Principes de fonctionnement des machines binaires

2019/2020

Pierluigi Crescenzi

Université de Paris, IRIF







- Tests et examens
 - CC : résultat des tests en TD / TP (semaine 4 et 10)
 - E0 : partiel (**samedi 26 octobre**)
 - E1 : examen mi décembre
 - E2 : examen fin juin
- Notes finales
 - Note session 1:25% CC + 25% E0 + 50% E1
 - Note session 2 : max(E2, 33% CC + 67% E2)
- Rappel
 - Pas de note ⇒ pas de moyenne ⇒ pas de semestre
- Site web
 - moodlesupd.script.univ-paris-diderot.fr

- Numération et arithmétique
- Numération et arithmétique en machine
- Numérisation et codage (texte, images)
- Compression, cryptographie, contrôle d'erreur
- Logique et calcul propositionnel
- Circuits numériques

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision
 - Choix de fixer la taille des représentations

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision
 - Choix de fixer la taille des représentations
 - Architectures communes 32 ou 64 bits (binary digit):
 arithmétique sur des nombres représentés sur 32 ou 64 bits

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision
 - Choix de fixer la taille des représentations
 - Architectures communes 32 ou 64 bits (binary digit):
 arithmétique sur des nombres représentés sur 32 ou 64 bits
 - $\circ \ 2^{32} = 4294967296 \approx 4,3 \times 10^9 \ {\sf et} \ 2^{64} \approx 18,4 \times 10^{18}$

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision
 - Choix de fixer la taille des représentations
 - Architectures communes 32 ou 64 bits (binary digit):
 arithmétique sur des nombres représentés sur 32 ou 64 bits
 - $\circ \ 2^{32} = 4294967296 \approx 4,3 \times 10^9 \ {\sf et} \ 2^{64} \approx 18,4 \times 10^{18}$
 - Un très grand nombre de choix pour représenter des entiers
 - $\circ \ 2^{32}! \ \mathsf{ou} \ 2^{64}!$

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

	base 2	base 10
00000000 00000000 00000000 00000000	0	0

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

	base 2	base 10
00000000 00000000 00000000 00000000	0	0
00000000 00000000 00000000 00000001	1	1

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

	base 2	base 10
$00000000\ 00000000\ 00000000\ 00000000$	0	0
$00000000\ 00000000\ 00000000\ 00000001$	1	1
0000000 00000000 0000000 00000010	10	2

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

	base 2	base 10
0000000 00000000 00000000 00000000	0	0
0000000 00000000 00000000 00000001	1	1
0000000 00000000 00000000 00000010	10	2
0000000 00000000 00000000 00000011	11	3

- Un choix parmi les 2^{32} ! choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

	base 2	base 10
00000000 00000000 00000000 00000000	0	0
$00000000\ 00000000\ 00000000\ 00000001$	1	1
$00000000\ 00000000\ 00000000\ 00000010$	10	2
$00000000\ 00000000\ 00000000\ 00000011$	11	3
• • •	• • •	• • •
11111111 11111111 11111111 11111101		4294967293
11111111 11111111 11111111 11111110		4294967294
11111111 11111111 11111111 11111111		4294967295

- Un choix parmi les $2^{32}!$ choix pour représenter les entiers de 0 à $2^{32}-1$
 - Mots sur $\{0,1\}$ dans l'ordre lexicographique

		base 2	base 10
on non signé	00000000 00000000 00000000 00000000	0	0
	00000000 00000000 00000000 00000001	1	1
	00000000 00000000 00000000 00000010	10	2
	00000000 00000000 00000000 00000011	11	3
ntati	• • •	• • •	• • •
orésen	11111111 11111111 11111111 11111101		4294967293
	11111111 11111111 11111111 11111110		4294967294
reg	11111111 11111111 11111111 11111111		4294967295

- Addition de deux nombres entiers non signés
 - Si la retenue à gauche est 1, un débordement arithmétique a eu lieu lors de l'addition

- Addition de deux nombres entiers non signés
 - Si la retenue à gauche est 1, un débordement arithmétique a eu lieu lors de l'addition
 - Exemple: 109+221 (8 bits, entiers de 0 à 251)

- Addition de deux nombres entiers non signés
 - Si la retenue à gauche est 1, un débordement arithmétique a eu lieu lors de l'addition
 - Exemple: 109+221 (8 bits, entiers de 0 à 251)

