Université Abdelmalek Essaadi, FST – Tanger
Département Génie Informatique

Année universitaire: 2020/2021

Cycle d'ingénieur: LSI (S1)

Module

Architecture des Ordinateurs & Systèmes d'Exploitation

Pr. Mohamed EL BRAK
Pr. Abderrahim GHADI

Université Abdelmalek Essaadi, FST – Tanger
Département Génie Informatique

Année universitaire: 2020/2021

Cycle d'ingénieur: LSI (S1)

Architecture des ordinateurs

Pr. Mohamed EL BRAK

Méthode pédagogique

- Le cours est proposé sous forme d'enseignement à distance:

- Live **(google meet)** à raison de séances de cours hebdomadaires, sur un semestre.

- Fournitures de supports ppt, pdf, séries TD/TP, (sur Google **classroom**)

Objectifs

- Clarifier l'architecture des ordinateurs

- Comprendre le principe de fonctionnement d'un ordinateur :

→ comment fonctionne cette machine sur laquelle vous passez des heurs et des heurs

Plan du cours

- **Chapitre I :** Représentation de l'information
- **Chapitre II:** Architecture de base d'un ordinateur
- **Chapitre III:** Le processeur 80x86
- **Chapitre VI:** Langage Assembleur

Introduction

- Les technologies numériques sont maintenant omniprésentes
  - Elles sont le moteur et l'objet de ce qu'on appelle la « révolution numérique »
- Elles sont basées sur l'interaction entre :
  - Des programmes, aussi appelés logiciels, décrivant des processus de traitement de l'information : biens immatériels
  - Des ordinateurs, capables d'exécuter ces programmes : biens matériels

Définition (1)

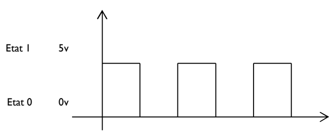
- **Ordinateur**
  - Machine automatique de traitement de l'information
  - Obéit à un programme formé par des suites d'opérations logiques et arithmétiques
- **Architecture d'un ordinateur**
  - Représente l'organisation des différentes unités de l'ordinateur et leurs interconnexions.

➢ **Information :** élément de connaissance susceptible d'être codé pour être conservé, traité ou communiqué.

- Unité de la quantité de l'information: **Bit**
  - ➢ Bit: Binairy digit (*chiffre binaire*)
  - ➢ Pour un chiffre binaire il y a deux valeurs possibles (0,1)
  - ➢ Langage binaire = langage dont l'alphabet se réduit à l'ensemble {0,1}

Définition (2)

➢ le choix du bit due au fait que les ordinateurs utilisés pour le stockage de l'information sont essentiellement des circuits logiques a deux états physiques stables:



➢ **Les multiples de bit:**

Le **bit** (pas de notation)
L'octet = 2^3 = 8 bits (noté **1 o**)
Le **Kilo-octet** = 2^{10} = 1024 o (noté **1 Ko**)
Le **Méga-octet** = 2^{20} = $(1024)^2$ o (noté **1 Mo**)
Le **Giga-octet** = 2^{30} = $(1024)^3$ o (**1 Go**)
Le **Téra-octet** = 2^{40} = $(1024)^4$ o (**1 To**)

Représentation de l'information

Représentation de l'information

- Un ordinateur manipule des données
- Besoin de coder et représenter ces données, pouvant être:
  - De nature différente
    - Des nombres
    - Des chaînes de caractères
    - Des informations de tout genre
  - De taille différente
    - Taille fixe de X chiffres : numéro de téléphone, code postal ...
    - De taille variable : nom, adresse, texte, film vidéo ...

