جامعة محمد الخامس
كلية العلوم الرباط
Université Mohammed V
Faculté des Sciences
Rabat

Architecture des Ordinateurs

Mme Soumia ZITI
Professeur d'Enseignement Supérieur
Département d'Informatique
Faculté des Sciences
Université Mohammed V
RABAT

2024/2025

1

Plan

- Introduction
- Codage d'Information
- Logique combinatoire et séquentielle
- Architecture de Von Neumann
- Processeur 80x86
- Assembleur
- Entrées/sorties

2

Introduction

But de l'enseignement

- Quels sont le codage de l'information et transcodage?
- De quoi est composé un ordinateur ?
- Quels sont les modèles sous-jacents au fonctionnement d'une machine ?
- Comment s'exécutent les programmes ?
- Quel est le lien entre le logiciel et le matériel ?
- Comment fonctionnent les divers périphériques ?
- Comment programmer en Assembleur?

3

Introduction

Ordinateur

- **Besoin**: Calcul plus complexe et plus rapide => Automatisation du calcul
- **Historique**:
  - XVIIe siècle et avant : les principes fondateurs
  - XIXe siècles : les calculateurs
  - XXe siècle : théorie de l'information + machine universelle
  - ~1945 : Architecture de Von Neumann et naissance de l'ordinateur
  - ~1950 : 1ere génération : tubes a vides
  - ~1960 : 2eme génération : transistors
  - ~1970 : 3eme génération : circuits intègres
  - ~1980 : 4eme génération : puces avec des millions de transistors

4

Introduction

Naissance de l'ordinateur

- **Claude Shannon (1948) :** chiffres binaires pour les relations logiques et les calculs logiques et arithmétiques (Tout calcul peut être réalisé avec les 3 opérations logiques de base ET, OU, NON
- **Alan Turing :** machine universelle ou Machine de Turing décrivant un modele abstrait du fonctionnement des appareils mécaniques de calcul => Invente les concepts de programmation et de programme
- **John Von Neumann (1945) :** Enregistrer le programme en mémoire => Architecture de l'ordinateur moderne : l'architecture de Von Neumann

5

Introduction

Ordinateur

- Une **machine** de traitement de **l'information** (acquérir, conserver, traiter et restituer). Il est capable d'effectuer **automatiquement** des opérations arithmétiques et logiques à partir de programmes définissant la séquence de ces opérations.
- C'est un ensemble de **circuits** électroniques permettant de manipuler des **données** sous forme **binaire**, ou bits afin d' exécuter des séquences de calculs ou des traitements de tout genre.

6

Introduction

Information

- Un ensemble de **données** qui a un sens précis
- Des **valeurs** numériques, textes, images, son, vidéos représentés sous forme de données.
- Des **instructions** composant un programme.
- Toute information est manipulée sous forme **binaire** (ou numérique) par un système informatique.

7

Introduction

Informatique

- Terme employé pour la première fois en 1962 et provenant des mots « **Information** » et « **automatique** ». C'est la **science** du **traitement rationnel et automatique de l'information**, considérée comme le support des connaissances dans différents domaines .

Objectifs

- Faciliter et accélérer le calcul,
- Automatiser les traitement des données
- Contrôler et commander des processus,
- Faciliter la communication entre plusieurs composants
- Partager des informations et des ressources.

8

Introduction

Système informatique

- Ensemble des moyens **logiciels** et **matériels** nécessaires pour **satisfaire** les besoins informatiques des **utilisateurs**.
- ***Un système informatique est capable de:***
  - **Acquérir** des informations nécessaires pour les **calculs** et les traitements
  - **Sauvegarder** les données d'une façon **permanente** pour des traitements ultérieurs sur des **supports** de stockage
  - **Effectuer** des traitements des **données** et des **calculs** simples ou complexes
  - **Restituer** les **données** au cas de besoin

9

Introduction

Programmation

- A partir d'un **problème donné**, réaliser un **programme** dont l'exécution apporte **une solution satisfaisante** au problème posé suivant un **algorithme** bien précis et moins complexe
- Elle est effectuer en utilisant un **langage** de **programmation** comme le **langage machine**, **l'assembleur** ou un **langage** évolués (traduction de l'algorithme)
- Elle fait partie de **l'ingénierie de développement logiciel** (**implémentation ou code**)

10

Introduction

Langage de programmation

- C'est **l'intermédiaire** entre **l'humain et la machine**, il permet d'écrire, dans un langage proche de la machine mais intelligible par l'humain, toutes les opérations que l'ordinateur doit effectuer.
- il doit donc respecter une **syntaxe stricte**. Un **algorithme** peut toutefois aboutir à **plusieurs programmes**.
- Un **langage informatique** est destiné à décrire l'ensemble des **actions consécutives** qu'un ordinateur doit **exécuter**. C'est une façon pratique de **donner** des **instructions à un ordinateur**.

11

Introduction

Familles de langage de programmation

- **Langages fonctionnels:** (ou langage procédural) est un langage dans lequel le programme est construit par fonctions, retournant un nouvel état en sortie et prenant en entrée la sortie d'autres fonctions par exemple

=> diviser un problème complexe en sous-problèmes plus simples. Lorsqu'une fonction s'appelle elle-même, on parle alors de récursivité.

- **Langages objets:** part du principe que des choses peuvent avoir des points communs, des similarités en elles-mêmes ou en leur façon d'agir. L'idée est regrouper de tels éléments afin d'en simplifier leur utilisation.

=>Un regroupement est appelé classe, les entités qu'il regroupe sont appelées objets (définition des actions pour toute une classe et chaque objet pourra les effectuer)

12

Introduction

Programme

- Suite **d'instructions** dans un **langage** donnée, définissant un des actions spécifiques exécutables par un ordinateur
  - programmes systèmes (système d'exploitation gérant différents ressources machine)
  - programmes d'application (des logiciels de traitements)
- Un programme est composé de deux partie:
  - La partie contenant les données
  - La partie contenant le code des instructions à exécuter
- Les **instructions** sont des **opérations** de base que l'ordinateur peut traiter comme l'addition, la multiplication la comparaison…

13

Introduction

Microprocesseur

- C'est un **circuit** électronique intégré complexe et miniaturisé contenant plusieurs millions de **transistors** interconnectés(ex : le Pentium).
- C'est le **cœur** de l'ordinateur qui permet de traiter et distribuer les informations.
- Il résulte de l'intégration sur une puce de **fonctions logiques combinatoires** (logiques et/ou arithmétique) et **séquentielles** (registres, compteur, etc…).
- Il exécute les instructions élémentaires au rythme de son **horloge interne** (ex : 300 Mhz ou mégahertz => 300 millions d'instructions par seconde).

14

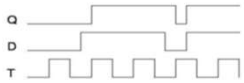
Introduction

Horloge

- Synchronisation de l'ensemble des dispositifs logiques d'un ordinateur.
- Cadencement des instructions à fréquence constante : l'horloge divise le temps en battements de même durée appelés cycles.
- une fréquence d'horloge à 500MHz: des cycles élémentaires de 2 nanosecondes.

un signal est une grandeur discrète appartenant à [0,1]

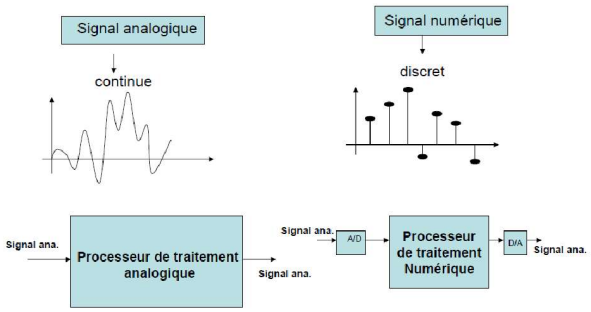
un chronogramme est la représentation graphique d'un signal évoluant dans le temps



15

Introduction

Représentation des grandeurs

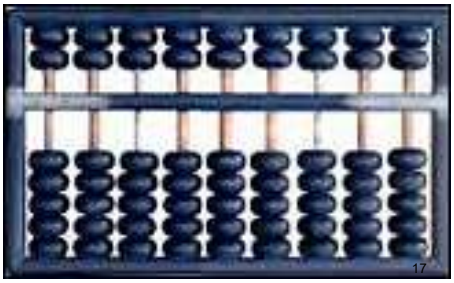


16

Introduction

Historique:80 AC

- **Abaques**, machine pour prédire le mouvement des astres
- Le **boulier** est un outil de calcul formé d'un cadre rectangulaire muni de tiges sur lesquelles coulissent des boules.

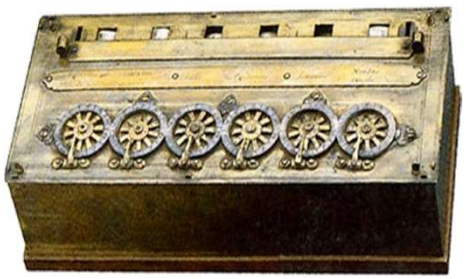


17

Introduction

Historique: 17ème

- **Pascal** : machine à calculer (Pascaline)
- **Leibniz** : système binaire pour le calcul



18

Introduction

Historique: 18ème

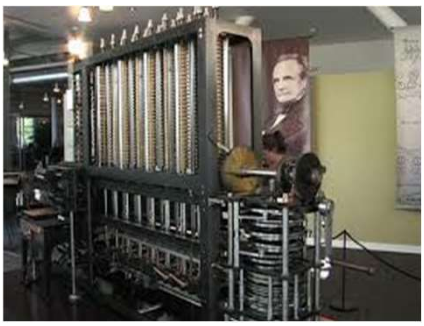
- **Jacquard** : métier à tisser en 1801, premier système mécanique programmable avec cartes **perforées**.

