

# Bases Informatique

Institut Supérieur du Numérique  
(SupNum)

# PLAN

1. Objectif, déroulement, Evaluation
2. Généralités et notions sur l'informatique
3. Composants d'un PC
4. Systèmes d'exploitation : Windows
5. Codage de l'information
6. Utilisation de Word et Excel

# Objectif

1. Comprendre le principe de fonctionnement et le rôle de chaque composant dans un PC
2. Comprendre le mécanisme de représentation de l'information
3. Comprendre le rôle d'un SE et s'initier en Windows
4. Avoir des compétence en Word et Excel, etc

# Déroulement évaluation

1. Cours , TD et TP
2. Contrôles de compétences: TP notés,  
Contrôles écrits et sur machine,  
Contrôle de connaissances : Ecrit et  
pratique

# Généralités

- Qu'est ce que c'est un ordinateur ?
- Qu'est ce que c'est l'informatique ?
- Internet ? Algorithme ?  
Programmation ?  
Informaticien?
- WINDOWS, WORD, EXCEL, E-MAIL
- ET d'autres questions



# Ordinateur

- ✓ ensemble de circuits électroniques permettant de manipuler des données numériques sous forme binaire : **1 bit = 0 ou 1. (C'est quoi bit réellement ? )**
- ✓ terme amélioré de "calculateur ", traduction de computer. Origine : la firme IBM.
- ✓ Un ordinateur est généralement assimilé à un **ordinateur personnel**. Le PC ( Personal Computer) est le plus courant.
- ✓ D'autres types d'ordinateurs : Apple Macintosh, stations Alpha, stations SUN, stations Silicon Graphics
- ✓ Toutes les données manipulées et traitées par un ordinateur (fichiers, images, sons, vidéo, documents, etc.) **sont Numériques**

# Informatique

- ✓ Science de la gestion, manipulation et exploitation des données sous format numérique
- ✓ Programme : des instructions numériques dont l'interprétation par l'ordinateur donne un résultat

# Composants d'un PC

- **Unité centrale** : Le boîtier principal de l'ordinateur, il contient tous les principaux composants internes (CPU, disques, carte mère, mémoire...)
- **Périphériques** : Éléments non vitaux pour l'ordinateur, ils sont internes (CDROM, disquette, carte réseau) ou externes (écran, clavier, souris, hauts parleurs)
- **Processeur** (*Central processing unit* (CPU)) : Le cœur (cerveau) de l'ordinateur, il effectue tous les calculs et les instructions des programmes

# Composants internes du PC

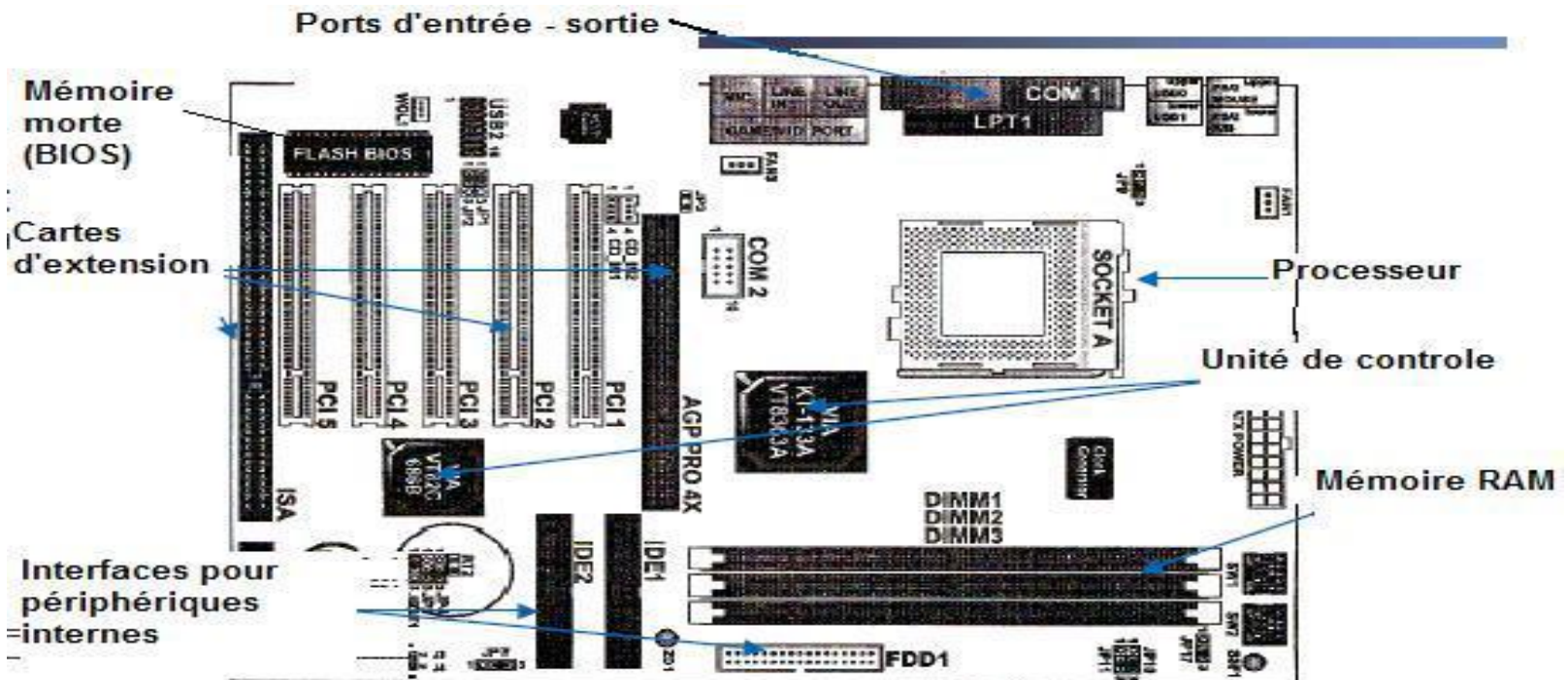
- **Microprocesseur** : un circuit intégré qui contient le CPU et de la mémoire auxiliaire rapide
- **Carte mère** : Le circuit imprimé qui supporte tous les composants (bus, mémoire, CPU, cartes)
- **Alimentation** : convertit le 220V secteur en 12V/5V continu
- **Unités disques** : sert au stockage des données (CDROM, HD)