Représentation des nombres

Codage des nombres: Système de numération

- **Qu'est-ce qu'un nombre ?**
  - ✓ Une quantité
  - ✓ La mesure de quelque chose

➢ Codage des nombres : dans un but de calcul

➢ **Sa représentation ?**

✓ **Les chiffres (symboles)**
Les briques de bases pour représenter un nombre

➢ Plusieurs bases de codage possibles
➢ **Bases les plus utilisées**
- ✓ Pour les êtres humains : base décimale
- ✓ **Pour un ordinateur**
  - o Base binaire (2) et dérivées : base hexadécimale (16) ou octale (8)

Numération et codage

- On a l'habitude de représenter les nombres en base décimale ou **base 10**.
- Ce système est donc composé de 10 symboles (ou chiffres ou digits : 0, 1, 2, 3...9) permettant de coder tous les nombres à partir des puissances de 10.

➢ **Les puissance de la base du système**

- Base 10 : 1 (un), 10 (dix), 100 (cent), 1000 (mille), etc.

Exemple:

$$101_{10} = 1 \times 10^2 + 0 \times 10^1 + 1 \times 10^0$$
$$101_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5_{10}$$

Poids des chiffres

➢ La position respective des chiffres représente leur **poids** (unité, dizaine, millier,…)

➢ **l'association** de chiffres est appelé **nombre.**

➢ Dans le cas d'un nombre codé en base 10, on parle de nombre **décimal.**

$N_{base\ b}$				
Poids du chiffre	b^n	b^2	b^1	b^0
Rang du chiffre	N	2	1	0

Numération et codage

- Il est important de distinguer le concept de nombre de sa représentation graphique
- La représentation graphique d'un nombre dépend :
  - des symboles utilisées (les chiffres)
  - de la base utilisées (le nombre de chiffres disponibles)

- Un même nombre peut être représenté dans plusieurs bases.
- **Exemple:** le nombre 123 est représenté graphiquement:

123 en base 10 (décimal)
1111011 en base 2 (binaire)
173 en base 8 (octale)
7B en base 16 (hexad´ecimale): notation plus économe en place

Codage en base B

❑ Pour une base *B*, il y a *B* symboles différents (les chiffres de cette base)

- **Base 10** – Décimal
  - 10 symboles [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
- **Base 2** – Binaire
  - 2 symboles [0, 1]
- **Base 8** – Octale
  - 8 symboles [0, 7]
- **Base 16** – Hexadécimal
  - 16 symboles [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F]

Codage en base B

- Dans une base B, un entier naturel N s'écrit sous la forme

  $(N)_B = a_n\ a_{n-1}\ a_{n-2} \ldots a_1\ a_0$

  - Avec a_x qui est un des B chiffres de la base
- Exemples
  - Base décimale : 1234
    - ✓ De droite à gauche : chiffre des unités, des dizaines, des centaines, des milliers…
  - Base binaire : 11001
  - Base hexadécimale : 1C04

Base B vers décimal (polynôme)

- Valeur en décimal (base 10) d'un entier naturel

  « $a_n\ a_{n-1} \ldots a_1\ a_0$ » codé dans une base B
  - $a_nB^n + a_{n-1}B^{n-1} + \ldots a_1B + a_0$
  - En prenant la valeur décimale de chaque chiffre a_x

- Exemples:
  - $(1234)_{10}$ = 1 x 103 + 2 x 102 + 3 x 10 + 4
  - $(11001)_2$ = 1 x 24 + 1 x 23 + 0 x 22 + 0 x 2 + 1 = 16 + 8 + 1 = 25
  - $(1C04)_{16}$ = 1 x 163 + 12 x 162 + 0 x 16 + 4

    = 4096 + 12 x 256 + 0 + 4 = 7172

  avec A = 10, B = 11, C = 12, D = 13, E = 14, F = 15

Conversion Binaire → Décimal

Exemple :

$(1\ 1\ 0\ 0\ \ 1)_2 = \ldots.)_{10}$?

On additionne les poids associés à chaque symbole

$$\begin{array}{ccccc} 16 & 8 & 4 & 2 & 1 \\ 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$$(1\ \ 1\ \ 0\ \ 0\ \ \ 1)_2 \quad = 1 + 8 + 16 = (25)_{10}$$

Conversion Hexadécimal → Décimal

Exemple : $(B\ \ 2\ \ 2)_{16} = \ldots)_{10}$?