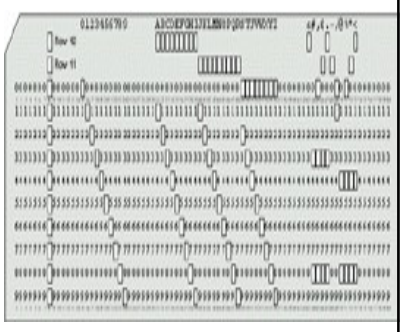
19

Introduction

• 18ème

- **Babbage** : machine à différence, est une calculatrice mécanique conçue pour calculer des tables de fonctions polynomiales





20

Introduction

Historique: 18ème

- **Babbage** : machine analytique différentielle, c'est une machine à calculer programmable imaginée en 1834

21

Introduction

• 19ème

- **Boole** : calcul **binaire** et calcul **logique**. l'algèbre de Boole trouve de nombreuses applications en **informatique** et dans la conception des **circuits électroniques**.

22

Introduction

Historique: Premier Ordinateur

- **L'ENIAC** (Electronic numerical Integrator And Computer) a été conçu et construit sous la direction de Mauchly et Eckert à l'université de Pennsylvanie.
  - La construction de l'ENIAC a commencé en 1943 et s'est achevé en 1946.
  - L'ENIAC a fonctionné jusqu'en 1955 date de désassemblage.

23

Introduction

Historique: L'ENIAC:

- **Poids** : 30 tonnes.
- **Encombrement** : 500m² au sol.
- 18 000 **tubes à vide.**
- **Consommation** : 140KW.
- **Performances** : 5 000 additions par seconde.

•➔ C'était un ordinateur décimal programmé manuellement en positionnant des commutateurs et en branchant et débranchant des câbles.

24

Introduction

Historique: Transistor

- Inventé par **Baarden**, **Brattain** et **Shockley** en 47
- Il est plus petit., moins cher.et produit moins de chaleur.
- Les premiers ordinateurs entièrement transistorisés ont vu le jour en 1950.
- De 1950 à 1960 les équipements électroniques étaient composés essentiellement de composants discrets : Transistors, résistances et condensateurs.

25

Introduction

Historique: Circuit intégré

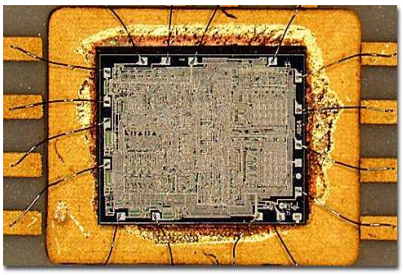
- Les tous premiers ordinateurs contenaient environ **10 000** transistors.
- L'ensemble du processus de fabrication, depuis le transistor jusqu'à la carte à circuits imprimés était **coûteux** et **complexe**.
- Chaque composant discret étaient fabriqués **séparément**, **intégrés** à son propre boîtier.
- Les différents composants étaient alors **soudés** et **câblés** ensemble sur des cartes à **circuits imprimés** que l'on installait à l'intérieur des ordinateurs.
- En 1958, nouvelle révolution dans le monde de l'électronique avec la naissance de l'ère de la **micro électronique** : l'invention du **circuit intégré**.

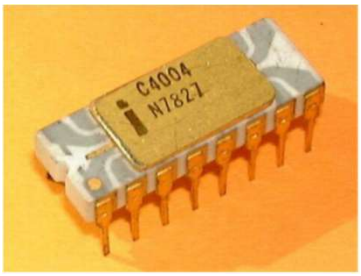
26

Introduction

Microprocesseur

- **1971** : Innovation majeure, développement par Intel du premier **microprocesseur** le 4004.
- Le **4004** pouvait ajouter **2** nombres de **4 bits** et ne pouvait multiplier que par répétition d'additions.,



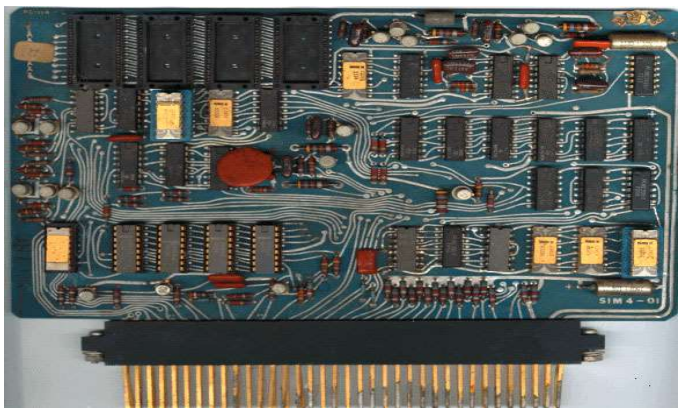


27

Introduction

Microprocesseur

- Ce microprocesseur intègre les opérations logiques, arithmétiques, la mémoire….



28

Introduction

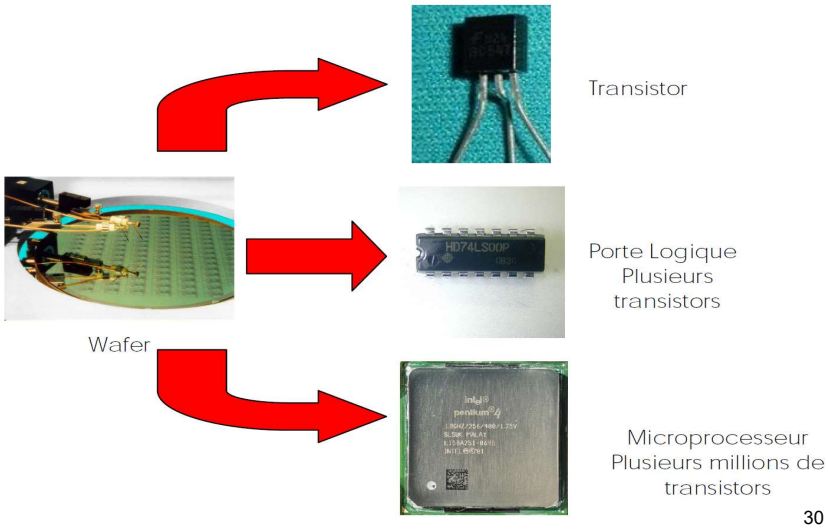
Microprocesseur

- Un circuit entier est fabriqué dans un petit morceau de silicium au lieu **d'assembler** dans un même circuit des composants discrets fabriqués à partir de différents morceaux de silicium
- De nombreux transistors peuvent être placés sur une tranche de **silicium**.
- Ces transistors peuvent être connectés par un processus de **métallisation** afin de former des circuits.
- Au début on arrivait à fabriquer et à assembler de façon fiable que **quelques** portes ou cellules mémoire.
- Avec le temps, il a été possible d'intégrer **de plus en plus** de composants au sein d'une **même puce**.

29

Introduction

Microprocesseur



30

Introduction

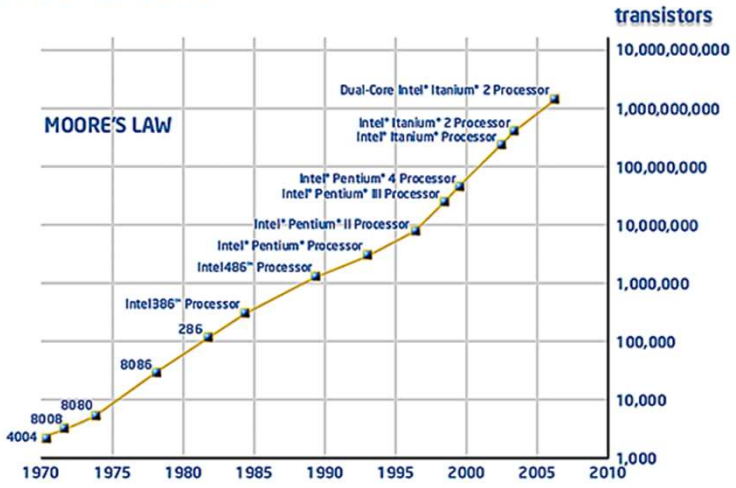
Loi de Moore

- Lors de la préparation de son discours en 1965, **Gordon Moore** (un des Présidents d'Intel) fit une remarque qui reste toujours d'actualité.
  - le nombre de transistors des processeurs devrait **doubler tous les 18 mois** et permettre ainsi une croissance exponentielle régulière des performances. Cette loi s'est vérifiée au fil du temps, et elle permet d'avoir un bon ordre de grandeur des performances des futurs processeurs.
  - Exemple 6000 mille transistors en 1974, 9,5M en1999

31

Introduction

Loi de Moore



32

Introduction

Microprocesseur

- **1972** : Introduction par Intel du 8008, le premier microprocesseur 8 bits.
- **1974** : Lancement par Intel du 8080, le premier microprocesseur «multi-usages».
  - Plus rapide.
  - Jeu d'instructions plus complexe.
- **1978** : Premier microprocesseur 16 bits, le 8086.

