OU EXCLUSIF

$a$	$b$	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

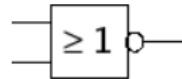
$$\begin{aligned}a \oplus b &= (a + b) \cdot \overline{(a \cdot b)} \\a \oplus b &= (a \cdot \bar{b}) + (\bar{a} \cdot b) \\a \oplus b &= ((a \cdot b) + (\bar{a} \cdot \bar{b})) \\a \oplus b &= (a + b) \cdot (\bar{a} + \bar{b})\end{aligned}$$



# NON-OU / NON-ET

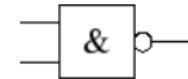
Porte NON-OU (*NOR*)

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



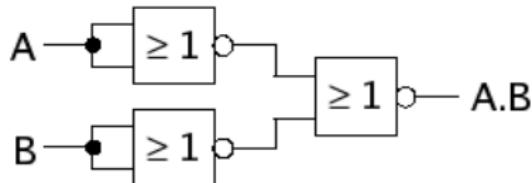
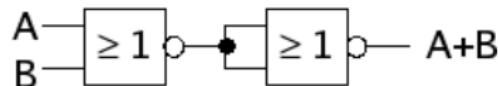
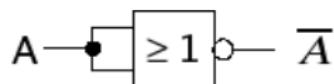
Porte NON-ET (*NAND*)

$a$	$b$	$\overline{(a.b)}$
0	0	1
0	1	1
1	0	1
1	1	0



Théorème : Toute fonction logique peut-être réalisée à partir de portes NON-ET et NON-OU.

# Application du théorème



# Table des matières

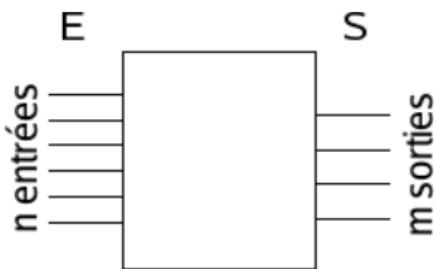
## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- **Logique binaire, circuits logiques, fonctions câblées**
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Circuit logique

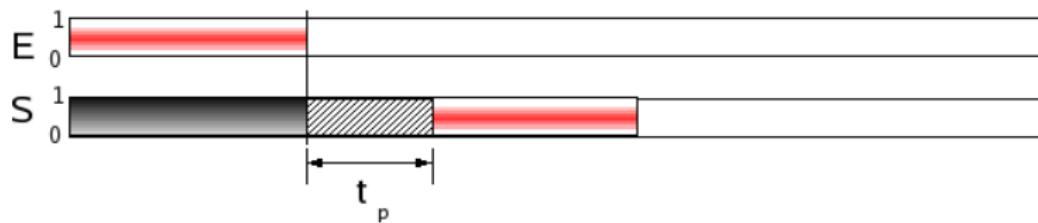
Un circuit logique est un dispositif électronique permettant la réalisation d'une fonction logique

Chaque variable de la fonction est matérialisée par un conducteur et sa valeur est définie par une tension



- $(s_1, \dots, s_m) = f(e_1, \dots, e_n)$
- Entrées : conducteurs qui amènent les opérandes
- Sorties : conducteurs qui fournissent le résultat

Il y a toujours un délai de propagation,  $t_p$  dans le circuit, diagramme temporel :



# Table des matières

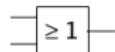
## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- **Types des circuits**
  - Circuits combinatoires
  - Circuits séquentiels
  - Architecture de Von Neumann
  - Le processeur
  - la mémoire
  - L'unité d'entrées/sorties
  - Représentation de l'information numérique
  - Bases de numération

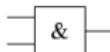
Il existe deux types de circuits :

- Les circuits **combinatoires** dont les sorties sont uniquement fonction des entrées (pas de mémoire interne),
- Les circuits **séquentiels** dont les sorties sont fonction des entées et de l'état antérieur du circuit (mémoire interne).

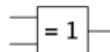
OU (OR)



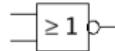
ET (AND)



OU-EXCLUSIF (XOR)



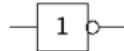
NON-OU (NOR)



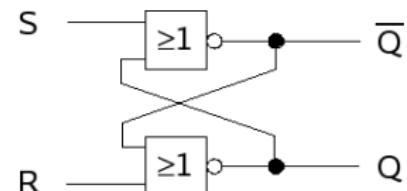
NON-ET (NAND)



NON (NOT)



Circuits combinatoires



Circuit séquentiel (à mémoire)

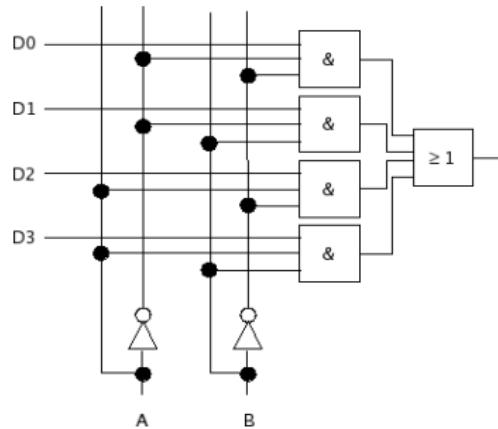
# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

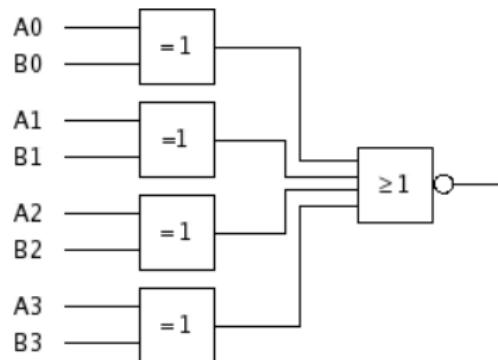
# Le multiplexeur

- Il possède  $2^n$  entrées, une sortie et  $n$  lignes de sélection.
- Il représente une fonction d'aiguillage.
- Une seule entrée passe en sortie selon les valeurs de la ligne de sélection.



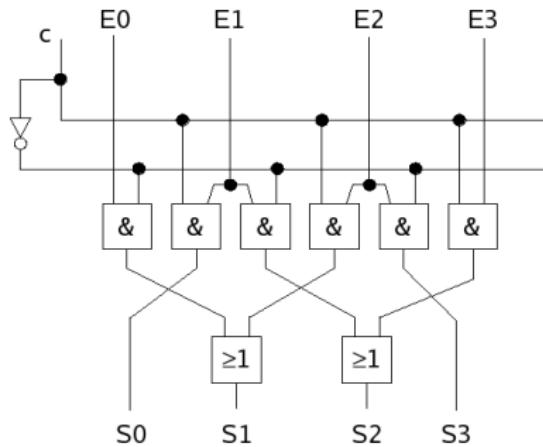
# Le comparateur

- Il sert à comparer deux nombres binaires.
- La sortie vaut 1 lorsque les deux nombres sont égaux et 0 sinon.
- Il utilise des portes XOR pour la comparaison et une porte NOR pour le résultat.



# Le décaleur

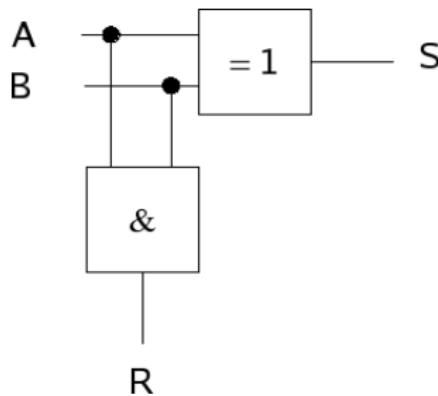
- Il permet le décalage à gauche ou à droite d'un nombre binaire :



Quand on décale un nombre binaire de  $n$  positions vers la gauche, on le multiplie par  $2^n$ ,

Quand on décale un nombre binaire de  $n$  positions vers la droite, on le divise par  $2^n$ .

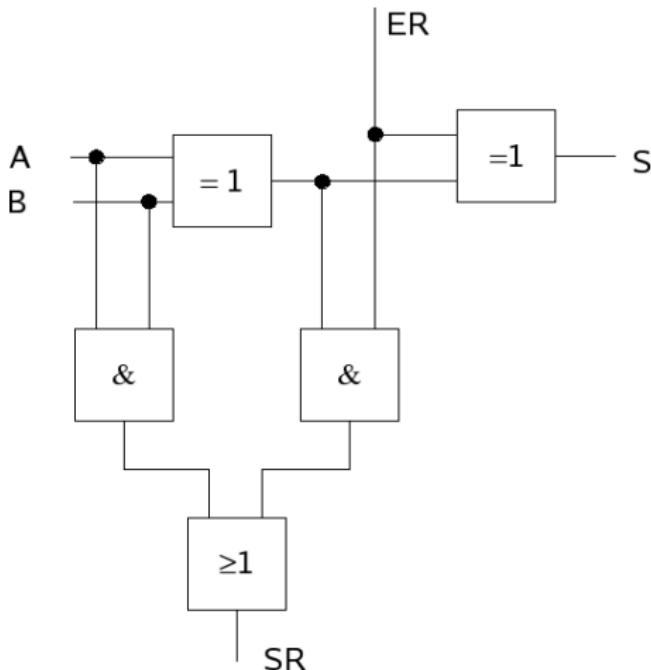
# Le semi-additionneur



Semi-additionneur

A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

# l'additionneur



Additionneur complet

A	B	ER	S	SR
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

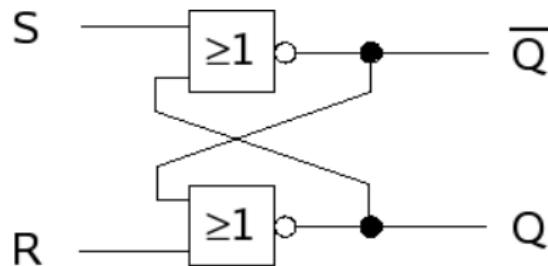
# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- **Circuits séquentiels**
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# La bascule RS

- La bascule RS est un circuit stable de mémorisation de 1 bit :



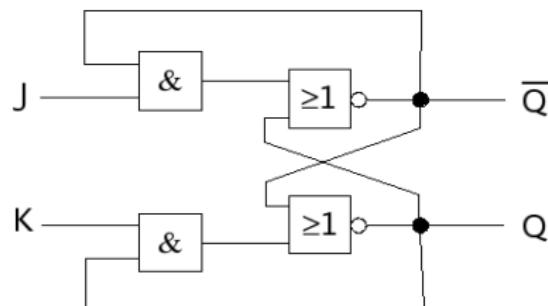
S (*Set*) met la sortie Q à 1,

R (*Reset*) met la sortie Q à 0.

Il y a instabilité lorsque R et S sont mis à 1 simultanément !

# La bascule JK

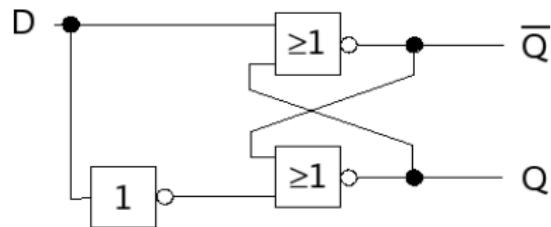
- La bascule JK est un circuit stable de mémorisation de 1 bit :



Il n'y a plus d'instabilité avec cette bascule, mais il y a toujours deux entrées.

# La bascule D

- La bascule D est un circuit stable de mémorisation de 1 bit :



Il n'y a plus d'instabilité avec cette bascule à une seule entrée.

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels

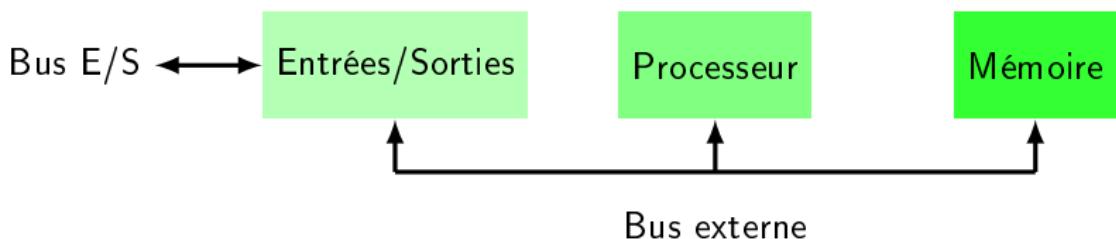
### • Architecture de Von Neumann

- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Architecture de Von Neumann

Un ordinateur est composé :

- d'un **processeur**,
- de **mémoire**,
- d'une **unité d'entrées-sorties**.



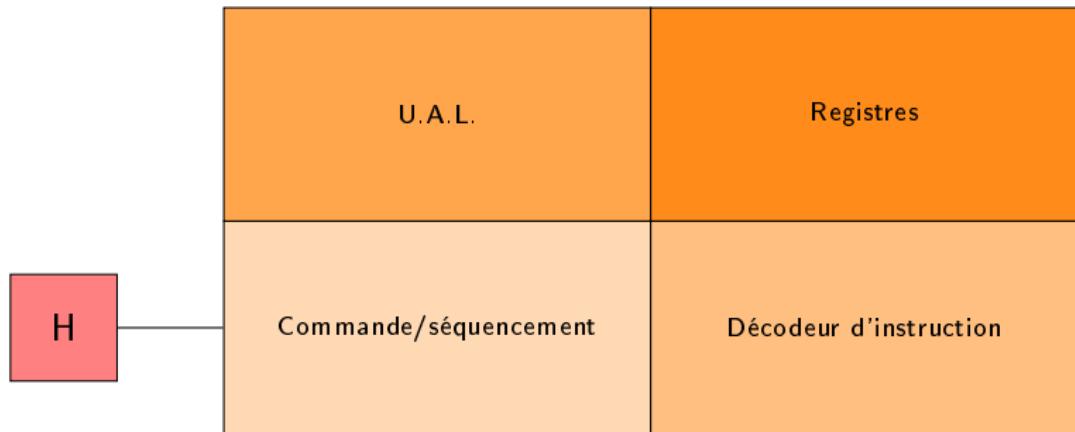
# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- **Le processeur**
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# Le processeur

C'est le cerveau de l'ordinateur, c'est lui qui est chargé des calculs. Il est aidé par un coprocesseur dédié aux calculs à virgule flottante (nombres réels).



U.A.L. : Unité arithmétique et logique — H : Horloge

# Horloge

Le processeur est un circuit **synchrone**, cadencé par une horloge.

Exemple : Processeur à 3,2 GHz → 3,2 milliards d'oscillations par seconde !

L'horloge permet l'exécution d'instructions à des intervalles de temps réguliers (front montant).



La fréquence  $f$  est exprimée en Hertz (Hz) si la période  $T$  est exprimée en secondes (s), on a :

$$f = \frac{1}{T} \text{ et } T = \frac{1}{f}$$

# Les registres

- Les registres sont des **emplacements mémoires internes au processeur** destinés à recevoir des données de tailles fixes,
- Les registres du Pentium 4 sont 32 bits contre 64 bits pour l'Intel Dual Core ou l'AMD64x2,
- Puisqu'ils sont internes au processeur, l'accès et le calcul sur ces mémoires sont très rapides,
- Il y a dans les processeurs un registre particulier appelé **accumulateur**, c'est en général vers lui que sont envoyés les résultats de calcul sur les registres.

# L'unité arithmétique et logique

- Elle est chargée d'exécuter des instructions simples sur les registres :
- Addition, soustraction, décalage, comparaison (opérations arithmétiques),
- et, ou, non, ou exclusif (opérations logiques).
- Elle ne fait que des choses **simples**, mais très rapidement (tout est **câblé**).

# Le décodeur d'instruction

- Il est chargé de décoder les instructions qui viennent d'un programme en cours d'exécution (processus),
- Il organise le transfert des données nécessaires aux calculs de l'extérieur (mémoire) vers l'intérieur (registres) du processeur.

# L'unité de commande et de séquencement

- L'unité de commande et traitement effectue le transfert des données nécessaires aux calculs à l'intérieur (registres) du processeur,
- Les instructions sont exécutées en séquence à chaque top d'horloge.

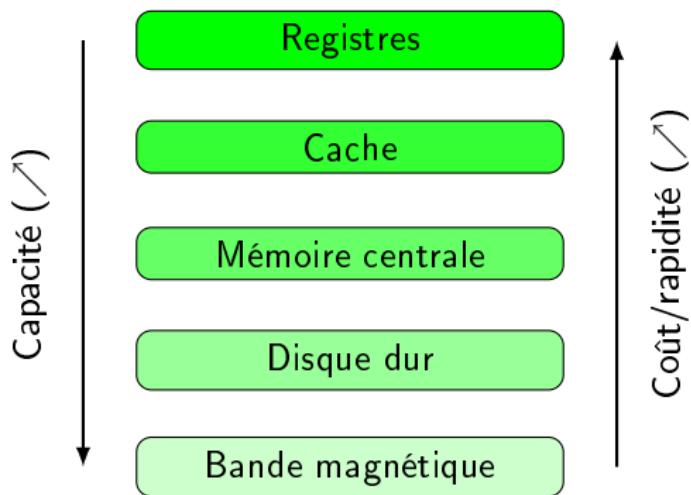
# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- Bases de numération

# La mémoire

- La mémoire est la zone de stockage des données de l'ordinateur,
- On distingue différents types de mémoires (voir schéma),
- Dans le schéma ci-dessous, les valeurs augmentent dans le sens de la flèche qui les porte :



# RAM et ROM

Dans un ordinateur, on trouve toujours deux mémoires principales :

- La **RAM** (*Random Access Memory*), mémoire vive qui est modifiable par l'utilisateur,
- La **ROM** (*Read Only Memory*), mémoire morte qui est accessible en lecture seulement.

La RAM est d'environ 1 giga-octets et contient le système d'exploitation et les applications lancées par l'utilisateur,

La ROM est beaucoup plus petite et contient les micro-programmes machines de base pour démarrer le système et détecter les périphériques (BOOT).

