Représentation des entiers

1 Entiers non signés

1.1 Entiers naturels

Exercice 1. Conversion en binaire.
Convertissez les nombres suivants du binaire vers le décimal.

1) 1111. 2) 100 1001. 3) 111 1011. 4) 1000 0000.

Correction.

$$\begin{aligned}
\overline{1111} &= (((1 \times 2 + 1) \times 2 + 1) \times 2) + 1 = 15\,,\\
\overline{100\ 1001} &= 2^6 + 2^3 + 2^0 = 64 + 8 + 1 = 73\,,\\
\overline{111\ 1011} &= 7 \times 16 + 11 = 123\,,\\
\overline{1000\ 0000} &= 2^7 = 128\,.
\end{aligned}$$

Convertissez les nombres suivants du décimal vers le binaire en effectuant des divisons euclidiennes successives.

5) 17. 6) 31. 7) 32. 8) 733.

Correction. On calcule

$$\begin{array}{r|l} 17 & 2 \\ \cline{2-2} 1 & 8 \end{array} \quad \begin{array}{r|l} 8 & 2 \\ \cline{2-2} 0 & 4 \end{array} \quad \begin{array}{r|l} 4 & 2 \\ \cline{2-2} 0 & 2 \end{array} \quad \begin{array}{r|l} 2 & 2 \\ \cline{2-2} 0 & 1 \end{array} \quad \begin{array}{r|l} 1 & 2 \\ \cline{2-2} 1 & 0 \end{array}$$

Ainsi, $17 = \overline{10001}$. On trouve de la même manière que $31 = \overline{11111}$ et $32 = 2^5 = \overline{100000}$. Enfin, pour écrire 733 en binaire, on calcule

$$\begin{array}{r|l} 733 & 2 \\ \cline{2-2} 1 & 366 \end{array} \quad \begin{array}{r|l} 366 & 2 \\ \cline{2-2} 0 & 183 \end{array} \quad \begin{array}{r|l} 183 & 2 \\ \cline{2-2} 1 & 91 \end{array} \quad \begin{array}{r|l} 91 & 2 \\ \cline{2-2} 1 & 45 \end{array} \quad \begin{array}{r|l} 45 & 2 \\ \cline{2-2} 1 & 22 \end{array}$$

$$\begin{array}{r|l} 22 & 2 \\ \cline{2-2} 0 & 11 \end{array} \quad \begin{array}{r|l} 11 & 2 \\ \cline{2-2} 1 & 5 \end{array} \quad \begin{array}{r|l} 5 & 2 \\ \cline{2-2} 1 & 2 \end{array} \quad \begin{array}{r|l} 2 & 2 \\ \cline{2-2} 0 & 1 \end{array} \quad \begin{array}{r|l} 1 & 2 \\ \cline{2-2} 1 & 0 \end{array}$$

Ainsi, $733 = \overline{10\,1101\,1101}$.

Exercice 2. Hexadécimal, même pas mal.
Convertissez les nombres suivant du binaire vers l'hexadécimal.

1) 1111 0000 0000 1101.
2) 1011 1010 1101 0100 0101 0101.
3) 1101 0011 1100 0000 1101 1110.

Correction.

$$\underbrace{1111}_{\texttt{F}}\,\underbrace{0000}_{\texttt{0}}\,\underbrace{0000}_{\texttt{0}}\,\underbrace{1101}_{\texttt{D}} = \texttt{F00D}\,, \qquad \underbrace{1011}_{\texttt{B}}\,\underbrace{1010}_{\texttt{A}}\,\underbrace{1101}_{\texttt{D}}\,\underbrace{0100}_{\texttt{4}}\,\underbrace{0101}_{\texttt{5}}\,\underbrace{0101}_{\texttt{5}} = \texttt{BAD455}\,,$$

$$\underbrace{1101}_{\texttt{D}}\,\underbrace{0011}_{\texttt{3}}\,\underbrace{1100}_{\texttt{C}}\,\underbrace{0000}_{\texttt{0}}\,\underbrace{1101}_{\texttt{D}}\,\underbrace{1110}_{\texttt{E}} = \texttt{D3C0DE}\,.$$

Convertissez les nombres suivants de l'hexadécimal vers le binaire.

