Informatique-Electronique - AGIL Elec B

Electronique numérique :
Représentation des nombres et systèmes numériques

Renato Martins, ICB UMR CNRS - Univ. Bourgogne
UFR Sciences & Techniques - IEM, 2024

Remerciements

- Les slides du cours sont basés pour la plupart sur le support gentillement mis à disposition par **Amira Bousselmi** et par de nombreuses autres personnes.

- Je n’ai pas crédité ces personnes dans la plupart des slides (ce qui n'est pas bien...mes excuses.)

Table des Matières

Chapitre 1 :
Numérisation et arithmétique binaire

C1. Systèmes logiques et numérisation

Question: Quelles quantités représentent ces nombres/textes ?

10

77

FF

FFD700

C1. Systèmes logiques et numérisation

Question: Quelles quantités représentent ces nombres/textes ?

10

77

FF

FFD700

Question en retour : ils sont représentés dans quelle(s) base(s) ?

C1. Systèmes logiques et numérisation

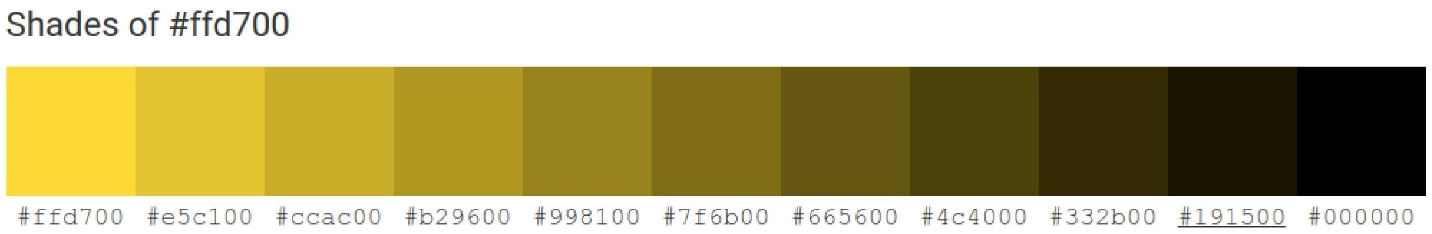
Question: Quelles quantités représentent ces nombres/textes ?

10

77

FF

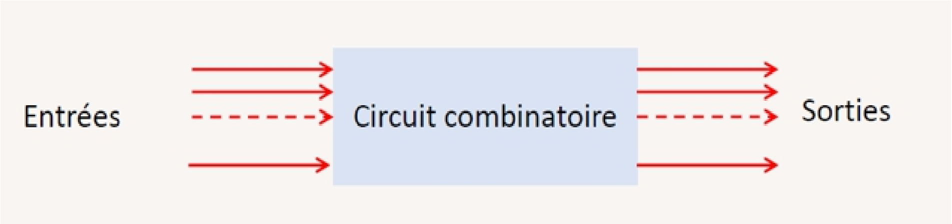
FFD700



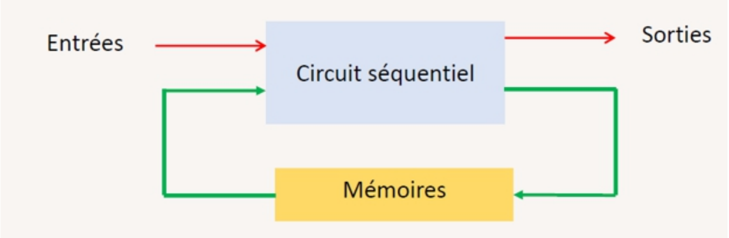
Question en retour : ils sont représentés dans quelle(s) base(s) ?

C1. Systèmes logiques et numérisation

1. Familles logiques



- 1. Logique combinatoire :
  - La fonction de sortie dépend uniquement des variables d'entrée.
  - Un circuit combinatoire est défini par une ou plusieurs fonctions logiques.
- 2. Logique séquentielle :
  - Les fonctions de sortie dependent non seulement de l'état des variables d'entrée mais également de l'état de certaines variables de sortie.



