

Architecture des ordinateurs

Corrigé du TD 1: Arithmétique des ordinateurs et codage

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

(a) 1011011 et 101010 en binaire (base 2);

Correction : $1011011_2 = 91_{10}$, $101010_2 = 42_{10}$.

(b) A1BE et C4F3 en hexadécimal (base 16);

Correction : $A1BE_{16} = 41\ 406_{10}$, $C4F3_{16} = 50\ 419_{10}$.

(c) 77210 et 31337 en octal (base 8).

Correction : $77210_8 = 32\ 392_{10}$, $31337_8 = 13\ 023_{10}$.

2. Coder l'entier 2 397 successivement en base 2, 8 et 16.

Correction : $2\ 397_{10} = 100101011101_2 = 4535_8 = 95D_{16}$.

3. Donner la valeur décimale du nombre 10101, dans le cas où il est codé en base 2, 8 ou 16.

Correction : $10101_2 = 21_{10}$, $10101_8 = 4\ 161_{10}$, $10101_{16} = 65\ 793_{10}$.

Même question avec le nombre 6535 codé en base 8 ou 16.

Correction : $6535_8 = 3\ 421_{10}$, $6535_{16} = 25\ 909_{10}$.

4. Combien d'entiers positifs peut-on coder en binaire sur un octet ?

Correction : *Un octet contient 8 bits, on peut donc coder $2^8 = 256$ entiers.*

Combien de bits faut-il pour représenter 65 563 entiers différents en binaire ?

Correction : *Avec b bits, on peut coder 2^b entiers différents. Pour coder n entiers, il nous faut donc m bits tels que $2^{m-1} < n \leq 2^m$, c.-à-d. $m-1 < \log_2 n \leq m$. On a donc $m = \lceil \log_2 n \rceil$.*

Pour $n = 65\ 563$, on a $m = \lceil \log_2 65\ 563 \rceil = 17$.

5. Soit un ordinateur dont les mots mémoire sont composés de 32 bits. Cet ordinateur dispose de 4 Mo de mémoire. Un entier étant codé sur un mot, combien de mots cet ordinateur peut-il mémoriser simultanément ?

Correction : *4 Mo = 4×2^{20} octets, un mot est composé de 4 octets. Cet ordinateur peut donc mémoriser $\frac{4 \times 2^{20}}{4} = 2^{20} = 1\ 048\ 576$ mots*

Quelle est la plus grande valeur entière (décimale) que cet ordinateur peut mémoriser, cette valeur étant représentée par son codage binaire pur ? Donner un ordre de grandeur du nombre de chiffres en codage décimal.

Correction : *La mémoire contient 4×2^{20} octets, c.-à-d. $4 \times 2^{20} \times 8 = 33\ 554\ 432$ bits. La plus grande valeur entière que cet ordinateur peut mémoriser est donc $2^{33\ 554\ 432} - 1$.*

Le nombre de chiffres en décimal¹ est de $\lceil \log_{10} 2^{32 \times 2^{20}} \rceil \simeq 2^{20} \log_{10} 2^{32} \simeq 10^6 \times 3,2 \times \log_{10} 2^{10} \simeq 10^7$. Le nombre exact de chiffres en décimal est 10 100 891.

6. Coder en binaire sur un octet les entiers 105 et 21 puis effectuer l'addition binaire des entiers ainsi codés. Vérifier que le résultat sur un octet est correct. Même question avec les entiers 184 et 72.

¹ $v_{\max} = \text{base}^{\text{chiffres}} - 1 \Leftrightarrow \text{chiffres} = \log_{\text{base}}(v_{\max} + 1)$

Correction :

$$\begin{array}{r} 1101001 \quad (105) \\ + \quad 10101 \quad (21) \\ \hline 1111110 \quad (126) \end{array}$$

Ce résultat est correct.

$$\begin{array}{r} 10111000 \quad (184) \\ + \quad 1001000 \quad (72) \\ \hline (1)00000000 \quad (0) \end{array}$$

Ce résultat n'est pas correct (sur 8 bits).

7. Coder en binaire sur un octet les entiers 79 et 52 puis effectuer la multiplication binaire des entiers ainsi codés. Même question avec les entiers 135 et 46.

Correction :

$$\begin{array}{r} \quad \quad \quad 1001111 \quad (79) \\ \times \quad \quad 110100 \quad (52) \\ \hline \quad \quad 1001111 \\ \quad 1001111 \\ + \quad 1001111 \\ \hline 100000001100 \quad (4108) \end{array}$$

$$\begin{array}{r} \quad \quad \quad 10000111 \quad (135) \\ \times \quad \quad 101110 \quad (46) \\ \hline \quad \quad 10000111 \\ \quad 10000111 \\ \quad 10000111 \\ + \quad 10000111 \\ \hline 1100001000010 \quad (6\ 210) \end{array}$$

8. Indiquer la valeur décimale codée par le mot de 16 bits 1101100101110101 suivant qu'il représente un entier non signé, ou un entier signé selon le codage « bit de signe et valeur absolue » d'abord et complément à deux ensuite.

Correction : *En non signé, la valeur est $1101100101110101_2 = 55\ 669_{10}$. En signé, le premier bit (bit de signe) vaut 1, c'est donc un nombre négatif dont la valeur est $-101100101110101_2 = -22\ 901_{10}$.*

En complément à deux : $C_1(1101100101110101) = 0010011010001010$

donc $C_2(1101100101110101) = C_1(1101100101110101) + 1 = 0010011010001011$

et $0010011010001011_2 = 9867_{10}$

Même question avec le mot 1001000011101101.

Correction : *En non signé, la valeur est $1001000011101101_2 = 37\ 101_{10}$. En signé, c'est un nombre négatif dont la valeur est $-1000011101101_2 = -4\ 333_{10}$.*

$C_2(1001000011101101) = 0110111100010010 + 1 = 0110111100010011_2 = 28435_{10}$