Débordement est facile à detecter

• Un choix parmi les 2³²! choix pour représenter les entiers de

$$-2^{31}+1$$
 à $2^{31}-1$

■ Mots sur $\{0,1\}$

• Un choix parmi les 2³²! choix pour représenter les entiers de

$$-2^{31}+1$$
 à $2^{31}-1$

• Mots sur $\{0,1\}$

base 10

0

2147483645

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

0 1111111 11111111 11111111 11111101

pierluigi.crescenzi@irif.fr

-2147483647

-2147483646

-2147483645

-1

-0

0

2147483645

2147483646

2147483647

• Un choix parmi les 2³²! choix pour représenter les entiers de

PF1

$$-{f 2^{31}}+{f 1}$$
 à ${f 2^{31}}-{f 1}$

$$-2^{31} + 1 a 2^{31} -$$

$$-2 + 102 - 1$$

$$-2 + 102 - 1$$

■ Mots sur
$$\{0,1\}$$

$$\mathsf{r}\left\{ 0,1
ight\}$$



1 1111111 11111111 11111111 11111111

1 1111111 11111111 11111111 1111110

1 1111111 11111111 11111111 11111101

1 0000000 00000000 00000000 00000001

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

0 1111111 11111111 11111111 11111101

0 1111111 11111111 11111111 11111111



-2147483647

-2147483646

-2147483645

0

2147483645

2147483646

2147483647

• Un choix parmi les 2³²! choix pour représenter les entiers de

$$-\mathbf{2^{31}} + \mathbf{1}$$
 à $\mathbf{2^{31}} - \mathbf{1}$

pierluigi.crescenzi@irif.fr

• Mots sur
$$\{0,1\}$$

1111111 11111111 11111111 11111110

1 0000000 00000000 00000000 00000001

1 0000000 00000000 00000000 00000000

0 000000 0000000 0000000 00000000

0 0000000 00000000 00000000 00000001

1111111 11111111 11111111 11111101

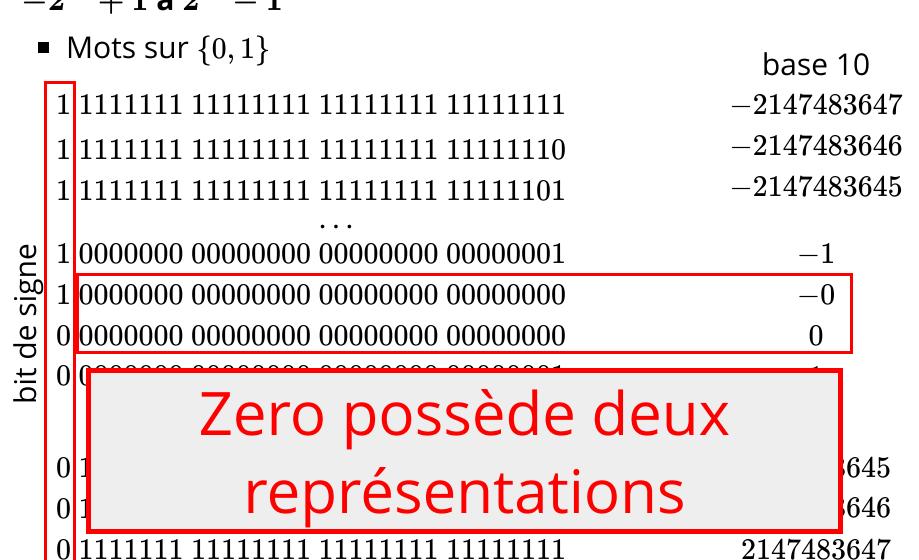
11111111 11111111 11111111

PF1



• Un choix parmi les 2³²! choix pour représenter les entiers de

$$-\mathbf{2}^{31}+1$$
 à $\mathbf{2}^{31}-1$



- Représentation signe / valeur absolue
 - Difficultés au niveau des opérations arithmétiques

- Représentation signe / valeur absolue
 - Difficultés au niveau des opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?