C1. Systèmes logiques et numérisation

2. Numérisation de l'information

- Un système de numération permet de coder une information (texte, image, nombre, parole ...) en lui attribuant un symbole ou combinaison de symboles compréhensible par le processeur.
- Le système conventionnel de comptage en base 10 est incompatible avec la machine, d'où la nécessité d'introduire d'autres systèmes de numération. Les systèmes de numération binaire et Hexadécimal sont les plus utilisés dans le domaine de l'électronique et de l'informatique.
- Il existe plusieurs systèmes de numérotation, appelés «**base**» :
  - Base Décimale, (10) : Les chiffres décimales sont : (0,1,2,3,4,5,6,7,8,9)
  - Base Binaire, (2) : Les chiffres binaires sont : (0,1) appelés bits ou digits
  - Base Octale, (8) : Les chiffres octales sont : (0,1,2,3,4,5,6,7)
  - Base Hexadécimale, (16) : Les chiffres hexadécimales sont : (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)

C1. Systèmes logiques et numérisation

- En général, dans le système de numérotation de base « B » le nombre $(N)_B$ s'écrit :

$$(N)_B = (a_{n-1} \ldots a_1\, a_0, a_{-1}\, a_{-2} \ldots)_B = (a_{n-1}.\, B^{n-1} + a_{n-2}.\, B^{n-2} + \ldots\ldots + a_1.\, B^1 + a_0.\, B^0 + a_{-1}.\, B^{-1} + a_{-2}.\, B^{-2} + \ldots)$$

- **Exemple :**
  - En numérotation décimale, le nombre $(78{,}13)_{10}$ s'écrit :

$$(78{,}13)_{10} = 7.10^1 + 8.10^0 + 1.10^{-1} + 3.10^{-2}$$

  - En numérotation binaire, le nombre $(1011{,}101)_2$ s'écrit :

$$(1011{,}101)_2 = 1.2^3 + 0.2^2 + 1.2^1 + 1.2^0 + 1.2^{-1} + 0.2^{-2} + 1.2^{-3}$$

3. Conversion d'un nombre d'une base à une autre

Il s'agit du processus de conversion d'un nombre écrit dans une base b1 à une autre base b2.

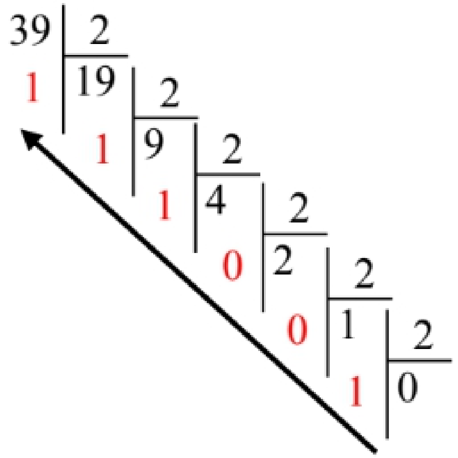
1- Le codage

La conversion d'un nombre décimal en une autre base non décimale est appelée codage

1.1 Codage en un nombre binaire

Le principe consiste à faire des divisions successives du nombre sur 2 et prendre le reste de la division dans l'ordre inverse.

- Exemple 1: $(39)_{10} = (?)_2$



- Après la division, le nombre recherché est : ?

1.2 Codage en un nombre octal

On utilise la méthode de divisions successives par 8 jusqu'à un quotient égale à 0. Les restes successifs pris de bas en haut forment le nombre codé en octal.

- Exemple 1: $(423)_{10} = (?)_8$

1.3 Codage en un nombre hexadécimal

On utilise la méthode de divisions successives par 16 jusqu'à un quotient égale à 0. Les restes successifs pris de bas en haut forment le nombre code en hexadécimal.

- Exemple 1: $(423)_{10} = (1A7)_{16}$

2- Le décodage

Le décodage est la conversion d'un nombre quelconque écrit en base **m** (octal, binaire ou hexadécimal) en base décimal. La forme générale est :

$$(y)_{10} = \sum_{n=0}^{i-1} X_n . m^n \qquad avec \begin{cases} m : la\ base \\ i : nombre\ de\ symbole \\ n : le\ rang \end{cases}$$

- Exemples

$(100111)_2 = 1*2^5+0*2^4+0*2^3+1*2^2+1*2^1+1*2^0=(39)_{10}$

$(647)_8$ =?