On additionne les poids associés à chaque symbole

$$\begin{array}{ccc} 256 & 16 & 1 \\ 16^2 & 16^1 & 16^0 \end{array}$$

$$(B\ \ 2\ \ 2)_{16} \quad \begin{aligned} &= Bx256 + 2x16 + 2x1 \\ &= 11x256 + 32 + 2 \\ &= (2850)_{10} \end{aligned}$$

Décimal vers base B

- On procède par une série de divisions entières par B
- Division du nombre décimal N par B : donne une valeur v_0 et un reste r_0
- On divise v_0 par B : donne v_1 et reste r_1
- On recommence pour v_1 et ainsi de suite
- Quand v_x < B, c'est fini
  - Le résultat de la prochaine division donnera 0
- $(N)_B = v_x\, r_{x-1} \ldots r_1 r_0$

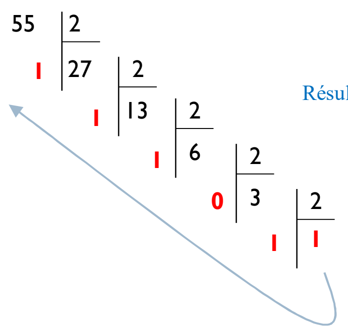
Exercice:
écrire un programme en qui permet de convertir un nombre décimale vers une base B?

Conversion Décimal → Binaire

Exemple :

$$(55)_{10} = (\ldots)_2$$

On effectue des divisions successives par 2



Résultat: $(55)_{10} = (110111)_2$

Conversion Décimal → Hexadécimale

- Exemple : $(7172)_{10}$ en hexadécimal

```
◆ 7172 | 16
    4  +----+
        448 | 16
         0  +----+
             28 | 16
              C +---
                  1
```

Résultat : $(1C04)_{16}$

Cas particuliers : Conversion du binaire à l'octal/hexadécimal ou inverse

- **Remarque:**
  - ✓ 1 chiffre octal = un groupe de 3 chiffres binaires
  - ✓ 1 chiffre hexadécimal = un groupe de 4 chiffres binaires

Nombre binaire	
Pour Octal	Pour Hexadécimal
000	0000
001	0001
010	0010
011	0011
100	0100
101	0101
110	0110
111	0111

Exemples

- Exemple : $(10110001101)_2$ en octal
- On regroupe par groupes de 3 bits :
  010 110 001 101
  - On rajoute des zéros au début au besoin
- $(010)_2 = 2$, $(110)_2 = 6$, $(001)_2 = 1$, $(101)_2 = 5$
- $(10110001101)_2 = (2615)_8$

- Exemple : $(10110001101)_2$ en hexadécimal
- On regroupe par groupes de 4 bits :
  0101 1000 1101
- $(0101)_2 = 5$, $(1000)_2 = 8$, $(1101)_2 = 13$
- $(10110001101)_2 = (58D)_{16}$

Codage des nombres

- On a vu le codage des entiers naturels (uniquement positifs) dans différentes bases
- Mais on doit aussi pouvoir manipuler des
  - Nombres réels
  - Nombres entiers négatifs

Codage de l'information

- Les informations directement traitées par un ordinateurs sont :
  - des données :
    - entiers : naturels et relatifs
    - flottants : nombres réels
    - caractères

      ✓ Le codage de ces trois types est défini par des standard (normes spécifiées par des organisations internationales)

  - des instructions :
    - ✓ leur codage est spécifique à un processeur

Cas des entiers naturels

- Les entiers naturels (positifs ou nuls) sont codés sur un nombre d'octets fixé (1, 2 ou 4 octets)
- Un codage sur n bits permet de représenter tous les nombres naturels compris entre 0 et 2^n -1.
- La représentation en machine est effectuée de la façon suivante :
- On représente le nombre en base 2 et on range les bits dans les cases mémoires binaires contiguës correspondant à leur poids binaire,

❑ **Exemple:**

•N= $144)_{10}$ représenté par $(0000000010010000)_2$ sur 16 bits

En mémoire nous avons donc :

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

Cas des entiers relatifs

- ❑ Les nombres entiers relatifs peuvent être positifs ou négatifs.
- ❑ Ils possèdent donc un signe.
- ❑ En machine nous devons également représenter le signe de ces nombres.