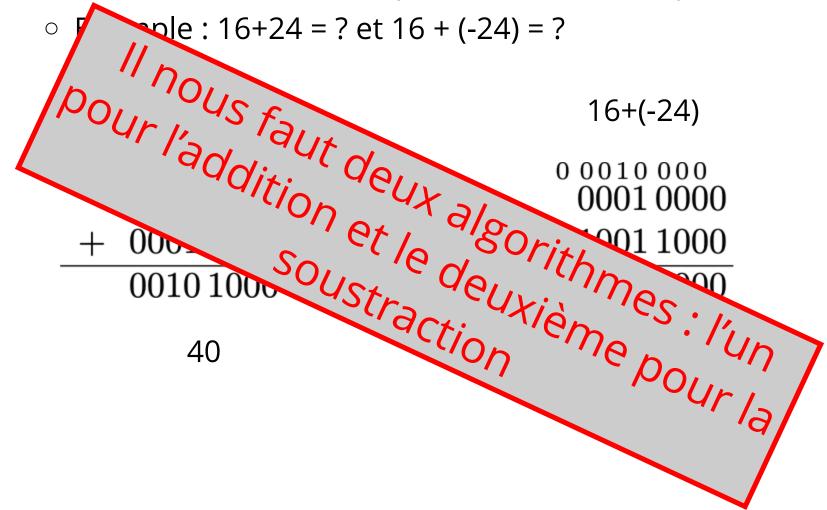
- Représentation signe / valeur absolue
 - Difficultés au niveau des opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?

$$\begin{array}{r} \begin{smallmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline + & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline & + & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ \hline & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ \hline & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

40

- Représentation signe / valeur absolue
 - Difficultés au niveau des opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?

- Représentation signe / valeur absolue
 - Difficultés au niveau des opérations arithmétiques



-2147483648

-2147483647

-3

-2

-1

0

2147483645

2147483646

2147483647

Amphi #03

• Un choix parmi les 2³²! choix pour représenter les entiers de

PF1

$$-2^{31}$$
 à $2^{31}-1$

pierluigi.crescenzi@irif.fr

Mots sur
$$\{0,1\}$$

• Mots sur
$$\{0,1\}$$

1 1111111 11111111 11111111 11111101

1 1111111 11111111 11111111 11111110

1 1111111 11111111 11111111 11111111

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

0 1111111 11111111 11111111 11111101

0 1111111 11111111 11111111 11111111

-2147483648

-2147483647

-3

-2

-1

0

2147483645

2147483646

2147483647

• Un choix parmi les 2³²! choix pour représenter les entiers de

$$-\mathbf{2^{31}}$$
 à $\mathbf{2^{31}}-\mathbf{1}$

pierluigi.crescenzi@irif.fr

• Mots sur
$$\{0,1\}$$

1 1111111 11111111 11111111 11111101

1 1111111 11111111 11111111 11111110

0 000000 0000000 0000000 00000000

0 0000000 00000000 00000000 00000001

1111111 11111111 11111111 11111101

11111111 11111111 11111111

PF1

ullet Un choix parmi les $2^{32}!$ choix pour représenter les entiers de

$$-\mathbf{2^{31}}$$
 à $\mathbf{2^{31}}-\mathbf{1}$

• Mots sur $\{0,1\}$ base 10 1 0000000 00000000 00000000 00000000 -21474836481 0000000 00000000 00000000 00000001 -2147483647Complément à 2 1 1111111 11111111 11111111 11111101 1 1111111 11111111 11111111 11111110 1111 11111111 11111111 11111111 0 000000 0000000 0000000 00000000 0 000000 0000000 0000000 00000001 1111111 11111111 11111111 11111101 2147483645

0 1111111 1111 pierluigi.crescenzi@irif.fr

2147483646

2147483647

• Complément à 2 : comment coder

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1 1111111 11111111 11111111 11111110

1 1111111 11111111 11111111 11111111

0 1111111 11111111 11111111 11111110

0 1111111 11111111 11111111 11111111

pierluigi.crescenzi@irif.fr

base 10

-2147483648

-2147483647

-2147483646

2147483646

2147483647

Amphi #03

 $1\ 0000000\ 00000000\ 00000000\ 00000001$

 $0\ 0000000\ 00000000\ 00000000\ 00000001$ $0\ 0000000\ 00000000\ 00000000\ 00000010$

PF1

PF1

base 10

-2147483648

-2147483647

-2147483646

2147483646

2147483647

Amphi #03

Numération en machine

divisions

pierluigi.crescenzi@irif.fr

• Complément à 2 : comment coder

 $1\ 0000000\ 00000000\ 00000000\ 00000001$

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1 1111111 11111111 11111111 11111110

