# INFORMATIQUE

# REPRÉSENTATION DES ENTIERS

PTSI – 2014 – 2015



Xavier Pessoles - PTSI - Rouvière

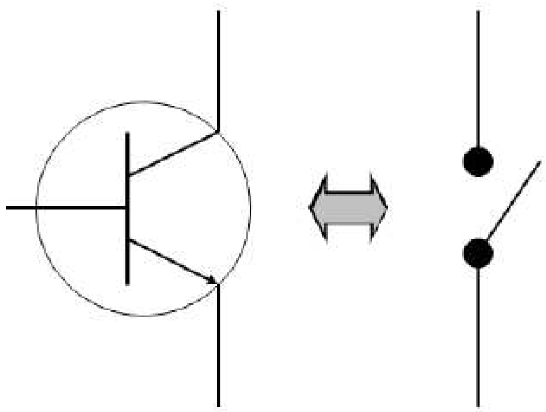
Rouvière PTSI

# GÉNÉRALITÉS
# LOGIQUE BINAIRE

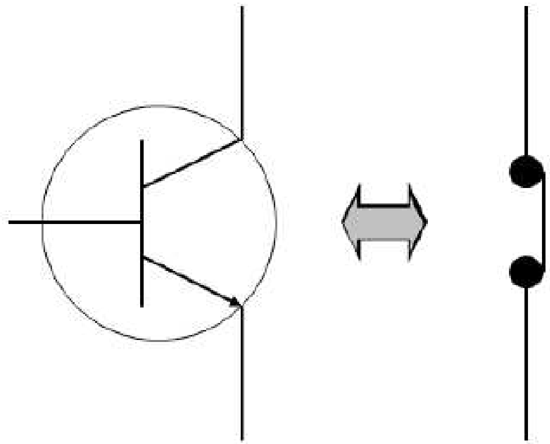
*Définition*

**Bit**

On appelle bit une information élémentaire de type 0 ou 1 (contraction de l'anglais *binary digit*)



*Transistor bloqué*
*Interrupteur ouvert)*
*Le courant ne passe pas*
*État logique : 0*



*Transistor saturé*
*(interrupteur fermé)*
*Le courant passe*
*État logique : 1*

# Généralités
# Notion de mot

Définition

**Mot** – *Word*

Ensemble de bits de longueur fixe.

Suivant le type de processeur, les mots peuvent avoir 32, 64 ou 128 bits pour les processeurs Intel Itanium.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

Quartet

Octet

Mot de 32 bits

Remarque

*byte* est la traduction anglaise du mot octet. En conséquence, un *byte* équivaut à une séquence de 8 bits.

# Généralités
# Notion de pondération

*Définition*

**Écriture d'un nombre dans une base**

Dans un système de numération en base $B$, un nombre noté $N_B$ peut s'écrire sous la forme :

$$N_B = \sum_{k=0}^{n} a_k \cdot B^k$$

s'écrit symboliquement sous la forme :

$$N_B = \underbrace{(a_n a_{n-1} \cdots a_2 a_1 a_0)_B}_{n+1 \text{ chiffres}}$$

On note :

- $B$ : la base ou nombre de chiffres différents qu'utilise le système de numération ;
- $a_k$ : chiffre de rang $k$ ;
- $B^k$ la pondération associée à $a_k$.

*Exemple*

$$247_{(10)} = 2 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0$$

On appelle :

- 2 est appelé digit de poids fort (*most significant digit*) ;
- 7 est appelé digit de poids faible (*least significant digit*) ;

$$1001_2 = 1 \cdot 2^{11_2} + 0 \cdot 2^{10_2} + 0 \cdot 2^{1_2} + 1 \cdot 2^{0_2} = 1 \cdot 2^{3_{10}} + 0 \cdot 2^{2_{10}} + 0 \cdot 2^{1_{10}} + 1 \cdot 2^{0_{10}}$$

# Représentation des nombres entiers naturels
# De la base k à la base 10

*Savoir*

**Savoir-Faire :** Représenter en base dix un entier naturel donné en base $k$.

Pour trouver la représentation en base dix d'un entier naturel donné en base $k$, on utilise le fait qu'en base $k$, le chiffre le plus à droite représente les unités, le précédent les paquets de k, le précédent les paquets de $k \cdot k = k^2$, le précédent les paquets de $k \cdot k \cdot k = k^3$, *etc.*

# Représentation des nombres entiers naturels
## De la base k à la base 10

- Convertir $(10101)_2$ en base 10.