→ Comment représenter des entiers négatifs ??

- Magnitude signée
- Complément à 1
- Complément à 2
- Biaisée

Cas des entiers signée (1)

Magnitude signée

❑ Réserver un bit pour le signe (le bit le plus à gauche); les autres bits codent la valeur absolue du nombre

- 0 = « + » et 1 = « - »

➢ pour N bits: valeur signé compris entre – $(2^{n-1} - 1)$ et $2^{n-1} - 1$

❑ Exemple:

- Représentation de +5 et -5 en valeur signée sur 6 bits

+5 → 000101 - 5 → 100101

❑ Inconvenants:

- Deux représentation pour la valeur 0

+0 → 000000 - 0 → 100000

- Opérations arithmétiques peu aisées

Exemple

- Addition avec l'opposé
- $1_{10} - 1_{10} = 1_{10} + (-1)_{10}$
- $= 00000001_2 + 10000001_2$ Sur 8 bits
- $= 10000010_2$
- $= (-2)_{10}$!

Cas des entiers signée (2)

- **Complément à 1:**

Complément à 1 de **x** → Il suffit <u>d'inverser</u> chaque bit de x : 0 -> 1 ou 1 -> 0

Exemple: représentation de $-11)_{10}$

	0	0	0	0	1	0	1	1
-11 =	1	1	1	1	0	1	0	0

➢ Remarque:
Deux représentations pour 0: 00000000_2 et 11111111_2

Cas des entiers signée (3)

- Complément à 2

  Complément à 2 de **x** = Complément à 1 de x + 1

  - **Exemple:** calcul de l'opposé de (11) onze avec son complément à 2=(complément à 1)+1

	0	0	0	0	1	0	1	1	motif binaire de onze
	↓	↓	↓	↓	↓	↓	↓	↓	
	1	1	1	1	0	1	0	0	complément à 1 de onze
+	0	0	0	0	0	0	0	1	addition de + 1
=	1	1	1	1	0	1	0	1	complément à 2 de onze = opposé de onze

Remarque: Une seule représentation pour 0: 00000000_2

- Exemple :

sur 1 octet, 8 bits : codage des nombres de $-2^7 = -128$ à $2^7 - 1 = +127$

Exercice

- 1. Représenter $-20)_{10}$ en c-à-2 sur 8-bits

-20_{10}: Valeur positive =		00010100
"Inverser":	Complément à 1	11101011
Ajouter 1:		+ 1
		11101100

- 2. 1100011 est une représentation en c-à-2 sur 7-bits. Donnez la valeur?

C-à-2: Nombre négatif	1100011	
"Inverser":	0011100	(Complément à 1)
Ajouter 1:	+ 1	
Valeur absolue	0011101 = 29	
Nombre:	= - 29	

L'arithmétique binaire

- La soustraction
  - Addition avec l'opposé (complément à 2)
- Exemple :
  - $5 - 2 = 5 + (-2)$
  - $= 0101_2 + (1 + compl(10_2))$
  - $= 0101_2 + (1 + 1101_2)$ *Sur 4 bits*
  - $= 0101_2 + 1110_2$
  - $= 0011_2 = 3$

Codage de l'information

- **Codage des nombres réels:**

→Il existe deux méthodes pour représenter les nombres réels :

o **Virgule fixe** : la position de la virgule est fixe

o **Virgule flottante** : la position de la virgule change (dynamique)

Virgule fixe

- **Le cas des Nombres fractionnaires** : On multiplie la partie fractionnaire par la base en répétant l'opération sur la partie fractionnaire du produit jusqu'a ce qu'elle soit nulle

Exemple:

Conversion de $(54,25)_{10}$ en base 2.