1 1111111 11111111 11111111 11111111

0 0000000 00000000 00000000 00000000

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

0 0000000 00000000 00000000 00000010

0 1111111 11111111 11111111 11111111

divisions

pierluigi.crescenzi@irif.fr

• Complément à 2 : comment coder

■ Nous utilisons
$$b + (1 - b) = 1$$
 et $\overbrace{11 \cdots 1}^{b} + 1 = 100 \cdots 0$

 $1\ 0000000\ 00000000\ 00000000\ 00000001$

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1 1111111 11111111 11111111 11111110

0 000000 0000000 0000000 00000000

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

 $0\ 0000000\ 00000000\ 00000000\ 00000010$

11 11111111 11111111 11111110

11 11111111 11111111 11111111

1111111 11111111 11111111 11111111

■ Nous utilisons
$$b + (1 - b) = 1$$
 et

lisons
$$b \perp (1 - b) - 1$$
 et 1

PF1

$$+1 = 100$$

$$00\cdots0$$
 base 10

-2147483646

2147483646

2147483647

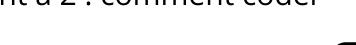
Amphi #03

$$-2147483648 \\ -2147483647$$

divisions

pierluigi.crescenzi@irif.fr

- Complément à 2 : comment coder





- Nous utilisons b + (1 b) = 1 et $11 \cdot \cdot \cdot \cdot 1 + 1 = 100 \cdot \cdot \cdot \cdot 0$

 $1\ 0000000\ 00000000\ 00000000\ 00000001$

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1 1111111 11111111 11111111 11111110

0 000000 0000000 0000000 00000000

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

 $0\ 0000000\ 00000000\ 00000000\ 00000010$

11111 11111111 11111111 11111110

11 11111111 11111111 11111111

1111111 11111111 11111111 1111111

PF1

base 10

-2147483648

-2147483647

-2147483646

2147483646

2147483647

Amphi #03

1. coder valeur absolue

2. inverser tous les bits

3. ajouter 1

• Complément à 2 : comment coder



■ Nous utilisons
$$b + (1 - b) = 1$$
 et

■ Nous utilisons
$$b + (1 - b) = 1$$
 et

 $1\ 0000000\ 00000000\ 00000000\ 00000001$

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1 1111111 11111111 11111111 11111110

1111111 11111111 11111111 1111111

0 000000 0000000 0000000 0000000

Nous utilisons
$$b+(1-b)=1$$
 et $\widehat{\mathbb{1}}$

■ Nous utilisons
$$b + (1 - b) = 1$$
 et $\overbrace{11 \cdots 1} + 1 = 100 \cdots 0$

utilisons
$$b+(1-b)=1$$
 et $\widehat{1}$

lisons
$$b+(1-b)=1$$
 et $\widehat{1}$

$$(11\cdots 1+1)$$

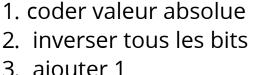
$$\cdot 1 + 1$$

 $0\ 0000000\ 00000000\ 00000000\ 00000001$ $0\ 0000000\ 00000000\ 00000000\ 00000010$



3. ajouter 1





base 10

-2147483648

-2147483647

-2147483646

2147483646

• Complément à 2 : comment coder



■ Nous utilisons
$$b + (1 - b) = 1$$
 et $11 \cdot \cdot \cdot \cdot 1 + 1 = 100 \cdot \cdot \cdot \cdot 0$

■ Nous utilisons
$$b + (1 - b) = 1$$
 et 1

 $1\ 0000000\ 00000000\ 00000000\ 00000010$

1111111 11111111 11111111 11111111

0 000000 0000000 0000000 00000000

 $0\ 0000000\ 00000000\ 00000000\ 00000001$

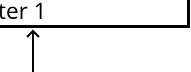
11111 11111111 11111111 11111110

• Nous utilisons
$$b+(1-b)=1$$
 et 1

s
$$b+(1-b)=1$$
 et $\widehat{1}$

2. inverser tous les bits 3. ajouter 1

1. coder valeur absolue



2147483646

base 10

-2147483648

-2147483647

-2147483646

PF1



2147483647

divisions

• Complément à 2 : comment coder



■ Nous utilisons
$$b + (1 - b) = 1$$
 et

• Nous utilisons
$$b+(1-b)=1$$
 et \hat{b}

■ Nous utilisons
$$b + (1 - b) = 1$$
 et $\overbrace{11 \cdots 1} + 1 = 1 \overbrace{00 \cdots 0}$