4) FB1.

5) F10A7.

6) E1FFE165DABADEE.

Correction.

$$\underbrace{1111}_{\texttt{F}}\,\underbrace{1011}_{\texttt{B}}\,\underbrace{0001}_{\texttt{1}}\,, \qquad \underbrace{1111}_{\texttt{F}}\,\underbrace{0001}_{\texttt{1}}\,\underbrace{0000}_{\texttt{0}}\,\underbrace{1010}_{\texttt{A}}\,\underbrace{0111}_{\texttt{7}}\,,$$

$$\underbrace{1110}_{\texttt{E}}\,\underbrace{0001}_{\texttt{1}}\,\underbrace{1111}_{\texttt{F}}\,\underbrace{1111}_{\texttt{F}}\,\underbrace{1110}_{\texttt{E}}\,\underbrace{0001}_{\texttt{1}}\,\underbrace{0110}_{\texttt{6}}\,\underbrace{0101}_{\texttt{5}}\,\underbrace{1101}_{\texttt{D}}\,\underbrace{1010}_{\texttt{A}}\,\underbrace{1011}_{\texttt{B}}\,\underbrace{1010}_{\texttt{A}}\,\underbrace{1101}_{\texttt{D}}\,\underbrace{1110}_{\texttt{E}}\,\underbrace{1110}_{\texttt{E}}\,.$$

Exercice 3. Calcul en binaire.

Convertissez les nombres suivants en binaire puis effectuez les opérations demandées.

1) $43 + 38$ 2) $65 + 63$ 3) 43×4 4) 43×38.

Correction.

$$\begin{array}{rrl} & 10\,1011 & 43 \\ + & 10\,0110 & 38 \\ \hline & 101\,0001 & 81 \end{array} \qquad \begin{array}{rrl} & 10\,1011 & 43 \\ \times & 100 & 4 \\ \hline & 1010\,1100 & 172 \end{array} \qquad \begin{array}{rrl} & 10\,1011 & 43 \\ \times & 10\,0110 & 38 \\ \hline & 101\,011. & \\ & 1010\,11.. & \\ + & 101\,011.\;.... & \\ \hline & 110\,0110\,0010 & \end{array}$$

$$\begin{array}{rrl} & 100\,0001 & 65 \\ + & 11\,1111 & 63 \\ \hline & 1000\,0000 & 128 \end{array}$$

1.2 Entiers positifs sur n bits

Exercice 4. L'invasion des uns.

1) Montrez par récurrence que pour tout entier naturel $n \geq 1$, on a $\sum_{k=0}^{n-1} 2^k = 2^n - 1$.

Correction. On remarque que $\sum_{k=0}^{0} 2^k = 2^0 = 1 = 2^1 - 1$. La propriété est donc vérifiée au rang $n = 1$. Soit n un entier non nul. Supposons que $\sum_{k=0}^{n-1} 2^k = 2^n - 1$. On a

$$\sum_{k=0}^{n} 2^k = \left(\sum_{k=0}^{n-1} 2^k\right) + 2^n = (2^n - 1) + 2^n = 2 \times 2^n - 1 = 2^{n+1} - 1\,.$$

Le résultat est donc démontré par récurrence.

2) Déduisez-en la valeur de l'entier $a = \overline{11\ldots1}$ dont la décomposition binaire est composée de n bits.

Correction

Correction. En utilisant la définition de l'écriture en binaire ainsi que la question précédente, on a

$$\overline{1\dots1} = \sum_{k=0}^{n-1} = 2^n - 1\,.$$

3) Quel est l'ensemble des entiers naturels que l'on peut coder sur n bits ?

