

## Codage numérique : du nombre au pixel - Cours 1

# Codage numérique ?

L1 - Université de Lorraine  
B. Girau et N. de Rugy Altherre

Transparents disponibles sur la plateforme de cours en ligne  
(bientôt)

# Motivations

Pourquoi l'informatique est d'abord une histoire de codage.

## Pourquoi représenter les informations dans un ordinateur ?

- Un ordinateur est essentiellement l'association d'un processeur et d'une mémoire
- La mémoire stocke les informations
- Le processeur va chercher des informations en mémoire, les transforme et place les résultats en mémoire.
- Tout ce que vous faites faire par vos ordinateurs, tablettes, smartphones, etc. passe par ce mécanisme répété des milliards de fois.

## Pourquoi représenter les nombres dans un ordinateur ?

- Beaucoup d'informations qu'un ordinateur manipule correspondent à des nombres.
- De nombreuses autres informations sont représentables par des nombres (exemple : les textes, voir plus loin dans le cours).
- Finalement, toutes les informations vont être représentées sous forme codée, ce qui revient à les représenter par des combinaisons de nombres.

## Pourquoi un ordinateur doit savoir calculer ?

- De nombreuses applications utilisent explicitement des calculs mathématiques :
  - Calcul scientifique (modélisation de phénomènes physiques, prédiction de phénomènes naturels, etc.)
  - Intelligence artificielle (jeux, reconnaissance de visages, etc.)
  - Santé (imagerie médicale, chimie moléculaire, bio-informatique)
  - Economie (modélisation financière, etc.)
  - etc.

... mais surtout ...

- Toutes les transformations opérées par le processeur sont basées sur des calculs arithmétiques et logiques.

## Pourquoi il faut savoir comment l'ordinateur code et calcule

- Etre conscient que l'ordinateur a des limites, fait des approximations.
- La plupart des concepts mathématiques portent sur des ensembles infinis (et souvent continus), alors qu'un ordinateur travaille dans un monde fini (sa mémoire).
- Ne pas faire une confiance aveugle aux résultats obtenus, ni à quelques tests positifs.
- Sinon ...

## Problèmes de spécifications

On code les informations par des valeurs qu'il faut interpréter correctement, sinon ...

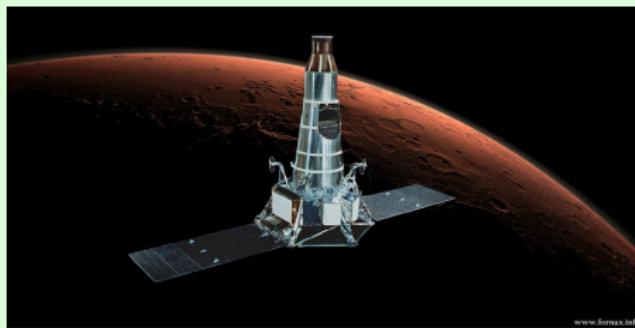
Exemple très connu : crash de la sonde Mars Climate Orbiter



Force de poussée exprimée en Newton ... ou en livres ? !!!

Pourtant il y avait eu des précédents ...

Exemple moins connu : échec de la propulsion vers Mars de Kosmos 419 en 1971.



Un technicien avait programmé la mise à feu du moteur du troisième étage après une durée de 1,5 ... années au lieu de 1,5 heures !

## Problèmes de conversion

Transition entre différents formats de codage : attention aux approximations ou valeurs non représentables, sinon ...

### Exemples

- calculatrice de Windows 3.1 : on tape « 2.01-2.00 », le résultat est « 0.0 »
- Excel (jusqu'à version 7) : on tape « 1,40737488355328 », la case affiche « 0.64 »
- En bien pire : premier lancement d'Ariane 5.

## Autodestruction Ariane 5



Cause : fausse alerte due à la conversion sur un format d'entier d'une valeur de vitesse élevée codée sur un format réel grande précision.