# Carte mère

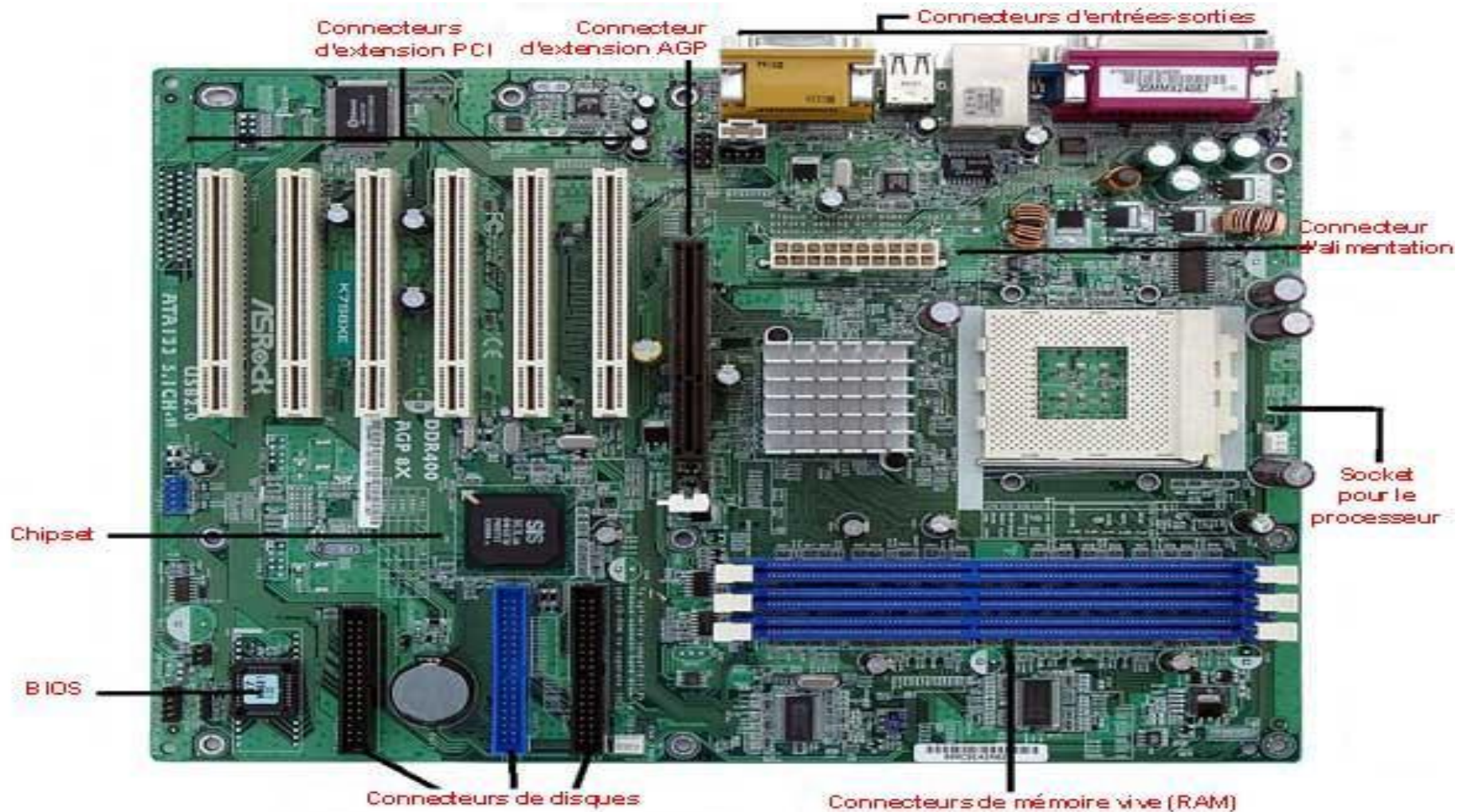
- La partie la plus importante de l'ordinateur
- Circuit imprimé qui relie tous les éléments de l'ordinateur.
- Elle réunit tous les composants matériels de l'ordinateur :
  - directement (microprocesseur, mémoire, carte graphique, carte son),
  - ou par branchement d'une nappe (disque dur, lecteur de disquette, lecteur de cd-rom).

Elle est étroitement liée au microprocesseur.

Marque de référence : **Abit, Asus, Gigabyte, MSI**



# Carte mère



# Carte mère : Composition

- **Le processeur** : exécute les instructions et les calculs. le nombre de processeurs est généralement d'un seul mais une carte mère peut accueillir plusieurs processeurs. Plus le nombre de processeurs est important, plus l'ordinateur est rapide. Attention à ne pas confondre « double processeur » avec la technologie Double Coeur (Dual Core).
- **Connecteur d'alimentation** : là où le câble d'alimentation de la carte mère est branché. Permet d'alimenter la carte mère et ses circuits en courant électrique
- **Le bios (Basic Input Output System)** : petit programme lancé automatiquement au démarrage de l'ordinateur afin d'inspecter la configuration matériel de l'ordinateur. Il se charge de faire le lien entre le système d'exploitation et les différents périphériques de l'ordinateur.
- **Pile** : elle permet de conserver les données enregistrées et modifiées du BIOS contenu dans une mémoire interne à la carte mère