$(1A7)_{16}$ =?

C1. Systèmes logiques et numérisation

- **Exercice 1** : Quelle est la valeur maximale représentée avec 8 bits (en base binaire) ?

- **Exercice 2** : Vous aimeriez representer la temperature de la salle de cours qui peut varier entre 20 et 25 dégrées avec une précision de 0.1 dégrées. Combien de chiffres avez vous besoin avec une base 2 ? Et avec une base 8 ?

- **Conversion Hexadécimal ➔ Binaire**

Chaque symbole de base hexadécimal s'écrit sur 4 bits

Hexadécimal ➔ $(1\ A\ F\ 3\ )_{16}$

Binaire ➔ $(0001\ 1010\ 1111\ 0011)_2$

- **Conversion Binaire ➔ Hexadécimal**

Binaire ➔ $(1101\ 0000\ 1100)_2$

Hexadécimal ➔ $(D0C)_{16}$

Exercice 1 : Convertir en binaire, octal et en hexadécimal, les nombres décimaux suivants :

a) 22_{10}

b) 63_{10}

c) 242_{10}

d) 3_{10}

Exercice 2 : Convertir les nombres suivants :

a) $(11011)_2 = (.....)_{10} = (.....)_8$

b) $(111101,101001)_2 = (.....)_{10}$

c) $(DA37)_{16} = (.....)_{10}$

4- Conversion des parties fractionnaires:

- **Multiplications successives par B.**
  - **Exemples**:

0,8125	
*2	
1,6250	MSB
*2	
1,2500	
*2	
0,5000	
*2	
1,0000	LSB

$(0,8125)_{10} = (0,1101)_2$

0,8125	
*8	
6,5000	MSB
*8	
4,0000	LSB

$(0,8125)_{10} = (0,64)_8$

Exercice 2 : Convertir en binaire les nombres décimaux suivants :

a) $(1{,}25)_{10}$

b) $(255{,}255)_{10}$

c) $(0{,}001)_{10}$

4. Calcul arithmétique

Dans cette partie, on définit :

- ✓ Les nombres binaires non signés (pas de bit de signe) et le calcul arithmétique se fait comme dans le système décimal (base 10).
- ✓ Les nombres binaires signés dont on tient comptent de signe positif ou négatif du nombre et on considère un bit de signe.

1- Opérations arithmétiques pour les nombres non signés

On procède de la même façon que celle utilisée dans la base décimale. Ainsi, on peut effectuer l'opération dans la base 10, ensuite convertir le résultat par colonne à la base B.

1.1 Opérations d'addition

Cas du système binaire:

- Exemple :

  $N1 = (254)_{10}$ ➔

  $N2 = (187)_{10}$ ➔

- N1 + N2 = ?

	somme	retenue
0+0	0	0
0+1	1	0
1+1	0	1

4. Calcul arithmétique

Cas du système octal

$$\begin{array}{r} \boxed{1} \\ 34_8 \\ +\,45_8 \\ \hline 101_8 \end{array} \qquad \begin{array}{l} 3_8+1_8 = 4_8 \\ 4_8+4_8 = 10_8 \end{array}$$

$$\begin{array}{r} (2\ \ 5\ \ 3)_8 \\ +\ (7\ \ 1\ \ 6)_8 \\ \hline \end{array}$$

Cas du système héxadecimal

$$\begin{array}{r} (5\ 3\ 0\ 4)_{16} \\ +\ (CC\ 3\ B)_{16} \\ \hline \boxed{} \end{array} \qquad \begin{array}{r} (89A27)_{16} \\ +\ (\ EE54)_{16} \\ \hline \boxed{} \end{array}$$

Exercice 1 :

a) $\begin{array}{r} 1111 \\ +\ 0101 \\ \hline \boxed{} \end{array}$

b) $\begin{array}{r} 10101010 \\ +\ 00110011 \\ \hline \boxed{} \end{array}$

c) $\begin{array}{r} 11001101 \\ +\ 11100011 \\ \hline \boxed{} \end{array}$

Exercice 2 :

$326_8 + 735_8 =$

$BB_{16} + 36_{16} \simeq$

$19B9_{16} + C7E6_{16} \simeq$

4. Calcul arithmétique

4.2 Opérations de soustraction

- Cas d'une opération de soustraction de deux opérandes non signées A et B, il faut que A<B.
- On peut soustraire les nombres dans les systèmes, binaire, octal ou hexadécimal, en utilisant la même méthode que le système décimal.