Nous utilisons
$$b+(1-b)=1$$
 et $\hat{1}$

1 0000000 00000000 00000000 00000010 ←

1 1111111 11111111 11111111 11111110

1111111 11111111 11111111 1111111

s utilisons
$$b+(1-b)=1$$
 et $\widehat{1}$

sons
$$b+(1-b)=1$$
 et $\widehat{1}$

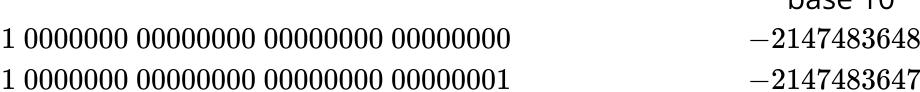
$$= \underbrace{11 \cdots 1}^{n} + 1$$

$$\mathsf{t}\, \widetilde{11\cdots 1} + 1 =$$

$$1\cdots 1+1=100 \cdot$$

base 10
$$-2147483648$$

3. ajouter 1



1. coder valeur absolue

2. inverser tous les bits

$$2147483646$$

- 0 000000 0000000 0000000 00000000 $0\ 0000000\ 00000000\ 00000000\ 00000001$ 0 0000000 00000000 00000000 00000010
- divisions 111 11111111 11111111 11111111 pierluigi.crescenzi@irif.fr

-2147483646

- Complément à 2 : comment coder



Nous utilisons
$$b+(1-b)=1$$
 et \hat{b}

- $1\ 0000000\ 00000000\ 00000000\ 00000001\ \leftarrow$
- $1\ 0000000\ 00000000\ 00000000\ 00000010$
- 1 1111111 11111111 11111111 11111110 1111111 11111111 11111111 11111111
- $0\ 0000000\ 00000000\ 00000000\ 00000001$ 0 0000000 00000000 00000000 00000010
- 0 000000 0000000 0000000 00000000
- - 1. coder valeur absolue
 - 2. inverser tous les bits
 - 3. ajouter 1

11111 11111111 11111111 11111110 21474836461111 11111111 11111111 11111111 2147483647

base 10

-2147483648

-2147483647

-2147483646

Comment decoder

Si bit de signe est 0

Si bit de signe est 1

3. Ajouter 1

pierluigi.crescenzi@irif.fr

Méthode d'Horner

1. Inverser tous les bits

2. Méthode de Horner

- Complément à 2 : comment coder
 - Nous utilisons b + (1 b) = 1 et $11 \cdot \cdot \cdot \cdot 1 + 1 = 100 \cdot \cdot \cdot \cdot 0$

1 11111111 11111111 11111111

PF1

Entiers en machine

- $1\ 0000000\ 00000000\ 00000000\ 00000001$

absolue

s les bits

2147483647

Amphi #03

base 10

-2147483648

-2147483647

-2147483646

- 2147483646

par 2

- Complément à 2 : opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?

- Complément à 2 : opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?
 - 16 = 00010000
 - 24 = 00011000
 - · -24 = 11101000

- Complément à 2 : opérations arithmétiques
 - Exemple : 16+24 = ? et 16 + (-24) = ?
 - 16 = 00010000
 - 24 = 00011000
 - -24 = 11101000

$$00010000 \\ 00010000$$

$$+ 00011000$$
 00101000

40

- Complément à 2 : opérations arithmétiques
 - Exemple : 16+24 = ? et 16 + (-24) = ?
 - 16 = 00010000
 - 24 = 00011000
 - · -24 = 11101000