- Méthode des divisions successives

*Méthode*

**Divisions successives**

On note $N_{10} = a_n \cdot k^n + a_{n-1} \cdot k^{n-1} + ... + a_1 \cdot k^1 + a_0 \cdot k^0 = a_n a_{n-1} ... a_1 a_0$

1. Calcul de la division de $N_{10}$ par k.
   - Le reste de la division correspond au terme $a_0$.
2. Le dividende de la division est divisé par k.
   - Le reste de la division correspond au terme $a_1$.
3. ....

La division s'arrête lorsque le dividende est nul.

- Convertir $(247)_{10}$ en binaire.

# Représentation des nombres entiers naturels
# De la base 10 vers la base k

- Méthode de la plus grande puissance

*Méthode*

Elle consiste à retrancher du nombre initial la plus grande puissance de k possible et ainsi de suite dans l'ordre décroissant des puissances. Si on peut retirer la puissance de k concernée, on note 1 sinon on note 0 et on continue de la sorte jusqu'à la plus petite puissance de k possible soit $k^0$ pour des entiers naturels.

- Convertir $(247)_{10}$ en binaire.

# Représentation des nombres entiers naturels
## Capacités de la représentation

*Résultat*

Si un mot est codé sur $n$ bits, on peut représenter un entier naturel compris entre 0 et $2^n - 1$, soit $2^n$ valeurs possibles.

*Exemple*

Si les mots sont codés sur un octet (8 bits), on peut compter de 0 à $2^8 - 1$, c'est-à-dire de 0 à 255.

| **Bits** | **Nombre de valeurs** | **de 0 à ...** |
|---|---|---|
| 4 | 16 | 15 |
| 8 | 256 | 255 |
| 16 | 65 536 | 65 535 |
| 32 | 4,29... milliards | 4,29... milliards |
| 64 | $1,84...10^{19}$ | $1,84...10^{19}$ |

## Capacités de la représentation

Remarque

**Dépassement de capacité –** ***Overflow***

Le résultat de l'addition de deux nombres codés sur le même nombre de bit n'est pas toujours possible car le résultat pourrait demander des bits supplémentaires.

Remarque

En effet, considérons un système où les mots sont codés sur un octet. Calculons $247_{(10)} + 53_{(10)}$

$$247_{(10)} + 53_{(10)} = 300_{(10)} = 1\underbrace{00101100}_{\text{octet retenu}}$$

Ainsi le résultat retenu est $001011100_{(2)} = 44_{(10)}$ au lieu de $300_{(10)}$.

On parle alors de dépassement de capacité (*overflow* en anglais). Sur certains ordinateurs, les calculs continuent. Sur d'autres, une erreur est signalée, d'une façon différente d'un constructeur à l'autre.

*Résultat*

**Addition en binaire**

En binaire, l'addition peut être considérée ainsi :

$$\begin{cases} 0_2 + 0_2 = 0_2 \\ 1_2 + 0_2 = 1_2 \\ 0_2 + 1_2 = 1_2 \\ 1_2 + 1_2 = 10_2 \end{cases}$$

On en déduit rapidement que l'addition en binaire s'effectuera sur le même "modèle" que l'addition en écriture décimale, à ceci près que la retenue se produit dès qu'on arrive à 2 (au lieu de 10).

- Faire la somme de (247)10 et (53)10 en binaire.
- Que se passe-t-il si les nombres sont codés sur 1 octet dans la machine ?

# EXERCICES

- Pour chacun des nombres suivants, donner leur conversion en binaire, hexadécimal ou décimal :
  - $(10050)_{10}$
  - $(10010001)_2$
  - $(A3F)_{16}$

- On désire utiliser 12 bits pour comptabiliser les des objets.
  - Quel est le nombre maximal d'objet qu'on peut compter ?
  - Indiquer le premier et le dernier nombre en binaire, décimal et hexadécimal.

- On désire compter 65000 objets.
  - Combien de bits sont nécessaires ?
  - Quel est le numéro du premier et du dernier objet ?

- Dans un système où les nombres sont codés sur 8 bits, réaliser en binaire la somme suivante :
  - 71 + 35

# REPRÉSENTATION DES ENTIERS RELATIFS
# PREMIÈRE APPROCHE

# Représentation des entiers relatifs
## Représentation en complément

# Représentation des entiers relatifs
# Capacité de représentation

# Représentation des entiers relatifs
# Notation Hexadécimale