

**INTRODUCTION AUX  
SYSTÈMES INFORMATIQUES**  
**REPRÉSENTATION FLOTTANTE**

*Victor Ostromoukhov*

Département d'Informatique et de Recherche Opérationnelle  
Http : [//www.iro.umontreal.ca/~ostrom/](http://www.iro.umontreal.ca/~ostrom/)  
E-mail : [ostrom@iro.umontreal.ca](mailto:ostrom@iro.umontreal.ca)

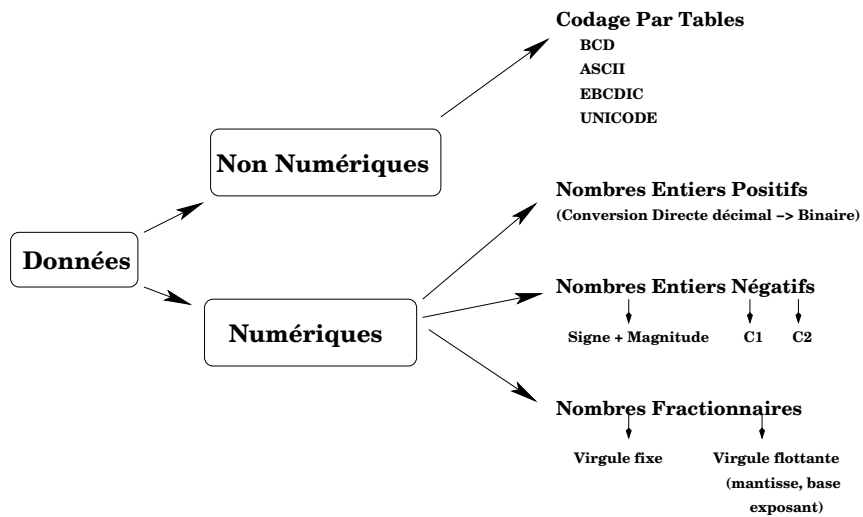
**REPRÉSENTATION FLOTTANTE**  
**SOMMAIRE**

Récapitulatif .....	2
Introduction .....	3
Conversion en Base 10 .....	4
Conversion en Flottant .....	5
Arithmétique en Flottant .....	6
IEEE 754 Standard .....	8
Packed Decimal Format .....	9
Programmation .....	10

# REPRÉSENTATION FLOTTANTE

## RÉCAPITULATIF

### Récapitulatif

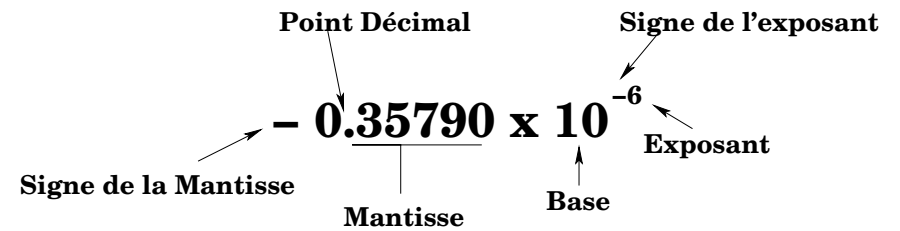


# REPRÉSENTATION FLOTTANTE

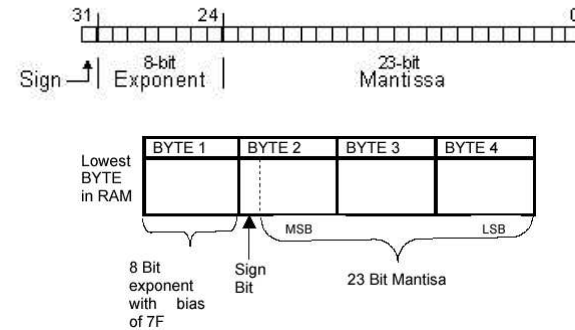
## INTRODUCTION

### Introduction

- La représentation flottante permet la représentation de nombres très petits ou très grands en utilisant très peu de digits, au prix d'une représentation avec une certaine précision



### Format Flottant sur 32 bits



- Bit de Signe (0 positive ou 1 négative)
- Exposant de  $-127$  à  $+128$  avec biais de  $127$  (exemple : vraie exposant de  $-6 \triangleright 127-6 = 121$ )

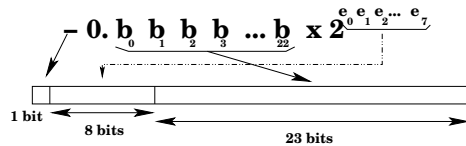
Biais de 50  $\triangleright$

Representation	0	49	50	99
Exponent being represented	-50	-1	0	49

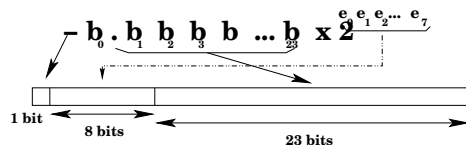
— Increasing value  $\rightarrow$  +

## REPRÉSENTATION FLOTTANTE CONVERSION EN BASE 10

1



2



- **Normalisation** : On peut toujours s'arranger pour que  $b_0 \neq 0$  (dans certain système il n'est pas stocké ▷ [bit caché])

### Exemple

IEEE Format **1** sans bit caché conversion en base 10

**0** | 0110 1000 | 101 0101 0100 0011 0100 0010

- Sign: 0 => positive
- Exponent:
  - 0110 1000<sub>two</sub> = 104<sub>ten</sub>
  - Bias adjustment: 104 - 127 = -23
- Significant:
  - + 1x2<sup>-1</sup> + 0x2<sup>-2</sup> + 1x2<sup>-3</sup> + 0x2<sup>-4</sup> + 1x2<sup>-5</sup> + ...
  - = 1 + 2<sup>-1</sup> + 2<sup>-3</sup> + 2<sup>-5</sup> + 2<sup>-7</sup> + 2<sup>-9</sup> + 2<sup>-14</sup> + 2<sup>-15</sup> + 2<sup>-17</sup> + 2<sup>-22</sup>
  - = + 0.666115
- Represents .666115 \* 2<sup>-23</sup>