Partie entière : $(54)_{10} = (110110)_2$

Partie fractionnaire :

$$\begin{array}{rcccccc} 0,25 \times 2 & = & 0,50 & \Rightarrow & a_{-1} & = & 0 \\ 0,50 \times 2 & = & 1,00 & \Rightarrow & a_{-2} & = & 1 \\ 0,00 \times 2 & = & 0,00 & \Rightarrow & a_{-3} & = & 0\,. \end{array}$$

Il en résulte donc que $(54,25)_{10} = (110110,010)_2$

Virgule Flottante

Chaque nombre réel peut s'écrire de la façon suivante : **N= ± M * b ^{e}**

- M : mantisse ,
- b : la base ,
- e : l'exposant

❖ Exemple :

- $15,6 = 0,156 * 10^{+2}$
- $-(110,101)_2 = -(0,110101)_2 * 2^{+3}$
- $(0,00101)_2 = (0,101)_2 * 2^{-2}$

Dans cette représentation sur n bits :

Signe mantisse	Exposant	Mantisse normalisée
1 bit	p bits	k bits

Représentation de l'information

- **Les différents types d'information**

❑ Les informations traitées par un ordinateur peuvent être de différents types
→ On distingue:

✓ Les **données discrètes,**
✓ Les **données continues ou analogiques :** Systèmes d'acquisition des données (micros, capteurs, cartes d'acquisition, …)

→ représentées et manipulées par un ordinateur sous forme binaire
❑ Toute information sera traitée comme une suite de 0 et de 1. (bit) (**codage).**

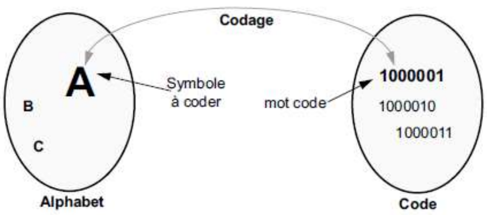
➢ **Le codage** est plus spécifiquement appelé:

✓ Codage de l'information pour les informations discrètes
✓ Numérisation de l'information pour les informations analogiques.

Le codage des informations

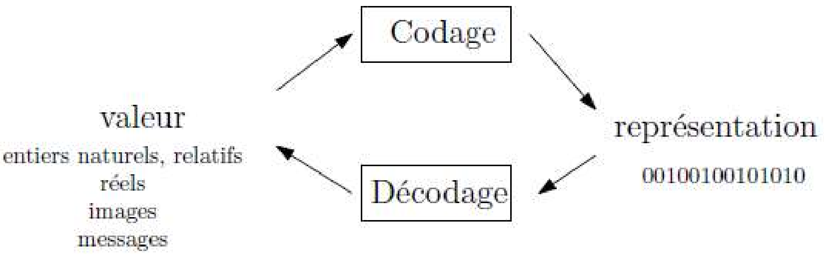
- Le codage d'une information consiste à établir une correspondance entre la représentation externe de l'information et la représentation interne dans la machine qui est une suite de bits

➢ codages alphanumérique



Représentation des données

- Une valeur, quelle que soit sa nature, doit être représenté en binaire :



Codage des données non numérique

➢ Codage des caractères :

Plusieurs formats de représentation binaire:

- EBCDIC (Extended Binary-Coded Decimal Interchange Code)
  - Représentation sur 8 bits (256 caractères possibles)
  - Utilisé autrefois sur les mainframes IBM
- ASCII (American Standard Code for Information Interchange)
  - représentation sur 7 bits (pas d'accents)
  - ASCII étendu : sur 8 bits (mais pas normalisé) (ex: OEM, ANSI)
- Unicode : encodage sur 16 bits (65536 possibilités) pour représenter tous les caractères de toutes les langues