# Table des matières

## 1 Architecture de l'ordinateur

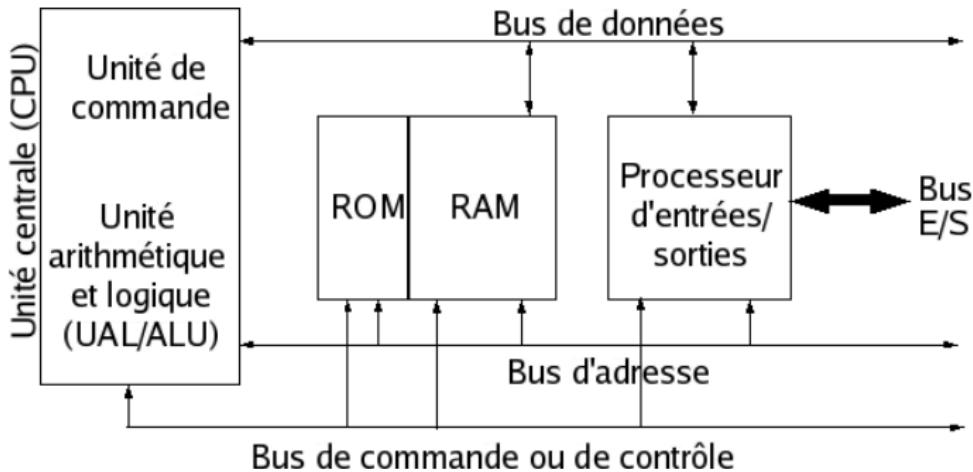
- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- **L'unité d'entrées/sorties**
- Représentation de l'information numérique
- Bases de numération

# L'unité d'entrées/sorties

- L'unité d'entrée/sortie permet à l'ordinateur d'échanger des informations entre les périphériques et - le microprocesseur ou la mémoire -
- Les périphériques sont les éléments extérieurs à l'ordinateur : L'écran, le clavier, l'imprimante, le disque dur...
- Les périphériques sont de trois types : d'entrée, de sortie et d'entrée/sortie

# Le micro-processeur

- Dans un microprocesseur, on distingue trois bus internes :
  - ▶ Bus d'adresse
  - ▶ Bus de données
  - ▶ Bus de commande ou de contrôle



# Les bus

Il existe différents bus d'entrée/sortie :

- ISA (16 bits)
- PCI (32/64 bits)
- AGP (64 bits)
- USB
- IEEE1394 (firewire)
- PCI Express (64 bits)

# Le bus AGP

- Accelerated Graphics Port
- Il est spécialisé pour les cartes graphiques (3D) qui nécessitent une bande passante très importante et des fréquences de bus élevées
- C'est un bus 32 bits à 33 MHz soit 133 Mo/s (AGP)
- Versions : 533 Mo/s (2X), 1,04 Go (4X) et 2,08 Go/s (8X)

# Le bus PCI

- Peripheral Component Interconnect
- C'est le bus standard des PC actuels
- Il est cadencé à 33 Mhz pour une largeur de 32 bits (PCI) et 64 bits (PCI 2.0)
- Il permet de relier des cartes sons, cartes réseaux, carte d'acquisition vidéo...
- PCI-X cadencé à 133 Mhz

# Le bus USB

- Universal Serial Bus, bus série plug and play
- Il permet de connecter jusqu'à 127 périphériques
- Taux de transfert de 12 Mbits/s (USB1) et 480Mbit/s soit 60 Mo/s (USB2)
- Il relie webcams, clés mémoires, imprimantes, scanners... au PC
- Il remplace le bus série classique (RS232) ainsi que le bus parallèle

# Le bus IEEE1394 (firewire)

- Bus série
- Plug and play
- Il autorise la connexion de 63 périphériques par bus et 1023 bus
- Intégré à de nombreux caméscopes numériques
- Taux de transfert très élevé : 12.5 Mo/s (100 Mbps), 25 Mo/s (200 Mbps) voire 50 Mo/s (400 Mbps).

# Le bus PCI Express

- Il tend à devenir le standard du marché
- Bus Plug and Play
- PCI 64 bits
- Taux de transfert : 8 Go/s

# Loi de Moore

- Le nombre de transistors sur une puce de circuit intégré double tous les 18 mois. On devrait atteindre le milliard de transistor vers 2010.
- Dès 1965, Gordon Moore, cofondateur avec Bob Noyce de la firme Intel, constatait que la capacité des microprocesseurs (c'est-à-dire, en fait, le nombre de transistors sur une surface donnée d'un circuit intégré), doublait pendant une période de temps constante. Cette constatation, connue désormais sous le nom de loi de Moore, a continué de se vérifier depuis lors, la durée observée de la période de doublement étant de 18 mois environ. Alors qu'en 1971 le premier processeur d'Intel, le 4004, comportait 2 200 transistors, le Pentium 4 en comporte 10 millions, le processeur Core 2 duo en contient 300 millions. Dès 1996, Intel prévoyait pour 2011 un microprocesseur avec 1 milliard de transistors et une fréquence de 10 Ghz pour une largeur de gravure de 0,07 microns (par comparaison, la largeur de gravure du Pentium 4 de 2 Ghz est de 0,18 microns).

# Changement de programme

En raison de problème d'évacuation de **chaleur** dans les processeurs, Intel et AMD (les deux principaux fabricants de micro-processeurs) ont renoncé à la course à la vitesse.

La technologie s'oriente vers les architectures multi-cœurs :

- Plusieurs cœurs physiques dans un seul et même processeur  $\implies$  VRAI multi-tâche,
- Optimisation de l'utilisation des ressources processeurs,
- Horloge à fréquence limitée (moins de consommation électrique, moins d'échauffement au niveau du processeur).

## En résumé

- L'ensemble des fonctions matérielles des ordinateurs est assuré par des composants électroniques,
- Un microprocesseur ne sait faire que des choses très simples, mais il les fait très vite,
- L'informatique est basée sur le système binaire qui permet le câblage des fonctions logiques,
- L'horloge assure la synchronisation de tous les composants du système informatique.

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- **Représentation de l'information numérique**
- Bases de numération

# Représentation de l'information numérique

## Information numérique

Les informations numériques sont **TOUTES** (textes, images et sons) codées en binaire car c'est le seul langage compris par l'ordinateur.

## Représentation de l'information

Voir le cours de [représentation de l'information](#) pour plus d'information.

# Représentation de l'information numérique

## Information numérique

Les informations numériques sont **TOUTES** (textes, images et sons) codées en binaire car c'est le seul langage compris par l'ordinateur.

## Représentation de l'information

Voir le cours de **représentation de l'information** pour plus d'information.

# Table des matières

## 1 Architecture de l'ordinateur

- Machines virtuelles
- Niveau physique
- Logique binaire, circuits logiques, fonctions câblées
- Types des circuits
- Circuits combinatoires
- Circuits séquentiels
- Architecture de Von Neumann
- Le processeur
- la mémoire
- L'unité d'entrées/sorties
- Représentation de l'information numérique
- **Bases de numération**

# Bases de numération

- Binaire : 2 états 0 ou 1
- Octal : 8 états 0, 1, 2, 3, 4, 5, 6, 7
- Décimal : 10 états 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Hexadécimal : 16 états 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

# Représentation d'un nombre dans une base

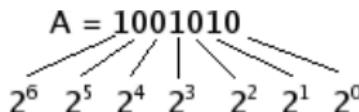
Représenter un nombre dans une base  $b$ , c'est décomposer ce nombre sur des puissances de la base.

On procède donc par divisions successives par la base  $b$  du nombre pour obtenir le résultat :

- $(617)_{10} = 6 * 10^2 + 1 * 10^1 + 7 * 10^0 = 6 * 100 + 1 * 10 + 7 * 1$
- $(21)_3 = 2 * 3^1 + 1 * 3^0 = 2 * 3 + 1 * 1 = (7)_{10}$

# Conversion binaire → décimal

- La représentation binaire est une décomposition d'un nombre décimal sur une base de puissances croissantes de 2.



Bit de  
poids fort

Bit de  
poids faible

$$\begin{aligned}A &= 1 * 2^6 + 0 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 \\A &= 1 * 64 + 0 * 32 + 0 * 16 + 1 * 8 + 0 * 4 + 1 * 2 + 0 * 1 \\A &= 1 * 64 + 1 * 8 + 1 * 2 \\A &= 64 + 8 + 2 \\A &= 74\end{aligned}$$

## Conversion décimal → binaire

- On souhaite transformer le nombre décimal 79 en nombre binaire, on procède par divisions successives par deux, c'est-à-dire par décomposition sur des puissances de deux.

$$\begin{array}{rcl} 79:2 = 39 \text{ reste } 1 & \xrightarrow{\hspace{1cm}} & 1 \\ 39:2 = 19 \text{ reste } 1 & \xrightarrow{\hspace{1cm}} & 1 \\ 19:2 = 9 \text{ reste } 1 & \xrightarrow{\hspace{1cm}} & 1 \\ 9:2 = 4 \text{ reste } 1 & \xrightarrow{\hspace{1cm}} & 1 \\ 4:2 = 2 \text{ reste } 0 & \xrightarrow{\hspace{1cm}} & 0 \\ 2:2 = \underline{1} \text{ reste } 0 & \xrightarrow{\hspace{1cm}} & 0 \\ & & \xrightarrow{\hspace{1cm}} 1 \end{array}$$

- $(79)_{10} \rightarrow (1001111)_2$

# Conversion hexadécimal ↔ binaire

Décimal	Binaire	Hexadécimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

- Le passage de binaire à hexadécimal et d'hexadécimal à binaire est effectué directement en remplaçant les valeurs du tableau binaire en hexadécimal et inversement
- $(2A)_{16} = (00101010)_2 = (42)_{10}$
- $(10100101001)_2 = (529)_{16} = (1321)_{10}$
- $(99)_{10} = (1100011)_2 = (63)_{16}$

# Table des matières

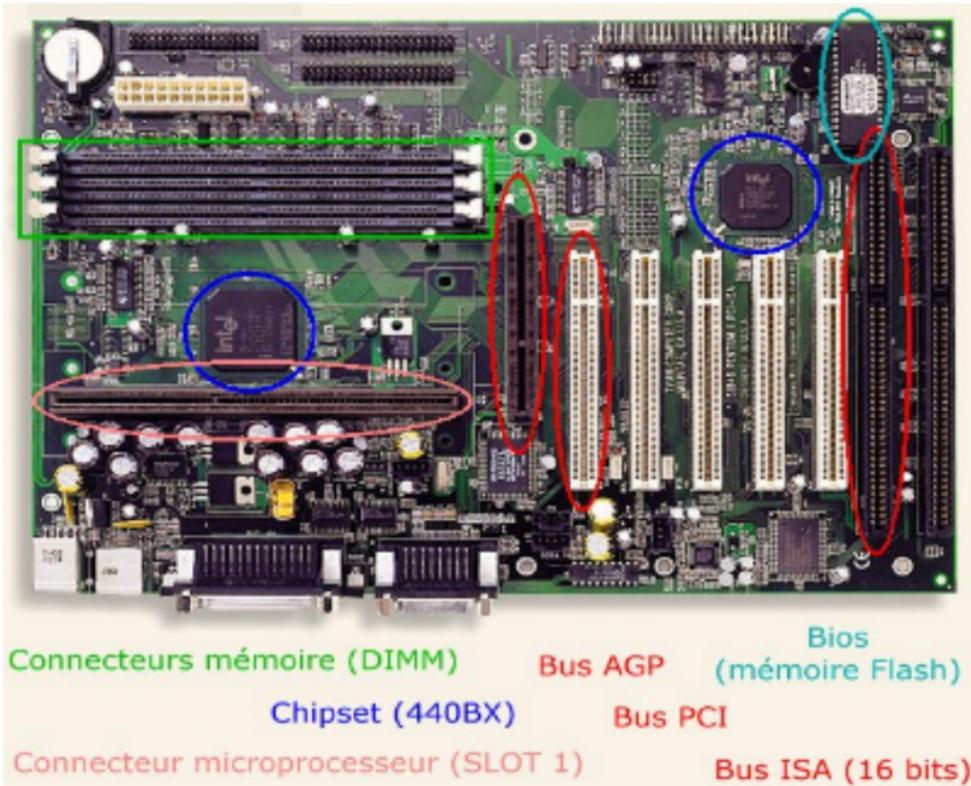
## 2 Architecture PC

# Compatible IBM PC



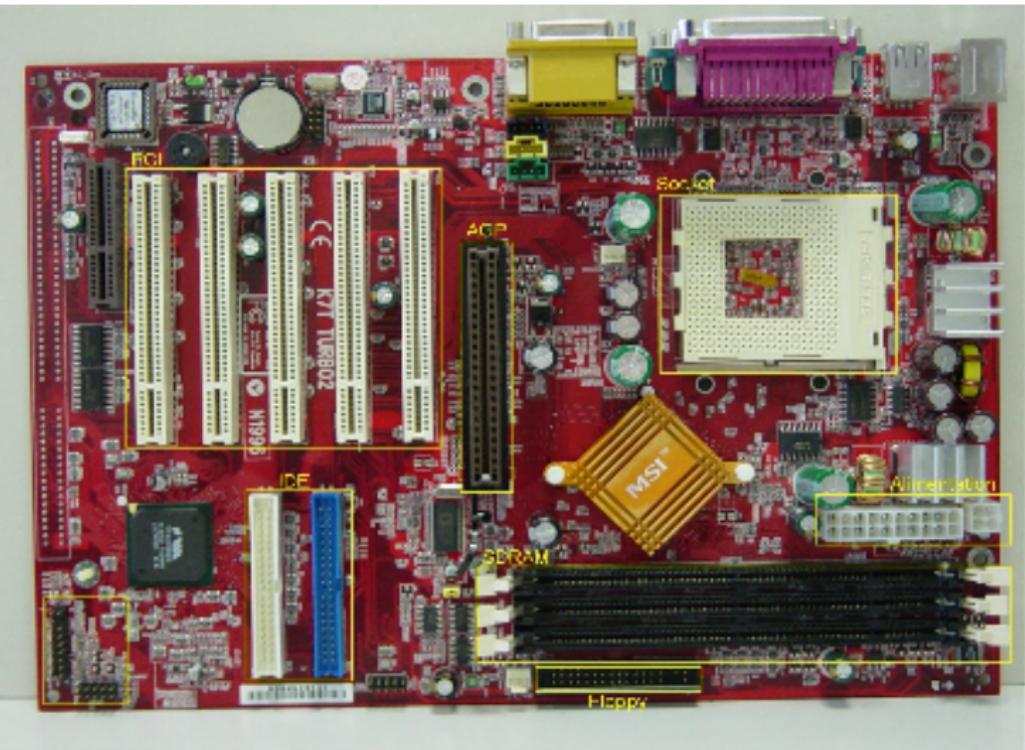
# Carte mère

## Carte mère

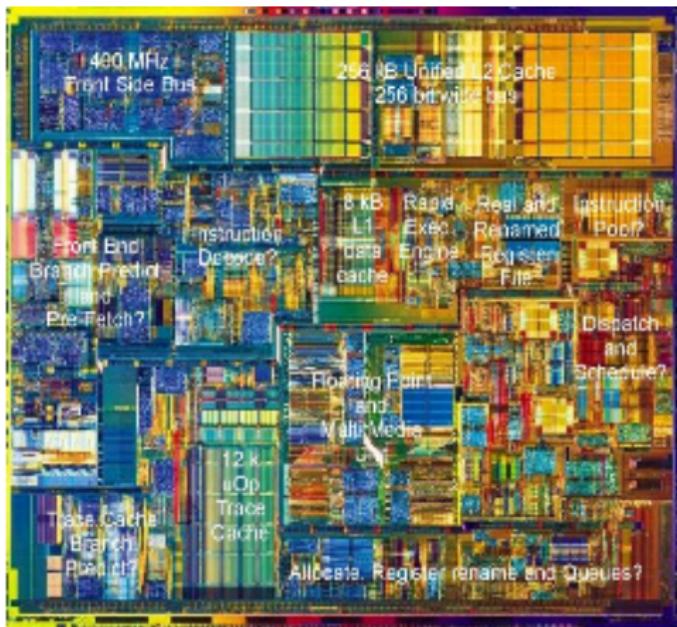


# Carte mère

## Carte mère



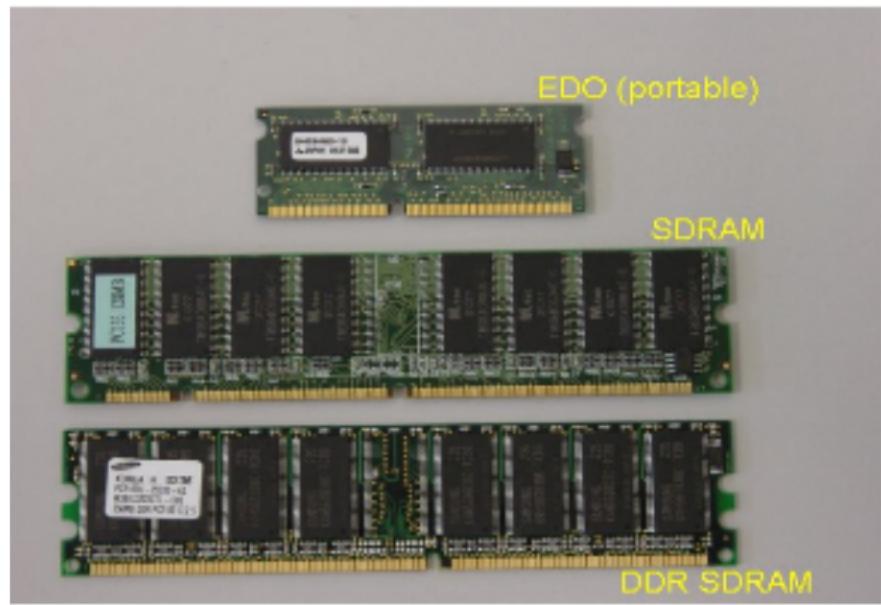
# Microprocesseur



# Barrettes mémoires



# Barrettes mémoires



# Cartes vidéos



PCI



AGP

# Disque dur

- Caractérisé par sa capacité d'environ 120 Go en standard et sa vitesse de rotation 5400, 7200 ou 10000 tours par minute.
- Il existe différentes normes pour les contrôleurs de disques durs : IDE, SCSI...
- La norme la plus répandue est l'IDE
- Il y a un maître et un esclave par nappe IDE
- Le serial-ATA tend à remplacer les disques durs IDE



# Disque dur



En norme IDE, il y a un maître et un esclave par nappe.

# Disques amovibles



# Connectique

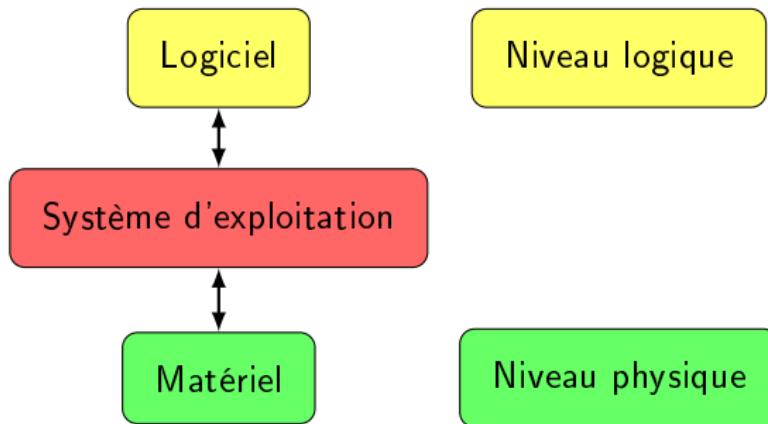


# BIOS



# Généralités

- La gestion du matériel d'un ordinateur est complexe et dépend du matériel à piloter,
- Il est nécessaire d'ajouter une couche entre le matériel et le logiciel,
- Cette couche est le **système d'exploitation**.