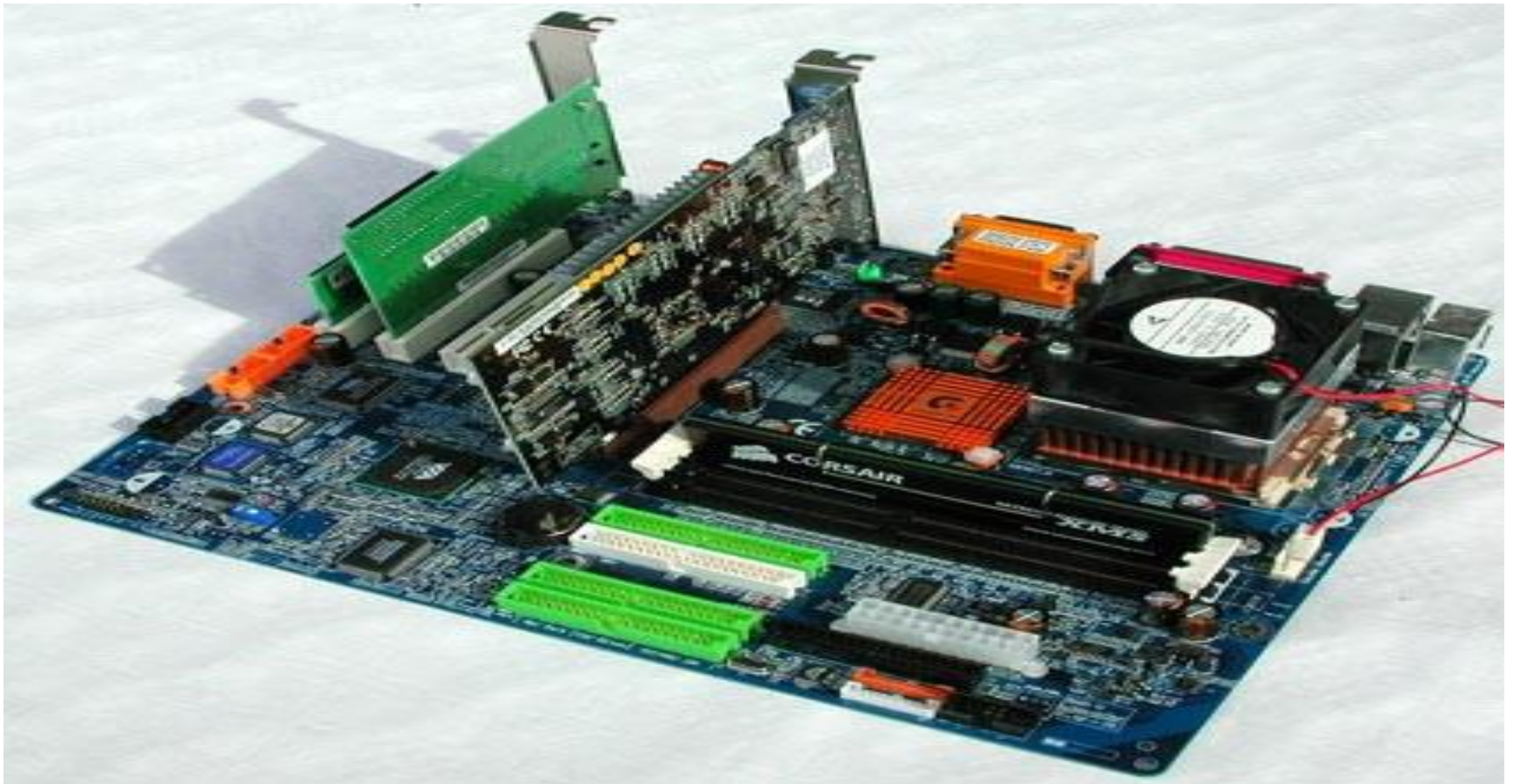
# Carte mère : composition

- **Connecteurs d'extension PCI** (Peripheral Component Interconnect) : C'est l'emplacement des différentes cartes de l'ordinateur (carte son, carte réseau, etc.).
- **Connecteur d'extension AGP** (Accelerated Graphics Port): c'est l'emplacement exclusivement réservé à la carte graphique. Ce type de connecteur tend à disparaître car les cartes graphiques récentes se connectent de plus en plus souvent via un connecteur PCI-Express.
- **Connecteurs d'entrée-sortie** : permettent de brancher les périphériques externes tel que le clavier, l'écran, l'imprimante, etc.
- **Ports IDE** : permettent de connecter des disques durs IDE ainsi que des lecteurs de CD-ROM, de lecteurs de DVD-ROM, de graveurs de CD-ROM et graveurs de DVD-ROM.

# Carte mère : Composition

- **Les connecteurs de mémoire vive** : permettent de placer les barrettes de mémoire vive (RAM). La mémoire vive sert à stocker temporairement des informations dont le processeur a besoin. Elle est volatile: son contenu est effacé à chaque arrêt de l'ordinateur.
- **Chipset** : circuit intégré à la carte mère dont la fonction est de contrôler les échanges de données entre les divers éléments d'une carte mère tels que le processeur, la RAM ou le disque dur. Il s'occupe aussi de la gestion d'énergie. Plus récent sera le chipset, plus la machine pourra intégrer des éléments récents.

# Carte mère



# Le processeur (1/3)

- ❑ A la base de tous les calculs, c'est le "noyau" ( ou le cerveau) de l'ordinateur.
- ❑ Placé sur la carte mère et il est en connexion avec les périphériques et les mémoires via les bus.
- ❑ Fréquence d'horloge d'un processeur = nombre d'instructions par seconde. La moyenne actuelle est de 2000-3000 Mhz, c-à-dire 2 à 3 milliards d'instructions par seconde.
- ❑ Exemples de processeurs : Intel 486, Intel Pentium, Intel Pentium IV, AMD Athlon, AMD Athlon XP,...

# Processeur(2/3)



- ❑ Le premier microprocesseur (Intel 4004) a été inventé en 1972. Depuis, la puissance des microprocesseurs augmente exponentiellement (en moyenne, elle double tous les 18 mois).
- ❑ Un processeur est constitué de :
  - ❑ **une horloge** qui rythme le processeur (donne la cadence):
    - ❑ A chaque impulsion (TOP) d'horloge le processeur effectue une instruction. Par exemple, un ordinateur de 2400 Mhz effectue 2 400 000 000 instructions (opérations élémentaires) par seconde.
  - ❑ **une unité de gestion des bus**
  - ❑ **une unité d'instruction** qui lit les données, les décode puis les envoie à l'unité d'exécution.
  - ❑ **une unité d'exécution**

# Processeur(3/3)

❑ Une opération simple comprend plusieurs instructions, par exemple :  $C \leftarrow A+B$  s'écrit en assembleur (langage de bas niveau, proche de la machine) comme suit:

- ❑ MOV R1, A ; R1 A
- ❑ MOV R2, B ; R2 B
- ❑ ADD R2, R1 ; R2 A + B
- ❑ MOV RS, R2 ; RS A + B

❑ Une instruction est décomposée en instructions élémentaires exécutées en un cycle.

❑ Si la division occupe 41 cycles, pour un processeur à 2.2 GHz, la division prend  $41/2200000000 = 0.0000000186$  s

# Fonctionnement du processeur

❑ Le **processeur** (CPU: Central Processing Unit) est donc rythmé par une **horloge** interne. A chaque top d'horloge (pour les instructions simples), le processeur:

- ❑ Lit l'instruction à exécuter en mémoire ;
- ❑ effectue l'instruction (par une UAL, Unité Arithmétique et Logique) ;
- ❑ passe à l'instruction suivante.

❑ **Une instruction** se décompose en 4 grandes étapes :

- ❑ **Lecture**
- ❑ **Décodage, Recherche d'opérandes**
- ❑ **Calcul, Branchement (if...then...else)**
- ❑ **Rangement du résultat**

# Fonctionnement du processeur : les registres

- ❑ Le processeur stocke temporairement les données dans des registres ("mémoires très rapides").
- ❑ Les registres les plus importants sont:
  - ❑ le **registre accumulateur**: stocke les résultats des opérations arithmétiques et logiques.
  - ❑ le **registre tampon**: stocke temporairement une des opérandes.
  - ❑ le **registre d'instruction**: il contient l'instruction en cours de traitement.
  - ❑ le **compteur ordinal**: il contient l'adresse de la prochaine instruction à traiter.
  - ❑ le **registre tampon mémoire**: stocke temporairement une donnée provenant de la mémoire.

# Fonctionnement du processeur, les signaux...

❑ **Les signaux de commande** : des signaux électriques pour communiquer avec le reste du système.

❑ Exemple : le signal Read/Write signale à la mémoire que le processeur veut lire ou écrire une donnée.

❑ Les signaux sont véhiculés via les bus (ensemble de fils électriques qui connectent, entre autres, le processeur avec les différents périphériques).

❑ C'est le processeur qui gère l'accès aux bus et a la priorité d'accès la plus élevée.

# Les mémoires: la mémoire vive (RAM)...

- ❑ La mémoire vive ou RAM (*Random Access Memory*). Elle est volatile, c'est-à-dire que son contenu est perdu dès que l'ordinateur passe hors tension.
- ❑ Les mémoires vives actuelles sont assez rapides : environ 70 ns (nano seconde =  $10^{-9}$  sec)
- ❑ La mémoire vive est constituée de condensateurs emmagasinant des charges (1 pour chargé, 0 sinon).
- ❑ Nécessité de rafraîchir périodiquement : cycle de rafraîchissement (d'une durée d'environ 15ms pour une mémoire DRAM).
- ❑ La mémoire est physiquement équivalente à une matrice. Chaque cellule (zone mémoire) est adressable par son numéro de ligne et son numéro de colonne.

# Les mémoires: la mémoire morte (ROM)...

❑ La **mémoire morte**, la ROM (*Read Only Memory*) permet de stocker des données nécessaires au démarrage de l'ordinateur.