$$00010000 \\ 00010000$$

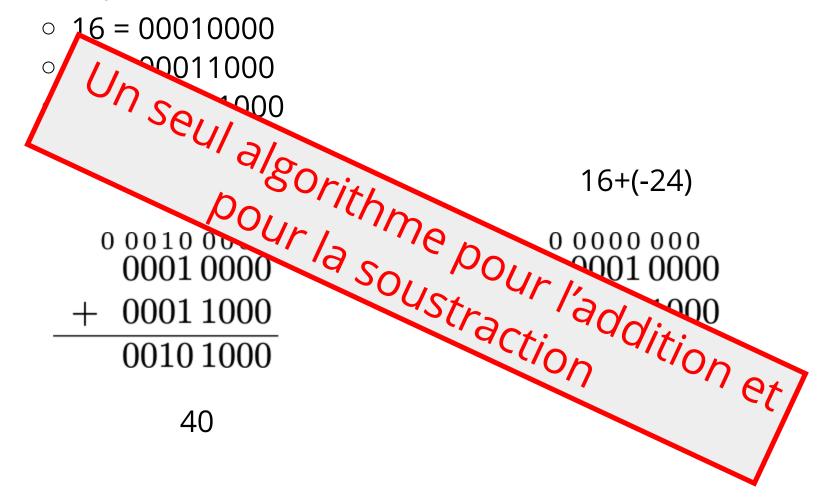
$$+ 00011000 \\ \hline 00101000$$

40

$$\frac{+\ 1110\,1000}{1111\,1000}$$

-8!

- Complément à 2 : opérations arithmétiques
 - Exemple: 16+24 = ? et 16 + (-24) = ?



- Complément à 2 et débordement
 - Retenue à gauche 1 n'implique pas nécessairement débordement

- Complément à 2 et débordement
 - Retenue à gauche 1 n'implique pas nécessairement débordement
 - Exemples (4 bits): 9-4, -9-4, -9+9
 - $\circ~9
 ightarrow 01001$ et -9
 ightarrow 10111
 - $\circ~4
 ightarrow 00100$ et -4
 ightarrow 11100

- Complément à 2 et débordement
 - Retenue à gauche 1 n'implique pas nécessairement débordement
 - Exemples (4 bits): 9-4, -9-4, -9+9
 - $\circ \ 9 \rightarrow 01001 \ \text{et} \ -9 \rightarrow 10111$
 - $\circ~4
 ightarrow 00100 ext{ et } -4
 ightarrow 11100$
 - 9-4
 - $\begin{smallmatrix}11&000\\0&1001\end{smallmatrix}$
 - + 11100

$0\,0101$

- retenue
- résultat positif (5)
- pas de débordement

- Complément à 2 et débordement
 - Retenue à gauche 1 n'implique pas nécessairement débordement
 - Exemples (4 bits): 9-4, -9-4, -9+9
 - $\circ~9 \rightarrow 01001 \text{ et } -9 \rightarrow 10111$
 - \circ 4 \rightarrow 00100 et $-4 \rightarrow$ 11100
 - 9-4
 - -9-4
 - $11\ 100$ 11 000
 - 10111 $0\,1001$
 - + 11100+ 111001001100101
- retenue
- résultat positif (5)
- pas de débordement
- retenue
- résultat négatif (-13)
- pas de débordement

- Complément à 2 et débordement
 - Retenue à gauche 1 n'implique pas nécessairement débordement
 - Exemples (4 bits): 9-4, -9-4, -9+9
 - $\circ~9
 ightarrow 01001$ et -9
 ightarrow 10111
 - $\circ~4
 ightarrow 00100$ et -4
 ightarrow 11100

 - 9-4
 - -9-4

 - 11 100 11 000
 - 10111 $0\,1001$
 - + 11100 $1\,1100$
 - 1001100101 retenue
- retenue résultat positif (5)
- pas de débordement
- résultat négatif (-13) • pas de débordement
- 00000
- retenue
- résultat positif (0)
- pas de débordement

-9+9

10111

 $11 \ 111$

+ 01001

Amphi #03

- Complément à 2 et débordement
 - Débordement
 - Si la somme de deux nombres positifs donne un nombre négatif
 - o Si la somme de deux nombres négatifs donne un nombre positif

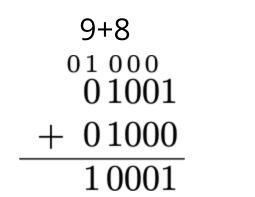
- Complément à 2 et débordement
 - Débordement
 - o Si la somme de deux nombres positifs donne un nombre négatif
 - Si la somme de deux nombres négatifs donne un nombre positif
 - Exemples (4 bits): 9+8, -9-8
 - $\circ 9 \to 01001 \text{ et } -9 \to 10111$

- Complément à 2 et débordement
 - Débordement
 - Si la somme de deux nombres positifs donne un nombre négatif
 - Si la somme de deux nombres négatifs donne un nombre positif
 - Exemples (4 bits): 9+8, -9-8
 - $\circ~9
 ightarrow 01001 ext{ et } -9
 ightarrow 10111$
 - $\circ~8
 ightarrow 01000 \ \text{et} -8
 ightarrow 11000$