8086

33

Introduction

Microprocesseur

- **1981** : Premier microprocesseur 32 bits en une seule puce développé par Bell Labs et Hewlett Packard.
- **1985** : Intel introduit son microprocesseur 32 bits le 80386.

- **1987**: Intel introduit son microprocesseur 80486 semblable au 80386 avec quelques instructions supplémentaires et une microarchitecture avancée

34

Introduction

Le Pentium

- La vitesse brute du microprocesseur ne peut atteindre son plein potentiel que si la puce est alimentée constamment par un flot de travaux à exécuter.
- Toute entrave à ce flot régulier réduit la puissance du processeur.
- Aujourd'hui, le principal problème est la vitesse à laquelle il est possible de transférer les données entre la mémoire et le processeur.
- Difficile dans ces conditions d'alimenter en instructions et en données le processeur.

35

Introduction

Le Pentium

- **En1993** Intel introduit son microprocesseur 64 bits le Pentium.
- La vitesse brute du microprocesseur ne peut atteindre son plein potentiel que si la puce est alimentée constamment par un flot de travaux à exécuter.
  - Toute entrave à ce flot régulier réduit la puissance du processeur.
  - Aujourd'hui, le principal problème est la vitesse à laquelle il est possible de transférer les données entre la mémoire et le processeur.
  - Difficile dans ces conditions d'alimenter en instructions et en données le processeur.

36

Introduction

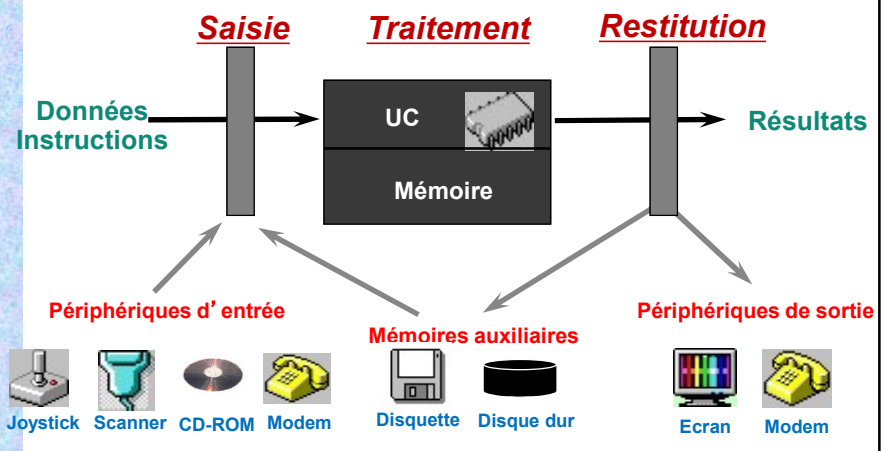
Le Pentium

- Augmenter la taille du **bus de données** : augmenter le nombre de bits récupérés simultanément.
- Utiliser plusieurs **types** de mémoires :
- Des mémoires **très rapides** mais de **faible capacité** à proximité du processeur.
- Des mémoires **moins rapides** mais de **grande** capacité.
- L'histoire et l'avenir des ordinateurs est donc lié à celui des **mémoires**.

37

Introduction

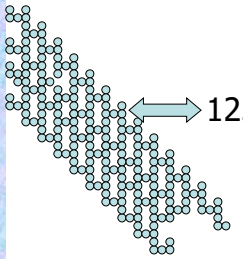
Schéma fonctionnel d'un ordinateur



38

Codage de l'information

Codage et transcodage



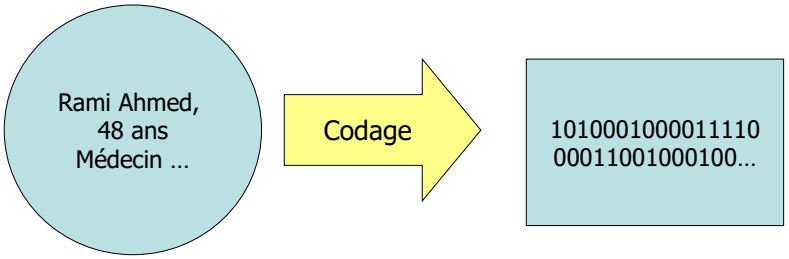
123 ⟺ Cent vingt trois ⟺ CXXIII

Règles permettant de passer d'une représentation à une autre

39

Codage de l'information

Fichier



Des données…

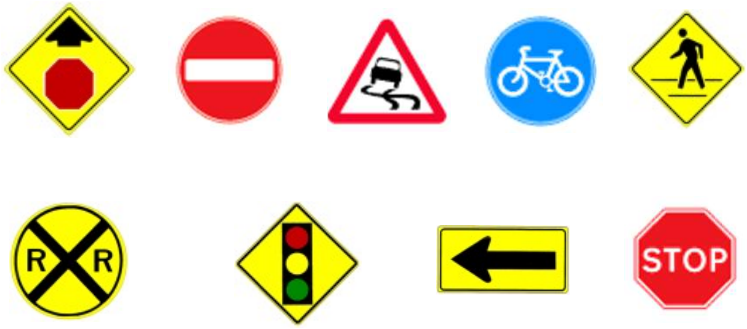
… stockées en machine dans un fichier

40

Codage de l'information

Signes

- *Les signes peuvent être visuels : couleur, forme, dessin*



41

Codage de l'information

Signes

- *les signes peuvent être sonores : sonnette, bruit, applaudissements, musique, discours*

42

Codage de l'information

Signes

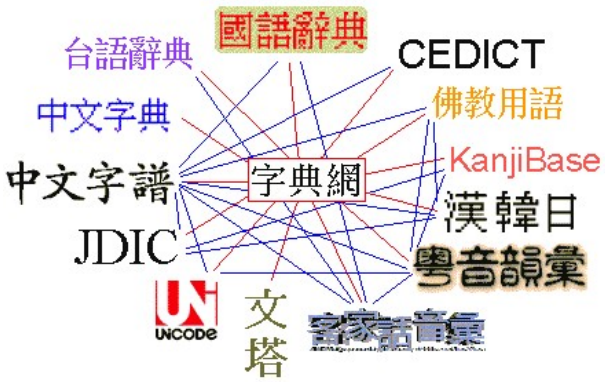
- *Les signes peuvent être olfactifs*

43

Codage de l'information

Ecriture

- *Le Chinois utilise plus de 80,000 caractères pour coder son langage*



44

Codage de l'information

Ecriture

- *Les Egyptiens utilisaient les hiéroglyphes pour coder les sons et les mots*

45

Codage de l'information

Ecriture

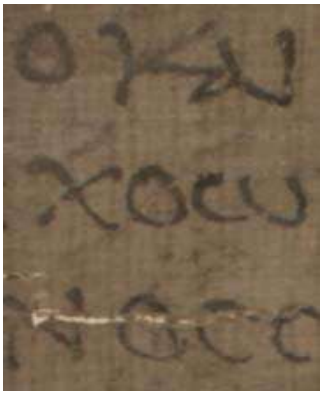
- *Le Japanais utilise les 96 caractères Hiragana pour coder les syllabes*

おこそとのほもよろを
えけせてねへめ　れ
うくすつぬふむゆる
いきしちにひみ　り
あかさたなはまやらわん

46

Codage de l'information

- **Ecriture**
  - *Les Phéniciens et les Grecs ont découvert qu'un alphabet de 23 caractères peuvent coder les sons élémentaires*



α β γ δ ε ζ η θ
ι κ λ μ ν ο π ρ
σ τ υ ϕ χ ψ ω

47

Codage de l'information

- **Ecriture**
  - *George BOOLE (1815-1864) utilisait seulement deux caractères pour coder les opérations logiques*

0 1

48

Codage de l'information

Ecriture

- *John von NEUMANN (1903-1957)*
  développa le concept de programmation utilisant aussi un système binaire pour coder toute information

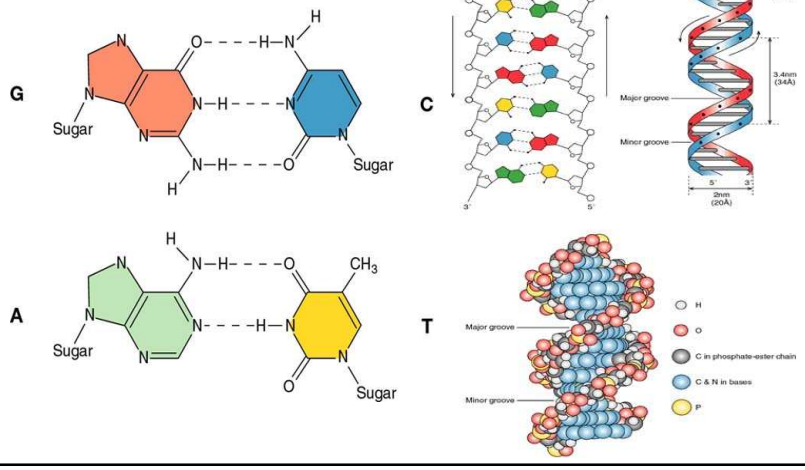
000111

49

Codage de l'information

Ecriture

- *La nature utilise 4 molécules*



Codage de l'information

représentation d'information

- Les **informations traitées par les ordinateurs** sont de différentes natures :
  - **nombres, texte,**
  - **images, sons, vidéo,**
  - **programmes, …**
- Dans un ordinateur, elles **sont toujours** représentées sous forme binaire (**BIT : Binary digIT**)
  - **une suite de 0 et de 1**

