LISV Laboratoire d'ingénierie des systèmes de Versailles
UVSQ université PARIS-SACLAY

Architecture des systèmes numériques et informatiques
TD 1 - Les systèmes de numération et arithmétique binaire

Halim Djerroud

révision 2.1

Exercice 1 : Conversion entre bases

a) Conversion en Base 10

Donner la valeur décimale des entiers suivants, la base dans laquelle ces entiers sont codés étant précisée.

1. 1011011 et 101010 en binaire (base 2)
2. A1BE et C4F3 en hexadécimal (base 16)
3. 77210 et 31337 en octal (base 8)

b) Conversion vers la Base 10

Convertir de la base décimale vers la base indiquée.

1. 1548 et 1484 en hexadécimal.
2. 254, 255, 256 et 721 en octal.
3. 127, 312, 1024 et 2048 en binaire.

c) Conversion entre bases puissance de 2

1. Convertir le binaire suivant `101101010101011101011001` en base 4, 8 et 16.
2. Convertir l'hexadécimale suivant `0x4D3F2FFC4275E` en base 2, 4 et 8.
3. Convertir l'octal suivant `0124356734` en base 2, 4 et 16.

d) Conversions diverses (exercice complémentaire)

1. Coder l'entier 2 397 successivement en base 2, 8 et 16.
2. Donner la valeur décimale du nombre 10101, dans le cas où il est codé en base 2, 8 ou 16. Même question avec le nombre 6535 codé en base 8 ou 16.

Exercice 2 : Arithmétique en complément à 2

On considère des entiers signés, représentés en compléments à 2 sur n bits.

1. Rappelez le principe et l'intérêt de cette représentation. Quel est l'intervalle des nombres représentables ? Application numérique : n = 5 bits.
2. On considère dans la suite les nombres binaires suivants, codés en complément à 2 sur 5 bits : X1 = 01100, X2 = 01101, X3 = 10111, X4 = 10100. Donnez leur valeur décimale.
3. Effectuez en binaire les opérations suivantes, en utilisant le complément à 2 pour effectuer les soustractions : X2 – X1, X1 – X2, X1 + X3, X1 + X2, X3 + X4
4. Quelle est l'explication du résultat des deux dernières additions ? A-t-elle un rapport avec l'existence d'une retenue sortante ?

   Soient deux nombres a et b en complément à 2 sur n bits, et leur somme binaire $s = a + b$. On désigne par a_{n-1}, b_{n-1} et s_{n-1} les bits de poids fort de a, b et s. On considère la valeur booléenne suivante :

   $V = \bar{a}_{n-1} * \bar{b}_{n-1} * s_{n-1} + a_{n-1} * b_{n-1} * \bar{s}_{n-1}$

5. Calculez la valeur de V pour les opérations de la question 3. Quelle est la signification de V ?

Solution :

1) Principe, intérêt et intervalle

Le complément à 2 code les entiers signés sur n bits. Le bit de poids fort est le bit de signe (0 pour positif, 1 pour négatif).
Pour un mot $x = x_{n-1} \dots x_0$:

$$\text{val}(x) = -x_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} x_i \cdot 2^i.$$

Intérêt : l'addition/soustraction signées se fait avec le même additionneur binaire (la soustraction $a - b$ se calcule par $a +$ (complément à 2 de b)).
Intervalle représentable : $[-2^{n-1},\ 2^{n-1} - 1]$.
Pour $n = 5$: $\boxed{[-16,\ 15]}$.

2) Valeurs décimales (sur 5 bits)

Binaire	Interprétation	Décimal
$X_1 = 01100$	positif	12
$X_2 = 01101$	positif	13
$X_3 = 10111$	négatif : $10111 - 2^5 = 23 - 32$	-9
$X_4 = 10100$	négatif : $10100 - 2^5 = 20 - 32$	-12

(Remarque : sur n bits, une valeur v avec $v_{n-1} = 1$ vaut $v - 2^n$.)

3) Opérations en binaire (sur 5 bits)

On utilise le complément à 2 pour les soustractions.

$$X_2 - X_1 = 01101 + (\overline{01100} + 1) = 01101 + 10100 = 100001 \Rightarrow \boxed{00001}\ (= 1).$$

$$X_1 - X_2 = 01100 + (\overline{01101} + 1) = 01100 + 10011 = \boxed{11111}\ (= -1).$$

$$X_1 + X_3 = 01100 + 10111 = 100011 \Rightarrow \boxed{00011}\ (= 3).$$

$$X_1 + X_2 = 01100 + 01101 = \boxed{11001}\ (\text{mod } 32) = -7 \ \text{(dépasse 15} \Rightarrow \text{overflow)}.$$

$$X_3 + X_4 = 10111 + 10100 = \boxed{01011}\ (\text{mod } 32) = 11 \ \text{(dépasse } -16 \Rightarrow \text{overflow)}.$$