## Problèmes d'exactitude

Même des valeurs « simples » peuvent ne pas être représentées de manière exacte dans l'ordinateur.

### Exemples

- Sur certains supercalculateurs Cray : multiplier  $x$  par 1 (de manière répétée) conduisait à un dépassement de capacité.
- Sur certaines versions du pentium, la division donnait des résultats avec seulement 3 chiffres décimaux justes.

## Problèmes d'exactitude

La résolution de certains problèmes fait appel à des méthodes mathématiques complexes, garanties théoriquement ... en cas de calculs exacts. A utiliser avec précaution, sinon ...

Erreur d'approximation dans le logiciel de calcul des éléments finis pour le dessin des balasts de la plate-forme pétrolière de Gandsfjorden.



## Catastrophe de Gandsfjorden



## Problèmes d'accumulation d'erreurs

Même des erreurs très faibles peuvent devenir critiques quand elles s'accumulent par répétition ou multiplication. Il faut savoir estimer la confiance qu'on peut avoir dans un résultat, sinon ...

- Cas d'école : le banquier infernal.
- Cas concret : en 1982, la bourse de Vancouver décide de faire une troncature à la 3ème décimale après chaque transaction - après 22 mois, une action de 1000\$ valait 525\$ au lieu de 1099\$.

## Patriot bug

Erreur de position d'un missile anti-missile par accumulation d'erreur au cours du temps.



Erreur de 500m en confondant ... 0,1 seconde et 0,100000095 seconde.



Pour quelques bugs célèbres, combien d'erreurs commises dans les programmes qui nous entourent au quotidien ?

D'où l'importance de savoir comment les ordinateurs codent et manipulent les informations.

Motivations  
ooooooooooooooo

Historique  
●ooooooooooo

Codage binaire  
oooooooooooo

Binaire/Hexadécimal  
oooooooo

Un peu d'histoire des nombres et du calcul . . .

*(illustrations tirées des présentations de Nathalie Revol et Arnaud Tisserand)*

## Petit retour en arrière

- origine du mot « calcul » : caillou (*calculus*)
- méthode initiale pour compter
- un caillou = 1
- ou même : les valeurs associées aux cailloux dépendent de leur forme

Motivations  
ooooooooooooooo

Historique  
○○●○○○○○○○○

Codage binaire  
○○○○○○○○○○○○

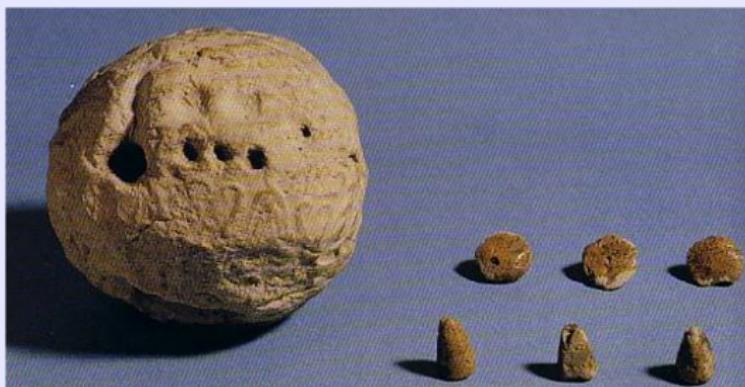
Binaire/Hexadécimal  
○○○○○○○

## Petit retour ... en Mésopotamie

- zone de l'Irak actuel
- différentes civilisations : des Sumériens (environ -3000) aux Babyloniens (environ -2000 à -1000)
- nombreuses traces (notamment via tablettes d'argile) d'une pratique évoluée des Mathématiques

## Petit retour ... en Mésopotamie

- Sumer :