51

Codage de l'information

représentation d'information

- **En informatique, tout s'exprime sous forme de bits (0 ou 1)**
- Le **BIT (Binary digIT)** a pour valeur **0 ou 1**
  - Au niveau électronique : **0 Volt / +5 Volts**
  - Au niveau magnétique : **champ magnétique / champ magnétique inverse**
- **Octet** (Byte) = regroupement de 8 bits

Exemple : 7 codé sur un octet

0	0	0	0	0	1	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

52

Codage de l'information

représentation d'information

- permet d'établir une **correspondance** qui permet **sans ambiguïté** de passer d'une représentation (dite **externe)** d'une information à une autre représentation (dite **interne** : sous forme binaire) de la même information, suivant un ensemble de **règles précises.**
- **Exemple :**
  - * Le nombre 35 : **35** est la représentation **externe** du nombre **trente cinq**
  - * La représentation **interne** de 35 sera une suite de 0 et 1 (**100011**)

53

Codage de l'information

Etapes de codage de l'information

- **En informatique, Le codage de l'information s'effectue principalement en trois étapes :**
  - **L'information sera exprimée par une suite de nombres (Numérisation)**
  - **Chaque nombre est codé sous forme binaire (suite de 0 et 1)**
  - **Chaque élément binaire est représenté par un état physique**

54

Codage de l'information

Elément binaire vers Etat physique

- Codage de l'élément binaire par un état physique
  - Charge électrique (RAM : Condensateur-transistor) : Chargé (bit 1) ou non chargé (bit 0)
  - Magnétisation (Disque dur, disquette) : polarisation Nord (bit 1) ou Sud (bit 0)
  - Alvéoles (CDROM): réflexion (bit 1) ou pas de réflexion (bit 0)
  - Fréquences (Modem) : dans un signal sinusoïdal
    - Fréquence f_1 (bit 1) : $s(t) = a \sin (2\pi f_1 t + \psi)$
    - Fréquence f_2 (bit 0) : $s(t) = a \sin (2\pi f_2 t + \psi)$

55

Codage de l'information

Système de numération

❑Système de numération décrit la façon avec laquelle les nombres sont représentés.

❑Un système de numération est défini par :

- Un alphabet A : ensemble de symboles ou chiffres,
- Des règles d'écritures des nombres : Juxtaposition de symboles

56

Codage de l'information

- **Numération Romaine**

système romain	I	V	X	L	C	D	M
valeur décimal	1	5	10	50	100	500	1000

- Lorsqu'un symbole est placé à la **droite** d'un symbole plus fort que lui, sa valeur **s'ajoute** : **CCLXXI → 271**
- Lorsqu'un symbole est placé à la **gauche** d'un symbole plus fort que lui, on **retranche** sa valeur : **CCXLIII → 243**
- On ne place jamais **4 symboles** identique à la suite : 9 s'écrit IX et non VIIII
- Le plus grand nombre exprimable est : **3999 (MMMCMXCIX)**➔Système inadapté au calcul

57

Codage de l'information

- **Numération babylonienne**

- Les Babyloniens (2000 ans av.J.C.), ont utilisé les symboles le **clou** pour l'**unité** et le **chevron** pour les **dizaines**. C'est un **système de position**.

2	9	12	53
𒐗	𒐝	𒐞	𒐞𒐗

- A partir de 60, la position des symboles entre en jeu :
  - **204 =3*60+24** 𒐗 𒐞
  - **7392 =2*60²+3*60+13** 𒐗 𒐗 𒐞

- Le nombre 60 constitue la base de ce système.

58

Codage de l'information

Les chiffres arabes

- Ce sont les arabes qui ont crée le **"cifre"** traduit par la suite en **"zéro''**.
- Le mot chiffre est un dérivatif du mot cifre
- Ils ont créer les chiffres (**1 2 3 4 5 6 7 8 9 0**) par opposition aux chiffres **romains** (**I II III IV V VI** ...) et aux chiffres **hindous** (١٢٣٤٥٦٧٨٩) qui manquent de ZERO,

59

Codage de l'information

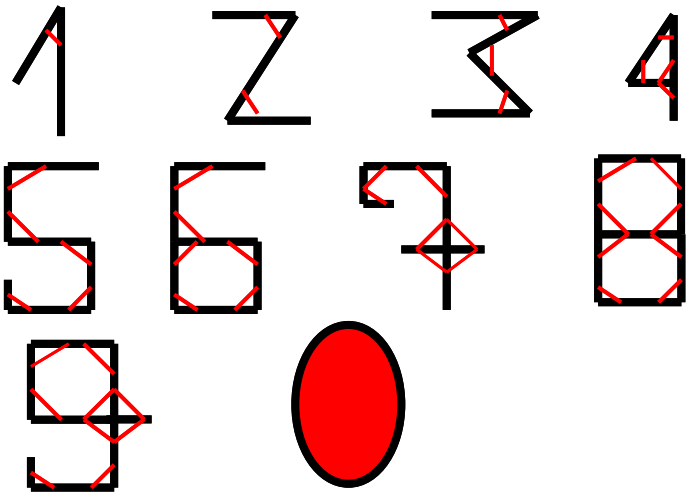
Les chiffres arabes

- Il fallait représenter les chiffres en arabe de manière à retrouver une **correspondance** entre la valeur du chiffre et sa **typographie** ou sa **représentation symbolique.**
- Suivre une logique des plus **scientifique** en utilisant les angles:
  - **un seul angle dans le UN (1)**
  - **deux angles dans le DEUX (2 = Z)**
  - **trois angles dans le TROIS (3)…**
- Et la seule forme géométrique pour représenter le rien en tant que chiffre est le **cercle** ou le **cifre** (zéro)

60

Codage de l'information

Les chiffres arabes



61

Codage de l'information

Numération décimale :

- ❑ **C'est le système de numération le plus pratiqué actuellement.**
- ❑ **L'alphabet est composé de dix chiffres :**
  - **A = {0,1,2,3,4,5,6,7,8,9}**
- ❑ **Le nombre 10 est la base de cette numération**
- ❑ **C'est un système positionnel. Chaque position possède un poids.**
- ❑ **Par exemple, le nombre 4134 s'écrit comme :**
  - $4134 = 4 \times 10^3 + 1 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$

62

Codage de l'information

Système de numération positionnel pondéré

❑Un système de numérotation positionnel pondéré à base b est défini sur un alphabet de b chiffres :

- $A = \{c_0, c_1, \ldots, c_{b-1}\}$ avec $0 \leq c_i < b$

❑Soit $N = a_{n-1}\, a_{n-2} \ldots a_1\, a_{0\,(b)}$: représentation en base b avec les chiffres

- a_i : est un chiffre de l'alphabet de poids i (position i).
- a_0 : chiffre de poids 0 appelé le chiffre de poids faible
- a_{n-1} : chiffre de poids n-1 appelé le chiffre de poids fort

63

Codage de l'information

Syntaxe de base

$$(N)_b = (a_{n-1} a_{n-2} \ldots a_0)$$

$$\text{avec } a_i = \{0, 1, \ldots, b-1\}$$

•La valeur de N en base 10 est donnée par :

$$(N)_{10} = a_{n-1}.b^{n-1} + a_{n-2}.b^{n-2} + \ldots + a_0.b^0{}_{(10)}$$

64

Codage de l'information

Bases de numération Binaire

Système binaire (b=2) utilise deux chiffres : {0,1}

- C' est avec ce système que fonctionnent les ordinateurs
  - Avec 1 bit : 2 (2^1) possibilités
    - 0 ⇒ 0
    - 1 ⇒ 1

65

Codage de l'information

Bases de numération Binaire de 4

- **Système binaire (b=4) utilise quatre chiffres : {0,1,2,3}**
  - Avec 2 bits : 4 ($2*2=2^2$) possibilités
    - 00 ⇒ 0
    - 01 ⇒ 1
    - 10 ⇒ 2
    - 11 ⇒ 3

66

Codage de l'information

Bases de numération Octale

•Système Octale (b=8) utilise huit chiffres : {0,1,2,3,4,5,6,7}

- Utilisé il y a un certain temps en Informatique.
- Elle permet de coder 3 bits par un seul symbole.

- Avec 3 bits : 8 ($2*2*2= 2^3$) possibilités
  - 000 ⇒ 0
  - 001 ⇒ 1
  - 010 ⇒ 2
  - 011 ⇒ 3
  - 100 ⇒ 4
  - 101 ⇒ 5
  - 110 ⇒ 6
  - 111 ⇒ 7

67

Codage de l'information

Bases de numération Hexadécimale

Système Hexadécimale (b=16) utilise 16 chiffres :

{0,1,2,3,4,5,6,7,8,9, A=$10_{(10)}$, B=$11_{(10)}$, C=$12_{(10)}$, D=$13_{(10)}$, E=$14_{(10)}$, F=$15_{(10)}$ }

- Cette base est très utilisée dans le monde de la microinformatique.
- Elle permet de coder 4 bits par un seul symbole.