Correction. Le plus petit entier naturel est 0 et il s'écrit $\overline{0\dots0}$ sur n bits. Le plus grand entier dont la décomposition binaire contient au plus n bits est $\overline{1\dots1}$ et nous venons de voir qu'il est égal à $2^n - 1$. Aussi, l'ensemble des entiers sur n bits est $[\![0, 2^n[\![$.

Exercice 5. Entiers positifs sur 4 bits.

On considère l'ensemble E des entiers non signés représentables sur 4 bits. Listez les éléments de E en décimal, en binaire puis en hexadécimal.

Correction.

Entier	Binaire	Hexa
0	0000	`0x0`
1	0001	`0x1`
⋮	⋮	⋮
7	0111	`0x7`

Entier	Binaire	Hexa
8	1000	`0x8`
⋮	⋮	⋮
14	1110	`0xE`
15	1111	`0xF`

Exercice 6. unsigned int.

Soit n un entier naturel non nul et a un entier défini sur m bits.

1) Déterminez, en binaire, le reste de la division euclidienne de $a = \overline{a_{m-1}\dots a_1 a_0}$ par 2^n.

Correction. Si a est inférieur strictement à 2^n, alors le reste est a. Supposons maintenant que a est supérieur ou égal à 2^n, c'est-à-dire que $m > n$. On remarque que

$$\begin{aligned} a &= \overline{a_{m-1}\dots a_1 a_0} = \sum_{k=0}^{m-1} a_k 2^k = \left(\sum_{k=n}^{m-1} a_k 2^k\right) + \left(\sum_{k=0}^{n-1} a_k 2^k\right) = 2^n\left(\sum_{k=n}^{m-1} a_k 2^{k-n}\right) + \overbrace{\left(\sum_{k=0}^{n-1} a_k 2^k\right)}^{\leq 2^n-1} \\ &= 2^n \times \overline{a_{m-1}\dots a_n} + \overline{a_{n-1}\dots a_0}\,. \end{aligned}$$

Par conséquent, le reste dans la division euclidienne de a par 2^n est $\overline{a_{n-1}\dots a_0}$, ce qui revient à ne garder que les n derniers bits de a.

On considère maintenant deux entiers a et b codés sur n bits. On effectue la somme $a + b$ et le produit ab, que l'on enregistre dans des entiers s et p codés sur n bits. Autrement dit, on tronque $a + b$ et ab en ne gardant que les n derniers bits.

2) En utilisant les notions d'arithmétique, quelles relations pouvez-vous établir entre $a+b$, ab, s et p ?

Correction. D'après la question précédente, s est le reste dans la division euclidienne de $a + b$ par 2^n et p est le reste de ab. On en déduit que

$$a + b \equiv s \ \ [2^n]\,, \qquad\qquad ab \equiv p \ \ [2^n]\,.$$

3) Sous quelle condition a-t-on $a+b = s$ (ne cherchez pas trop compliqué) ? Même question pour $ab = p$.

Correction. Comme s et p sont des restes modulo 2^n, ils sont inférieurs strictement à 2^n. Par conséquent, $a + b = s$ si et seulement si $a + b < 2^n$. De même, $ab = p$ si et seulement si $ab < 2^n$.

1.3 Pratique

Exercice 7. Calculs sur un octet.
Considérons les entiers non signés représentables sur 8 bits. Effectuez les calculs suivants en binaire puis exprimez le résultat en hexadécimal. Vérifiez ensuite vos calculs sur machine en utilisant des `unsigned char`. Toutes les expressions suivantes sont exprimées en hexadécimal.

1) `73 + 8D`. 2) `AB + CD`. 3) `01 * 01`. 4) `37 * 7`.