Codage des données non numérique

- Codage des caractères :
  - Code ASCII

Caractères spéciaux

Chiffres

Lettres

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

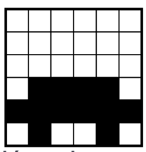
Source: www.asciitable.com

Codage de l'information

- Exemple : Codage d'une image

Image matricielle = matrice de points élémentaires

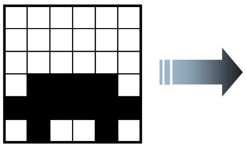
= PIcture ELement = pixel



- Chaque pixel est codé en binaire sur un certains nombre de bits

Codage de l'information

- Image noir et blanc :
  - Chaque pixel est codé sur 1 bit : 0 = blanc
    1 = noir



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

36 bits

303 x 303 x 1 bit = 91809 bits

Codage de l'information

- Image Niveaux de gris
  - Chaque pixel est codé sur plusieurs bits
  - Si on code sur 8 bits = 1 pixel = 1 octet

303 x 303 x 1 octet = 91809 octets
= 734 472 bits

Codage de l'information

- Image couleur 24 bits
  - Code RVB = Rouge, Vert, Bleu
  - Chaque couleur est codée sur 8 bits
  - La couleur du pixel est l'association des 3 couleurs
    - Chaque pixel est codé sur 24 bits (true color)

303 x 303 x 3 octets = 275 424 octets
= 2 203 392 bits

- **Exercices**

Exercices

- 1. Combien de nombre qu'on peut représenter avec n chiffres ? En décimale ? Et en binaire ?

Solution:

> - En décimal, avec n chiffres, on obtient 10^n combinaisons possibles, → [0 à 10^n -1]
>   - o Exemple : Avec 3 chiffres, on a 10^3 = 1000 combinaisons possibles → [000 a 999].
> - En binaire, avec n bits, on obtient 2^n combinaisons possibles, → [0 à 2^n -1]
>   - o Exemple : avec 8 bits, on a 2^8 = 256 combinaisons possibles → [00000000 a 11111111,] i.e. De [0 a 255].

Exercices

- **2. Convertir** $(254)_8$ en binaire ?

> - 2 = $(010)_2$, 5 = $(101)_2$, 4 = $(100)_2$
> - On concatène dans l'autre base ces groupes de 3 bits : $(254)_8$ = $(10101100)_2$

3. Convertir (D46C)16 en binaire ?

> D = 13 = $(1101)_2$, 4 = $(0100)_2$, 6 = $(0110)_2$,
> C = 12 = $(1100)_2$
> On concatène dans l'autre base ces groupes de 4 bits : $(D46C)_{16}$ = $(1101010001101100)_2$

Exercice

- 1. coder un réel positifs sur 8 bits : 1011,1101 = ...)$_{10}$

L'octet 1011,1101 code donc le réel $2^3 + 2^1 + 2^0 + 2^{-1} + 2^{-2} + 2^{-4}$ c'est-à-dire 11, 8125

- 2. traduire en binaire le nombre 78,347

  - ✓ Partie entière : 78 → 1001110
  - ✓ Partie fractionnelle : 0,347

0,347 . 2 = 0,694 < 1 je pose 0 : 0,347 = 0,0...
0,694 . 2 = 1,388 > 1 je pose 1 : 0,347 = 0,01...
0,388 . 2 = 0,766 < 1 je pose 0 : 0,347 = 0,010...
0,766 . 2 = 1,552 > 1 je pose 1 : 0,347 = 0,0101...
0,552 . 2 = 1,104 > 1 je pose 1 : 0,347 = 0,01011...

0,375 décimal est arrondi à 0,01011 en binaire avec une précision absolue de 2^{-5}

78,347)$_{10}$ =1001110,0101100011)$_2$

A suivre …

- Architecture de Base d'un ordinateur