68

Codage de l'information

Bases de numération Hexadécimale

- Avec 4 bits : 8 (2*2*2= 2^3) possibilités
  - 0000 ⇒ 0
  - 0001 ⇒ 1
  - 0010 ⇒ 2
  - 0011 ⇒ 3
  - 0100 ⇒ 4
  - 0101 ⇒ 5
  - 0110 ⇒ 6
  - 0111 ⇒ 7
  - 1000 ⇒ 8
  - 1001 ⇒ 9
  - 1010 ⇒ A
  - 1011 ⇒ B
  - 1100 ⇒ C
  - 1101 ⇒ D
  - 1110 ⇒ E
  - 1111 ⇒ F

69

Codage de l'information

Le code BCD

- **BCD** est la contraction de **Binary Coded Decimal** se traduisant par décimal codé en binaire. L'homme étant habitué au système décimal, il a été nécessaire de créer un code permettant de conserver les avantages du système décimal sans sacrifier la simplicité de conversion directe en binaire.
- Le BCD n'utilise que les **10 premières combinaisons**. Pour chaque chiffre décimal, nous avons besoin de **4 bits.**

70

Codage de l'information

Le code BCD

- Les combinaisons supérieures à 9 sont interdites

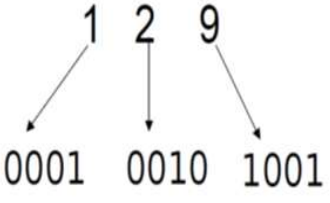
Décimal	Binaire
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

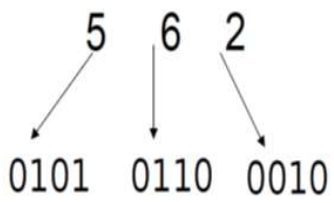
71

Codage de l'information

Le code BCD: exemple

- Chaque chiffre est codé sur quatre bits





$129_{10} = (0001\ 0010\ 1001)_{2d}$

$562_{10} = (0101\ 0110\ 0010)_{2d}$

72

Codage de l'information

Le code Gray ou Binaire réfléchi

- C'est un code à **distance minimale** car on passe d'une ligne à la suivante en ne changeant qu'un **seul bit**. On ne peut affecter aucun poids aux bits dans les groupes codés : ***ce code est non pondéré.***

Décimal	Gray			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1
10	1	1	1	1
11	1	1	1	0
12	1	0	1	0
13	1	0	1	1
14	1	0	0	1
15	1	0	0	0

73

Codage de l'information

Méthode de création du code Gray

- Les nombres 0 et 1 est codé par 0 et 1 en code Gray
- Les nombres suivant son codés en se basant sur la parité du nombre de 1 dans la nombre précédent
  - Si le nombre de 1 est **pair**, il faut inverser le dernier chiffre.
  - Si le nombre de 1 est **impair**, il faut inverser le chiffre situé à gauche du 1 le plus à droite.

74

Codage de l'information

Méthode de création du code Gray

- Le nombre de 1 est **pair**
  - 110 1100 => 110 1101
  - 111 1011 => 111 1010
  - 110 1001 => 110 1000
- Le nombre de 1 est **impair**
  - 110 1101 => 110 1111
  - 101 1000 => 100 1000
  - 110 0010 => 110 0110

75

Codage de l'information

Transcodage (ou conversion de base)

- C'est l'opération qui permet de passer de la représentation d'un nombre exprimé dans une base à la représentation du même nombre mais exprimé dans une autre base.
- Par la suite, on verra les conversions suivantes:
  - **Décimale vers Binaire, Octale et Hexadécimale**
  - **Binaire vers Décimale, Octale et Hexadécimale**

76

Codage de l'information

Techniques de conversion

- **Techniques pour convertir $(N)_b$ entre systèmes de numérotation bin-dec-hex:**

Type de conversion	Technique de conversion
binaire → décimal	Somme pondérée des contributions
hexadécimal → décimal	
décimal → binaire	Division par la base
décimal → hexadécimal	
binaire → hexadécimal	Substitution hex-bits
hexadécimal → binaire	

77

Codage de l'information

Changement de base de la base 10 vers une base b

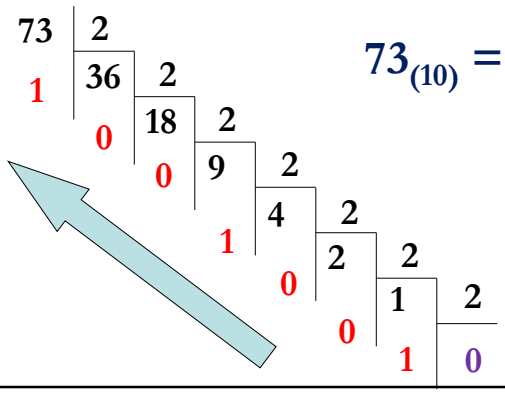
- **La règle à suivre est la division successive :**
  - On divise le nombre par la base b
  - Puis le quotient par la base b
  - Ainsi de suite jusqu'à l'obtention d'un quotient nul
  - La suite des restes correspond aux symboles de la base visée.
  - On obtient en premier le chiffre de poids faible et en dernier le chiffre de poids fort.

78

Codage de l'information

Exemple : décimale vers binaire

- Soit N le nombre d'étudiants d'une classe représentée en base décimale par : $N = 73_{(10)}$
- Représentation en Binaire?



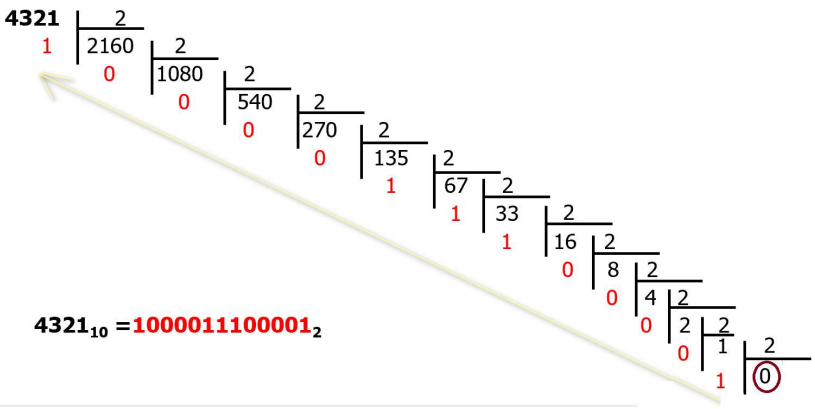
$73_{(10)} = 1001001_{(2)}$

79

Codage de l'information

Décimale vers Binaire

On prend les restes de la division successive de n par 2,
Exemple:

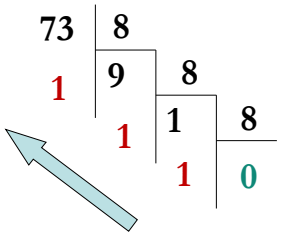


$4321_{10} = 1000011100001_2$

Codage de l'information

Exemple : décimale vers octale

- Soit N le nombre d'étudiants d'une classe représenté en base décimale par : $N = 73_{(10)}$
- Représentation en Octale?



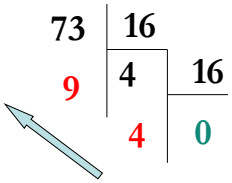
- **73(10) = 111(8)**

81

Codage de l'information

Exemple : décimale vers Hexadécimale

- Soit N le nombre d'étudiants d'une classe représenté en base décimale par : $N = 73_{(10)}$
- Représentation en Hexadécimale?



- **$73_{(10)} = 49_{(16)}$**

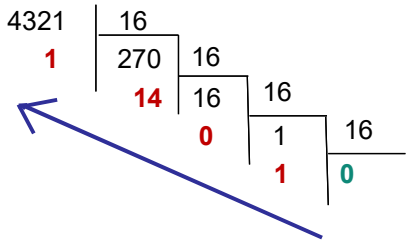
82

Codage de l'information

Décimale vers Hexadécimale

Prendre les restes de la division successive de n par 16,

Exemple:



$4321_{10} = 10E1_{16}$

83

Codage de l'information

Conversion Gray vers Binaire.

- Le bit de gauche du nombre binaire est le même que le bit de gauche du code Gray.
- Ajouter le MSB du nombre binaire obtenu au voisin de droite immédiat du code Gray.
- Continuer les additions jusqu'à atteindre le LSB.

84

Codage de l'information

Conversion Binaire vers Gray.

- Le bit de gauche du code Gray est le même que le bit de gauche du nombre binaire.
- Ajouter le MSB du nombre binaire à son voisin immédiat et reporter la somme en négligeant une retenue éventuelle sur la ligne inférieure correspondante au code Gray.
- Continuer l'addition des bits à leur voisin de droite et reporter les sommes ainsi obtenues jusqu'à atteindre le LSB.

85

Codage de l'information

Conversion Binaire vers Gray.

- Le nombre en code Gray comportera toujours le même nombre de bits que le binaire original

➔ **Addition décalée sans retenu (XOR)**

Exemple : soit à convertir le binaire 10110.

1	0	1	1	0	Nombre binaire
	+	+	+	+	
↓	↓	↓	↓	↓	
1	1	1	0	1	Code Gray

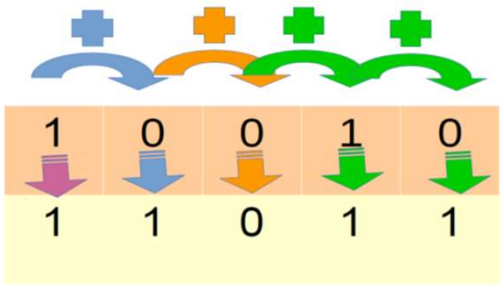
86

Codage de l'information

Conversion Binaire vers Gray.