# Logiciel

## Définition

Un logiciel est un ensemble de programmes qui permettent d'effectuer des traitements spécifiques (ex : traitement de texte, tableur, logiciel de base de donnée, etc...)

En première approche, un système d'exploitation :

- est un logiciel particulier,
- qui permet de gérer les différents composants de l'ordinateur au moyen de commandes simples.

# Système d'exploitation

## Définition

Le système d'exploitation d'un ordinateur (Operating System) est un ensemble de programme de base qui :

- permettent d'utiliser tous les services disponibles sur une machine (variable en fonction des capacités du matériel),
- En assurant la gestion des travaux, les opérations d'entrées sorties et l'affectation des ressources.

## Définition (suite)

- Un SE est un ensemble de programmes qui :
  - ▶ Exécutent les tâches de base de gestion de l'ordinateur,
  - ▶ Qui mettent des outils à disposition des utilisateurs, des programmeurs et des logiciels.

## En résumé :

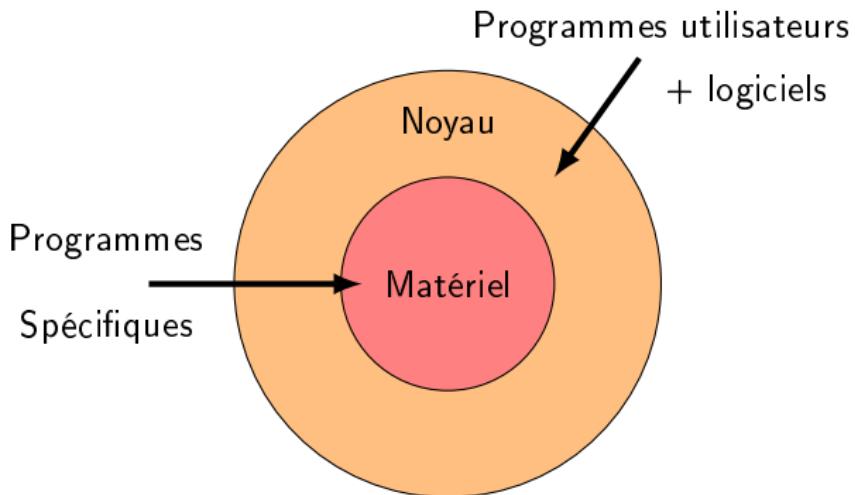
- Un SE permet :
  - ▶ De gérer les ressources matérielles internes,
  - ▶ De présenter une machine virtuelle à chaque utilisateur,
  - ▶ La **machine virtuelle** réalise une abstraction des ressources, elle inclut le noyau qui est l'abstraction la plus basique du matériel.

# Exemples de SE

- Windows (Microsoft),
- MacOS (Apple),
- Linux (GNU),
- UNIX,
- Solaris (SUN Microsystems),
- BeOS,
- ...

# Noyau

- Le noyau englobe le matériel,
- Le matériel est cachée aux utilisateurs,
- Le noyau fournit une machine virtuelle d'abstraction du matériel.



# Gestion des ressources matérielles

## Objectif :

Fournir une interface simple avec le matériel.

## Problème :

- Plusieurs utilisateurs mais une machine unique.
- Comment répartir les ressources pour répondre au mieux aux demandes des utilisateurs ?

## Solution :

- Répartir les ressources entre les divers utilisateurs

Le SE doit assurer cette fonction à plusieurs niveaux :

- Processeurs, mémoire, périphériques (ex. : disques) mais aussi au niveau des travaux soumis par les utilisateurs (processus)
- En pratique, chaque travail reçoit :
  - ▶ Une tranche de la mémoire,
  - ▶ Une fraction de temps de traitement processeur.

# Machine virtuelle

## Objectif :

Cacher la réalité (complexité) de la machine réelle aux utilisateurs

- Contrairement aux apparences, tout utilisateur d'une machine ne dialogue pas directement avec la machine mais avec un intermédiaire, la machine virtuelle
- Cet intermédiaire est un ensemble de programmes du SE qui assurent le rôle d'interface entre l'utilisateur et l'électronique du matériel

- Le SE fournit à chaque utilisateur une machine virtuelle,
- Ainsi chaque utilisateur à l'impression d'avoir la machine à sa seule disposition,
- La machine virtuelle possède plus de fonctionnalités que la machine réelle : langage de programmation, entrées/sorties standardisées.

# Machine virtuelle

- L'architecture de l'ordinateur et le système d'exploitation sont liés, ils sont complémentaires,
- On retrouve le modèle en couches,
- Bas-niveau : électronique, architecture,
- Haut-niveau : Systèmes d'exploitation, applicatifs utilisateurs.

# Super-utilisateur

- Certaines fonctions de la machine virtuelle considérées comme dangereuses sont réservées à un utilisateur spécial : le super-utilisateur (administrateur, root)
- Exemple : le redémarrage d'un serveur, l'installation de pilotes de périphériques...

# Critères de qualité d'un système d'exploitation

- ① Efficacité maximale : Économie en temps et moyen, ressources minimales utilisées, temps de réponse court, multitâches (passage facile d'une tâche à une autre),
- ② Fiabilité : Résistance aux perturbations : répondre à toutes les éventualités logicielles, notion de sécurité (vis-à-vis de l'extérieur du système),
- ③ Souplesse : Différentes architecture matérielle, différentes versions (compatibilité ascendante), personnalisation de l'environnement de travail (bureau),
- ④ Ouverture : Capacité de communiquer avec d'autres systèmes,
- ⑤ Ergonomie : Qualité des dialogues, interface graphique, communication homme/machine.

# Histoire des systèmes d'exploitation

50-60	60-70	70-90	90-...
<b>Programmes</b> Utilisateur/spécialisés	<b>Programmes</b> Utilisateur/spécialisés	Logiciels applicatifs	<b>Services</b>
Matériel	<b>Logiciels</b> <b>Système (SE)</b>	<b>Programmes</b> <b>spécialisés</b>	Logiciels applicatifs
Matériel	Matériel	<b>Logiciels</b> <b>Système (SE)</b>	Programmes spécialisés
Matériel	Matériel	Réseau	Logiciels Système (SE) Services Réseau
Matériel	Matériel	Matériel	Réseau
Matériel	Matériel	Matériel	Matériel

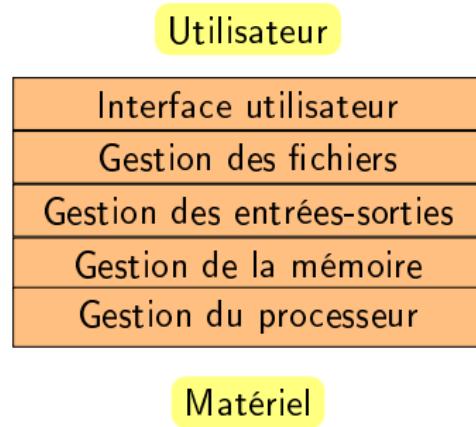
# Principes des systèmes d'exploitation

- Modularité : On réunit les fonctionnalités similaires dans des modules (exemple : son, graphiques...).
- Hiérarchisation : Les éléments du système sont organisés de façon arborescente. Exemple : Structure des fichiers, processus, ...
- Abstraction : Séparation de la logique de fonctionnement de la réalité du travail (pilote de périphérique), icônes du bureau, tâche d'impression, ...

# Architecture d'un système d'exploitation

- Le noyau : gestion du matériel, E/S, processus, mémoire...
- Les services : gestion des fichiers, services réseaux, interface graphique...

Selon le principe d'abstraction, un système d'exploitation est composé de cinq couches fonctionnelles.



# Classement des systèmes d'exploitation

- Degré de complexité
- Nombre de programmes exécutés simultanément
- Nombre de processeur
- Nombre d'utilisateur(s)
- Nombre d'applications
- Nombre d'ordinateurs
- Interaction utilisateur

# Interfaces

On distingue deux interfaces :

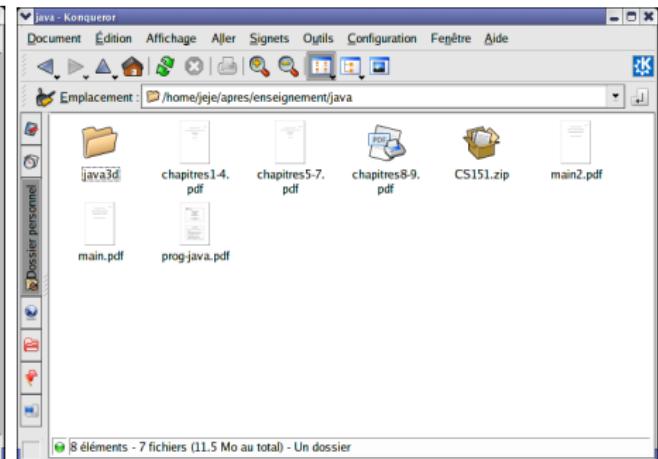
- Les shells : interpréteurs de commandes
  - ▶ Mode texte : CLI (*Common Language Interpreter*),
  - ▶ Mode graphique : GUI (*Graphical User Interface*)
- Les appels systèmes :
  - ▶ Bibliothèques de fonctions : API (*Application Programming Interface*)

# Les shells (coquille)

- Tous les utilisateurs emploient un shell,
- C'est un langage de communication avec l'ordinateur (texte ou graphique),
- Il masque la réalité du système à l'utilisateur.
- Il représente tout ce qui est visible (autorisé) à l'utilisateur.

# Exemples de shells

```
[jeje@localhost ~]$ cd enseignement/java - Terminal - Konsole
Session Édition Affichage Signets Configuration Aide
[jeje@localhost enseignement]$ ls -l
total 48
drwxr-xr-x 2 jeje jeje 4096 nov 25 18:24 divers
-rw-r--r-- 1 jeje jeje 10752 nov 25 20:04 heures.xls
drwxr-xr-x 3 jeje jeje 4096 nov 24 06:04 java
drwxr-xr-x 6 jeje jeje 4096 nov 20 21:16 ri_S1
drwxr-xr-x 4 jeje jeje 4096 nov 24 06:04 se_ar_S1
[jeje@localhost enseignement]$ cd java
[jeje@localhost java]$ ll
total 11816
-rwxr--r-x 1 jeje jeje 1044782 nov 24 06:04 chapitres1-4.pdf
-rwxr--r-x 1 jeje jeje 960128 nov 24 06:04 chapitres5-7.pdf
-rwxr--r-x 1 jeje jeje 1066163 nov 24 06:04 chapitres8-9.pdf
-rwxr--r-x 1 jeje jeje 6864786 nov 24 06:04 CS151.zip
drwxr-xr-x 2 jeje jeje 4096 nov 24 06:04 java3d
-rwxr--r-x 1 jeje jeje 617040 nov 24 06:04 main2.pdf
-rwxr--r-x 1 jeje jeje 735807 nov 24 06:04 main.pdf
-rwxr--r-x 1 jeje jeje 721382 nov 24 06:04 prog-Java.pdf
[jeje@localhost java]$ kghostview main.pdf &
[1] 6480
[jeje@localhost java]$
```



# Appels systèmes

- Les appels au système se font via une API (Application Programming Interface)
- Une API regroupe des fonctions de même nature (gestion des fichiers, primitives graphiques)
- Au niveau des fichiers on parle de bibliothèques systèmes (DLL sous Windows, .a ou .so sous Unix)
- Les appels au système se font depuis l'intérieur des programmes
- Les appels système de bas niveau (lecture disque, transfert réseau, etc.) sont gérés par des interruptions logicielles

# Interruptions

- Lorsqu'une machine est en marche, contrôlée par le SE, le mécanisme d'interruption permet (aux périphériques) de signaler au microprocesseur qu'il doit tenir compte d'événement extérieurs
- Lorsque le microprocesseur est occupé le mécanisme d'interruption permet de suspendre l'exécution du programme et de traiter un événement
- Exemples : une touche est enfoncée, l'imprimante n'a plus de papier, le disque a terminé sa lecture, ...

# Les différents types d'interruption

## Interruptions logicielles :

- Interruptions spéciales destinées à un programme utilisateur pour faire une requête au système

## Interruptions matérielles :

- Générées par les composants pour signaler un événement au processeur

## Portée des appels au système :

- Tous les services ne peuvent pas être appelés par l'utilisateur : Exemple des appels réseau ou disque

# Interface ligne de commandes

- Interpréteur de commandes (différent de compilateur)
- Les commandes internes : opérations de base de la CLI (dir)
- Les commandes externes : opérations annexes du SE ou programmes utilisateur (copy, format, fdisk, etc.)
- Le langage de commande peut s'employer comme un langage de programmation
- Un programme est alors une séquence de commandes placées dans un fichier texte

# Interface graphique

- GUI = graphical user interface
- Exemples : MAC OS, WINDOWS, X-WINDOW
- La GUI encapsule la couche supérieure du SE
- C'est un niveau supplémentaire qui est beaucoup plus complexe que la CLI
- Plus simple d'utilisation, plus ergonomique
- Mais moins directe, moins souple et moins efficace que la CLI pour les actions répétitives

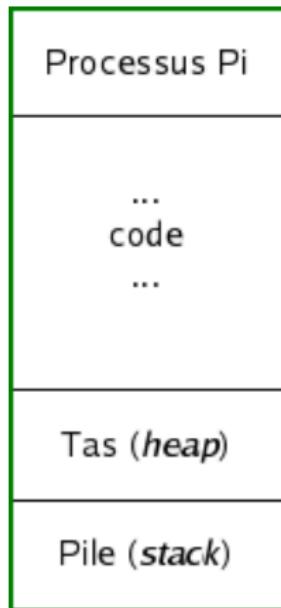
# Interface graphique (2)

- Une interface graphique réalise une abstraction des objets manipulés par l'OS : par exemple, les fichiers, répertoires, périphériques, programmes, commandes sont représentés sous la forme de pictogrammes appelés icônes
- Un système de gestion de zones permet de séparer l'écran en plusieurs parties et de distinguer les programmes actifs
- Les zones sont des fenêtres, on désigne cette approche par le terme multi-fenêtrage
- Les éléments principaux de la GUI sont :
  - ▶ Un système Multi-fenêtrage
  - ▶ Un système de gestion des événements
  - ▶ Une interface d'application standardisée
  - ▶ Une boîte à outils (API)
  - ▶ Un langage de description et de présentation (ex : thèmes de bureau sous windows)

# Gestion des ressources

- Notion de processus : **Un processus est un programme en exécution**
- On parle de multi-programmation pour dire que chaque processus (tâche) reçoit l'usage du processeur pendant une fraction de temps (moins d'une seconde)
- La rapidité de commutation des tâches fait que les utilisateurs ont l'impression d'une simultanéité dans l'exécution du programme

# Notion de processus



Un processus est composé d'une zone mémoire allouée par le système d'exploitation contenant :

- Un code à exécuter (en langage machine),
- Une pile (*stack*) pour le stockage de l'environnement d'exécution,
- Un tas (*heap*) contenant les variables et objets du programme

Chaque processus s'exécute dans son espace mémoire propre,

Ainsi un processus ne peut pas écraser un autre processus en exécution,

On peut interrompre un processus sans entraver le déroulement des autres.

# Création d'un processus

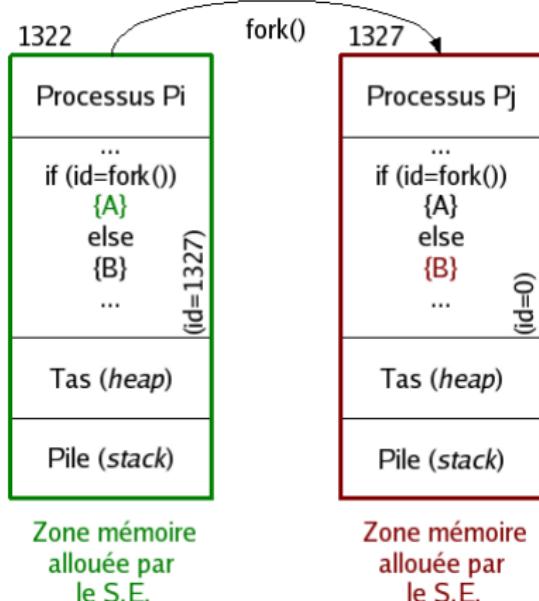
Dans un système d'exploitation, les processus sont créés par copie de l'environnement d'exécution d'un processus existant

La fonction fork() permet de créer un processus

```
#include <stdio.h> // Entrée-sorties standards
#include <unistd.h> // Fonctions sur les processus
#include <sys/types.h> // Types de processus

int main()
{
    pid_t id=0;
    printf ("Bonjour\u00e0\u00e0"); // Texte commun pere/fils
    if (id=fork())
    { printf ("Processus\u00e0 pere\u00e0:\u00e0%d\n", id); } // pere
    else
    { printf ("Processus\u00e0 fils\u00e0:\u00e0%d\n", id); } // fils
    return (0);
}
```

# Création de processus par copie



- Tout processus est créé par recopie à l'aide de la fonction `fork()`.
- Le processus fils créé hérite des propriétés du père.
- C'est la valeur renvoyée par la fonction `fork()` qui permet de savoir si on se trouve dans le père ou dans le fils.
- Les processus s'exécutent en parallèle !

## Les tâches (*threads*)

Une tâche (*thread*) est un programme s'exécutant à l'intérieur d'un processus (même plage mémoire que le processus créateur).

Il peut y avoir plusieurs tâches exécutées en parallèle dans un même processus.