❑ La ROM contient:

- ❑ Le BIOS: un programme pilotant les interfaces d'entrée-sortie principales du système ;
- ❑ Le chargeur d'amorce: un programme chargeant le système d'exploitation en mémoire (vive) et permettant de le lancer ;

# Les mémoires: capacité de stockage

**L'élément de base de mémoire, c'est le bit.**

- Un octet fait 8 bits**
- un kilo-octet (ko) fait 1024 octets**
- un méga-octet (Mo) fait 1024 kilooctets**
- un giga-octet (Go) fait 1024 mégaoctets**
- un tera-octet (To) fait 1024 gigaoctets**

# Les mémoires: temps d'accès

- ❑ Pour une RAM, le temps d'accès à une donnée = délai de cycle + temps de latence.
  - ❑ Exemple : pour une mémoire de type DRAM, le temps d'accès = 1 + 10 = 11 nanosecondes.
- ❑ Pour un processeur, temps de cycle =  $1/f$ ,  $f$  étant la fréquence de l'horloge.
  - ❑ Pour un ordinateur de 2000 Mhz, le temps de cycle =  $1/(2000 \times 10^6) = 0.5 \text{ ns}$
  - ❑ nécessité de cycles d'attente (*wait state*), car les processeurs sont beaucoup plus rapides que les mémoires.
- ❑ Dans l'exemple précédent, il y a 20 cycles d'attente.

→ importance des mémoires rapides.

# Communication avec les périphériques: les ports d'E/S

- ❑ Les processeurs communiquent avec les périphériques via des **slots d'extension** directement installés sur la carte mère et des **ports** d'entrée-sortie installés sur le boîtier de l'unité centrale.
- ❑ Il existe plusieurs types de slots: **ISA** (le plus lent, 16 bits), **PCI** (plus rapide, 32 bits et 64 bits) et **AGP, PCI-E**(pour le multimédia ).
- ❑ Les instructions et les données sont acheminées via des **bus**. Les bus sont des fils électriques formant des "autoroutes" transportant des millions de bits par seconde.
- ❑ Les données (instructions + données) sont véhiculées sur un bus de données et les adresses sont transportées sur un autre bus: le bus d'adresses.

## Les ports d'E/S (suite)

- ❑ Il existe différents types de ports:
  - ❑ Les ports **série** (communication avec la souris, le modem,...). COM1, COM2, COM3 ...
  - ❑ Les ports parallèles (LPT1, LPT2) généralement réservés à l'imprimante.
  - ❑ Les ports IDE0 et IDE1 destinés aux disques durs, CDROM,
  - ❑ Les ports USB (les plus utilisés maintenant et pour quasiment tous les périphériques : souris, clavier, clés usb, disque dur externes, etc.)

# Les périphériques de stockage:

## le disque Dur

❑ Un **disque dur** est constitué de plusieurs disques rigides en métal, en verre ou en céramique empilés

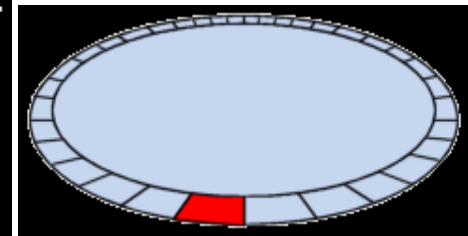
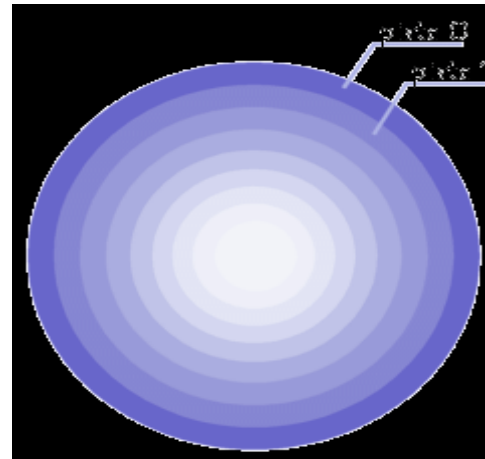
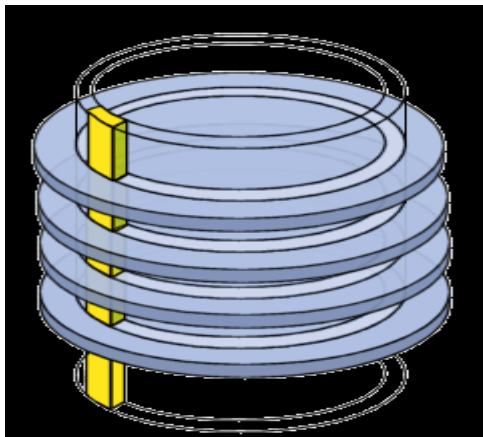
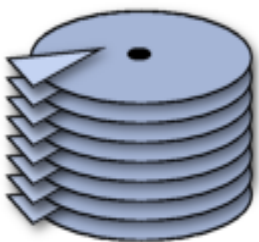
- ❑ Les disques sont recouverts d'une couche magnétique de quelques microns d'épaisseur recouverte d'un film protecteur.
- ❑ Des **têtes** effectuent la **lecture et l'écriture**. Elles sont situées de part et d'autre de chacun des **plateaux**.
- ❑ Les têtes sont séparées de quelques microns de la surface par une couche d'air. Les têtes balaient latéralement la surface du disque



# Les périphériques de stockage:

## le disque Dur

- ❑ Les têtes sont liées entre-elles, une seule tête peut lire ou écrire à un moment donné.
- ❑ Le cylindre est l'ensemble des disques placés verticalement.
- ❑ Un lecteur de disque dur doit être contenu dans un boîtier totalement hermétique.
- ❑ Les têtes de lecture/écriture commencent à inscrire des données à la périphérie du disque (piste 0), puis avancent vers le centre.
  - ❑ Les données sont organisées en cercles concentriques appelés "pistes" créées par le formatage de bas niveau.
  - ❑ Les pistes sont séparées en secteurs.
  - ❑ Un secteur est une zone de stockage de données (512 octets en général).



Secteur

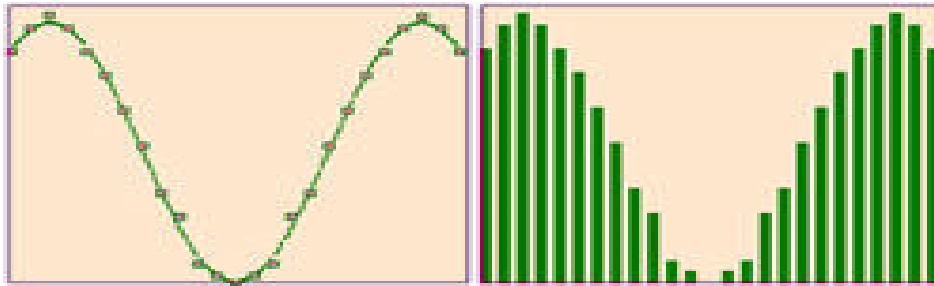
# Les périphériques de stockage:

## le disque Dur

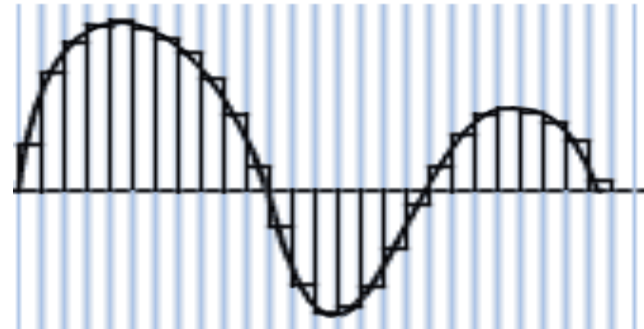
- ❑ On appelle ***cluster*** la zone minimale que peut occuper un ***fichier sur le disque***.
- ❑ Lors du formatage, on définit la taille des clusters (4ko, 8ko, 16ko, ...). Elle dépend du système d'exploitation et équivaut à plusieurs secteurs (entre 1 et 16).
- ❑ Un fichier, aussi minuscule soit-il, devra donc occuper plusieurs secteurs (un cluster).