- De gauche à droite faire la somme des bits adjacents sans retenue

```
10010
 10010
11011
```

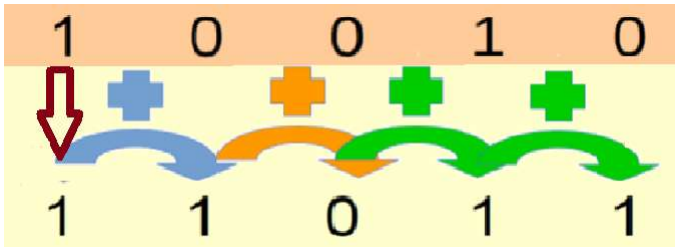


87

Codage de l'information

Conversion Gray vers Binaire

- De gauche à droite, reporté le premier bit, puis faire la somme de bit de résultat et le bit suivant sans retenue



88

Codage de l'information

Base binaire vers base b

- **Première solution :**
  - convertir le nombre en base binaire vers la base décimale puis convertir ce nombre en base 10 vers la base b.
- **Exemple :**
  - $10010_{(2)} = ?_{(8)}$
  - $10010(2) = 2^4+2_{(10)}=18_{(10)}=2*8^1+2*8^0{}_{(10)}=22(8)$

89

Codage de l'information

Binaire vers Décimale

Utiliser la formule de développement

Exemple :

$10011010_2 = 1\times2^7 + 0\times2^6 + 0\times2^5 + 1\times2^4 + 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$

$= 128 + 16 + 8 + 2$

$= 154_{10}$

90

Codage de l'information

Base binaire vers base b

- **Binaire vers octale : regroupement des bit en des sous ensemble de trois bits puis remplacé chaque groupe par le symbole correspondant dans la base 8.**
- **Binaire vers Hexadécimale : regroupement des bits en des sous ensembles de quatre bits puis remplacer chaque groupe par le symbole correspondant dans la base 16.**

91

Codage de l'information

Correspondance Octale \Binaire

Symbole Octale	suite binaire
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

92

Codage de l'information

- **Correspondance Octale \Binaire**

S. Hexad.	suite binaire	S. Hexad.	suite binaire
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

93

Codage de l'information

- **Les nombres en Hexadécimale**

0 1 2 3 4 5 6 7 8 9 A B C D E F
10 11 12 13 14 15 16 17 18 19 1A
1B 1C 1D 1E 1F 20 21 22 23 24 25
26 27 28 29 2A 2B 2C 2D 2E 2F 30
31 32 33 34 35 36 37 38 39 3A
3B 3C 3D 3E 3F 40 41 42 43 44
45 46 47 48 49 4A 4B 4C 4D 4E
4F … …

94

Codage de l'information

Exemple : binaire vers décimale

- Soit N un nombre représenté en binaire par :
- $N = 1010011101_{(2)}$
- Représentation Décimale?

$N=1.2^9+0.2^8+1.2^7+0.2^6+0.2^5+1.2^4+1.2^3+1.2^2+0.2^1+1.2^0$

$=512 + 0 + 128 + 0 + 0 + 16 + 8 + 4 + 0 + 1$

$=669_{(10)}$

$1010011101_{(2)}=669_{(10)}$

95

Codage de l'information

Binaire vers Octale

- On regroupe les bits par blocs de trois en allant vers la gauche (on complète par des zéros à gauche si nécessaire),

Exemple :

$n = 10110101100111_2 = 010\ 110\ 101\ 100\ 111$

$= 2\ 6\ 5\ 4\ 7$

$= 26547_8$

96

Codage de l'information

Exemple : binaire vers octale

- Soit N un nombre représenté en base binaire par :
  $N = 1010011101_{(2)}$
- Représentation Octale?

N = 001 010 011 $101_{(2)}$

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

$1010011101_{(2)} = 1235_{(8)}$

97

Codage de l'information

Binaire vers Hexadécimale

- On regroupe les bits par blocs de quatre en allant vers la gauche (on complète par des zéros à gauche si nécessaire),

Exemple :

$n = 10110101100111_2$ = 0010 1101 0110 0111

= 2 D 6 7

= $2D67_{16}$

98

Codage de l'information

Binaire vers Hexadécimale

❑ Soit N un nombre représenté en base binaire par :

- **N = $1010011101_{(2)}$**

❑ Représentation Hexadécimale?

N = 0010 1001 $1101_{(2)}$

= 2 9 $D_{(16)}$

$1010011101_{(2)}$= $29D_{(16)}$

99

Codage de l'information

Hexadécimale vers Binaire

❑ Chaque chiffre sera remplacé par un bloc de quatre bits (l'inverse de la méthode précédente),

❑ Exemple :

A 1 7 B

$n = A17B_{16}$ = 1010 0001 0111 1011

= 1010000101111011_2

100

Codage de l'information

Exercice

Décimal	Binaire	Hexadécimal	Octal
1	00000001	001	001
10			
	01100100		
		065	
			764

101

Codage de l'information

Exercice: correction

Décimale	Binaire	Héxa.	Octale
10	00001010	0A	012
100	01100100	064	144
101	01100101	065	145
500	111110100	1F4	764

102

Codage de l'information

Codage des nombres entiers

Codage des entiers naturels

- **Utilisation du code binaire pur :**
  - **L'entier naturel (positif ou nul) est représenté en base 2,**
  - **Les bits sont rangés selon leur poids, on complète à gauche par des 0.**
  - **Exemple: sur un octet, $10_{(10)}$ se code en binaire pur**

$$0\ 0\ 0\ 0\ 1\ 0\ 1\ 0_{(2)}$$

103

Codage de l'information

Codage des entiers naturels

- Etendu du codage binaire pur :
  - Codage sur n bits : représentation des nombres de 0 à $2^n - 1$
  - sur 1 octet (8 bits): codage des nombres de 0 à $2^8 - 1 = 255$
  - sur 2 octets (16 bits): codage des nombres de 0 à $2^{16} - 1 = 65535$
  - sur 4 octets (32 bits) : codage des nombres de 0 à $2^{32} - 1 = 4\ 294\ 967\ 295$

104

Codage de l'information

Nombre de valeurs possibles

- **Avec des mots de n bits, il est possible de représenter 2^n valeurs différentes.**
- pour **n=1**, nous pouvons représenter deux valeurs **0 et 1.**
- Avec **deux bits**, nous pouvons représenter 4 valeurs codées, **00, 01, 10 et 11**.
- Avec **trois bits**, nous pouvons représenter 8 valeurs codées, **000, 001, 010, 011, 100, 101, 110 et 111**.
- Avec **n bits**, nous pouvons représenter **2^n valeurs différentes.**

105

Codage de l'information

Nombre de valeurs possibles

- **Inversement, pour coder n valeurs différentes, il faudra lg(n) bits,**
  - **où lg(n) est le plus petit entier k tel que .**
  - **$2^{k-1} < n \leq 2^k$**
  - **Ainsi, pour coder 11 valeurs différentes, il faudra 4 bits car $2^3 < 11 \leq 2^4$.**
  - **On peut noter que dans ce cas 5 suites de bits seront inutilisées. (Car avec 4 bits on peut coder $2^4 = 16$ valeurs différentes)**

106

Codage de l'information

Arithmétique en base 2

- **Les opérations sur les entiers s'appuient sur des tables d'addition et de multiplication :**

Addition

0	0	0
0	1	1
1	0	1
1	1	(1)0

Retenu

Multiplication

0	0	0
0	1	0
1	0	0
1	1	1

107

Codage de l'information

Addition

la règle en binaire est :

0 + 0 = 0,
0 + 1 = 1,
1 + 0 = 1,
1 + 1 = 0 avec **1** comme **retenue**
1 + 1 + 1 = 1 avec **1** comme **retenue**.

Exemple :

$$\begin{array}{rr} & 1\ \ 1 \\ & 1\,1\,0\,1\,0\,1\,1\,0\,0_2 \\ + & 1\,1\,1\,0\,0\,1\,0\,1\,1_2 \\ \hline = & \mathbf{1\,1\,0\,1\,1\,1\,0\,1\,1\,1_2} \end{array} \qquad \begin{array}{r} 1\ \ 1\ \ 1 \\ 4\ A\ 5\ C_{16} \\ +\ E\ B\ 9\ 5_{16} \\ \hline \mathbf{= 1\ 3\ 6\ F\ 1_{16}} \end{array}$$

108

Codage de l'information

Exemple (Addition)

- **Addition binaire** (8 bits) sans débordement (overflow)

```
  1 0 0 1 0 1 1 0
+ 0 1 0 1 0 1 0 1
  1 1 1 0 1 0 1 1
```

- **Addition binaire** (8 bits) avec débordement

```
  1 0 0 1 0 1 1 0
+ 1 1 1 1 0 1 0 1
1 1 0 0 0 1 0 1 1
```

overflow

109

Codage de l'information

Exemple (Multiplication)

- Multiplication binaire

```
        1 0 1 1  (4 bits)
      * 1 0 1 0  (4 bits)
        0 0 0 0
      1 0 1 1 .
    0 0 0 0 .
  1 0 1 1 .
0 1 1 0 1 1 1 0
```

Sur 4 bits le résultat est faux

Sur 7 bits le résultat est juste

Sur 8 bits on complète à gauche par un 0

110

Codage de l'information

Codage des entiers relatifs

- Il existe au moins trois façons pour coder :
  - code binaire signé (*par signe et valeur absolue*)
  - code complément à 1
  - code complément à 2 (Utilisé sur ordinateur)

111

Codage de l'information

Codage des entiers relatifs en binaire signé

- **Le bit le plus significatif est utilisé pour représenter le signe du nombre :**
  - **Si** le bit le plus fort **= 1 alors nombre négatif**
  - **Si** le bit le plus fort **= 0 alors nombre positif**
- **Les autres bits codent la valeur absolue du nombre**
- **Exemple : Sur 8 bits, codage des nombres : -24 et -128 en (bs)**
  - **-24 est codé en binaire signé par : 1 0 0 1 1 0 0 $0_{(bs)}$**
  - **-128 hors limite → nécessite 9 bits au minimum**

112

Codage de l'information

Codage des entiers relatifs en binaire signé

- **Etendu de codage :**
  - Avec n bits, on code tous les nombres entre
    - $-(2^{n-1}-1)$ et $(2^{n-1}-1)$
  - Avec 4 bits : -7 et +7
- **Limitations du binaire signé:**
  - Deux représentations du zéro : + 0 et – 0 Sur 4 bits :
    - $+0 = 0000_{(bs)}$, $-0 = 1000_{(bs)}$
  - Multiplication et l'addition sont moins évidentes.