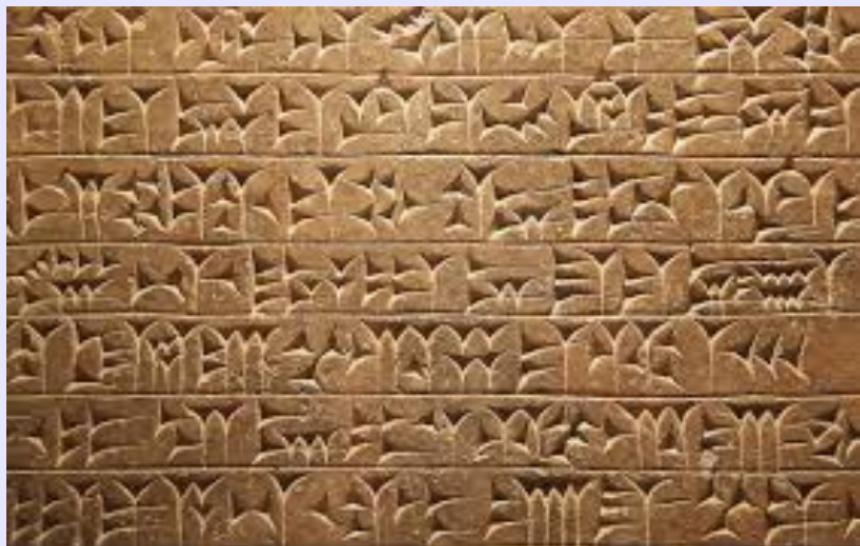


mémorisation exacte (pas vraiment de calcul)

- cône (petit) = 1, bille (petite) = 10, cône (grand) = 60, cône percé = 600, bille (grosse) = 3600, bille percée = 36000
- fondement de la mise au point d'un système sexagésimal ( $\ll$  base  $\gg$  60)

## Petit retour ... en Mésopotamie

- Babylone :



- premier système de position (contrairement au système romain par exemple)

## Système de position

(ou système positionnel)

- Ecriture des nombres constituée de symboles placés à des positions prédéfinies.
- Chaque position est associée à une valeur.
- La valeur associée à une position est liée à celles des positions voisines par un multiplicateur, ou base.
- La valeur associée à un symbole dépend de ce symbole et de la valeur associée à sa position.
- La valeur associée à l'écriture d'un nombre est la somme des valeurs associées aux symboles.

## Intérêt d'un système de position

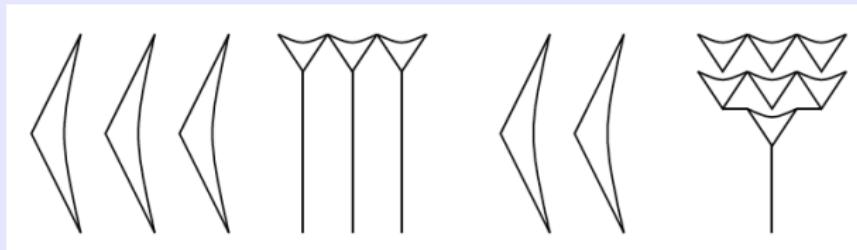
- Exercice (en chiffres romains) : multipliez *CXXI* par *XXV*
- ...
- ... sans tricher ...
- ...
- ... heuuuu ! ...

## Petit retour ... en Mésopotamie

- Babylone :

1	2	3	4	5	6	7	8	9	10

- écriture base 60 de 2007 ( $= 1980 + 27 = 33 \times 60 + 27$ ) :



## Petit retour ... en Mésopotamie

- Table de multiplication par 25 :





## Petit retour ... sur l'intérêt d'un système de position

- Exercice : multipliez 121 par 25 dans le système babylonien
- ...
- ... sans tricher ...
- ...
- Convaincus ?

## Calculs dans un système de position

Description par des mécanismes systématiques et généralisables, donc des algorithmes.

## Petit retour ... jusqu'à nous

- Héritage : 60 secondes dans une minutes, 60 minutes dans une heure, tour complet de 360 degrés, etc.
- Après les babyloniens :
  - chez les Mayas : système de position base 20
  - en Chine : esquisse d'un système de position décimal (base 10, pratique avec 10 doigts)
  - en Inde : finalisation, écriture qui a inspiré les chiffres arabes (et donc notre écriture des nombres depuis le X<sup>ème</sup> siècle)
- Et les ordinateurs ? ... ils n'ont pas dix doigts !