# Numérique vs Analogique

- Signaux analogiques
  - Continu dans le temps
  - Représentés par des courbes qui varient dans le temps sans discontinuité



- Signaux numériques
  - sous-ensemble de valeurs du signal analogique
  - Obtenus par échantionnage du signal analogique
  - Formés par l'ensemble des valeurs du signal à des instants bien déterminés



# Avantages du numérique

La qualité du signal numérique dépendra de deux facteurs :

- la fréquence d'échantillonnage: plus celle-ci est grande (i.e: les échantillons sont relevés à des intervalles de temps plus petits), plus le signal numérique sera fidèle à l'original ;
- Le nombre de bits sur lequel on code les valeurs (appelé **résolution**): **il détermine le nombre de valeurs différentes qu'un échantillon peut prendre. Plus celui-ci est grand, meilleure est la qualité.**

## Avantages:

- ❖ le signal numérique est moins sensible aux « bruits » (i.e: les signaux parasites qui peuvent altérer le signal).
- ❖ Plus facile à reproduire et à manipuler;
- ❖ Plus facile à corriger si des erreurs de transmission surviennent

# Numérisation

Transformation d'un signal analogique en numérique

Deux étapes :

1. **l'échantillonnage** : prélever périodiquement des échantillons d'un signal analogique.
2. **La quantification** : affecter une valeur numérique à chaque échantillon prélevé, à partir d'un ensemble de valeurs pré-définies.

Ces valeurs dépendent du nombre de bits sur lequel chaque échantillon est représenté (qu'on appelle: la **résolution**).

# Numérisation : Exemple

Si on décide de représenter notre signal sur **8 bits (1 octet)**, alors on dispose de 256 valeurs possibles ( **$2^8=256$** ), allant de 00000000 à 11111111, pour chaque échantillon.

Si notre signal électrique analogique d'origine fluctue entre 0V (valeur minimale) et 5V (valeur maximale) alors on peut diviser cet intervalle de 5V en 256 niveaux. Chaque niveau correspond à l'une des valeurs pouvant être représentées par 8 bits.

Ainsi, chaque échantillon prendra l'une de ces 256 valeurs (la valeur la plus proche).

# Représentation de l'information numérique

- Dans un ordinateur, l'information est numérisée (digitale)
- *L'information est la connaissance d'un état parmi un nombre fini d'états possibles.*
- Une information non numérisée est une information **analogique** : une grandeur physique continue (tension ou courant).
- L'unité de mesure de l'information est **le bit**
- *Un bit est la quantité d'information liée à la connaissance d'un état parmi deux.*
- Un bit d'information peut être représenté par un digit binaire prenant les valeurs **0 ou 1**.
- Rappel : avec  **$n$**  bits, on peut représenter  **$2^n$**  états

## Représentation de l'information (2)

- Codage = représentation avec un nombre limité de bits

→ Problème pour le dépassement de capacité

- Exemple pour les entiers naturels

$$\begin{array}{r} 10101010 \\ + 11000000 \\ = 101101010 \end{array}$$

$$\begin{array}{r} 170 \\ + 192 \\ = 362 \end{array}$$

- Représentation par codage sur 8 bits

$$\begin{array}{r} 10101010 \\ + 11000000 \\ = 101101010 \end{array}$$

→ 362 est trop grand pour être codé sur 8 bits

# Numération en base $b$

- **Numérotation romaine:** MCMXCIX :  
 $I=1$ ;  $V=5$ ;  $X=10$ ;  $L=50$ ;  $C=100$ ;  $D=500$ ;  $M=1000$
- **Numérotation arabe:** représentation positionnelle  
Le rang de chaque chiffre indique son **poids**: dans 1999 le 9 le plus à droite vaut 9, le deuxième vaut 90...  
→ la valeur d'un chiffre dépend de sa position.
- **Représentations courantes des nombres:**  
plusieurs bases :  $XXX_b$  indique que le nombre  $XXX$  est écrit en base  $b$ .
- Représentation usuelle : base 10
- Représentations courantes en informatique : base 2 (**binaire**), base 8 (**octal**) et base 16 (**hexadécimal**)

# Codage des entiers en base $b$

- Soient  $B$  un entier  $\geq 2$  et  $N$  un entier  $\geq 1$
- Tout entier  $p$  compris entre  $0$  et  $B^N - 1$  s'écrit de façon unique sous la forme :

$$p = \sum_{i=0}^{N-1} d_i B^i$$

où les  $d_i$  sont entre  $0$  et  $B-1$

- Le codage de  $p$  en base  $B$  est noté:

$$P = d_{N-1} \dots d_2 d_1 d_0$$

| Bases                   | ai $\in$ {Chiffres}             |
|-------------------------|---------------------------------|
| $b = 2$ ou Binaire      | 0,1                             |
| $b = 8$ ou Octal        | 0,1,2,3,4,5,6,7                 |
| $b = 10$ ou Décimal     | 0,1,2,3,4,5,6,7,8,9             |
| $b = 16$ ou Hexadécimal | 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F |

# Changement de base : passage d'une base b à la base 10

$$N_{(10)} = d_0.b_0 + \dots + d_i.b_i + \dots + d_n.b_n$$

i : rang

$b_i$  : poids du chiffre

Le chiffre le plus à gauche est le chiffre de **poids fort**, celui le plus à droite le chiffre de poids faible.

Exemple :

$$143_{(5)} = 3*5^0 + 4*5^1 + 1*5^2 = 3+20+25 = 48_{(10)}$$

$$AB_{(16)} = 11*16^0 + 10*16^1 = 11+160 = 171_{(10)}$$

$$1001_{(2)} = 1*2^0 + 0*2^1 + 0*2^2 + 1*2^3 = 1+0+0+8 = 9_{(10)}$$

le poids des chiffres en binaire est 1, 2, 4, 8, 16, 32, 64...

→ Avec **n** chiffres on peut donc coder  $b^n$  valeurs en base **b**

Dans un ordinateur on code les nombres sur 8, 16 ou 32 bits

- Avec 8 bits :  $N_{\max} = 2^8 - 1 = 255$ , on peut donc coder 256 valeurs.