113

Codage de l'information

Codage des entiers relatifs en binaire signé

- **Malheureusement, avec cette représentation, une soustraction de deux nombres a-b ne pourra pas se faire par l'addition de a et -b.**
- **En effet, avec ce codage, -5 serait codé sur 8 bits par 10000101 et si on effectue l'opération 9+(-5), on obtient comme résultat**

```
  00001001
+ 10000101
= 10001110   soit -14 et non 4
```

114

Codage de l'information

Exercices Binaire signé

- Coder 100 et -100 en binaire signé sur 8 bits

$100_{(10)} = (01100100)_{(bs)}$

$-100_{(10)} = (11100100)_{(bs)}$

- Décoder en décimal $(11000111)_{(bs)}$ et $(00001111)_{(bs)}$

$(11000111)_{(bs)} = -71_{(10)}$

$(00001111)_{(bs)} = 15_{(10)}$

- Calculer : 1– 2 en binaire signé sur 8 bits

115

Codage de l'information

Exercices Binaire signé

- Calculer : 1– 2 en binaire signé sur 8 bits

1 = 0000 0001

-2 = 1000 0010

1-2 = 1+(-2) :

```
  0000 0001
+ 1000 0010
= 1000 0011
```

On obtient -3 au lieu de -1

116

Codage de l'information

Codage des entiers relatifs en complément à 1

- Aussi appelé Complément Logique (CL) ou Complément Restreint (CR) :
  - les nombres positifs sont codés de la même façon qu'en binaire pure.
  - un nombre négatif est codé en inversant chaque bit de la représentation de sa valeur absolue
- Le bit le plus significatif est utilisé pour représenter le signe du nombre :
  - si le bit le plus fort = 1 alors nombre négatif
  - si le bit le plus fort = 0 alors nombre positif

117

Codage de l'information

Codage des entiers relatifs en complément à 1

- Exemple : -24 en complément à 1 sur 8 bits
  - |-24|en binaire pur → 0 0 0 1 1 0 0 0 $_{(2)}$ puis
  - on inverse les bits → 1 1 1 0 0 1 1 1 $_{(cà1)}$
- Limitation :
  - deux codages différents pour 0 (+0 et -0)
  - Sur 8 bits : +0=0 0 0 0 0 0 0 0$_{(cà1)}$ et -0=1 1 1 1 1 1 1 1$_{(cà1)}$
  - Multiplication et l'addition sont moins évidentes.

118

Codage de l'information

Exercices en code complément à 1

- Coder 100 et -100 par complément à 1 (cà1) sur 8 bits
  - $100_{(10)} = (01100100)_{(cà1)}$
  - $-100_{(10)} = (10011011)_{(cà1)}$
- Décoder en décimal $(11000111)_{(cà1)}$ et $(00001111)_{(cà1)}$
  - $(11000111)_{(cà1)} = -56_{(10)}$
  - $(00001111)_{(cà1)} = 15_{(10)}$
- Calculer : – 1 – 2 en complément à 1 sur 8 bits

119

Codage de l'information

Codage des entiers relatifs en complément à 1

- Calculer : -1– 2 en complément à 1 sur 8 bits

-1 = 1111 1110

-2 = 1111 1101 .

-1-2 = -1+(-2) :

```
   1111 1110
+  1111 1101
=  1111 1011
```

On obtient -4 au lieu de -3

120

Codage de l'information

Codage des entiers relatifs en complément à 2

- Aussi appelé Complément Vrai (CV) :
  - Les nombres positifs sont codés de la même manière qu'en binaire pure.
  - Un nombre négatif est codé en ajoutant la valeur 1 à son complément à 1
- Le bit le plus significatif est utilisé pour représenter le signe du nombre
- Exemple : -24 en complément à 2 sur 8 bits
  - 24 est codé par $0\,0\,0\,1\,1\,0\,0\,0_{(2)}$
  - -24 → $1\,1\,1\,0\,0\,1\,1\,1_{(cà1)}$
  - donc -24 est codé par $1\,1\,1\,0\,1\,0\,0\,0_{(cà2)}$

121

Codage de l'information

Codage des entiers relatifs en complément à 2

- Un seul codage pour 0. Par exemple sur 8 bits :
  - +0 est codé par $00000000_{(cà2)}$
  - -0 est codé par $11111111_{(cà1)}$
  - Donc -0 sera représenté par $00000000_{(cà2)}$
- Etendu de codage :
  - Avec n bits, on peut coder de $-(2^{n-1})$ à $(2^{n-1}-1)$
  - Sur 1 octet (8 bits), codage des nombres de -128 à 127
    - +0 = 00000000 -0=00000000
    - +1 = 00000001 -1=111111111
    - … …
    - +127= 01111111 -128=10000000

122

Codage de l'information

Exercices en Code Complément à 2

- Coder $100_{(10)}$ et $-100_{(10)}$ par complément à 2 sur 8 bits
  - $100_{(10)} = 01100100_{(Cà2)}$
  - $-100_{(10)} = 10011010_{(Cà2)}$
- Décoder en décimal $11001001_{(Cà2)}$ et $01101101_{(Cà2)}$

  $11001001_{(Cà2)} = -55_{(10)}$

  $01101101_{(Cà2)} = 109_{(10)}$
- Calculer : 1-2 en complément à 2 sur 8 bits

123

Codage de l'information

Exercices en Code Complément à 2

- **Calculer : 1– 2 en complément à 2 sur 8 bits**
  - **1 = 0000 0001**
  - **-2 = 1111 1110**
  - 1-2 = 1+(-2) :
  - 0000 0001
  - \+ 1111 1110
  - = 1111 1111
  - On obtient -1

124

Codage de l'information

Codage des nombres réels

- Format virgule fixe (utilisé par les premières machines):
  possède une partie 'entière' et une partie 'décimale' séparés par une virgule dont la position est fixe
  - Exemple : $54,25_{(10)}$; $10,001_{(2)}$; $A1,F0B_{(16)}$
- Format virgule flottante (utilisé actuellement sur machine)
  Défini par : $\pm\ m \cdot b^{e}$
  - un signe + ou –
  - une mantisse m (en virgule fixe)
  - un exposant e (un entier relatif)
  - une base b (2,8,10,16,…)
  - Exemple : $0,5425 \cdot 10^{2}{}_{(10)}$; $10,1 \cdot 2^{-1}{}_{(2)}$; $A0,B4.16^{-2}{}_{(16)}$

125

Codage de l'information

Codage en Virgule Fixe

- Etant donné une base b, un nombre x est représenté par :
  - $x = a_{n-1}a_{n-2}\ldots a_1a_0,a_{-1}a_{-2}\ldots a_{-p}$ (b)
  - a_{n-1} est le chiffre de poids fort
  - a_{-p} est le chiffre de poids faible
  - n est le nombre de chiffre avant la virgule
  - p est le nombre de chiffre après la virgule
  - La valeur de x en base 10 est : $x = \sum_{-p}^{n-1} a_i b^i{}_{10)}$
  - Exemple :

$101,01_{(2)} = 1.2^2+0.2^1+1.2^0+0.2^{-1}+1.2^{-2} = 5,25_{(10)}$

126

Codage de l'information

Codage en Virgule Fixe base 10 vers 2

- Le passage de la base 10 à la base 2 est défini par :
  - Partie entière est codée sur p bits (division successive par 2)
  - Partie décimale est codée sur q bits en multipliant par 2 successivement jusqu'à ce que la partie décimale soit nulle ou le nombre de bits q est atteint.