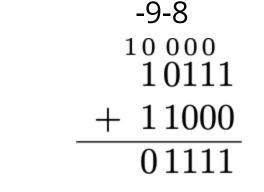
$$\begin{array}{r} 9+8 \\ 01\ 000 \\ 0\ 1001 \\ +\ 0\ 1000 \\ \hline 1\ 0001 \end{array}$$

- pas de retenue
- résultat négatif (-15)
- débordement

- Complément à 2 et débordement
 - Débordement
 - Si la somme de deux nombres positifs donne un nombre négatif
 - Si la somme de deux nombres négatifs donne un nombre positif
 - Exemples (4 bits): 9+8, -9-8
 - $\circ \ 9 \rightarrow 01001 \ \mathsf{et} \ -9 \rightarrow 10111$
 - $\circ~8
 ightarrow 01000$ et -8
 ightarrow 11000



- pas de retenue
- résultat négatif (-15)
- débordement



- retenue
- résultat positif (15)
- débordement

- Complément à 2 et débordement
 - Débordement
 - Si la somme de deux nombres positifs donne un nombre négatif
 - Si la somme de deux nombres négatifs donne un nombre positif
 - Exemples (4 bits): 9+8, -9-8
 - $\circ~9
 ightarrow 01001$ et -9
 ightarrow 10111
 - 0 8

Jamais un débordement si les deux nombres sont de signes différents

 $+ 01000 \over 10001$

 $\frac{+\ 11000}{01111}$

- pas de retenue
- résultat négatif (-15)
- débordement

- retenue
- résultat positif (15)
- débordement

- (Si) on manipule rarement les très très grands nombres
- (Si) on manipule rarement les nombres avec une grande précision
 - Choix de fixer la taille des représentations
 - Architectures communes 32 ou 64 bits :
 - Une infinité de choix pour représenter des réels
 - Problème : comment indiquer à la machine la position de la virgule ?
 - Deux méthodes
 - Virgule fixe : la position de la virgule est fixe
 - Virgule flottante : la position de la virgule change

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur p bits
- Un bit est utilisé pour le signe

- Partie entière est représentée sur *n* bits
- Partie fractionnelle sur *p* bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur *p* bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

0 000 00 base 10 0 000 00

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur *p* bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

	base 10
0 000 00	0
$0\ 000\ 01$	0,25

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur *p* bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

	base 10
$0\ 000\ 00$	0
$0\ 000\ 01$	0,25
0 000 10	0,5

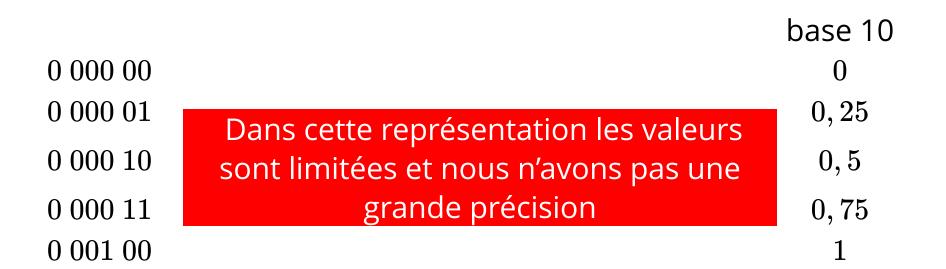
- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur p bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

	base 10
0 000 00	0
$0\ 000\ 01$	0,25
0 000 10	0,5
$0\ 000\ 11$	0,75

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur p bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2

	base 10
0 000 00	0
$0\ 000\ 01$	0,25
0 000 10	0,5
0 000 11	0,75
$0\ 001\ 00$	1

- lacktriangle Partie entière est représentée sur n bits
- Partie fractionnelle sur p bits
- Un bit est utilisé pour le signe
- Exemple : n = 3 et p = 2



• Représentation en virgule flottante

Chaque nombre réel peut s'écrire de la façon suivante :

$$n = \pm m imes b^e$$

- \blacksquare m: mantisse
- *b* : base
- e: exposant
- Exemple
 - \blacksquare $(13,11)_{10} = +1,311 \times 10^1 = +0,1311 \times 10^2 = +131,1 \times 10^{-1}$
 - $lacksquare (-110,101)_2 = -1,10101 imes 2^2 = -0,110101 imes 2^3 = 11010,1 imes 2^{-2}$