Avec 16 bits :  $N_{\max} = 65\,535$

# Changement de base : passage de la base 10 à une base $b$

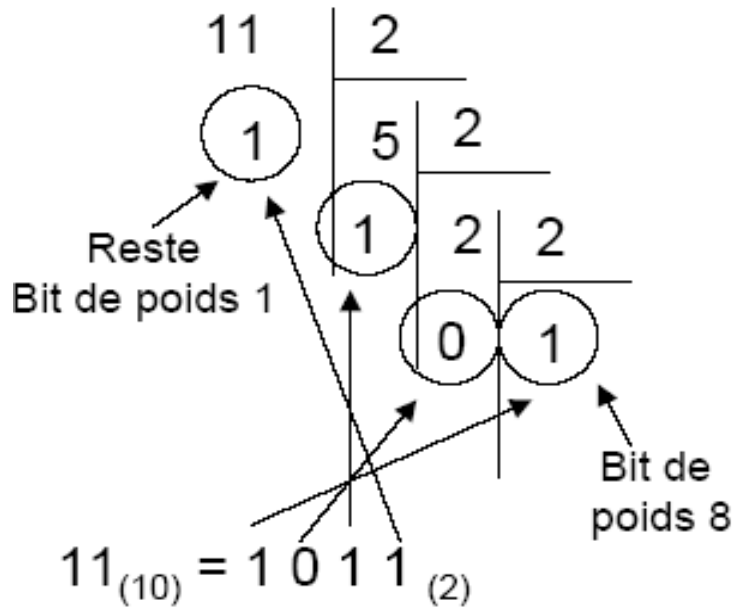
## Algorithme des divisions successives

- On divise le nombre par la base
- puis le quotient par la base
- et ainsi de suite jusqu'à l'obtention d'un quotient nul.
- la suite des restes correspond aux chiffres dans la base visée. Le bit obtenu en premier est celui de poids faible et celui obtenu en dernier est celui de poids fort.

→ Pour convertir un nombre d'une base  $b_1$  vers une base  $b_2$ , il suffit de convertir le nombre de la base  $b_1$  en base 10, puis ce nombre en base 10 en base  $b_2$ .

# Changement de base : passage de la base 10 à une base b

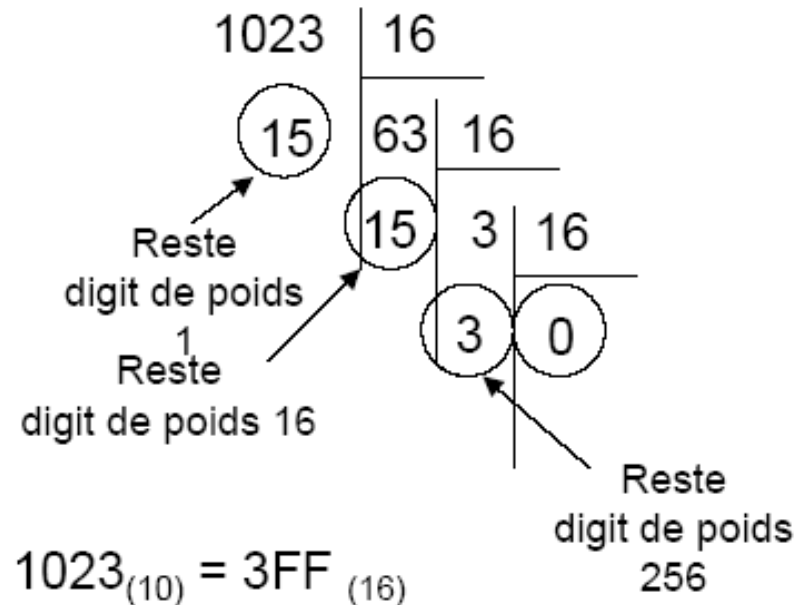
## Conversion de $11_{(10)}$ en base 2



Vérification:

$$1*1 + 1*2 + 0*4 + 1*8 = 11$$

## Conversion de $1023_{(10)}$ en base 16



Vérification :

$$3*16^2 + 15*16 + 15*1 = 1023$$

## Cas particuliers: bases en puissances de 2 (8 et 16)

- Pour les bases en puissances de 2, la conversion est facile.
- Exemple : **base 8 : découpage par blocs de 3 chiffres.**

001 010 011 101

1 2 3 5 =  $1235_{(8)}$

- Exemple : **base 16 : découpage par blocs de 4 chiffres**

0010 1001 1101

2 9 D =  $29D_{(16)}$

- Pour simplifier la notation binaire on utilise en informatique la notation hexadécimale, car en représentation binaire, les nombres peuvent être **très longs**.

→ simplification de la lecture et écriture des adresses mémoires...

## Cas particuliers: bases en puissances de 2 (8 et 16)

- Pour simplifier la notation binaire on utilise en informatique la notation hexadécimale, car en représentation binaire, les nombres peuvent être **très longs**.

→ simplification de la lecture et écriture des adresses mémoires...

```
00000000 fc 31 c0 8e c0 8e d8 8e d0 bc 00 7c 89 e6 bf 00 |.l.....|
00000010 06 b9 00 01 f3 a5 89 fd b1 08 f3 ab fe 45 f2 e9 |.....E..|
00000020 00 8a f6 46 bb 20 75 08 84 d2 78 07 80 4e bb 40 |...F. u...x..N.@|
00000030 8a 56 ba 88 56 00 e8 fc 00 52 bb c2 07 31 d2 88 |.V..V...R...l..|
00000040 6f fc 0f a3 56 bb 73 19 8a 07 bf 87 07 b1 03 f2 |o...V.s.....|
00000050 ae 74 0e b1 0b f2 ae 83 c7 09 8a 0d 01 cf e8 c5 |.t.....|
00000060 00 42 80 c3 10 73 d8 58 2c 7f 3a 06 75 04 72 05 |.B...s.X,...u.r.|
00000070 48 74 0d 30 c0 04 b0 88 46 b8 bf b2 07 e8 a6 00 |Ht.O....F.....|
00000080 be 7b 07 e8 b2 00 8a 56 b9 4e e8 8e 00 eb 05 b0 |.{.....V.N.....|
00000090 07 e8 b0 00 30 e4 cd 1a 89 d7 03 7e bc b4 01 cd |....0.....~....|
000000a0 16 75 0d 30 e4 cd 1a 39 fa 72 f2 8a 46 b9 eb 16 |.u.O...9.r..F...|
000000b0 30 e4 cd 16 88 e0 3c 1c 74 f1 2c 3b 3c 04 76 06 |0.....<.t.;<.v.|
000000c0 2c c7 3c 04 77 c9 98 0f a3 46 0c 73 c2 88 46 b9 |..<.w....F.s..F.|
000000d0 be 00 08 8a 14 89 f3 3c 04 9c 74 0a c0 e0 04 05 |.....<.t.....|
000000e0 be 07 93 c6 07 80 53 f6 46 bb 40 75 08 bb 00 06 |.....S.F.@u....|
000000f0 b4 03 e8 59 00 5e 9d 75 06 8a 56 b8 80 ea 30 bb |...Y.^..u...V...O.|
00000100 00 7c b4 02 e8 47 00 72 86 81 bf fe 01 55 aa 0f |.|...G.r.....U..|
00000110 85 7c ff be 85 07 e8 19 00 ff e3 b0 46 e8 24 00 |.|.....F.$..|
00000120 b0 31 00 d0 eb 17 0f ab 56 0c be 78 07 e8 eb ff |.l.....V..x....|
00000130 89 fe e8 03 00 be 85 07 ac a8 80 75 05 e8 04 00 |.....u....|
00000140 eb f6 24 7f 53 bb 07 00 b4 0e cd 10 5b c3 8a 74 |..$.S.....[.t|
00000150 01 8b 4c 02 b0 01 56 89 e7 f6 46 bb 80 74 13 66 |..L...V...F..t.f|
00000160 6a 00 66 ff 74 08 06 53 6a 01 6a 10 89 e6 48 80 |j.f.t..Sj.j...H.|
00000170 cc 40 cd 13 89 fc 5e c3 20 20 a0 0a 44 65 66 61 |.|@.....^.....Defa|
00000180 75 6c 74 3a a0 0d 8a 00 05 0f 01 06 07 0b 0c 0e |ult:.....|
00000190 83 a5 a6 a9 0d 0c 0b 0a 09 08 0a 0e 11 10 01 3f |.....?|
000001a0 bf 44 4f d3 4c 69 6e 75 f8 46 72 65 65 42 53 c4 |.DO.Linu.FreeBS.|
000001b0 66 bb 44 72 69 76 65 20 00 00 80 8f b6 00 00 00 |f.Drive .....|
000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
000001f0 00 00 00 00 00 00 00 00 00 00 00 00 55 aa |.....U.|
00000200
```