- Exemple : $4,25_{(10)} = ?_{(2)}$ format virgule fixe
  - ✓ $4_{(10)} = 100_{(2)}$
  - ✓ 0,25 x 2= 0,5 → 0
  - ✓ 0,5 x 2 = 1,0 → 1
  - ✓ donc $4,25_{(10)} = 100,01_{(2)}$
- Exercice : Coder $7,875_{(10)}$ et $5,3_{(10)}$ avec p = 8 et q = 8

Codage en Virgule Flottante

$$x = \pm M \cdot 2^{E}$$

- où M est la mantisse (virgule fixe) et E l'exposant (signé).
- Le codage en base 2, format virgule flottante, revient à coder le signe, la mantisse et l'exposant.
- Exemple: Codage en base 2, format virgule flottante, de (3,25)

  $3,25_{(10)} = 11,01_{(2)}$ (en virgule fixe)

  $= 1,101 \cdot 2^{1}_{(2)}$

  $= 110,1 \cdot 2^{-1}_{(2)}$
- Pb : différentes manières de représenter E et M → Normalisation

Codage de l'information

Normalisation du Codage en Virgule Flottante

$$x = \pm 1,M \cdot 2^{Eb}$$

- Le signe est codé sur 1 bit avant le poids fort :
  - le signe – : bit 1
  - Le signe + : bit 0
- Exposant biaisé (Eb)
  - placé avant la mantisse pour simplifier la comparaison
  - Codé sur p bits et biaisé pour être positif (ajout de $2^{p-1}-1$)
- Mantisse normalisé(M)
  - Normalisé : virgule est placé après le bit à 1 ayant le poids fort
  - M est codé sur q bits
  - Exemple : 11,01 → 1,101 donc M =101

SM	Eb	M
1 bit	p bits	q bits

129

Codage de l'information

Standard IEEE 754 (1985)

- Simple précision sur 32 bits :
  - 1 bit de signe de la mantisse
  - 8 bits pour l'exposant
  - 23 bits pour la mantisse

SM	Eb	M
1 bit	8 bits	23 bits

- Double précision sur 64 bits :
  - 1 bit de signe de la mantisse
  - 11 bits pour l'exposant
  - 52 bits pour la mantisse

SM	Eb	M
1 bit	11 bits	52 bits

130

Codage de l'information

Conversion décimale - IEEE754

❑ **Codage d'un réel**

- $35,5_{(10)} = ?_{\text{(IEEE 754 simple précision)}}$

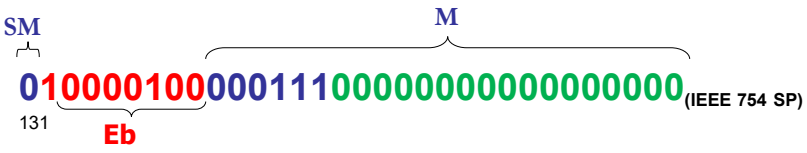
Nombre positif, donc SM = 0

$35,5_{(10)} = 100011,1_{(2)}$ (virgule fixe)

$= 1,000111 \cdot 2^{5}{}_{(2)}$ (virgule flottante)

Exposant = Eb-127 = 5, donc Eb = 132

1,M = 1,000111 donc M = 00011100...

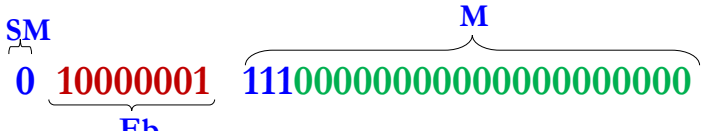


Codage de l'information

Conversion IEEE754 - Décimale

❑ **Évaluation d'un réel**

- $01000000111100000000000000000000_{\text{(IEEE 754 SP)}}$



S = 0, donc nombre positif

Eb = 129, donc exposant = Eb-127 = 2

1,M = 1,111

$+ 1,111 \cdot 2^{2}{}_{(2)} = 111,1_{(2)} = 7,5_{(10)}$

132

Codage de l'information

Caractéristiques des nombres flottants standard au IEEE

	Simple précision	Double précision
Bit de signe	1	1
Bit d'exposant	8	11
Bit de mantisse	23	52
Nombre total de bits	32	64
Codage de l'exposant	Excédant l27	Excédant l023
Variation de l'exposant	-126 à +127	-1022 à +1023
Plus petit nombre normalisé	2^{-126}	2^{-1022}
Plus grand nombre normalisé	Environ 2^{+128}	Environ 2^{+1024}
Echelle des nombre décimaux	Environ 10^{-38} à 10^{+38}	Environ 10^{-308} à 10^{+308}
Plus petit nombre dénormalisé	Environ 10^{-45}	Environ 10^{-324}

Codage de l'information

Codage des caractères

- Caractères : Alphabétique (A-Z , a-z), numérique (0 ,…, 9), ponctuation, spéciaux (&, $, %,…) …
- Données non numérique (addition n'a pas de sens) → concaténation
- Comparaison ou tri → très utile
- Le codage revient à créer une Table de correspondance entre les caractères et des nombres.

134

Codage de l'information

Les Standards

- Code (ou Table) ASCII (American Standard Code for Information Interchange)
  - 7 bits pour représenter 128 caractères (0 à 127)
  - 48 à 57 : chiffres dans l'ordre (0,1,…,9)
  - 65 à 90 : les alphabets majuscules (A,…,Z)
  - 97 à 122 : les alphabets minuscule (a,…z)

135

Codage de l'information

Les Standards

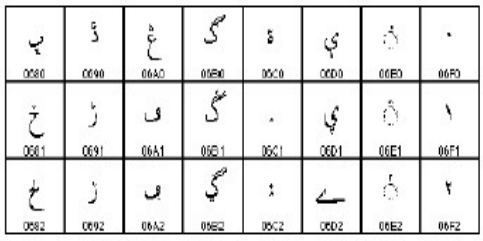
- Table ASCII Etendu
  - 8 bits pour représenter 256 caractères (0 à 255)
  - Code les caractères accentués : à, è,…etc.
  - Compatible avec ASCII
- Code Unicode (mis au point en 1991)
  - 16 bits pour représenter 65 536 caractères (0 à 65 535)
  - Compatible avec ASCII
  - Code la plupart des alphabets : Arabe, Chinois, ….
  - On en a défini environ 50 000 caractères pour l'instant

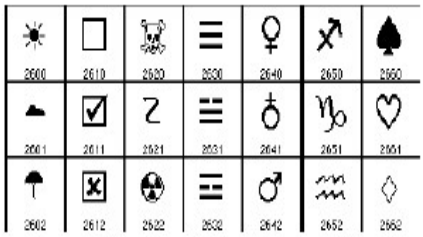
136

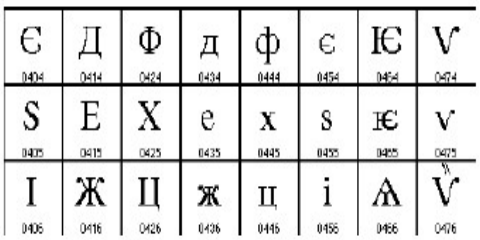
Code ASCII Etendu

DECIMAL VALUE	HEXA DECIMAL VALUE	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	BLANK (NULL)	►	SP	0	@	P	‘	p	Ç	É	á	⁞	└	┴	∝	≡
1	1	☺	◄	!	1	A	Q	a	q	ü	æ	í	▒	┴	┬	β	±
2	2	☻	↕	"	2	B	R	b	r	é	Æ	ó	▓	├	├	Γ	≥
3	3	♥	‼	#	3	C	S	c	s	â	ô	ú	│	─	─	π	≤
4	4	♦	¶	$	4	D	T	d	t	ä	ö	ñ	┤	┬	┼	Σ	⌠
5	5	♣	§	%	5	E	U	e	u	à	ò	Ñ	╡	┴	┼	σ	⌡
6	6	♠	▬	&	6	F	V	f	v	å	û	ª	╢	├	├	μ	÷
7	7	BEL	↨	’	7	G	W	g	w	ç	ù	º	╖	┬	┬	τ	≈
8	8	BS	↑	(	8	H	X	h	x	ê	ÿ	¿	╕	┤	┼	Φ	°
9	9	HT	↓	)	9	I	Y	i	y	ë	Ö	⌐	╣	┬	┬	Θ	•
10	A	LF	→	*	:	J	Z	j	z	è	Ü	¬	╗	├	├	Ω	·
11	B	VT	←	+	;	K	[	k	{	ï	¢	½	╝	┤	┼	δ	√
12	C	FF	FS	,	<	L	\	l	¦	î	£	¼	╜	┬	┬	∞	n
13	D	CR	GS	—	=	M	]	m	}	ì	¥	¡	╛	┤	┼	φ	²
14	E	♫	RS	.	>	N	^	n	~	Ä	₧	«	┐	├	├	∈	■
15	F	☼	US	/	?	O	_	o	⌂	Å	ƒ	»	└	┴	┴	∩	NBSP

Unicode







Codage de l'information

Codage des caractères

Suite de caractères

- Caractères codés en ASCII Etendu (8 bits)

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

➔ INFORMATIQUE

- Entiers codés en binaire pur sur 1 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

73 ; 78 ; 70 ; 79 ; 82 ; 77 ; 65 ; 84 ; 73 ; 81 ; 85 ; 69 (base 10)

139

Codage de l'information

Suite de caractères

- entiers codés en binaire pur sur 2 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

18766 ; 17999 ; 21069 ; 16724 ; 18769 ; 21829 (base 10)

- entiers codés en binaire pur sur 4 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

1 229 866 575 ; 1 380 794 708 ; 1 230 067 013 (base 10)

140

Codage de l'information

Suite de caractères

- **nombres en flottant simple précision (32 bits)**

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

+ (1,10011100100011001001111) . 2^{19} ;
+ (1,10011010100000101010100) . 2^{37} ;
+ (1,10100010101010101000101) . 2^{19} ;
844 900,9375;
220 391 079 936 ;
857 428,3125 (base 10)

141