Plan

- Introduction
- Systèmes de numération et Représentation des nombres
  - Systèmes de numération
  - Système de numération décimale
  - Représentation dans une base b
  - Représentation binaire, Octale et Hexadécimale
  - Transcodage ou changement de base
- Codage des nombres
  - Codage des entiers positifs (binaire pur)
  - Codage des entiers relatifs (complément à 2)
  - Codage des nombres réels (virgule flottante)
- Codage des caractères :
  - ASCII et
  - ASCII étendu,
  - Unicode , …
- Codage du son et des images

Codage 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

Codage d'information : -Définition-

Codage de l'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 règle précise.

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)

Codage d'information (suite)

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

(Elément binaire 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) : $s(t) = a \sin (2 \; f_1 t + \;)$
  - Fréquence f_2 (bit 0) : $s(t) = a \sin (2 \; f_2 t + \;)$
- ….

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

Exemples de Système de numération (1)

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)

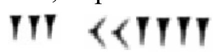
Système inadapté au calcul

Exemples de Système de numération (2)

Numération babylonienne

Chez les Babyloniens (environ 2000 ans avant J.C.), les symboles utilisés sont le clou pour l'unité et le chevron pour les dizaines. C'est un système de position.

A partir de 60, la position des symboles entre en jeu :

204 : 

7392 : 

Le nombre 60 constitue la base de ce système.

Exemples de Système de numération (3)

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 :

$$4134 = 4 \times 10^3 + 1 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$$

Système de numération positionnel pondéré à base b

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

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

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

La valeur de N en base 10 est donnée par :

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

Bases de numération (Binaire, Octale et Hexadécimale)

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

C'est avec ce système que fonctionnent les ordinateurs

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.

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 micro informatique.

Elle permet de coder 4 bits par un seul symbole.

Transcodage (ou conversion de base)

Le transcodage (ou conversion de base) 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:

- <u>Décimale vers Binaire, Octale et Hexadécimale</u>
- <u>Binaire vers Décimale, Octale et Hexadécimale</u>

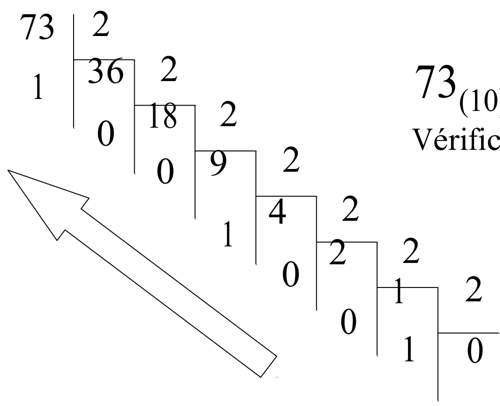
Changement de base de la base 10 vers une base b

La règle à suivre est les divisions successives :

- 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.

Exemple : décimale vers binaire

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

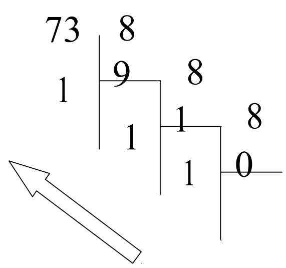


$73_{(10)} = 1001001_{(2)}$

Vérification

Exemple : décimale vers octale

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



$73_{(10)} = 111_{(8)}$

Vérification

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 | 16
9 | 4 | 16
4 | 0

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

Vérification

De la base binaire vers une base b -Solution 1-

Première solution :

Convertir le nombre en base binaire vers la base décimale puis convertir ce nombre en base 10 vers la base b.

Exemple :

$10010_{(2)} = ?_{(8)}$

$10010_{(2)} = 2^4+2_{(10)}=18_{(10)}=2*8^1+2*8^0{}_{(10)}=22_{(8)}$

De la base binaire vers une base b

-Solution 2-

Deuxième solution :

Binaire vers décimale : par définition ($\sum_{i=0}^{n-1} a_i b^i$)

Binaire vers octale : regroupement des bits en des sous ensembles de trois bits puis remplacer chaque groupe par le symbole correspondant dans la base 8.(Table)

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.(Table)

Correspondance Octale \Binaire

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

Correspondance Hexadécimale \Binaire

Hexadécimale\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

Exemple : binaire vers décimale

Soit N un nombre représenté en binaire par :

$N = 1010011101_{(2)}$

Représentation Décimale?