4) Explication des deux dernières additions

$X_1 + X_2$ additionne deux **positifs** $(12 + 13 = 25) > 15$: le résultat sort de l'intervalle ⇒ **overflow** et le mot 5 bits obtenu (11001) est interprété comme un négatif (-7).
$X_3 + X_4$ additionne deux **négatifs** $(-9 + (-12) = -21) < -16$: **overflow** et le mot 5 bits (01011) est interprété comme un positif $(+11)$.
Lien avec la retenue sortante : en complément à 2, la simple présence d'une retenue sortante n'indique **pas** l'overflow. Le critère correct est la différence entre la retenue entrant dans le bit de signe et la retenue sortante *ou*, de manière équivalente, la formule logique V ci-dessous.

5) Calcul et signification de V

On pose $s = a + b$ (addition binaire sur 5 bits) et

$$V = \overline{a_4}\ \overline{b_4}\ s_4 \ +\ a_4\ b_4\ \overline{s_4},$$

où a_4, b_4, s_4 sont les bits de signe (bit 4). $V = 1$ ssi l'addition de deux **positifs** donne un **négatif** (premier terme) ou l'addition de deux **négatifs** donne un **positif** (second terme), c.-à-d. **overflow de signe**.

Opération	a_4	b_4	s_4	V
$X_2 - X_1 \ = \ X_2 + (\text{cpl2}(X_1)) : \ 01101 + 10100 = 00001$	0	1	0	0
$X_1 - X_2 \ = \ X_1 + (\text{cpl2}(X_2)) : \ 01100 + 10011 = 11111$	0	1	1	0
$X_1 + X_3 : \ 01100 + 10111 = 00011$	0	1	0	0
$X_1 + X_2 : \ 01100 + 01101 = 11001$	0	0	1	1
$X_3 + X_4 : \ 10111 + 10100 = 01011$	1	1	0	1

Ainsi, $V = 1$ exactement pour les deux cas en overflow ($X_1 + X_2$ et $X_3 + X_4$).

Exercice complémentaire :

1. Effectuer en complément à 2 les opérations suivantes, les valeurs sont données en décimal. N'oubliez pas d'indiquer quelles opérations ont provoquées un débordement. L'application numérique est toujours sur 5 bits) :
   — 5+8
   — -5-8
   — 8+8
   — -8-8
   — 19-13
   — -14-1

Exercice 3 :

Combien d'entiers positifs peut-on coder en binaire sur un octet ? Combien de bits faut-il pour représenter 65 563 entiers différents.

Exercice 4 : (exercice complémentaire)

1. Indiquer la valeur codée par le mot de 16 bits 1101100101110101 suivant qu'il représente un entier non signé, ou un entier signé. Même question avec le mot 1001000011101101.
2. Indiquer la valeur codée par le mot de 16 bits 0x7F5D suivant qu'il représente un entier non signé, ou un entier signé. Même question avec le mot 0xABCD.

Exercice 5 :

Coder en IEEE 754 les nombres suivantes :
— 0,578425
Solution :

On veut coder $x = 0{,}578425$ au format IEEE 754 *simple précision* (32 bits).
1) Conversion en binaire (partie fractionnaire)

On multiplie la fraction par 2 et on relève la partie entière à chaque étape (méthode standard) :

Étape	$\times 2$	bit
1	$0{,}578425 \times 2 = 1{,}15685$	1
2	$0{,}15685 \times 2 = 0{,}3137$	0
3	$0{,}3137 \times 2 = 0{,}6274$	0
4	$0{,}6274 \times 2 = 1{,}2548$	1
5	$0{,}2548 \times 2 = 0{,}5096$	0
6	$0{,}5096 \times 2 = 1{,}0192$	1
7	$0{,}0192 \times 2 = 0{,}0384$	0
8	$0{,}0384 \times 2 = 0{,}0768$	0
9	$0{,}0768 \times 2 = 0{,}1536$	0
10	$0{,}1536 \times 2 = 0{,}3072$	0
11	$0{,}3072 \times 2 = 0{,}6144$	0
12	$0{,}6144 \times 2 = 1{,}2288$	1
$\vdots$	$\vdots$	$\vdots$

Ainsi, on obtient (en regroupant les bits successifs) :

$$0{,}578425_{10} \approx 0,\underbrace{10010100000100111010 1001\ldots}_{\text{bits de fraction}}{}_2$$

2) Normalisation

On écrit la forme scientifique binaire (un seul 1 avant la virgule) :

$$0{,}10010100000100111010 1001\ldots_2 = 1{,}00101000001001110101001\ldots_2 \times 2^{-1}$$

Donc l'exposant non biaisé vaut $e = -1$.