# Création de tâches

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

volatile char theChar = '\0';
volatile char affiche = 0;

void* lire (void* name) {
    do {
        while (affiche == 1) ; //attendre tour
        theChar = getchar();
        affiche = 1; //donner tour
    }
    while (theChar != 'F');
    return NULL;
}

void* afficher (void* name) {
    int cpt = 0;
    do {
        while (affiche == 0) cpt++; //attendre
        printf("cpt=%d , uchar=%c\n", cpt, theChar);
        affiche = 0; //donner tour
    }
    while (theChar != 'F');
    return NULL;
}
```

```
int main (void) {
    pthread_t filsA, filsB;

    if (pthread_create(&filsA, NULL, afficher, "AA"))
    { perror("pthread_create"); exit(-1); }
    if (pthread_create(&filsB, NULL, lire, "BB"))
    { perror("pthread_create"); exit(-1); }

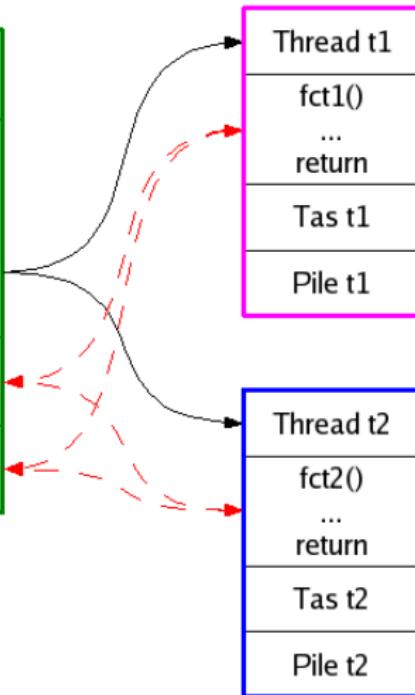
    if (pthread_join(filsA, NULL))
    perror("pthread_join");

    if (pthread_join(filsB, NULL))
    perror("pthread_join");