# Représentation des nombres fractionnaires

- Même méthode pour la partie entière
- On garde la virgule
- La partie décimale se décompose en puissances négatives de 2

**Ex :  $(0,75)_{10} = ?$**

$0,75 * 2 = 1,5$  (on garde 1, reste 0,5)

$0,5 * 2 = 1,0$  (on garde 1, reste 0 : terminé)

•  $\rightarrow (0,75)_{10} = (0,11)_2 = 1 * 2^{-1} + 1 * 2^{-2} (= 0,5 + 0,25)$

• **Ex :  $(0,65)_{10} = ?$**

$0,65 * 2 = 1,3$  (1)       $0,4 * 2 = 0,8$  (0)

$0,3 * 2 = 0,6$  (0)       $0,8 * 2 = 1,6$  (1)

$0,6 * 2 = 1,2$  (1)       $0,6 \dots$

$0,2 * 2 = 0,4$  (0)

$\rightarrow (0,65)_{10} = (0,101001)_2$

# Représentation des entiers relatifs

- Problème : comment représenter le signe (-)
- Première méthode : signe ("+" = 0, "-" = 1) + valeur absolue.
  - Exemple: codage de  $(-22)_{10}$  sur 8 bits (1 bit de signe + 7 bits)  
= **1 0010110**
  - Avantage: facile à lire pour l'utilisateur
  - Problèmes:
    - 2 représentations possibles pour 0 (+0 et -0)
    - comment additionner 2 nombres de signes différents ?

## Représentation des entiers relatifs (2)

- **Seconde méthode : complément à 2**

technique de passage de  $X$  à  $-X$  (en complément à 2) :

–on part de  $X$  :

+22  

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|

+1  

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|

on complémente : (0 devient 1, et inversement) ;

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|

-22  

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

-1  

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|

–Inconvénient : plus difficile à interpréter

–Avantages :

- un seul codage pour 0

- pas de problème d'addition

# Représentation des nombres réels (Norme IEEE754)

- Il s'agit de représenter un nombre binaire à **virgule** (par exemple **101,01** qui ne se lit pas *cent un virgule zéro un* puisque c'est un nombre binaire mais **5,25 en décimale**) sous la forme **1,XXXXX... \* 2<sup>n</sup>** (c'est-à-dire dans notre exemple **1,0101\*2<sup>2</sup>**). La norme *IEEE* définit la façon de coder un nombre réel.
- Cette norme se propose de coder le nombre sur **32 bits** et définit trois composantes:
  - **le signe** est représenté par **un seul bit**, le bit de poids fort (celui le plus à gauche).
  - **l'exposant** est codé sur les **8 bits** consécutifs au signe
  - **la mantisse** (les bits situés après la virgule) sur les **23 bits** restants
- Ainsi le codage se fait sous la forme suivante:

seeeeeeeemmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm

le  $s$  représente le bit relatif au signe

les **e** représentent les bits relatifs à l'exposant

les **m** représentent les bits relatifs à la mantisse

# Représentation des nombres réels (Norme IEEE754), suite...

- Certaines conditions sont toutefois à respecter pour les exposants:

Les exposants peuvent varier de

-126 à 127

**Exemple:** Soit à coder la valeur **525,5**

525,5 s'écrit en base 2 de la façon suivante: **1000001101,1**

on normalise cette écriture sous la forme: **1.0000011011** x  $2^9$

Par conséquent: **le bit s vaut 0** (nombre positif)

**l'exposant vaut 9**, soit 1001, auquel on **ajoute 127 (un décalage)**, on obtient (136)<sub>décimal</sub> = **10001000**

la mantisse est **0000011011** qu'on complète avec des zéros pour remplir les **23 bits** qui lui sont réservés, on obtient: **00000110110000000000000**

La représentation du nombre 525.5 en binaire avec la norme IEEE est:

**0 100010000000011011000000000000**

*Pour la reconversion en décimale **enlever** 127 de l'exposant*

# Représentation des caractères (Code ASCII)

Idée = associer un nombre à un caractère

Codage **ASCII** (American Standard Code for Information Exchange)

☐ Codage sur 7 bits

Combien donc de possibilités ?

# Représentation des caractères (Code ASCII)

Idée = associer un nombre à un caractère

Codage **ASCII** (American Standard Code for Information Exchange)

☐ Codage sur 7 bits

Combien donc de possibilités ?

☐ Les caractères 0 à 31 sont des caractères spéciaux : saut de ligne, tabulation, espace, etc.

☐ Les caractères 32 à 127 représentent :

- les minuscules
- les majuscules
- les signes de ponctuation ...