## REPRÉSENTATION FLOTTANTE CONVERSION EN FLOTTANT

Exemples : Base 10 Format 1

**Ex1** : Convertir +9.9520<sub>10</sub> en notation flottante avec un exposant biaisé de 50 (2 digits) et 5 digits pour la mantisse

- Notation flottante normalisée  
▷ 0.99520 × 10<sup>1</sup>
- Chiffre Positif, exposant de 50 + 1 = 51

**Ex2** : Convertir +0.067850<sub>10</sub> pour le même format

- Notation flottante normalisée  
▷ 0.67850 × 10<sup>-1</sup>
- Chiffre Positif, exposant de 50 - 1 = 49

**Ex3** : Que représente ▷ 0 52 10020 dans ce format ?

- Chiffre positif, exposant de 52 - 50 = 2  
▷ + 0.10020 × 10<sup>2</sup> = 10.020

## REPRÉSENTATION FLOTTANTE ARITHMÉTIQUE EN FLOTTANT

### Addition & Soustraction

---

**Ex :** Faire  $9.9520_{10} + 0.067850_{10}$   
en notation flottante avec un exposant biaisé de 50 (2  
digits) et 5 digits pour la mantisse

–1– *Conversion*  $9.9520_{10} \triangleright 0\ 51\ 99520$ ,  
 $0.067850_{10} \triangleright 0\ 49\ 67850$

–2– *Aligner le plus faible exposant sur le plus fort*  
 $0\ 49\ 67850 \triangleright 0\ 51\ 0067850$

–3– Additionner les deux mantisses

0 51	99520
0 51	0067850
-----	
0 51	(1)0019850
0 52	10019850

–4– On arrondi car la mantisse à seulement 5 chiffres  
 $0\ 52\ 10020$

- Perte de Précision !

## REPRÉSENTATION FLOTTANTE ARITHMÉTIQUE EN FLOTTANT

### Multiplication & Division

---

- On multiplie les mantisses et on ajoute les exposants

**Ex :** Faire  $9.9520_{10} \times 0.067850_{10}$   
en notation flottante avec un exposant biaisé de 50 (2  
digits) et 5 digits pour la mantisse

–1– *Conversion*  $9.9520_{10} \triangleright 0\ 51\ 99520$ ,  
 $0.067850_{10} \triangleright 0\ 49\ 67850$

–2– On ajoute les exposants

51 = 50	+ 1
49 = 50	- 1
-----	
50 + 0	= 50

–3– On multiplie les deux mantisses  
 $0.99520 \times 0.67850 = 0.6752432$

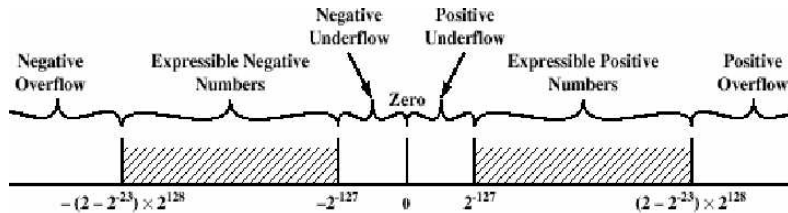
–4– On normalise et on tronque  $0\ 50\ 67524$

## REPRÉSENTATION FLOTTANTE IEEE 754 STANDARD

### IEEE 754 Standard

#### Simple Précision

- Bit de signe 1 bit
- Exposant sur 8 bits (biais de 127)
- Mantisse de 23 bits



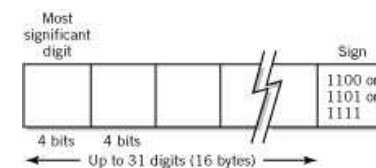
#### Double Précision

- Bit de signe 1 bit
  - Exposant sur 11 bits (biais de 1023)
  - Mantisse de 52 bits
- Format 2  
i.e., avec normalisation du style  $1.MMMMM...$
  - Bit caché
  - Le nombre 0 est défini par une mantisse remplie de 0 et un exposant remplie de 0
  - Le nombre  $\infty$  est défini par une mantisse remplie de 0 et un exposant remplie de 1
  - NaN est représenté par  
0 11111111 011 0111 0000 0000 0000 0000

## REPRÉSENTATION FLOTTANTE PACKED DECIMAL FORMAT

### Format décimal empaqueté

- Dans certaines applications, il est important de maintenir une précision parfaite pour la représentation des nombres réels (ex : dollars et cents)
- Quelque langage de haut niveau (ex : Cobol) fournit un format qui permet de spécifier un nombre de décimal désiré
- Chaque digit du nombre en base 10 est codé en BCD
- Utilisé par IBM System 370/390 et Compaq Alpha



Exemple :  $-324.6 = 0000\ 0011\ 0010\ 0100\ 0110\ 1101$

# REPRÉSENTATION FLOTTANTE

## PROGRAMMATION

### Programmation

---

#### Calcul avec des Entiers

- Plus facile pour l'ordinateur
- Très haute précision ou même sans perte de précision
- Plus rapide (moins d'accès mémoire, circuit logique plus simple, etc.)
- Langage C : *short* (16bits), *long* (32 bits)

#### Calcul avec des flottants

- Variable ou constante avec une partie fractionnaire
- Perte de précision
- Moins rapide
- Langage C : simple précision *float*, double précision *double*