    printf("Fin du pere\n");
    return (0);
}
```

# Création d'une tâche

1322



- Une tâche (*thread*) est créée par un processus à partir d'une fonction unique (pas de recopie complète),
- La tâche à son propre tas et sa propre pile,
- Mais elle a aussi accès au tas et à la pile du processus qui l'a créée.

# Arborescence des processus

- Arborescence des processus : de façon générale le SE alloue un processus pour l'interface utilisateur (shell) puis d'autres processus (fils du premier pour chaque commande utilisateur ou pour chaque programme lancé)
- Ainsi, il se crée une hiérarchie de processus (père fils) prenant la forme d'un arbre
- Exemple : Word et Excel ainsi que Delphi sont lancés à partir de l'interface graphique ce qui crée trois processus du même père. Si vous imprimez depuis Word, celui-ci crée un nouveau processus (fils) pour exécuter cette tâche

# Ressources

- Ressource non partageable : ne peut être utilisée que par un seul processus à la fois
- Processus en exclusion mutuelle
- Pour accéder à une ressource non partageable, il faut réserver puis libérer la ressource.
- Création d'une file d'attente de processus

# Sémaphores d'exclusion mutuelle

On dispose d'une ressource  $s$  à partager.

Un sémaphore sur  $s$  est composé :

- D'un entier  $e(s)$  qui représente le nombre de places disponibles dans la ressource,
- D'une file d'attente  $f(s)$  pour les processus avant leur accès à la ressource,
- De deux fonction  $P(s)$  pour la réservation et  $V(s)$  pour la libération

$P(s)$	$V(s)$
$e(s) = e(s) - 1$ On a : si $(e(s) < 0)$ alors { état(p)=bloqué entrer(p,f(s)) }	$e(s) = e(s) + 1$ si $(e(s) \leq 0)$ alors { sortir(p,f(s)) état(p)=éligible }

# Gestion des sémaphores

Pour réserver une ressource critique (imprimante, processeur,...), il faut procéder selon le pseudo-code :

$e(s) \leftarrow 1$

...

$P(s)$  (réservation)

<section critique>

$V(s)$  (libération)

...

À l'aide des sémaphores, on peut aussi proposer des rendez-vous entre processus :

$e(s) \leftarrow 0$

Processus 1

...

1<sup>er</sup> travail

$V(s)$

...

Processus 2

...

$P(s)$

2<sup>ème</sup> travail

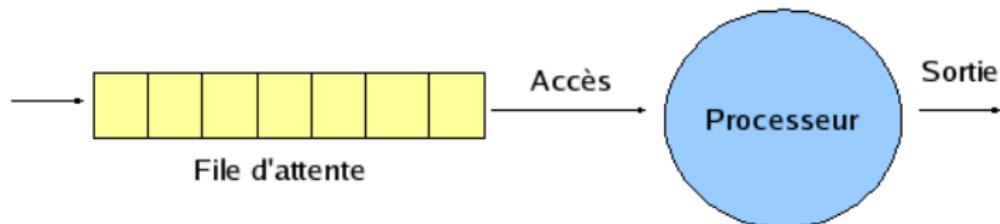
...

# Gestion du processeur

- Problème : un grand nombre de processus se partagent un seul processeur
- Il faut définir une politique d'accès au processeur (ordonnancement ou scheduling)
- Le mécanisme d'ordonnancement définit les critères selon lesquels les processus ont accès au processeur

# Ordonnancement

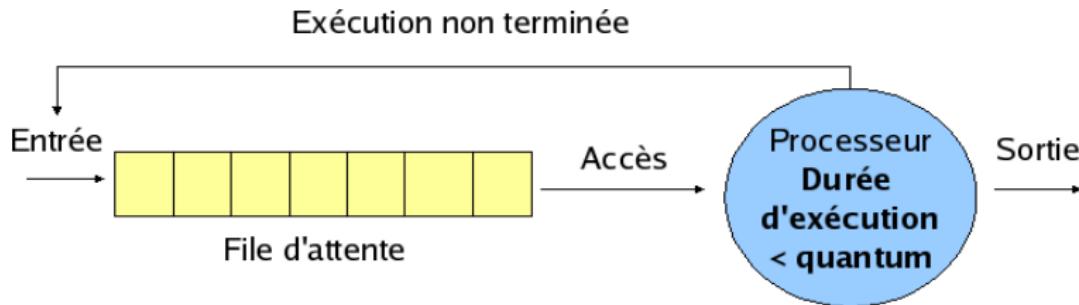
- Une politique d'ordonnancement est nécessaire lorsque plusieurs utilisateurs (demandeurs) se partagent une ressource essentielle
- Les demandeurs se retrouvent dans une file d'attente
- FIFO (*First In First Out*)



Problème avec cette solution : un processus bloqué prend tout le temps processeur et paralyse l'ordinateur.

# Politique d'ordonnancement

Solution : le tourniquet



Il y a trois principaux principes d'ordonnancement :

- Ordre d'arrivée : premier arrivé premier servi (FIFO, *First In First Out*)
- Urgence : le premier servi est celui dont la durée est la plus faible
- Importance : le premier servi est celui dont l'accès à la ressource est le plus important (notion de priorité)

# Le tourniquet

Principe : vider les processus qui s'attardent trop dans le processeur

Le système passe d'un processus à un autre

Un processus est remis en file d'attente dès que sa durée d'occupation du processeur dépasse une durée prédéfinie : quantum de temps

L'efficacité du système dépend de la valeur du quantum

- Trop petit, la machine perd du temps à changer de contexte (*swapping*)
- Trop long, les performances du système chutent

# Gestion de la mémoire

Problème : Plusieurs dizaines de processus doivent se partager une mémoire commune, la RAM

- La capacité de la mémoire centrale est restreinte
- Si les processus sont nombreux, ils vont occuper plus de place que ne peut leur offrir la mémoire centrale

# Mémoire virtuelle

- La mémoire secondaire (disque dur) est beaucoup plus importante (40Go) + 100 fois la mémoire centrale
- Le SE propose un mécanisme de mémoire virtuelle qui consiste à utiliser le disque dur pour simuler de la mémoire réelle

# Stratégie d'allocation et de libération

- Les opérations associées à la mémoire sont :
  - ▶ Opération sur les mots : lecture ou écriture
  - ▶ Opérations sur les zones : allocations ou libération
- Dans certains cas on ajoute les opérations :
  - ▶ Extension ou diminution de zones
  - ▶ Division d'une zone (partition)
- Pour un espace donné on peut choisir deux modes d'allocation :
  - ▶ Allocation contigüe consiste à placer la totalité d'un programme à des adresses consécutives
  - ▶ Allocation non-contigüe consiste à fractionner le programme et à placer les différents fragments à des adresses dispersées

# Notion de bloc

- On appelle bloc un groupe d'un nombre fixe de mots (16, 256, 512, 1024, 2048, 4096, ...)
- La taille des blocs est fixée par les caractéristiques du matériel
- Les zones allouées comportent toujours un nombre entier de blocs

Exemple d'allocation et de libération de blocs :

Mémoire de 20 blocs après allocations successives

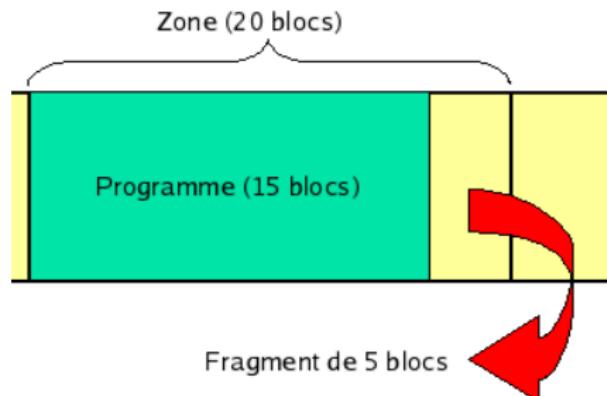
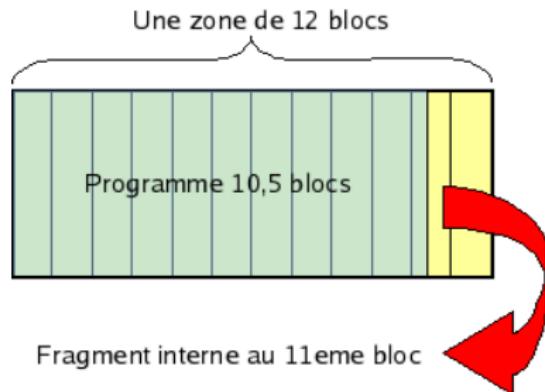
P1 3	P2 4	P3 2	P4 3	P5 2	P6 1	P7 3	2 libre
---------	---------	---------	---------	---------	---------	---------	------------

Libération de trois zones qui étaient allouées

P1 3	4 libre	P3 2	3 libre	2 libre	P6 1	P7 3	2 libre
---------	------------	---------	------------	------------	---------	---------	------------

# Fragmentation des blocs alloués

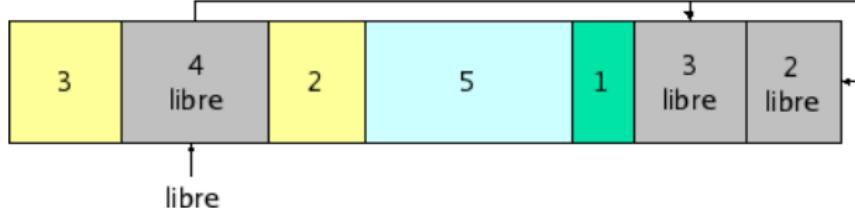
Lors de l'allocation de blocs, le système d'exploitation doit minimiser le nombre de blocs donnés à un processus afin d'optimiser les ressources en blocs mémoire.



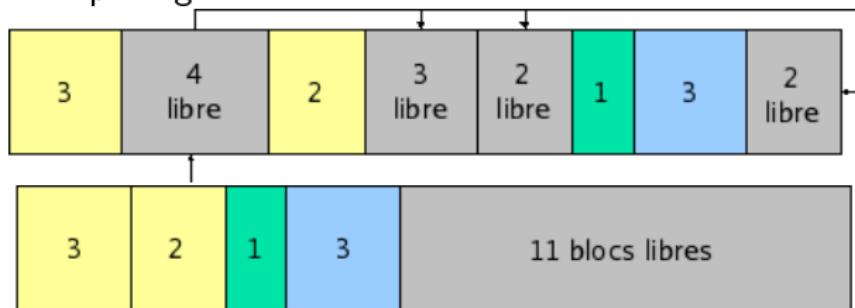
Le système d'exploitation doit, dans la mesure du possible, détecter et réduire les fragmentations internes et externes.

# Exemple d'allocation contigüe

Exemple 1 : demande de 5 blocs satisfaite en fusionnant 3+2

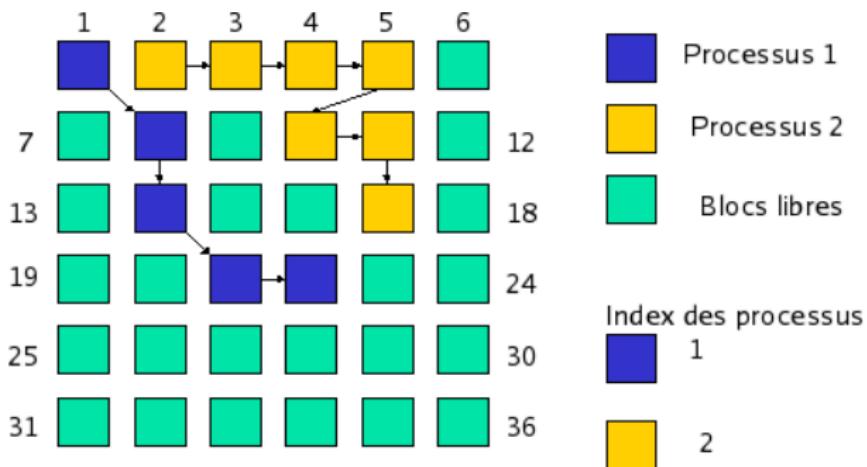


Exemple 2 : demande de 11 blocs non satisfaite car il n'existe pas 11 blocs consécutifs → compactage



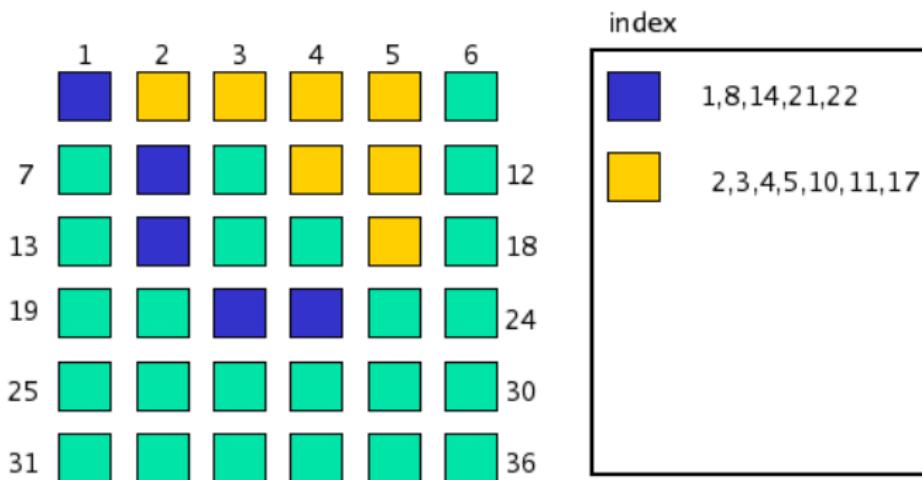
# Exemple d'allocation non-contigüe par chaînage

Les zones mémoires sont allouées et chaînées les unes aux autres.



# Exemple d'allocation non-contigüe par indexation

Les zones mémoires sont allouées et leur état est conservé dans une table d'index.



## Échange de zones mémoire (*swapping*)

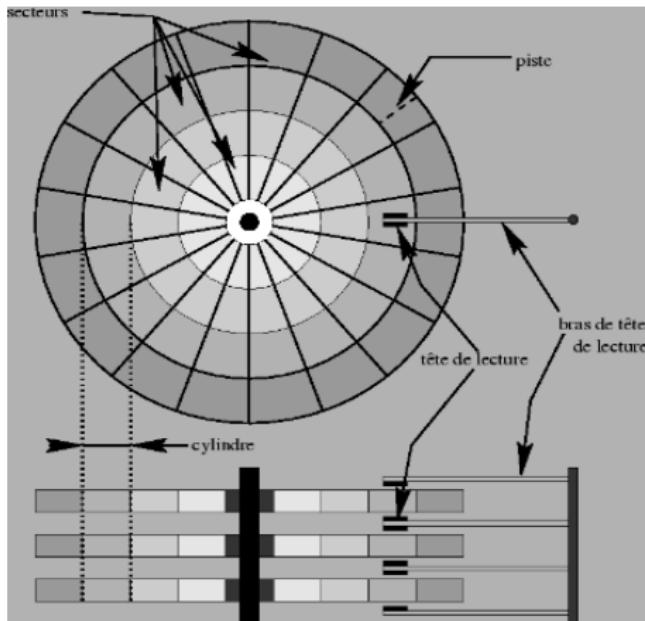
Le mécanisme déchange de zone swapping consiste à échanger des zones mémoires entre la mémoire principale (RAM) et la mémoire secondaire (disque dur). Le *swapping* est très long, temps d'accès entre mémoire et disque, même rapport que d'une seconde à un jour → Il faut éviter à tout prix l'échange de zones mémoires !

# Gestion du disque dur

La gestion des fichiers recouvre deux activités :

- Une gestion statique : taille, date de création, localisation
- Une gestion dynamique : ouverture, gestion des adresses de zones allouées sur le disque

# Disque dur



- Formatage : le disque est découpé en zones logiques appelées secteurs regroupés sur des pistes
- L'ensemble des pistes de même numéro sur différents plateaux forme un cylindre
- Le premier secteur contient un tableau T qui s'étend sur la globalité de la première piste et qui associe à chaque nom de fichier le numéro du secteur qui le contient qui permet d'ouvrir un fichier par son nom

# Notion de fichier

- Au niveau logique l'utilisateur perçoit des fichiers organisés dans des répertoires
- Cette structure arborescente est souple et conserve la même structure quelque soit le niveau (réursive)
- Les répertoires sont considérés comme des fichiers particuliers : cette abstraction permet d'unifier les procédures de traitement de fichier avec celle de traitement de répertoire
- Deux informations sont fondamentales :
  - ▶ Le nom du fichier
  - ▶ L'adresse du premier bloc sur le disque
- On ajoute en général :
  - ▶ Les droits d'accès ou protection
  - ▶ La date de création
  - ▶ La date et heure de la dernière modification
  - ▶ La taille du fichier
  - ▶ Le propriétaire

# Organisation

- Les fichiers sont découpés en blocs non-contigus sur le disque.
- Par contre les données à l'intérieur du fichier sont contiguës.
- Sous UNIX : tout est géré à partir d'une seule et même racine ("/")
- Les utilisateurs sont tous possesseurs d'une branche dans laquelle il sont maîtres (home directory), ils peuvent la développer : créer de nouveaux fichiers et de nouveaux répertoires
- Sous Windows : tout est géré à partir d'une seule et même racine (\) par unité (disquette A :, disque dur C :, ...)
- Les utilisateurs ont des droits définis à partir des définitions de l'administrateur. Possibilité de développer l'arborescence : créer de nouveaux fichiers et de nouveaux répertoires (en fonction de leurs droits).

# système de fichiers

- Sous Windows ces informations sont regroupées dans l'arborescence au niveau du répertoire dans lequel figure le fichier
- Un déplacement de fichier implique alors une recopie de ces informations
- Sous Unix ces informations sont enregistrées de façon indépendante dans de petites tables descriptives de fichier appelées i-nodes ou i-noeuds
- Un déplacement sous UNIX : un déplacement du i-node

# Fichiers physiques, fichiers logiques

Fichier physique :

- Ensemble de données stockées réellement sur le disque.

Fichier logique :

- Vue particulière d'un fichier
- Exemple : raccourci Windows ou lien Unix.

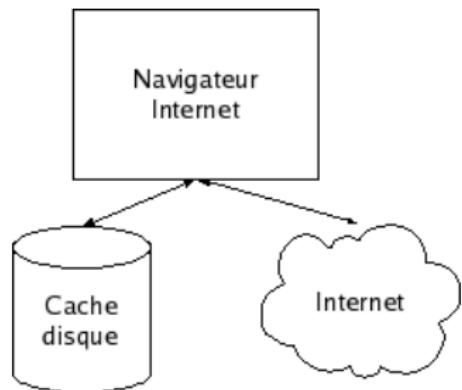
# Architectures multi-cœurs

- Il est techniquement impossible de construire un processeur à 10GHz à cause d'un problème de **dissipation thermique** : le processeur surchauffe,
- Les nouvelles architectures des processeurs possèdent plusieurs cœurs (**architecture multi-cœurs**) à une fréquence volontairement basse.

On a donc un gain de vitesse non pas par augmentation de la fréquence du processeur, mais par exécution simultanée des applications sur plusieurs cœurs (on réalise un vrai **parallélisme** par opposition au temps partagé d'un système mono-processeur).

# Mémoire cache

- Un cache est une zone de stockage locale d'information provenant d'une source, de taille réduite mais dont l'accès est beaucoup plus rapide que l'accès direct à la source.



- À chaque lecture d'une page, le navigateur stocke les images et objets contenus dans cette page dans une zone du disque dans laquelle le contenu est indexé,
- À chaque visite d'une page, le navigateur regarde dans le cache si les objets présents sur cette page sont disponibles, si oui, il y a un gain de temps énorme car les objets sont téléchargés directement depuis le cache local, sans passer par une ligne Internet lente.

# La mémoire cache des microprocesseurs

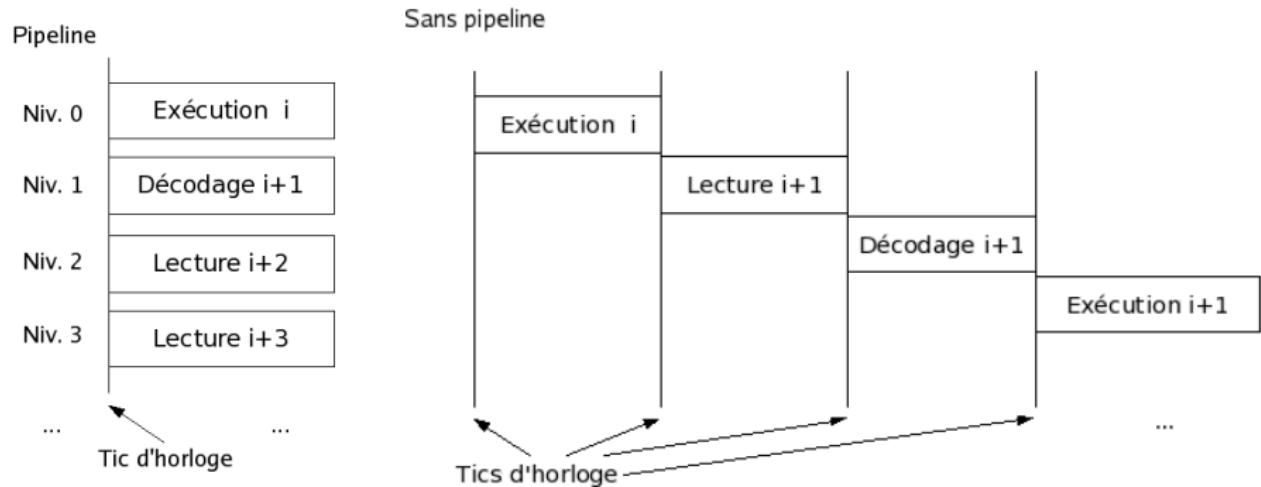
- Un microprocesseur possède trois mémoires caches :
  - ▶ La mémoire **cache interne** (de l'ordre de 8 à 32ko), interne au microprocesseur,
  - ▶ La mémoire **cache externe de premier niveau** (*Level 1, L1*) de l'ordre de 64 à 256ko,
  - ▶ La mémoire **cache externe de second niveau** (*Level 2, L2*) de l'ordre de 512ko à 4Mo.

Lors de l'exécution d'une instruction, le processeur cherche d'abord la zone mémoire dans son cache interne, puis dans son cache de niveau 1 et enfin dans son cache de niveau 2 avant de passer par le bus d'entrée/sortie pour récupérer la valeur en mémoire RAM standard.

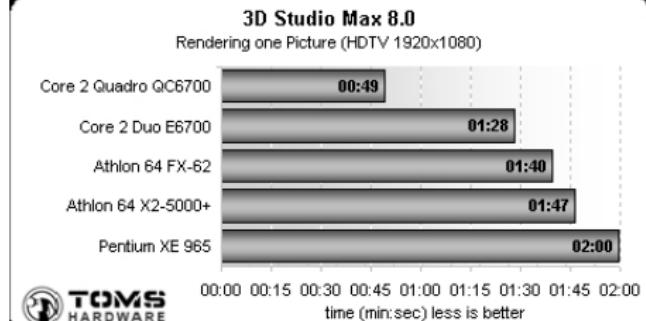
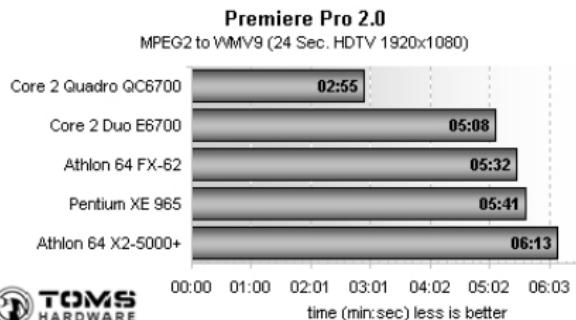
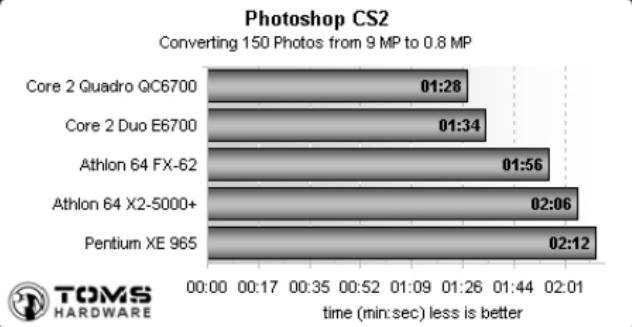
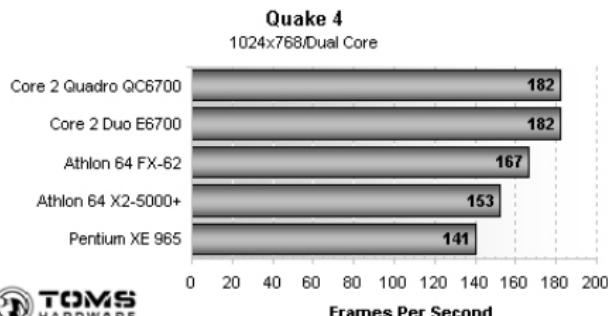
# Le pipeline (1/2)

- Le **pipeline** correspond à l'exécution de plusieurs instructions micro-codes dans le processeur de façon simultanée,
- À chaque cycle d'horloge, on exécute l'instruction  $i$ , on décode l'instruction  $i+1$ , on charge l'instruction  $i+2$ , ...
- On a ainsi une lecture, un décodage et une exécution pendant le même cycle d'horloge.

## Le pipeline (2/2)



# Intel Core 2 Duo contre AMD Turion 64 X2 (1/2)



# Intel Core 2 Duo contre AMD Turion 64 X2 (2/2)

Modèle de processeur	Intel Core 2 Duo	AMD Athlon 64 FX-62
Support du processeur	775	AM2
Fréquence du processeur	2660 MHz	2800 MHz
Fréquence du bus	FSB 1066	FSB 400
Finesse de gravure	0,065 Micron	0,09 Micron
Nom du core	Conroe	Windsor
Cache L1	64 ko	256 ko
Cache L2	4 Mo	2048 ko
Architecture	Intel Core	AMD K8
Nombre de core	2	2
MMX	oui	oui
3D Now !	non	oui
SSE 1, 2 et 3	oui	oui
SSE4	oui	non
Intel Speedstep	oui	non
EM64T	oui	non
AMD64	non	oui

# Intel Core 2 Duo (1/2)

Avant	Maintenant
Performance = fréquence	Performance = IPC ( <i>Instructions per clock cycle</i> )
Consommation d'énergie = peu importante	Consommation optimisée
Mono-cœur	multi-cœurs

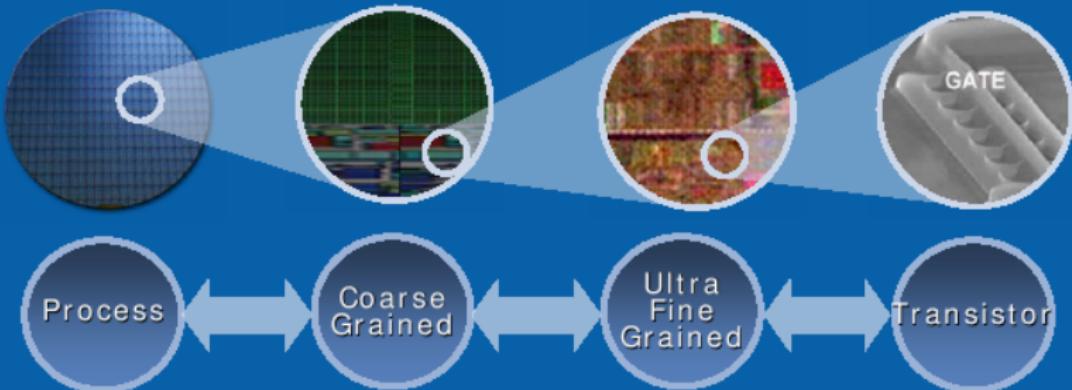
Buts :

- ↗ Performances,
- ↘ Baisse de la consommation électrique (↘ température),
- Vitesse d'horloge.

## Intel Core 2 Duo (2/2)

- Vrai 64 bits (registres, adresses, données),
- 128 bits (instructions SSE),
- Pipeline à 14 niveaux,
- Endommagement des composants électroniques non sollicités,
- Optimisation et fusion des opérations,
- Mémoire cache de second niveau partagée entre les cœurs.

# Intel® Intelligent Power Capability



- 65nm
- Strained Silicon
- Low-K Dielectric
- More Metal Layers

- Aggressive Clock Gating
- Enhanced Speed-Step

- Low VCC Arrays
- Blocks Controlled Via Sleep Transistors

- Low Leakage Transistors