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

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

$=669_{(10)}$

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

Exemple : binaire vers octale

Soit N un nombre représenté en base binaire par :

$N = 1010011101_{(2)}$

Représentation Octale?

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

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

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

Exemple : binaire vers Hexadécimale

Soit N un nombre représenté en base binaire par :

$N = 1010011101_{(2)}$

Représentation Hexadécimale?

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

$= \quad 2 \quad 9 \quad D_{(16)}$

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

Exercice

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

Plan

- Introduction
- Systèmes de numérotation et Codage des nombres
  - Systèmes de numérotation
  - Système de numération décimale
  - Représentation dans une base b
  - Représentation binaire, Octale et Hexadécimale
  - Transcodage ou changement de base
- Codage des nombres
  - Codage des entiers positifs (binaire pur)
  - Codage des entiers relatifs (complément à 2)
  - Codage des nombres réels (virgule flottante)
- Codage des caractères :
  - ASCII et
  - ASCII étendu,
  - Unicode , …
- Codage du son et des images

Codage des entiers naturels (1)

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?

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

Codage des entiers positifs (2)

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 $2^{32} - 1 = 4\ 294\ 967\ 295$

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

Exemple (Addition)

Addition binaire (8 bits)

```
   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 ou overflow) :

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

overflow

Exemples

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

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)

Codage des nombres relatifs -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

Codage des nombres relatifs -Binaire signé- (suite)

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.

Binaire signé (Exercices)

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

Codage des entiers relatifs (code 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

Codage des entiers relatifs -code complément à 1- (suite)

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 $_{(\text{cà1})}$

Limitation :

deux codages différents pour 0 (+0 et -0)

Sur 8 bits : +0=0 0 0 0 0 0 0 0$_{(\text{cà1})}$ et -0=1 1 1 1 1 1 1 1$_{(\text{cà1})}$

Multiplication et l'addition sont moins évidentes.

Code Complément à 1 (Exercices)

Coder 100 et -100 par complément à 1 (cà1) sur 8 bits

$100_{(10)} = (01100100)_{(\text{cà1})}$

$-100_{(10)} = (10011011)_{(\text{cà1})}$

Décoder en décimal $(11000111)_{(\text{cà1})}$ et $(00001111)_{(\text{cà1})}$

$(11000111)_{(\text{cà1})} = -56_{(10)}$

$(00001111)_{(\text{cà1})} = 15_{(10)}$

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

Codage des entiers relatifs -code complément à 2- (1)

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 $1\ 1\ 1\ 0\ 0\ 1\ 1\ 1_{(\text{cà1})}$

donc -24 est codé par $1\ 1\ 1\ 0\ 1\ 0\ 0\ 0_{(\text{cà2})}$

Codage des entiers relatifs -code complément à 2- (2)

Un seul codage pour 0. Par exemple sur 8 bits :

+0 est codé par $00000000_{(\text{cà2})}$
-0 est codé par $11111111_{(\text{cà1})}$
Donc -0 sera représenté par $00000000_{(\text{cà2})}$

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

Code Complément à 2
-Exercices-

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

Exercices

Quel est l'entendu de codage sur 6 et 9 bits :
Binaire pur, Binaire signé, complément à 2

Quelle est la valeur décimale des suites binaires (1010, 10010110 et 1011010011101001), s'elles sont codées en :
binaire pur, Binaire signé, Complément à 1, Complément à 2

Sur 4, 8 et 16 bits, coder les nombres +20 et -15 en :
Binaire pur, Binaire signé, Complément à 1, Complément à 2

Calculer 20-15 sur 8 et 16 bits en :
Complément à 2