Exemples :

- Cas du système binaire

$$\begin{array}{r} 11111110 \\ -\ 10111011 \\ \hline \end{array} \qquad \begin{array}{r} 101101 \\ -\ 11011 \\ \hline \end{array}$$

0-0	0
1-0	1
1-1	0
0-1	1, Il faut prendre 1 de la gauche

4. Calcul arithmétique

4.2 Opérations de soustraction

- Cas d'une opération de soustraction de deux opérandes non signées A et B, il faut que A<B.
- On peut soustraire les nombres dans les systèmes, binaire, octal ou hexadécimal, en utilisant la même méthode que le système décimal.

Cas du système octal :

$$\begin{array}{r} (7\ 3\ 5)_8 \\ -\ (4\ 7\ 6)_8 \\ \hline \end{array}$$

Cas du système hexadecimal :

$$\begin{array}{r} (1\ 5\ C\ E)_{16} \\ -\ (7\ D\ 5)_{16} \\ \hline \end{array}$$

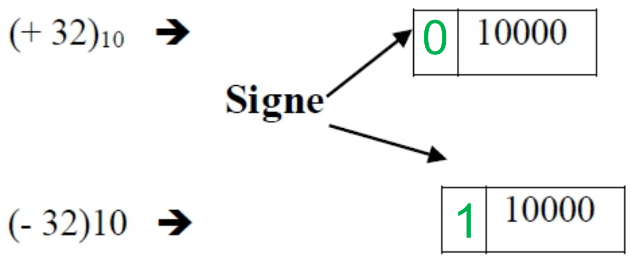
2- Opérations arithmétiques pour les nombres signés

Pour représenter des nombres négatifs et positifs dans une machine, on utilise généralement des représentations spécifiques. Dans cette partie, on s'intéresse au système binaire.

2.1 Représentation exacte

Elle consiste à affecter un bit pour le signe. Par convention : + ➔ 0 et - ➔ 1

Exemple:



2.2 Représentation en complément à 2

Cette représentation n'est pas valable par des calculs arithmétiques car elle possède 2 représentions possibles de zéro (+0 ➔00) et (-0 ➔ 10). D'ou il y a recours à l'utilisation de la notion complément (CMP).

D'une manière générale, le complément à "b" d'un nombre écrit dans la base "b" est la valeur qu'il faut ajouter à ce nombre pour obtenir la valeur maximale +1 que l'on peut exprimer (en tenant compte du format de la base).

2.2 Représentation en complément à 2

Méthode 1

Pour calculer le complément à 2 d'un nombre N, on effectue le CMP_1+1. Sachant le complément à 1 se détermine comme suit :

CMP_1(N) =$\overline{(N)}$ cad 0 si 1 et 1 si 0

Exemple :

CMP_1(10110) =...

CMP_2(10110) =...

Remarque: Maintenant on doit dire combien de bits on répresente le nombre (5, 8, 16 ou 32 bits ?)

2.3 Utilisation du complément à 2

Les opérations des nombres signés.

Exemple 1 : $+9_{10}$ -4_{10}, dans ce cas le résultat s'écrit sur 5 bits.

$$+9_{10} \rightarrow \overbrace{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots}^{5bits}$$

$$+$$

$$-4_{10} \rightarrow \underline{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots}$$

$$en\ effet : CMP_2(+4) = CMP_2(00100) = \ldots\ldots\ldots$$

Exemple 2 : -9_{10} -4_{10}, dans ce cas le résultat s'écrit sur 5 bits.

$$-9_{10} \rightarrow \quad CMP_2(+9) = CMP_2(01001) = \overbrace{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots}^{5bits}$$

$$+$$

$$-4_{10} \rightarrow \quad CMP_2(+4) = CMP_2(00100) = \underline{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots}$$

On remarque que le signe de bit est **1** alors le résultat est négatif. Il faut complémenter à 2 pour retrouver la valeur absolue.

$$10011 \ \xrightarrow{CMP_2}\ 01101 \rightarrow$$