  - Sleep Transistors

Energy

ADVANTAGE

- Mobile-Level Power Management
- Energy Efficient Performance

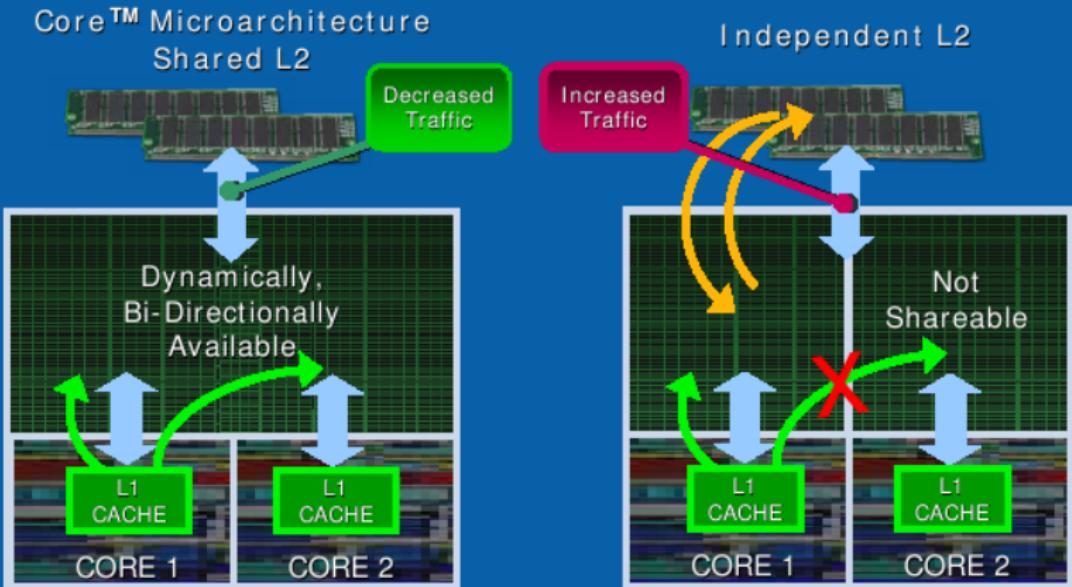
Intel Developer  
FORUM

\*Graphics not representative of actual die photo or relative size



# Intel® Advanced Smart Cache

## DYNAMIC L2 CACHE USAGE



Perf ↑

Energy ↓

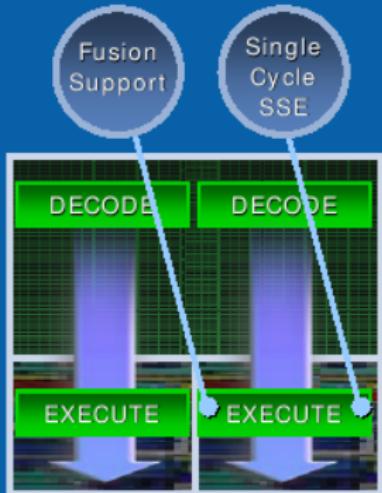
### ADVANTAGE

- Higher Cache Hit Rate
- Reduced BUS Traffic
- Lower Latency to Data

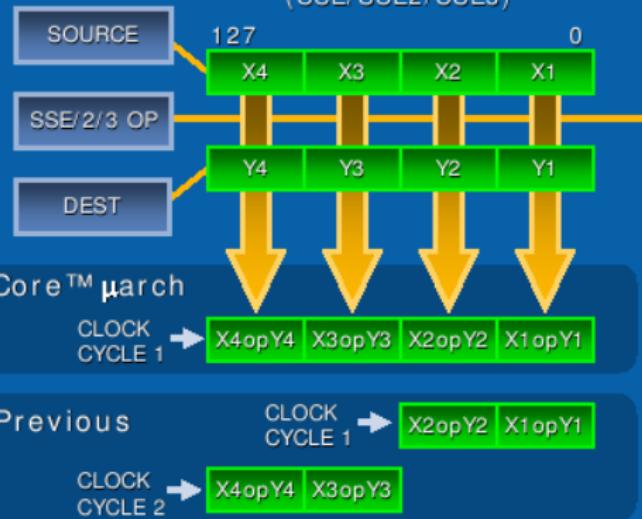
# Intel® Advanced Digital Media Boost

## Single Cycle SSE

In Each Core



SSE Operation  
(SSE/SSE2/SSE3)



Perf ↑

Energy ↓

ADVANTAGE

- Increased Performance
- 128 bit Single Cycle in each core
- Improved Energy Efficiency

Intel Developer  
FORUM

\*Graphics not representative of actual die photo or relative size

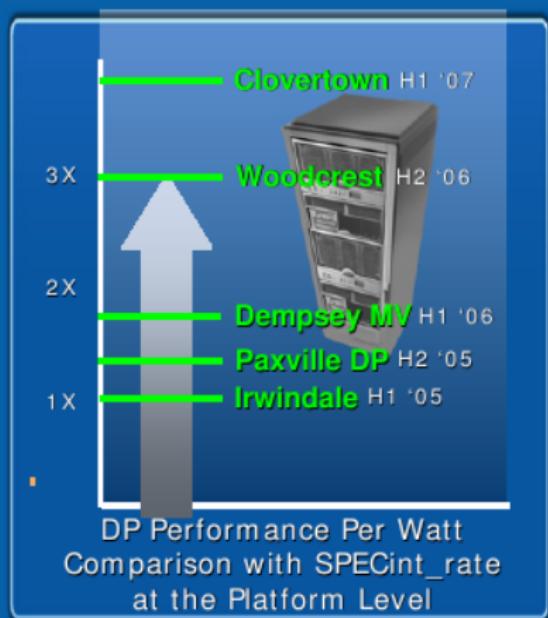
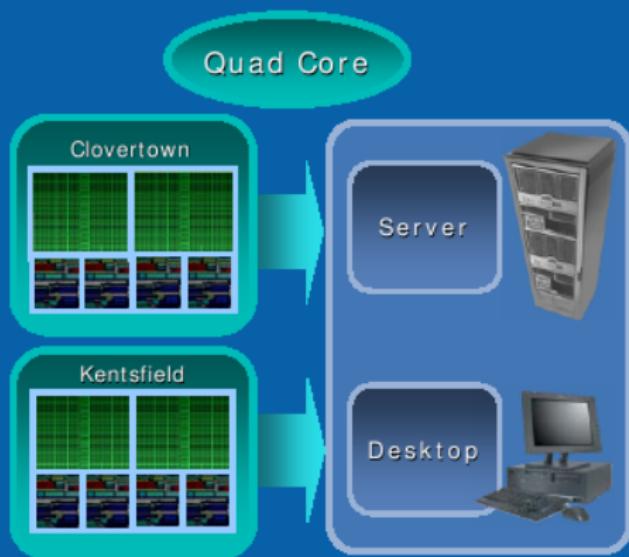


# Core™ Microarchitecture Advances With Quad Core

Energy

Efficient

Performance



Source: Intel®

Intel Developer  
FORUM

\*Graphics not representative of actual die photo or relative size



# Roadmap AMD

## AMD Client Processor Roadmap: 2008-2011

Segment	2008	2009	2010	2011 NEW
Enthusiast Desktop	Agena 4 cores 4M cache, DDR2	Deneb 4 cores 8M cache DDR2/3		Orochi > 4 cores > 8M cache DDR3
Mainstream Desktop		Propos 4 Core 2M cache DDR2/3		Llano *APU 4 cores 4M cache DDR3 GPU
Mainstream Notebook	Griffin 2 cores 2M cache DDR2	Caspian NEW 2 cores 2M cache DDR2	Champlain NEW 4 cores 2M cache DDR3	
Ultraportable		Conesus NEW 2 cores 1M cache DDR2	Geneva NEW 2 cores 2M cache DDR3	Ontario *APU 2 cores 1M cache DDR3 GPU
Mini-Notebook		BGA	BGA	BGA
* Accelerated Processing Unit				
65nm process		45nm process		32nm process

# Applications clients/serveurs

- Dans les débuts de l'informatique, les applications étaient situées sur un ordinateur central très puissant (**le serveur**) et de nombreux terminaux (**les clients**) se connectaient au serveur pour utiliser ses ressources.
- À partir de l'IBM PC, **l'ordinateur personnel** a repris le dessus sur le modèle client/serveur,
- Aujourd'hui, avec le progrès des débits sur les réseaux, la tendance est de proposer des services en mode client/serveur.

# Services collaboratifs

- De nouveaux services réseaux permettent de partager les connaissances et les applications,
- Exemple de services disponibles :
  - ▶ Le Wiki : encyclopédie multilingue, libre et ouverte, modifiable par tous "<http://fr.wikipedia.org>" ,
  - ▶ Delicious : marque-page Internet collaboratif "<http://del.icio.us>" ,
  - ▶ Les services Google, "<http://www.google.com/intl/en/options>" :
    - ★ Earth : vues satellites de la Terre,
    - ★ CodeSearch : codes de programmes dans tous les langages : C, C++, Java, Lisp, ...
    - ★ Gmail : service de messagerie électronique (1 Go),
    - ★ Calendar : calendrier partageable avec d'autres utilisateurs,
    - ★ Writely : traitement de textes en ligne,
    - ★ Spreadsheet : tableur en ligne,
    - ★ ...

[accueil](#)[discussion](#)[voir le texte source](#)[historique](#)[Créer un compte ou se connecter](#)

Vos dons permettent à Wikipédia de continuer à exister ! Merci de votre soutien.

## Bienvenue sur Wikipedia, encyclopédie librement distribuable que chacun peut améliorer

407 765 articles en français, plus de 5 millions dans 250 langues



### Recherche et consultation

[Articles de qualité](#) • [Catégories](#) • [Index](#) • [Liste des listes](#) •  
[Portails thématiques](#) • [Sélections](#)



### Participation et communauté

[Accueil des nouveaux arrivants](#) • [Premiers pas](#) •  
[Participez aux projets](#) • [Poser une question](#) • [Livre d'or](#)



### Arts

[ameublement](#) • [Architecture](#) • [Art contemporain](#) •  
[Arts du spectacle](#) • [Bande dessinée](#) • [Cinéma](#) • [Littérature](#) •  
[Musique](#) • [Peinture](#) • [Photographie](#) • [Sculpture](#)



### Sciences de la Terre

[Agriculture et Agronomie](#) • [Écologie](#) • [Géographie](#) •  
[Géologie](#) • [Hydrologie](#) • [Liste des pays](#) • [Météorologie](#) •  
[Monde maritime](#) • [Montagne](#) • [Sylviculture](#)



### Vie quotidienne et loisirs

[Aquariophilie](#) • [Bricolage](#) • [Gastronomie](#) • [Jardinage](#) •  
[Jeux](#) • [Jeu vidéo](#) • [Mode](#) • [Numismatique](#) • [Œnologie](#) •  
[Philatélie](#) • [Sexualité](#) • [Sport](#) • [Télévision](#) • [Tourisme](#)



### Sciences humaines et sociales

[Anthropologie](#) • [Archéologie](#) • [Droit](#) • [Économie](#) •  
[Généalogie](#) • [Histoire](#) • [Information et bibliothèques](#) •  
[Langues](#) • [Philosophie](#) • [Psychologie](#) • [Sociologie](#)



### Société

[Culture populaire](#) • [Défense et sécurité](#) •  
[Développement durable](#) • [Éducation](#) • [Femmes](#) •  
[Humanitaire](#) • [Minorités](#) • [Politique](#) • [Presse](#) • [Syndicalisme](#)



### Sciences exactes et naturelles

[Astronomie](#) • [Biochimie](#) • [Biologie](#) • [Botanique](#) • [Chimie](#) •  
[Cryptologie](#) • [Logique](#) • [Mathématiques](#) • [Médecine](#) •  
[Pharmacologie](#) • [Physique](#) • [Zoologie](#)



### Religions et croyances

[Athéisme](#) • [Bouddhisme](#) • [Christianisme](#) • [Ésotérisme](#) •



### Téchniques

[Astronautique](#) • [Électricité et électronique](#) • [Énergie](#) •

# Google Calendar

Google Agenda tous mes services » J.landre@laposte.net | Paramètres | Aide | Déconnexion

**Google** Agenda BETA

[Créer un événement](#)

Rechercher dans mes agendas [Afficher les options de recherche](#)

Aujourd'hui 11 - 17 déc. 2006

lun. 11 décembre 2006, 11:00 - 12:00

Objet :   
par exemple, cinéma avec Stéphanie

[Créer un événement](#) [modifier les détails de l'événement](#)

Agendas

Mes agendas +  Jérôme Landré

Autres agendas +

Rech. d'agendas publics [Ouvrir les agendas](#)

10:00

11:00

12:00

13:00

14:00

15:00

16:00

17:00

jeu. 14/12 ven. 15/12 sam. 16/12 dim. 17/12

08:00 TP RIF (2) - Gr.D

11:00 TP RIF (2) - Gr.C

14:00 TP RIF (1) - Gr.A

15:30 Cours SEA

This screenshot shows the Google Calendar interface. On the left, there's a sidebar with a calendar for December 2006, a section for 'Agendas' (including 'Mes agendas' and 'Autres agendas'), and a link to 'Ouvrir les agendas'. The main area displays a weekly view from Monday, December 11, to Sunday, December 17, 2006. A modal dialog box is open over the calendar for creating a new event on Monday at 11:00. The dialog has fields for 'Objet' (Subject) and a placeholder text 'par exemple, cinéma avec Stéphanie'. Below these are buttons for 'Créer un événement' and 'modifier les détails de l'événement'. The weekly view shows several scheduled events: 'TP RIF (2) - Gr.D' at 08:00 on Thursday, 'TP RIF (2) - Gr.A' at 11:00 on Friday, 'TP RIF (1) - Gr.A' at 14:00 on Saturday, and 'Cours SEA' at 15:30 on Sunday. The days of the week are labeled at the top: 'Jour', 'Semaine', 'Mois', '4 jours suivants', and 'Mon planning'. The date labels 'jeu. 14/12' through 'dim. 17/12' are also present.

# Google Docs (Writely)

Google Docs & Spreadsheets

j.landre@laposte.net | Docs Home | Help | Sign out

Voice edited on December 11, 2006 10:29 PM by J

[Save](#) [Save & close](#) [Discard changes](#)

[File](#) [Edit](#) [Insert](#) [Revisions](#) [Edit HTML](#)

[Preview](#) [Print](#) [Email](#) [Collaborate](#) [Publish](#)

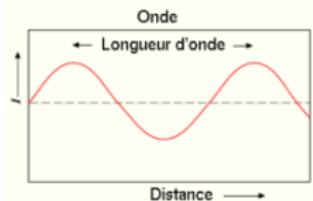
[Image](#) [Link](#) [Comment](#) [Table](#) [Bookmark](#) [Separator](#) [Special character](#)

## Services Google : WRITELY

Temps (s)	Longueur d'onde (nm)
1	450

Tableau 1 : évolution des longueurs d'ondes en fonction du temps

Voici un exemple de texte. L'image ci-dessous a été insérée sans problème...



No one else is editing this document. [Add collaborators](#)

[Check spelling](#) ▾

# Web 2.0

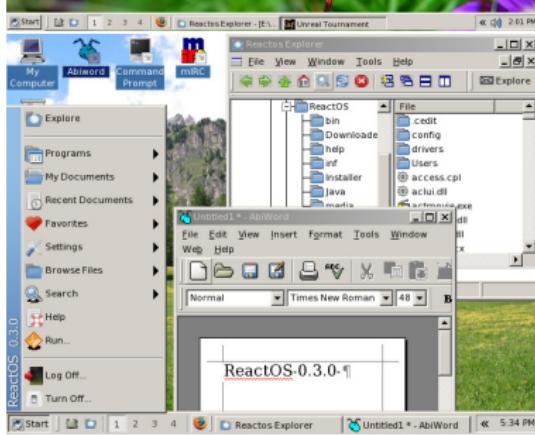
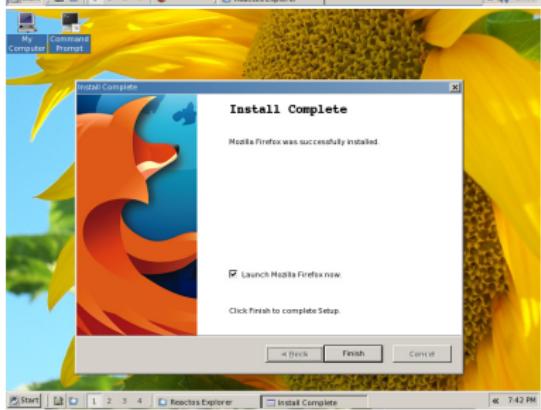
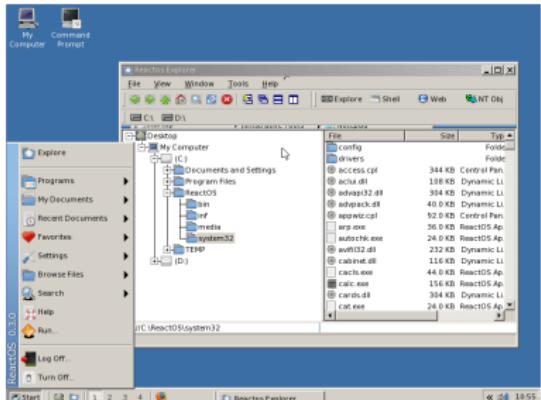
- Depuis quelques temps, on parle du **web 2.0**,
- Ce terme ne cache pas une révolution du web, mais un ensemble de technologies qui permettent une meilleure utilisation du web par l'orientation "contenu" des pages :
  - ▶ XML : **eXtended Markup Language** pour la description de l'information contenue dans les pages,
  - ▶ XSLT : **XML Stylesheet Language Transformation**, qui permet de générer les pages webs dynamiquement à partir du contenu XML,
  - ▶ XHTML : HTML réécrit en XML,
  - ▶ CSS : **Cascading Style Sheets**, les feuilles de styles en cascades qui autorise la séparation de l'information (fond) de la façon de la représenter (forme),
  - ▶ DOM : **Document Object Model**, qui spécifie la façon d'accéder aux balises XHTML à partir d'un langage de script,
  - ▶ AJAX : **Asynchronous Javascript with XML**, qui permet de ne charger qu'une partie de page web et pas la page entière.

# Les SE spécifiques 1/3

Il existe des systèmes d'exploitation (SE) destinés à des usages particuliers :

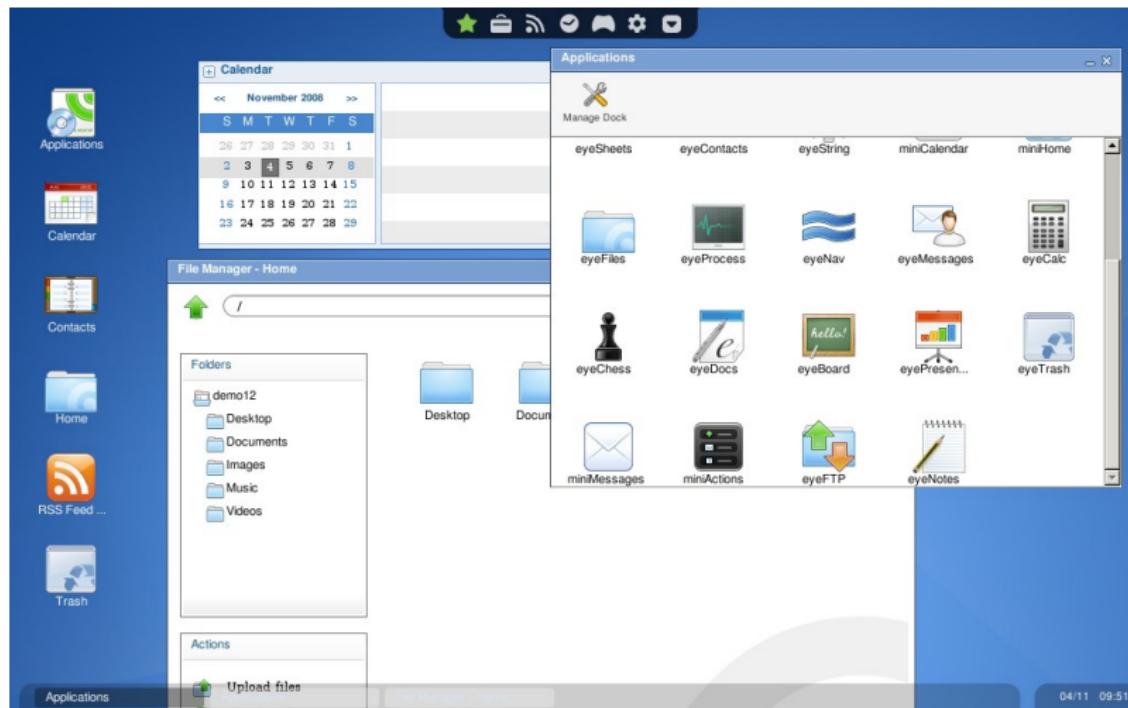
- SE temps réel (RTLinux,...),
- SE embarqués (Embedded Windows, Linux Embedded),
- SE pour tabletPC (Windows TabletPC edition),
- SE alternatifs : Linux, FreeBSD, OpenBSD, NetBSD, ...
- SE de remplacement : ReactOS (Windows sans windows),
- SE virtuels (ou bureau virtuel, dans un navigateur),
- ...

# Les SE spécifiques 2/3 - ReactOS



# Les SE spécifiques 3/3 - Web Operating System (WOS)

EyeOS - <http://eyeos.org.fr> - plateforme OpenSource



# Les SE spécifiques 3/3 - Web Operating System (WOS)

G.ho.st - <http://g.ho.st>



Aujourd'hui...

Les ordinateurs multi-cœurs permettent  
d'atteindre une puissance de calcul

**EXCEPTIONNELLE**

Le web devient un web orienté

**INFORMATION et SERVICES**

## Qu'est-ce que Linux ?

- Unix créé par un étudiant finlandais : Linus B. Torvalds,
- Ecriture d'un système d'exploitation Unix fonctionnant sur PC,
- Première version stable en mars 1992,
- Puis développement très rapide grâce à une communauté de développeurs.

## Originalité par rapport à d'autres Unix commerciaux :

- Linux n'a pas été développé dans un but commercial,
- Aucune ligne de code n'a été copiée des systèmes UNIX originaux,
- Tout le monde, depuis sa création, est libre de l'utiliser mais aussi de l'améliorer (OpenSource).

# Linux visuel

Linux possède sa mascotte (manchot) : Tux.



# Distribution

- Linux est architecturé autour d'un **noyau** (en anglais *kernel*) chargé de prendre en charge le matériel,
- Une **distribution** est l'assemblage d'un ensemble de logiciels autour d'un noyau Linux afin de fournir un système clé en main.

Quelques distributions :

Ubuntu, Mandriva, Knoppix, Slackware, Zenwalk, RedHat, Fedora Core, Debian, SuSe...

La plupart des distributions proposent également une installation graphique qui leur est propre ainsi qu'un système de gestion de paquetages permettant d'installer automatiquement des logiciels en gérant les dépendances (les logiciels sous Linux sont parfois liés à des bibliothèques externes ou s'appuient sur d'autres logiciels).

# Définition

- Linux est un système d'exploitation proche des systèmes UNIX pouvant être exécuté sur différentes plates-formes matérielles : x86 (Intel, AMD, etc.), Sparc, PowerPC, Alpha, ARM, etc.
- Ainsi le système Linux peut fonctionner aussi bien sur des ordinateurs personnels que des consoles de jeu ou des assistants personnels !
- Système **multi-plateformes**, **multi-utilisateurs** (plusieurs personnes peuvent en même temps travailler sur le même ordinateur), **multi-tâches** (plusieurs applications peuvent être lancées en même temps sans qu'aucune n'affecte les autres) et **multi-processeurs**.
- Linux est un système fiable, robuste et puissant. Il est d'ailleurs capable de fonctionner avec très peu de ressources sur des ordinateurs peu puissants.
- L'utilisateur **root** possède des droits étendus (administrateur) par rapport aux autres utilisateurs du système.

# Logithèque

Linux possède un très grande logithèque. La plupart des applications sont gratuites.

- OpenOffice, suite bureautique,
- Inkscape, dessin vectoriel,
- GIMP, retouche d'images,
- Blender, modélisation 3D,
- Firefox, navigateur web,
- ...

Il possède de très nombreux outils de développement pour de nombreux langages :  
GCC, C et C++, Java, Ruby, Scheme, Python, Perl...

# Logithèque (suite)

Linux offre des outils de calcul scientifique très performants :

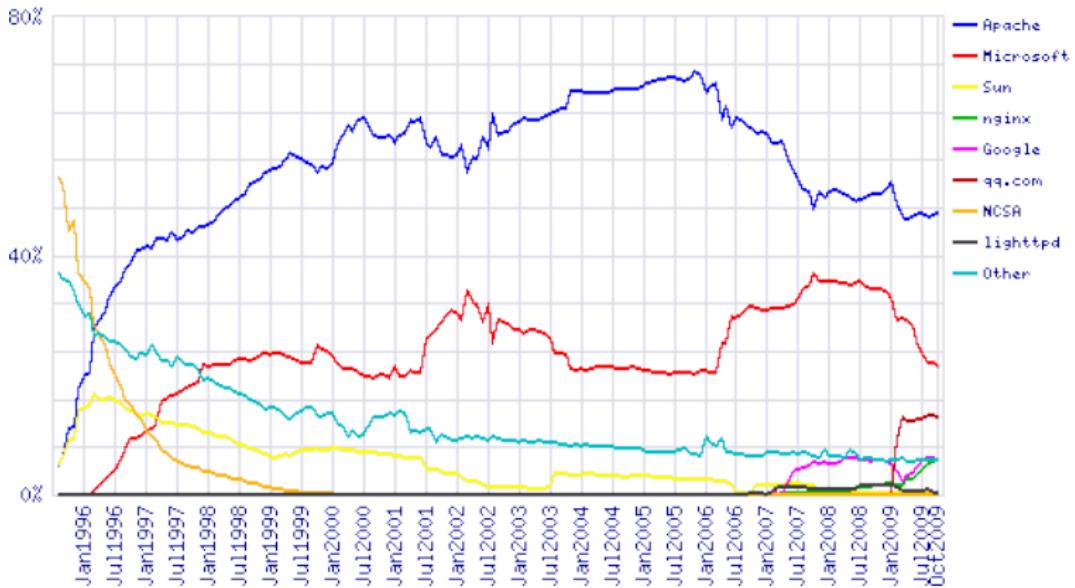
- sage, logiciel de mathématiques,
- octave, logiciel de calcul scientifique,
- scilab, logiciel de calcul scientifique,
- gnuplot, logiciel de création de graphiques,
- ...

Linux est aussi multimédia :

- VLC, lecture de DVD, sons, vidéos,
- audacity, manipulation de sons,
- dvdauthor, création de DVD,
- ...

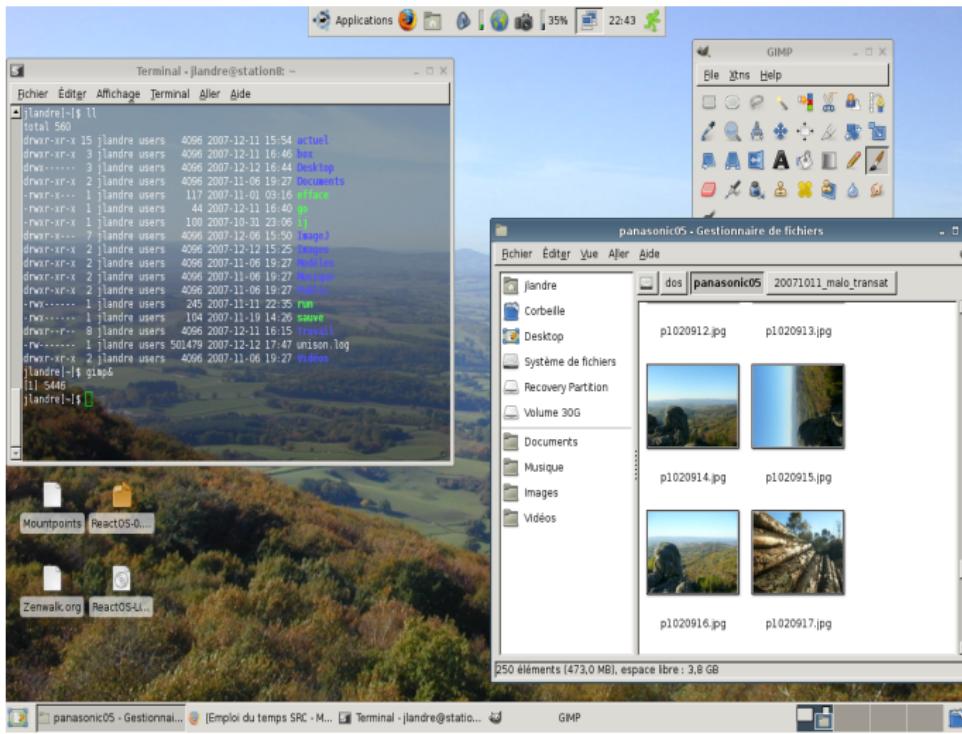
# Linux comme serveur

Linux peut être utilisé comme poste de travail ou comme serveur : Serveur web (Apache), serveur de bases de données (MySQL, Postgres), serveur de mail (postfix, sendmail), serveur réseau (DNS, DHCP, ftp, ssh, ...)



# Bureau Linux

Linux est entièrement configurable, nombreux gestionnaires de fenêtres...



# Super-utilisateur

Il existe dans tout système Linux un utilisateur particulier qui a des droits étendus de gestion de la machine, c'est le **super-utilisateur** (root).

Le shell du super-utilisateur est indiqué par un "#" alors qu'il s'agit d'un "\$" pour les autres utilisateurs.

La commande "sudo" permet de lancer des commandes comme si on était le super-utilisateur (les utilisateurs ayant ces droits sont définis dans le fichier /etc/sudoers) :