3) Signe

$x > 0 \Rightarrow$ bit de signe $s = 0$.

4) Exposant biaisé

Biais $= 127$. $\quad E = e + 127 = -1 + 127 = 126 = \texttt{01111110}_2$.

5) Mantisse (23 bits)

On enlève le 1. implicite et on garde 23 bits de fraction :

$$M = \texttt{00101000001001110101001}$$

(Le bit suivant est 0, donc pas d'arrondi.)

6) Mot IEEE 754 (32 bits)

$$\underbrace{0}_{s}\ \underbrace{\texttt{01111110}}_{E}\ \underbrace{\texttt{00101000001001110101001}}_{M}$$

En hexadécimal :

`0x3F1413A9`

Exercice complémentaire :

— 0,3215
— -10
— 1/3 ou 0,3333333...
— 40
— 13.625
— 0 **Solution :**

En IEEE 754 simple précision (32 bits), le zéro est un *cas particulier* : il ne peut pas être normalisé. On le représente avec un exposant nul et une mantisse nulle.

$$\underbrace{s}_{1\text{ bit}}\ \underbrace{E}_{8\text{ bits}}\ \underbrace{M}_{23\text{ bits}}$$

+0 :

$$\underbrace{0}_{s}\ \underbrace{00000000}_{E}\ \underbrace{00000000000000000000000}_{M}$$

–0 :

$$\underbrace{1}_{s}\ \underbrace{00000000}_{E}\ \underbrace{00000000000000000000000}_{M}$$

En hexadécimal :

$$+0 = \texttt{0x00000000}, \quad -0 = \texttt{0x80000000}.$$

— Nan

Solution :

En IEEE 754 simple précision (32 bits), un *NaN* ("Not a Number") est utilisé pour représenter une valeur indéfinie (par exemple $0/0$, $\sqrt{-1}$, $\infty - \infty$, etc.).

La structure est la suivante :

$$\underbrace{s}_{1\text{ bit}}\ \underbrace{E}_{8\text{ bits}}\ \underbrace{M}_{23\text{ bits}}$$

Les règles pour un NaN sont : - L'exposant est **tous les 1** : $E = \texttt{11111111}$. - La mantisse est **non nulle** : $M \neq 0$. - Le signe s peut être 0 ou 1 (il n'a pas de signification pour un NaN).

$$\underbrace{s}_{0\text{ ou }1}\ \underbrace{11111111}_{E}\ \underbrace{mantisse \neq 0}_{M}$$

On distingue généralement : - **qNaN** (quiet NaN) : mantisse dont le bit le plus à gauche est 1 (par ex. $M = 1000\ldots0$). - **sNaN** (signaling NaN) : mantisse dont le bit le plus à gauche est 0, mais où $M \neq 0$.

Exemple en hexadécimal (quiet NaN) :

`0x7FC00000`

— + ∞ et - ∞

Solution :

En IEEE 754 simple précision (32 bits), l'infini est un cas particulier. Il est utilisé pour représenter une valeur trop grande (dépassement) ou une division par zéro.

La structure est :

$$\underbrace{s}_{1\text{ bit}}\ \underbrace{E}_{8\text{ bits}}\ \underbrace{M}_{23\text{ bits}}$$

Les règles pour $\pm\infty$ sont : - L'exposant est **tous les 1** : $E = \texttt{11111111}$. - La mantisse est **tous les 0** : $M = \texttt{000...0}$. - Le signe s indique $+\infty$ ou $-\infty$.

$$+\infty : \quad \underbrace{0}_{s}\ \underbrace{11111111}_{E}\ \underbrace{00000000000000000000000}_{M}$$

$$-\infty : \quad \underbrace{1}_{s}\ \underbrace{11111111}_{E}\ \underbrace{00000000000000000000000}_{M}$$

En hexadécimal :

$$+\infty = \texttt{0x7F800000}, \quad -\infty = \texttt{0xFF800000}.$$

Exercice 6 :

Décoder les réels exprimés en norme IEEE 754 (32 bits)

— 0x41FE8000

exercice complémentaire :

— 0x3EA80000

— 0xC5E00000

— 0x00380000

Exercice 7 :

En utilisant le code ASCII, écrivez votre nom et votre prénom sans oublier de mettre les initiales en majuscule et terminer la chaîne de caractère par 0.

Exercice 8 : (exercice complémentaire)

En utilisant le code ASCII, décoder la phrase suivante : 76 39 97 114 99 104 105 44 32 99 39 101 115 116 32 102 97 99 105 108 101 46

Exercice 9 : (exercice complémentaire)

En utilisant le code ASCII, décoder la phrase suivante données en hexadécimal : 4A 27 41 49 20 54 52 4F 55 56 45 20 21