# Codage binaire

ou . . . comment compter avec un seul doigt ?

## Binaire

- C'est bien connu, dans un ordinateur, il n'y a que des 0 et des 1, ...
- ... et pourtant, n'importe quel informaticien sait compter de 0 à l'infini.
- La preuve :

## Binaire

- C'est bien connu, dans un ordinateur, il n'y a que des 0 et des 1, ...
- ... et pourtant, n'importe quel informaticien sait compter de 0 à l'infini.
- La preuve : 0,

## Binaire

- C'est bien connu, dans un ordinateur, il n'y a que des 0 et des 1, ...
- ... et pourtant, n'importe quel informaticien sait compter de 0 à l'infini.
- La preuve : 0, 1,

## Binaire

- C'est bien connu, dans un ordinateur, il n'y a que des 0 et des 1, ...
- ... et pourtant, n'importe quel informaticien sait compter de 0 à l'infini.
- La preuve : 0, 1, ... fini.

## Etape primordiale : coder les informations

- codage des informations sur ordinateurs : entiers, réels, textes, sons, images, vidéos, ...
- base commune : codage des entiers, opérations sur les entiers

## Un entier, c'est quoi ?

- Un entier (naturel) est une valeur définie mathématiquement comme un « ordinal fini ».
- Plus simplement, les entiers (naturels) sont les successeurs de « zéro » (identifié à l'ensemble vide).

## Exemple

- « Trois » est le successeur du successeur du successeur de zéro.
- « 3 » n'en est qu'une écriture (codage) à laquelle nous sommes habitués
- pourquoi ne pas l'écrire « 11 », ou même « 100001 » ?

## codage binaire

- système de position, base 2
- permet des manipulations algorithmiques
- 2 chiffres (binaires)
- attention : une écriture binaire (à base de 0 et 1 comme dans un ordinateur) n'implique pas un codage binaire

Pourquoi les systèmes de position sont possibles :

## Décomposition en base $B > 1$

- Soit  $B$  un entier naturel  $> 1$ . Pour tout entier naturel  $x$  il existe une unique suite d'entiers  $(x_i)$  telle que :
  - $\forall i \quad 0 \leqslant x_i < B$
  - $x = x_0 + x_1.B + x_2.B^2 + x_3.B^3 + \dots$
- C'est la décomposition de  $x$  en base  $B$ .
- $(B-1)+(B-1).B+(B-1).B^2+\dots+(B-1).B^{n-1}=B^n-1$

## Exemple

- En base 10 :  $1789 = 9 \times 1 + 8 \times 10^1 + 7 \times 10^2 + 1 \times 10^3$
- En base 2  
 $84 = 0 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6$

Ecriture en base  $B$  : les chiffres sont les  $x_i$  de 0 à  $B - 1$

## Poids forts/Poids faibles

Si la décomposition de  $x < B^n - 1$  dans la base  $B$  est la suite  $x_0, x_1, \dots, x_{n-1}$ , alors deux écritures existent :

- Poids fort en premier (en tête) :  $x = (x_{n-1}x_{n-2} \dots x_0)$
- Poids faible en premier (en tête) :  $x = (x_0x_1 \dots x_{n-1})$

La notation usuelle est poids fort en premier :

$$1666 = (1 \times 10^3 + 6 \times 10^2 + 6 \times 10^1 + 6)$$

On parle de notation grandboutienne (ou big endian en anglais).