```
$ sudo mount /dev/cdrom /media/cdrom  
$ ls -l /media/cdrom
```

# Processus

- Le système Linux en fonctionnement est un **arbre de processus**.
- Le processus est représenté au niveau système par le *processus identifiant (PID)*.
- Le processus initial de Linux s'appelle INIT, il porte toujours le PID 1.
- La commande "ps" permet de lister les processus en cours d'exécution.

```
$ ps
 PID TTY      TIME CMD
 4634 pts/0    00:00:00 bash
 7138 pts/0    00:00:12 texmaker
 7156 pts/0    00:00:00 ps
```

# Processus

Avec l'option "aux", tous les processus actuellement en exécution s'affichent :

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.1	3084	1888	?	Ss	07:42	0:01	/sbin/init
root	2	0.0	0.0	0	0	?	S<	07:42	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S<	07:42	0:00	[migration/0]
root	4	0.0	0.0	0	0	?	S<	07:42	0:00	[ksoftirqd/0]
[...]										
root	3921	0.0	0.1	4324	1156	?	Ss	07:42	0:00	/usr/bin/system-tools-backends
daemon	3995	0.0	0.0	2096	452	?	Ss	07:42	0:00	/usr/sbin/atd
root	4023	0.0	0.1	3480	1032	?	Ss	07:42	0:00	/usr/sbin/cron
root	4065	0.0	0.6	33072	7128	?	Ss	07:42	0:00	/usr/sbin/apache2 -k start
www-data	4077	0.0	0.3	33012	3772	?	S	07:42	0:00	/usr/sbin/apache2 -k start
www-data	4079	0.0	0.3	33012	3772	?	S	07:42	0:00	/usr/sbin/apache2 -k start
www-data	4081	0.0	0.3	33012	3772	?	S	07:42	0:00	/usr/sbin/apache2 -k start
www-data	4083	0.0	0.3	33012	3772	?	S	07:42	0:00	/usr/sbin/apache2 -k start
www-data	4085	0.0	0.3	33012	3772	?	S	07:42	0:00	/usr/sbin/apache2 -k start
[...]										
jlandre	4604	0.1	0.3	17388	3184	?	Ss	07:43	0:03	gnome-screensaver
jlandre	4611	0.0	0.8	70972	8704	?	S1	07:43	0:00	/usr/lib/evolution/evolution-da
jlandre	4633	0.0	0.0	2036	704	?	S	07:43	0:00	gnome-pty-helper
jlandre	4634	0.0	0.3	6064	3280	pts/0	Ss	07:43	0:01	/bin/bash
jlandre	5796	0.0	0.3	6124	3340	pts/1	Ss+	07:46	0:01	/bin/bash
jlandre	7292	0.0	0.1	2768	1032	pts/0	R+	08:19	0:00	ps aux

# Arbre des processus

La commande "pstree" affiche l'arbre des processus :

```
$ pstree -A
init+-NetworkManager---2*[{NetworkManager}]
| -NetworkManagerD
|-acpid
|-artsd
|-atd
|-avahi-daemon---avahi-daemon
|-console-kit-dae---61*[{console-kit-dae}]
|-cron
|-cupsd
|-3*[dbus-daemon]
|-2*[dbus-launch]
|-gdm---gdm-+Xorg
|           `--awesome---seahorse-agent
|-6*[getty]
|-syslogd
|-xterm---bash---kile-+--bash
|           `--evince---{evince}
`-xterm---bash-+-firefox-+-npviewer.bin---5*[{npviewer.bin}]
|           |
|           `--6*[{firefox}]
`-pstree
```

# Destruction d'un processus

Un processus peut être arrêté grâce à la commande "kill". Cette commande permet d'envoyer un signal au processus en utilisant son PID.

```
$ ps
 PID TTY      TIME CMD
18749 pts/2    00:00:00 bash
18765 pts/2    00:07:38 firefox
19335 pts/2    00:00:00 ps

$ kill -9 18765

$ ps
 PID TTY      TIME CMD
18749 pts/2    00:00:00 bash
19336 pts/2    00:00:00 ps
[1]+  Processus arrete      firefox
```

# Mémoire

- La commande "free" permet de connaître l'espace mémoire libre du système.

```
$ free
      total        used        free      shared      buffers      cached
Mem:    1019776     776212     243564          0      95960     338348
-/+ buffers/cache:  341904    677872
Swap:        0          0          0
```

# État du système

- La commande "top" propose une vision d'ensemble en temps réel des processus en exécution sur le système.

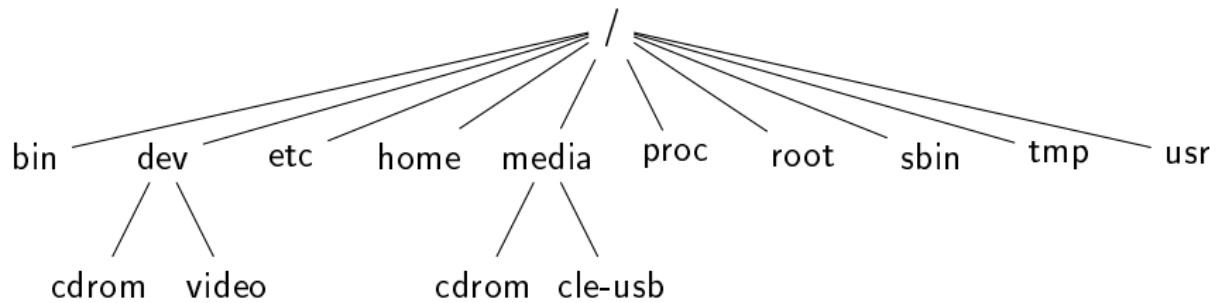
```
$ top
top - 13:31:56 up 34 min, 3 users, load average: 0.82, 0.82, 1.03
Tasks: 142 total, 1 running, 141 sleeping, 0 stopped, 0 zombie
Cpu(s): 30.4%us, 5.8%sy, 11.9%ni, 50.9%id, 0.6%wa, 0.2%hi, 0.2%si, 0.0%st
Mem: 1019776k total, 790008k used, 229768k free, 97256k buffers
Swap:      0k total,      0k used,      0k free, 348540k cached

 PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
3820 root      20   0  308m  33m  10m S   11  3.3  5:42.75 Xorg
8033 jlandre  20   0 2444 1108  828 R    4  0.1  0:00.04 top
  1 root      20   0  3084 1872  548 S    0  0.2  0:01.18 init
  2 root      15  -5     0     0  0 S    0  0.0  0:00.00 kthreadd
  3 root      RT  -5     0     0  0 S    0  0.0  0:00.01 migration/0
  4 root      15  -5     0     0  0 S    0  0.0  0:00.19 ksoftirqd/0
  5 root      RT  -5     0     0  0 S    0  0.0  0:00.00 watchdog/0
  6 root      RT  -5     0     0  0 S    0  0.0  0:00.03 migration/1
```

On sort de "top" en tapant la lettre q (*quit*).

# Arbre de répertoire

Le système Linux est organisé sous forme d'arbre logique sur lequel on vient "monter" des systèmes de fichiers. La racine de l'arbre s'appelle "/".



pwd

Dans l'arbre, il y a toujours un répertoire de travail ou répertoire courant (*working directory*) dans lequel l'utilisateur se trouve. La commande “pwd” (*print working directory*) permet de connaître le répertoire de travail :

```
$ pwd  
/home/jlandre
```

# mount

La commande “mount” sans argument donne les systèmes de fichiers actuellement montés sur la racine “/”.

```
$ mount
/dev/sda3 on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw)
none on /sys type sysfs (rw,noexec,nosuid,nodev)
udev on /dev type tmpfs (rw,mode=0755)
none on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
none on /dev/shm type tmpfs (rw,nosuid,nodev)
none on /var/run type tmpfs (rw,nosuid,mode=0755)
none on /var/lock type tmpfs (rw,noexec,nosuid,nodev)
none on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
/dev/sda4 on /home type ext4 (rw)
```

# Montage de systèmes de fichiers

La commande “mount” permet de venir monter un système de fichier sur un nœud de l’arbre.

```
# mount /dev/cdrom /media/cdrom  
# ls -l /media/cdrom
```

# Lister les fichiers et répertoires

La commande “ls” permet de lister les entrées d'un répertoire.

```
$ cd /
$ ls -l
total 92
drwxr-xr-x  2 root root  4096 2009-11-09 19:01 bin
drwxr-xr-x  3 root root  4096 2009-11-09 20:08 boot
drwxr-xr-x 15 root root  3780 2009-11-15 03:35 dev
drwxr-xr-x 150 root root 12288 2009-11-15 04:59 etc
drwxr-xr-x  8 root root  4096 2009-11-13 18:59 home
lrwxrwxrwx  1 root root    33 2009-11-09 18:59 initrd.img -> boot/initrd.img-2.6.31-14-generic
drwxr-xr-x 18 root root 12288 2009-11-14 15:04 lib
drwx----- 2 root root 16384 2009-11-09 18:37 lost+found
drwxr-xr-x  5 root root  4096 2009-10-28 21:55 media
drwxr-xr-x  2 root root  4096 2009-10-20 02:04 mnt
drwxr-xr-x  2 root root  4096 2009-10-28 21:55 opt
dr-xr-xr-x 142 root root     0 2009-11-15 04:34 proc
drwx----- 12 root root  4096 2009-11-15 03:41 root
drwxr-xr-x  2 root root  4096 2009-11-09 20:06 sbin
drwxr-xr-x  2 root root  4096 2009-10-20 01:05 selinux
drwxr-xr-x  2 root root  4096 2009-10-28 21:55 srv
drwxr-xr-x 12 root root     0 2009-11-15 04:34 sys
drwxrwxrwt 18 root root  4096 2009-11-15 05:03 tmp
drwxr-xr-x 12 root root  4096 2009-11-09 20:00 usr
drwxr-xr-x 16 root root  4096 2009-11-15 04:01 var
lrwxrwxrwx  1 root root   30 2009-11-09 18:59 vmlinuz -> boot/vmlinuz-2.6.31-14-generic
```

# Lister les fichiers et répertoires

La commande “ls” suivie par un argument donne la liste des répertoires dans le chemin indiqué par l’argument :

```
$ ls -l /home
total 72
d---rwx---  2 nobody  users      4096 2009-05-13 12:01 commun
drwxr-xr-x  2 pulse    nogroup   4096 2008-12-10 10:58 ftp
drwxr-xr-x 118 jlandre  jlandre  12288 2009-11-16 15:48 jlandre
drwx-----  2 root     root     16384 2007-06-12 15:54 lost+found
drwxr-x---  17 sl      users     4096 2008-04-18 12:12 sl
```

# Déplacement dans l'arbre

- Le répertoire dans lequel on se situe est le répertoire courant. Par défaut, c'est "/home/jlandre" pour l'utilisateur "jlandre".
- Dans chaque répertoire, on trouve deux répertoires spéciaux ":" qui indique le répertoire courant et ".." qui indique le répertoire parent du répertoire courant.
- Le déplacement dans l'arbre s'effectue grâce à la commande "cd" (change directory).
- Il y a deux façons d'indiquer un chemin :
  - ▶ Chemin absolu : on part de la racine / et on indique les répertoires traversés en les séparant par des "/",
  - ▶ Chemin relatif : on part du répertoire courant ("..") et on indique les répertoires traversés en les séparant par des "/".

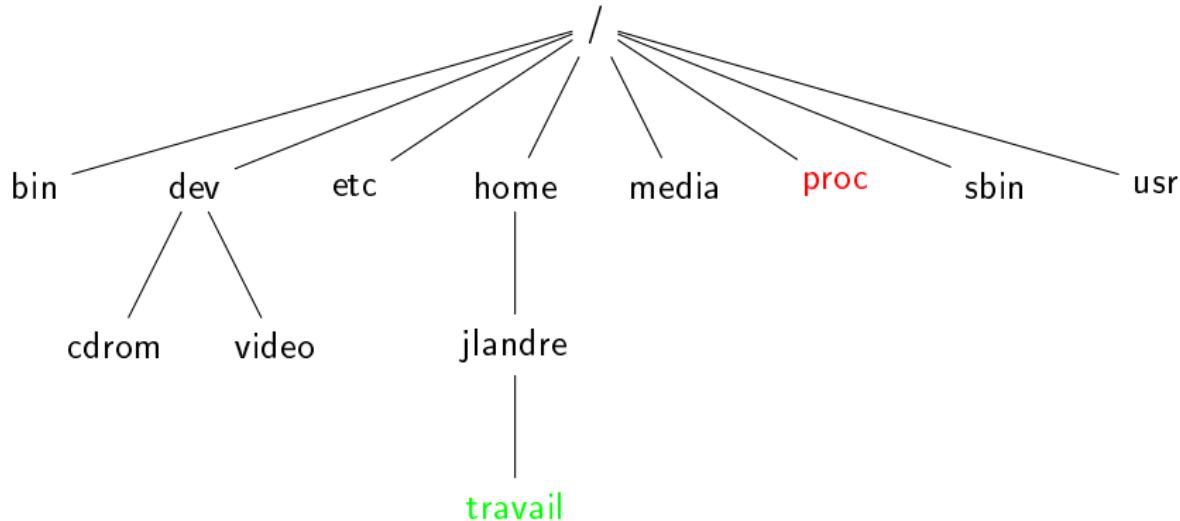
# Chemin absolu

On indique un chemin absolu en commençant par “/”.

Par exemple, on veut aller dans le répertoire “travail” (en vert).

Depuis n’importe quel point de l’arborescence, on va se retrouver dans le répertoire “travail” en tapant :

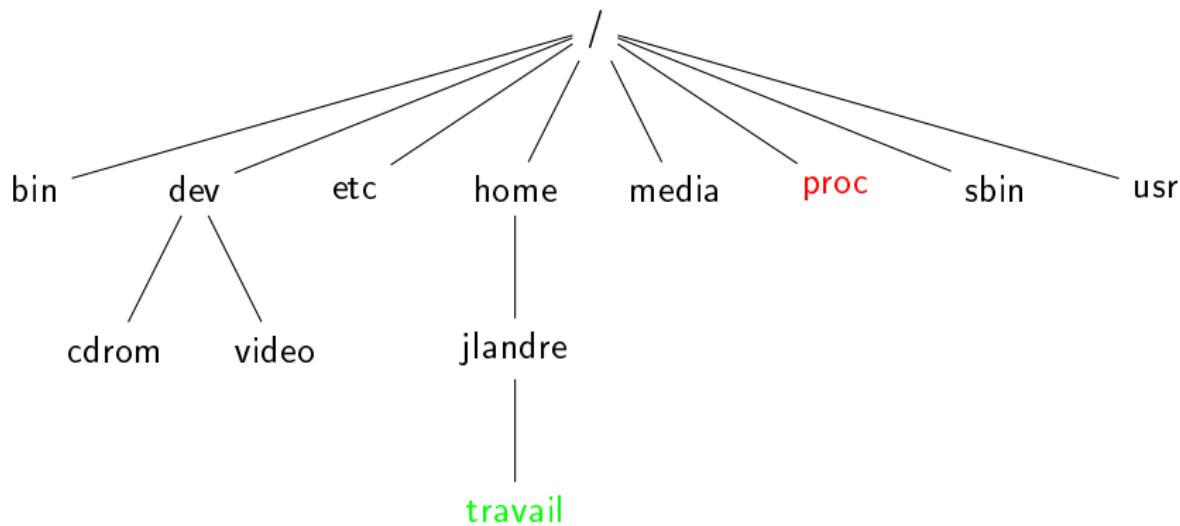
```
$ cd /home/jlandre/travail
```



# Chemin relatif

On indique un chemin relatif en partant du répertoire courant et en décrivant les sauts à effectuer pour se rendre à la destination.

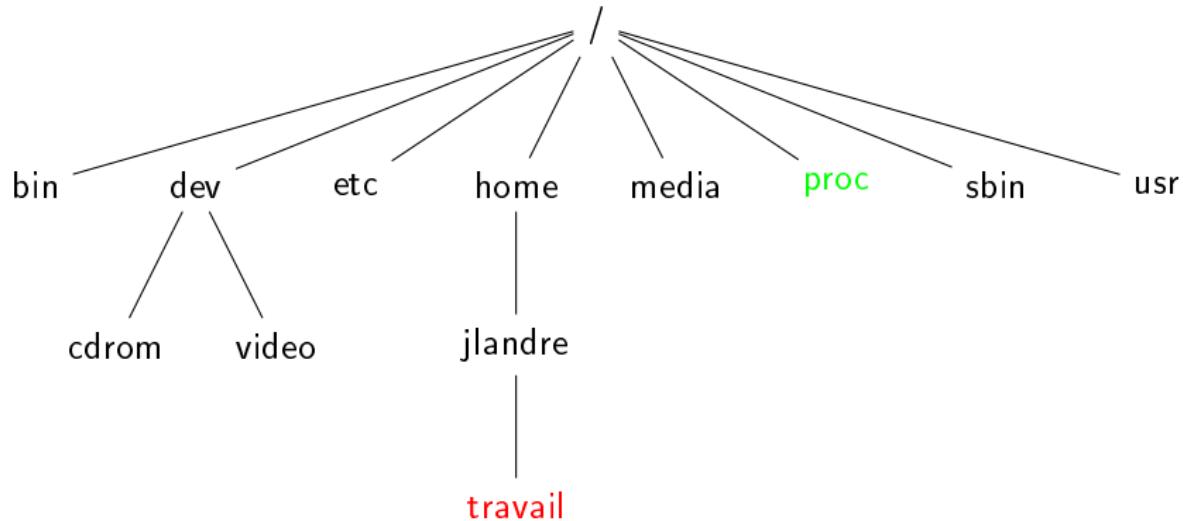
```
$ cd ../../home/jlandre/travail
```



# Chemin relatif

Pour aller du répertoire rouge dans le répertoire vert en chemin relatif :