# Table ASCII

| Dec | Hx | Oct | Char                               | Dec | Hx | Oct | Html  | Chr          | Dec | Hx | Oct | Html  | Chr      | Dec | Hx | Oct | Html   | Chr        |
|-----|----|-----|------------------------------------|-----|----|-----|-------|--------------|-----|----|-----|-------|----------|-----|----|-----|--------|------------|
| 0   | 0  | 000 | <b>NUL</b> (null)                  | 32  | 20 | 040 | &#32; | <b>Space</b> | 64  | 40 | 100 | &#64; | <b>@</b> | 96  | 60 | 140 | &#96;  | <b>`</b>   |
| 1   | 1  | 001 | <b>SOH</b> (start of heading)      | 33  | 21 | 041 | &#33; | <b>!</b>     | 65  | 41 | 101 | &#65; | <b>A</b> | 97  | 61 | 141 | &#97;  | <b>a</b>   |
| 2   | 2  | 002 | <b>STX</b> (start of text)         | 34  | 22 | 042 | &#34; | <b>"</b>     | 66  | 42 | 102 | &#66; | <b>B</b> | 98  | 62 | 142 | &#98;  | <b>b</b>   |
| 3   | 3  | 003 | <b>ETX</b> (end of text)           | 35  | 23 | 043 | &#35; | <b>#</b>     | 67  | 43 | 103 | &#67; | <b>C</b> | 99  | 63 | 143 | &#99;  | <b>c</b>   |
| 4   | 4  | 004 | <b>EOT</b> (end of transmission)   | 36  | 24 | 044 | &#36; | <b>\$</b>    | 68  | 44 | 104 | &#68; | <b>D</b> | 100 | 64 | 144 | &#100; | <b>d</b>   |
| 5   | 5  | 005 | <b>ENQ</b> (enquiry)               | 37  | 25 | 045 | &#37; | <b>%</b>     | 69  | 45 | 105 | &#69; | <b>E</b> | 101 | 65 | 145 | &#101; | <b>e</b>   |
| 6   | 6  | 006 | <b>ACK</b> (acknowledge)           | 38  | 26 | 046 | &#38; | <b>&amp;</b> | 70  | 46 | 106 | &#70; | <b>F</b> | 102 | 66 | 146 | &#102; | <b>f</b>   |
| 7   | 7  | 007 | <b>BEL</b> (bell)                  | 39  | 27 | 047 | &#39; | <b>'</b>     | 71  | 47 | 107 | &#71; | <b>G</b> | 103 | 67 | 147 | &#103; | <b>g</b>   |
| 8   | 8  | 010 | <b>BS</b> (backspace)              | 40  | 28 | 050 | &#40; | <b>(</b>     | 72  | 48 | 110 | &#72; | <b>H</b> | 104 | 68 | 150 | &#104; | <b>h</b>   |
| 9   | 9  | 011 | <b>TAB</b> (horizontal tab)        | 41  | 29 | 051 | &#41; | <b>)</b>     | 73  | 49 | 111 | &#73; | <b>I</b> | 105 | 69 | 151 | &#105; | <b>i</b>   |
| 10  | A  | 012 | <b>LF</b> (NL line feed, new line) | 42  | 2A | 052 | &#42; | <b>*</b>     | 74  | 4A | 112 | &#74; | <b>J</b> | 106 | 6A | 152 | &#106; | <b>j</b>   |
| 11  | B  | 013 | <b>VT</b> (vertical tab)           | 43  | 2B | 053 | &#43; | <b>+</b>     | 75  | 4B | 113 | &#75; | <b>K</b> | 107 | 6B | 153 | &#107; | <b>k</b>   |
| 12  | C  | 014 | <b>FF</b> (NP form feed, new page) | 44  | 2C | 054 | &#44; | <b>,</b>     | 76  | 4C | 114 | &#76; | <b>L</b> | 108 | 6C | 154 | &#108; | <b>l</b>   |
| 13  | D  | 015 | <b>CR</b> (carriage return)        | 45  | 2D | 055 | &#45; | <b>-</b>     | 77  | 4D | 115 | &#77; | <b>M</b> | 109 | 6D | 155 | &#109; | <b>m</b>   |
| 14  | E  | 016 | <b>SO</b> (shift out)              | 46  | 2E | 056 | &#46; | <b>.</b>     | 78  | 4E | 116 | &#78; | <b>N</b> | 110 | 6E | 156 | &#110; | <b>n</b>   |
| 15  | F  | 017 | <b>SI</b> (shift in)               | 47  | 2F | 057 | &#47; | <b>/</b>     | 79  | 4F | 117 | &#79; | <b>O</b> | 111 | 6F | 157 | &#111; | <b>o</b>   |
| 16  | 10 | 020 | <b>DLE</b> (data link escape)      | 48  | 30 | 060 | &#48; | <b>0</b>     | 80  | 50 | 120 | &#80; | <b>P</b> | 112 | 70 | 160 | &#112; | <b>p</b>   |
| 17  | 11 | 021 | <b>DC1</b> (device control 1)      | 49  | 31 | 061 | &#49; | <b>1</b>     | 81  | 51 | 121 | &#81; | <b>Q</b> | 113 | 71 | 161 | &#113; | <b>q</b>   |
| 18  | 12 | 022 | <b>DC2</b> (device control 2)      | 50  | 32 | 062 | &#50; | <b>2</b>     | 82  | 52 | 122 | &#82; | <b>R</b> | 114 | 72 | 162 | &#114; | <b>r</b>   |
| 19  | 13 | 023 | <b>DC3</b> (device control 3)      | 51  | 33 | 063 | &#51; | <b>3</b>     | 83  | 53 | 123 | &#83; | <b>S</b> | 115 | 73 | 163 | &#115; | <b>s</b>   |
| 20  | 14 | 024 | <b>DC4</b> (device control 4)      | 52  | 34 | 064 | &#52; | <b>4</b>     | 84  | 54 | 124 | &#84; | <b>T</b> | 116 | 74 | 164 | &#116; | <b>t</b>   |
| 21  | 15 | 025 | <b>NAK</b> (negative acknowledge)  | 53  | 35 | 065 | &#53; | <b>5</b>     | 85  | 55 | 125 | &#85; | <b>U</b> | 117 | 75 | 165 | &#117; | <b>u</b>   |
| 22  | 16 | 026 | <b>SYN</b> (synchronous idle)      | 54  | 36 | 066 | &#54; | <b>6</b>     | 86  | 56 | 126 | &#86; | <b>V</b> | 118 | 76 | 166 | &#118; | <b>v</b>   |
| 23  | 17 | 027 | <b>ETB</b> (end of trans. block)   | 55  | 37 | 067 | &#55; | <b>7</b>     | 87  | 57 | 127 | &#87; | <b>W</b> | 119 | 77 | 167 | &#119; | <b>w</b>   |
| 24  | 18 | 030 | <b>CAN</b> (cancel)                | 56  | 38 | 070 | &#56; | <b>8</b>     | 88  | 58 | 130 | &#88; | <b>X</b> | 120 | 78 | 170 | &#120; | <b>x</b>   |
| 25  | 19 | 031 | <b>EM</b> (end of medium)          | 57  | 39 | 071 | &#57; | <b>9</b>     | 89  | 59 | 131 | &#89; | <b>Y</b> | 121 | 79 | 171 | &#121; | <b>y</b>   |
| 26  | 1A | 032 | <b>SUB</b> (substitute)            | 58  | 3A | 072 | &#58; | <b>:</b>     | 90  | 5A | 132 | &#90; | <b>Z</b> | 122 | 7A | 172 | &#122; | <b>z</b>   |
| 27  | 1B | 033 | <b>ESC</b> (escape)                | 59  | 3B | 073 | &#59; | <b>;</b>     | 91  | 5B | 133 | &#91; | <b>[</b> | 123 | 7B | 173 | &#123; | <b>{</b>   |
| 28  | 1C | 034 | <b>FS</b> (file separator)         | 60  | 3C | 074 | &#60; | <b>&lt;</b>  | 92  | 5C | 134 | &#92; | <b>\</b> | 124 | 7C | 174 | &#124; | <b> </b>   |
| 29  | 1D | 035 | <b>GS</b> (group separator)        | 61  | 3D | 075 | &#61; | <b>=</b>     | 93  | 5D | 135 | &#93; | <b>]</b> | 125 | 7D | 175 | &#125; | <b>}</b>   |
| 30  | 1E | 036 | <b>RS</b> (record separator)       | 62  | 3E | 076 | &#62; | <b>&gt;</b>  | 94  | 5E | 136 | &#94; | <b>^</b> | 126 | 7E | 176 | &#126; | <b>~</b>   |
| 31  | 1F | 037 | <b>US</b> (unit separator)         | 63  | 3F | 077 | &#63; | <b>?</b>     | 95  | 5F | 137 | &#95; | <b>_</b> | 127 | 7F | 177 | &#127; | <b>DEL</b> |

Source: [www.LookupTables.com](http://www.LookupTables.com)