## Notations

Si la décomposition de  $x < B^n - 1$  dans la base  $B$  est la suite  $x_0, x_1, \dots, x_{n-1}$ , alors on note, en notation poids forts d'abord :

$$x = x_{n-1} \dots x_1 x_0 B$$

Par exemple :

$$\text{"1987"} = 1987_{10} = 011111000011_2$$

## bit/octet

- en codage binaire, les chiffres (bits) sont 0 et 1
- un octet est un groupe de 8 bits

## Valeurs remarquables

- sur  $n$  bits :  $10 \dots 0_2 = "2^{n-1}"$
- sur  $n$  bits :  $11 \dots 1_2 = "2^n - 1"$
- sur  $n$  bits : on code donc tous les entiers de 0 à  $2^n - 1$

## Exemples

- sur un octet : les entiers naturels de 0 à  $255_{10}$
- sur 4 octets :  $100000000000000000000000000000_2 = 2^{31} = 2147483648_{10}$

## Binaire/hexadécimal

- $B = 2$  /  $B = 16$
- chiffres hexadécimaux : 0, 1, 2, ..., 9, A, B, C, D, E, F

## Exemples

$$1492_{10} = 5D4_{16} = (5 \times 16^2 + 13 \times 16 + 4)$$

$$1001_{16} = (1 \times 16^3 + 1) = 4097_{10}$$

L'écriture hexadécimale est souvent utilisée pour écrire de manière plus compacte et lisible des contenus binaires.

## Binaire/hexadécimal

Différentes notations existent pour noter le binaire/hexadécimal :

- $(x_{n-1} \dots x_1 x_0)_{16} = 0x(x_{n-1} \dots x_1 x_0)$
- $(x_{n-1} \dots x_1 x_0)_2 = 0b(x_{n-1} \dots x_1 x_0)$
- $(x_{n-1} \dots x_1 x_0)_{10} = 0d(x_{n-1} \dots x_1 x_0)$

## Exemple

$$0d1802 = 0x70A = 0b11100001010$$

## Les conversions : décimal à binaire

On cherche le plus grand  $n$  tel que  $2^n \leq x$ . Alors  $x_n = 1$  et on ré-applique le processus à  $x - 2^n$  jusqu'à trouver 0.

### Exemple

Si  $x = 11_{10}$ ,  $n = 3$  car  $2^3 = 8$  et  $2^4 = 16$ . D'où  $x_3 = 1$  et on recommence avec  $x = 11_{10} - 8_{10} = 3_{10}$ . On trouve alors  $n = 1$ , donc  $x_1 = 1$  et on recommence avec  $x = 1$ .

En conclusion  $11_{10} = 1011_2$ .

## Les conversions : décimal à hexadécimal

On cherche le plus grand  $n$  tel que  $16^n \leq x$ . Alors  $x_n = x / 16^n$  et on ré-applique le processus à  $x - x_n \times 16^n$  jusqu'à trouver 0.

### Exemple

Si  $x = 1111_{10}$ ,  $n = 2$  car  $16^2 = 256$  et  $16^3 = 4096$ . D'où  $x_2 = 4$  et on recommence avec  $x = 1111_{10} - 4 \times 256_{10} = 87_{10}$ . On trouve alors  $n = 1$  et  $x_1 = 5$  et on recommence avec  $x = 7$ .

En conclusion  $1111_{10} = 457_{16}$ .

## Les conversions : binaire à hexadécimal

On rassemble les bits par 4, en commençant par les poids faibles.  
Chaque groupe de 4 bits est converti en un chiffre hexadécimal  
indépendamment.

### Exemple

$0b1111010101 = 0b0011\ 1101\ 0101 = 0x3D5$

## Les conversions : hexadécimal à binaire

Les chiffres hexadécimaux sont convertis individuellement.

### Exemple

$$0xD90 = 0b1101\ 1001\ 0000$$

## Les conversions : vers décimal

Les conversions d'une base  $B$  vers le décimal s'effectue en reprenant la définition.

### Exemple

$$0x5B0 = 5 \times 16^2 + 11 \times 16 = 1456$$

Motivations

ooooooooooooooo

Historique

ooooooooooooo

Codage binaire

ooooooooooooo

Binaire/Hexadécimal

ooooooo●

Il ne reste plus qu'à s'entraîner.