```
$ cd ../../proc
```



# Droits sur les fichiers et les répertoires

Il existe trois droits sur les fichiers et les répertoires : **lecture** (r pour *read*), **écriture** (w pour *write*) et **exécution** (x pour *execute*).

Pour un fichier les droits ont la signification suivante :

- r : autorisation de consulter le contenu d'un fichier
- w : autorisation de modifier le contenu d'un fichier
- x : autorisation d'exécuter un fichier

Pour un répertoire les droits ont la signification suivante :

- r : autorisation de consulter le contenu d'un répertoire
- w : autorisation d'ajouter ou de supprimer des fichiers dans le répertoire
- x : autorisation de se déplacer dans le répertoire (*cd*)

# Affichage des droits

Il y a trois droits distincts pour un fichier ou un répertoire :

- L'utilisateur propriétaire (u pour *user*)
- Le groupe propriétaire (g pour *group*)
- Les autres (o pour *others*)

Les droits sont affichés par la commande *ls* avec l'option -l (pour *long*)

Exemple pour un fichier :

```
-rwxr-x--- 23 jlandre users 4096 2009-11-16 23:11 essai.txt
    u   g   o   utilisateur   groupe
          propriétaire   propriétaire
                                nom
```

# Création des utilisateurs

La commande “adduser” permet d’ajouter un utilisateur. La commande “passwd” permet de lui affecter un mot de passe. Les informations sur les utilisateurs sont dans le fichier /etc/passwd et dans /etc/shadow si l’option shadow est choisie.

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
[...]
gdm:x:112:119:Gnome Display Manager:/var/lib/gdm:/bin/false
jlandre:x:1000:1000:jlandre,,,:/home/jlandre:/bin/bash

# cat /etc/shadow
root:$6$TavAu13j$qczFmV9iLm10nFSKhQlc5Lm3jn3IWcHyFPLKpe6ND0:14562:0:99999:7:::
daemon:*:14545:0:99999:7:::
bin:*:14545:0:99999:7:::
sys:*:14545:0:99999:7:::
[...]
gdm:*:14545:0:99999:7:::
jlandre:$6$/J0t5DjE$uZw20sfCnXBM12.T28ubiETgQEAdydkCb0Q50:14562:0:99999:7:::
```

# Groupes

Les utilisateurs par défaut sont partie d'un groupe qui est créé lors de leur création. Le groupe par défaut porte le même nom que l'utilisateur. Par exemple, l'utilisateur jlandre est dans le groupe jlandre par défaut.

La commande "groups" permet de connaître les groupes auquel appartient l'utilisateur.

```
$ groups jlandre
jlandre : jlandre adm dialout cdrom plugdev lpadmin admin sambashare

$ ls -l /home
total 36
drwxr-xr-x 59 jlandre jlandre 4096 2009-11-16 22:25 jlandre
drwx----- 2 root     root    16384 2009-11-09 18:37 lost+found
```

On peut ajouter un groupe avec la commande "addgroup" :

```
$ sudo addgroup srcusers
Ajout du groupe << srcusers >> (identifiant 1005)...
Termine.
```

# Insertion dans un groupe

Un utilisateur peut être inséré dans un groupe avec la commande "usermod".

```
$ groups jlandre
jlandre : jlandre adm dialout cdrom plugdev lpadmin admin sambashare
$ usermod -aG users jlandre
$ groups jlandre
jlandre : jlandre adm dialout cdrom plugdev users lpadmin admin sambashare
```

# Changement de propriétaire

Chaque répertoire et chaque fichier possède un **propriétaire** et un **groupe propriétaire**. La fonction “chown” permet de changer le propriétaire et le groupe propriétaire sur un fichier ou un répertoire :

```
jlandre@home-jl:/home$ ls -l
total 36
drwxr-xr-x 59 jlandre jlandre 4096 2009-11-16 23:11 jlandre
$ sudo chown jlandre:users jlandre
$ ls -l
total 36
drwxr-xr-x 59 jlandre users 4096 2009-11-16 23:11 jlandre
$ sudo chown jlandre:jlandre jlandre
<$ ls -l
total 36
drwxr-xr-x 59 jlandre jlandre 4096 2009-11-16 23:11 jlandre
```

# Changement des droits

Chaque répertoire et chaque fichier possède un **propriétaire** et un **groupe propriétaire**. On peut changer les droits lecture (r), écriture (w) et exécution (x) pour le propriétaire, le groupe propriétaire et tous les autres utilisateurs. La fonction “chmod” permet de changer les droits sur un fichier ou un répertoire :

```
jlandre@home-jl:/home$ ls -l
total 36
drwxr-xr-x 59 jlandre jlandre 4096 2009-11-16 23:11 jlandre
$ chmod 744 jlandre
$ ls -l
total 36
drwxr--r-- 59 jlandre jlandre 4096 2009-11-16 23:11 jlandre
$ chmod 755 jlandre
$ ls -l
total 36
drwxr-xr-x 59 jlandre jlandre 4096 2009-11-16 23:11 jlandre
```

user			group			others		
4	2	1	4	2	1	4	2	1
r	w	x	r	w	x	r	w	x

# Succession de commandes (tube)

Grâce au caractère “|”, on peut exécuter une commande sur le résultat d'une précédente commande.

```
$ ls -l  
total 11976  
drwxr-xr-x 3 jlandre users      4096 2009-09-16 10:40 cv  
drwxr-xr-x 4 jlandre users      4096 2009-10-23 14:13 enseignement  
-rw-r--r-- 1 jlandre users     94458 2009-10-21 09:33 iticlycee.pdf  
-rwx----- 1 jlandre jlandre   474336 2009-11-13 08:21 P1070806.JPG  
-rwx----- 1 jlandre jlandre   570178 2009-11-13 08:20 P1080203.JPG  
-rwx----- 1 jlandre jlandre   612433 2009-11-13 08:20 P1080556.JPG  
-rw-r--r-- 1 jlandre jlandre   47969 2009-11-15 04:35 prog-info-lycee-src.rtf  
-rw-r--r-- 1 jlandre users     33792 2009-10-29 09:48 Programme d.doc  
-rw-r--r-- 1 jlandre users     13539 2009-11-13 07:43 programme-info-lycee.odt  
drwxr-xr-x 9 jlandre users     4096 2009-11-16 09:09 recherche  
-rw-r--r-- 1 jlandre users    1855060 2009-09-28 15:04 sdarticle2.pdf  
-rw-r--r-- 1 jlandre users    573809 2009-09-28 15:03 sdarticle.pdf  
-rwxr-xr-x 1 jlandre users    174928 2009-09-04 17:13 tiddlymath.jlandre.html  
drwxr-xr-x 4 jlandre users     4096 2009-11-16 09:09 todo  
jlandre@jl-crestic:~/actuel$ ls -l | grep JPG  
-rwx----- 1 jlandre jlandre   474336 2009-11-13 08:21 P1070806.JPG  
-rwx----- 1 jlandre jlandre   570178 2009-11-13 08:20 P1080203.JPG  
-rwx----- 1 jlandre jlandre   612433 2009-11-13 08:20 P1080556.JPG
```

# Autres exemples

```
$ ls -lR / | more
/:
total 92
drwxr-xr-x  2 root root  4096 2009-11-09 19:01 bin
drwxr-xr-x  3 root root  4096 2009-11-09 20:08 boot
drwxr-xr-x 15 root root  3780 2009-11-16 20:49 dev
drwxr-xr-x 150 root root 12288 2009-11-16 22:22 etc
drwxr-xr-x  8 root root  4096 2009-11-13 18:59 home
lrwxrwxrwx  1 root root    33 2009-11-09 18:59 initrd.img -> boot/initrd.img-2.6.31-14-generic
drwxr-xr-x 18 root root 12288 2009-11-14 15:04 lib
drwxr-xr-x  5 root root  4096 2009-10-28 21:55 media
dr-xr-xr-x 123 root root     0 2009-11-16 21:48 proc
drwx----- 12 root root  4096 2009-11-15 03:41 root
drwxr-xr-x  2 root root  4096 2009-11-09 20:06 sbin
drwxrwxrwt 16 root root  4096 2009-11-16 22:51 tmp
drwxr-xr-x 12 root root  4096 2009-11-09 20:00 usr
drwxr-xr-x 16 root root  4096 2009-11-15 04:01 var
lrwxrwxrwx  1 root root   30 2009-11-09 18:59 vmlinuz -> boot/vmlinuz-2.6.31-14-generic

/bin:
total 5456
-rwxr-xr-x 1 root root 875596 2009-09-14 07:09 bash
-rwxr-xr-x 1 root root  30192 2009-06-30 02:18 bunzip2
--Plus--
```

# Expressions régulières

Une expression régulière permet de travailler sur un nom de fichier en utilisant des jokers représentant des ensembles de lettres. Ce mécanisme permet de sélectionner facilement dans les chemins des informations intéressantes.

- ? Correspond à tout caractère
- \* Correspond à toute chaîne de caractères
- ^ Correspond au début de la ligne
- \$ Correspond à la fin de la ligne
- Représente un intervalle de valeurs

# Exemples

```
$ ls -ld /home/jlandre/*.pdf
-rw----- 1 jlandre users 330035 2009-09-14 11:52 /home/jlandre/B02.Bourbaki.pdf
-rw-r--r-- 1 jlandre users 40298 2009-09-11 11:54 /home/jlandre/lettre-retour-dijon.pdf
-rw-r--r-- 1 jlandre users 222072 2009-09-14 11:56 /home/jlandre/P11.Boltzmann.pdf
-rw-r--r-- 1 jlandre users 3758007 2009-09-24 11:16 /home/jlandre/pgfmanual.pdf
-rw-r--r-- 1 jlandre users 11883 2009-09-22 16:59 /home/jlandre/test.pdf

$ ls -ld /home/jlandre/?e?t*
-rwxr-xr-x 1 jlandre users 19091 2009-09-11 11:45 /home/jlandre/lettre-retour-dijon.odt
-rw-r--r-- 1 jlandre users 40298 2009-09-11 11:54 /home/jlandre/lettre-retour-dijon.pdf
-rw-r--r-- 1 jlandre users 1280 2009-09-22 17:00 /home/jlandre/test.dvi
-rw-r--r-- 1 jlandre users 11883 2009-09-22 16:59 /home/jlandre/test.pdf
-rw-r--r-- 1 jlandre users 519 2009-09-22 17:23 /home/jlandre/test.tex
```

# Avec grep

```
$ ls /home/jlandre | grep ^[a-zA-C]  
actuel  
beast  
Bureau  
$ ls /home/jlandre | grep [0-9]  
sage-4.2  
sage-4.2.tar  
$ ls /home/jlandre | grep ^[s]  
sage-4.2  
sage-4.2.tar  
sauve  
$ ls /home/jlandre | grep ^[s] | grep 2$  
sage-4.2
```

# Avec grep

```
$ ls /home/jlandre | grep [a-l]$  
actuel  
Musique  
Public  
sauve  
$ ls -l /home/jlandre | grep tar$  
-rw-r--r-- 1 jlandre jlandre 271779840 2009-11-14 15:29 sage-4.2.tar  
$ ls -l /home/jlandre | grep s$  
drwxr-xr-x 2 jlandre jlandre 4096 2009-11-09 19:07 Documents  
drwxr-xr-x 3 jlandre jlandre 4096 2009-11-13 18:23 Images  
drwxr-xr-x 2 jlandre jlandre 4096 2009-11-09 19:07 ModÃ"les  
drwxr-xr-x 2 jlandre jlandre 4096 2009-11-15 05:11 TÃ©lÃ©chargements  
drwxr-xr-x 2 jlandre jlandre 4096 2009-11-09 19:07 VidÃ©os
```

# Chemin de recherche

Lors de la saisie d'une commande au clavier, l'interpréteur de commandes cherche la commande tapée dans un chemin spécial (path). Ce chemin est écrit dans une variable d'environnement qu'on peut afficher et faire évoluer.  
Pour savoir où se trouve la commande réellement lancée, on utilise "which".

```
$ echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
```

```
$ which firefox  
/usr/bin/firefox
```

```
$ which ls  
/bin/ls
```

# dmesg

La commande “dmesg” permet d'avoir des informations sur le noyau linux et sur les modifications sur le matériel.

```
$ dmesg
[...]
[ 14.409247] alloc irq_desc for 21 on node -1
[ 14.409255] alloc kstat_irqs on node -1
[ 14.409268] Intel ICH 0000:00:02.7: PCI INT C -> GSI 21 (level, low) -> IRQ 21
[ 14.828031] intel8x0_measure_ac97_clock: measured 54735 usecs (2633 samples)
[ 14.828039] intel8x0: clocking to 48000
[ 19.033717] eth0: link up, 100Mbps, full-duplex, lpa 0x45E1
[ 21.228260] ppdev: user-space parallel port driver
[ 26.893643] agpgart-sis 0000:00:00.0: AGP 2.0 bridge
[ 26.893675] agpgart-sis 0000:00:00.0: putting AGP V2 device into 4x mode
[ 26.893720] nvidia 0000:01:00.0: putting AGP V2 device into 4x mode
[ 29.356015] eth0: no IPv6 routers present
```

# cat

La commande “cat” affiche le contenu d'un fichier.

```
$ cat /etc/hosts
127.0.0.1      localhost
127.0.1.1      jl-crestic

# The following lines are desirable for IPv6 capable hosts
::1      localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

La commande “vi” ouvre un éditeur de fichiers textes. Cet éditeur est présent sur de nombreux systèmes Unix (pas uniquement Linux).

Il y a seulement quatre commandes minimales à connaître pour pouvoir éditer des fichiers :

Raccourci	Effet
i	Passage en mode Insertion
ESC	Sortie du mode Insertion
:x	Sauver le fichier et sortir
:q !	Quitter sans sauver le fichier

# head

La commande “head” donne le début d'un fichier.

```
$ head -n 3 /etc/hosts
127.0.0.1 localhost
127.0.1.1 home-jl

$ dmesg | head -n 10
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.31-14-generic (buildd@rothera) (gcc version 4.4.1 (Ubuntu 4.4.1-12ubuntu10) 20100424)
[    0.000000] KERNEL supported cpus:
[    0.000000]     Intel GenuineIntel
[    0.000000]     AMD AuthenticAMD
[    0.000000]     NSC Geode by NSC
[    0.000000]     Cyrix CyrixInstead
[    0.000000]     Centaur CentaurHauls
[    0.000000]     Transmeta GenuineTMx86
```

# tail

La commande “tail” affiche les lignes de fin d'un fichier.

```
$ tail -n 3 /etc/hosts
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
$ dmesg | tail -n 10
[ 13.064410] input: ImPS/2 Generic Wheel Mouse as /devices/platform/i8042/serio1/input/input4
[ 18.310794] ppdev: user-space parallel port driver
[ 18.990298] eth0: link up, 100Mbps, full-duplex, lpa 0x45E1
[ 26.495684] agpgart-sis 0000:00:00.0: AGP 2.0 bridge
[ 26.495715] agpgart-sis 0000:00:00.0: putting AGP V2 device into 4x mode
[ 26.495760] nvidia 0000:01:00.0: putting AGP V2 device into 4x mode
[ 29.172025] eth0: no IPv6 routers present
[ 5594.737581] agpgart-sis 0000:00:00.0: AGP 2.0 bridge
[ 5594.737612] agpgart-sis 0000:00:00.0: putting AGP V2 device into 4x mode
[ 5594.737658] nvidia 0000:01:00.0: putting AGP V2 device into 4x mode
```

# ln

La commande “ln” permet de créer des raccourcis vers un chemin.

```
$ ls -l
total 4
-rwxr-xr-x 1 jlandre jlandre 1270 2009-11-16 22:59 efface
$ ln -s efface eff
$ ls -l
total 4
lrwxrwxrwx 1 jlandre jlandre      6 2009-11-16 23:00 eff -> efface
-rwxr-xr-x 1 jlandre jlandre 1270 2009-11-16 22:59 efface
$ ln -s /usr/bin/firefox fx
$ ls -l
total 4
lrwxrwxrwx 1 jlandre jlandre      6 2009-11-16 23:00 eff -> efface
-rwxr-xr-x 1 jlandre jlandre 1270 2009-11-16 22:59 efface
lrwxrwxrwx 1 jlandre jlandre     16 2009-11-16 23:00 fx -> /usr/bin/firefox
```

# Recherche de fichiers

Un mécanisme d'indexation des fichiers sur le disque permet de retrouver un fichier par son nom.

La commande "updatedb" met à jour l'index de tous les fichiers sur le disque (peut être long).

La commande "locate" permet de retrouver un fichier ou un répertoire par son nom.

```
$ sudo updatedb  
[sudo] password for jlandre:  
$ locate src | grep pdf  
/home/jlandre/sauve/jerome/perso/oma-src.pdf  
/usr/share/doc/texlive-doc/latex/srcltx/srcltx.pdf  
/usr/share/doc/texlive-latex-extra-doc/latex/srcltx/srcltx.pdf  
/usr/share/man/man1/pdftosrc.1.gz
```

# ifconfig

ifconfig donne des informations et permet de paramétrer la carte réseau.

```
$ ifconfig
eth0      Link encap:Ethernet HWaddr 00:e0:18:7d:70:28
          inet adr:192.168.1.15 Bcast:192.168.1.255 Masque:255.255.255.0
                  adr inet6: fe80::2e0:18ff:fe7d:7028/64 Scope:Lien
                  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                  Packets recus:10573 erreurs:0 :0 overruns:0 frame:0
                  TX packets:10696 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 lg file transmission:1000
                  Octets recus:11691831 (11.6 MB) Octets transmis:1540667 (1.5 MB)
                  Interruption:17 Adresse de base:0xb000

lo        Link encap:Boucle locale
          inet adr:127.0.0.1 Masque:255.0.0.0
                  adr inet6: ::1/128 Scope:Hôte
                  UP LOOPBACK RUNNING MTU:16436 Metric:1
                  Packets recus:16 erreurs:0 :0 overruns:0 frame:0
                  TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 lg file transmission:0
                  Octets recus:960 (960.0 B) Octets transmis:960 (960.0 B)
```