Correction.

$$\begin{array}{rcl} & 0111\,0011 & 73 \\ + & 1000\,1101 & 8D \\ \hline & 0000\,0000 & 00 \end{array} \qquad \begin{array}{rcl} & 0000\,0001 & 01 \\ \times & 0000\,0001 & 01 \\ \hline & 0000\,0001 & 01 \end{array} \qquad \begin{array}{rcl} & 0011\,0111 & 37 \\ \times & 0000\,0111 & 07 \\ \hline & 0011\,0111 & \\ & 0110\,111. & \\ + & 1101\,11.. & \\ \hline & 1000\,0001 & 81 \end{array}$$

$$\begin{array}{rcl} & 1010\,1011 & AB \\ + & 1100\,1101 & CD \\ \hline & 0111\,1000 & 78 \end{array}$$

Exercice 8. Opérations binaires.
Considérons les entiers non signés représentables sur 8 bits. Effectuez les calculs suivants en binaire puis exprimez le résultat en hexadécimal. Vérifiez ensuite vos calculs sur machine en utilisant des `unsigned char`. Toutes les expressions suivantes sont exprimées en hexadécimal.

1) `~42` 3) `37 | 42` 5) `01 << 4` 7) `42 << 2`
2) `37 & 42` 4) `37 ^ 42` 6) `03 >> 2` 8) `3C >> 2`

Correction.
1) `~0x42 = ~01000010 = 10111101 = 0xBD`
2) `0x37 & 0x42 = 00110111 & 01000010 = 00000010 = 0x02`
3) `0x37 | 0x42 = 00110111 | 01000010 = 01110111 = 0x77`
4) `0x37 ^ 0x42 = 00110111 ^ 01000010 = 01110101 = 0x75`
5) `0x01 << 4 = 00000001 << 4 = 00010000 = 0x10`
6) `0x03 >> 2 = 00000011 >> 2 = 00000000 = 0x00`
7) `0x42 << 2 = 01000010 << 2 = 00001000 = 0x08`
8) `0x3C >> 2 = 00111100 >> 2 = 00001111 = 0x0F`

Exercice 9. Manipulations binaires.
Pour chacune des questions suivantes, vous devez utiliser les opérations binaires et/ou classiques pour obtenir le résultat souhaité en *une* instruction en langage C. Il est donc interdit d'avoir recours aux fonctions ou aux boucles.

1) Obtenir le nombre ayant uniquement le `n`-ième bit à 1.

Correction. `unsigned int x = 1 << n;`

2) Obtenir le nombre ayant uniquement le `n`-ième bit à 0.

Correction

Correction. `unsigned int x = ~(1 << n);`

3) Obtenir 2 à la puissance `n`.

Correction. `unsigned int x = 1 << n;`

4) Obtenir le nombre ayant les `n` bits de poids faible à 1 et tous les autres à 0.

Correction. `unsigned int x = (1 << n) - 1;`

5) Obtenir le quotient de la division d'un entier `a` par 2, 4, puis 2 à la puissance `n`.

Correction.
```
unsigned int x = a >> 1; // quotient par 2
unsigned int y = a >> 2; // quotient par 4
unsigned int z = a >> n; // quotient par 2^n
```

6) Obtenir le reste de la division d'un entier `a` par 2, 4, puis 2 à la puissance `n`.

Correction.
```
unsigned int x = a & 1; // reste par 2
unsigned int y = a & 3; // reste par 4
unsigned int z = a & ((1 << n) - 1); // reste par 2^n
```

2 Entiers signés

Exercice 10. Erreurs d'addition.

Selon les conditions de représentation des entiers signés présentées dans le cours (c'est à dire avec un bit de signe seulement), que valent les opérations suivantes en conservant la méthode de calcul de l'addition et de la soustraction ?

1) $21 + 73$; 2) $73 + (-21)$; 3) $73 + 73$; 4) $21 + (-73)$; 5) $73 + (-73)$;

Correction. On calcule tout d'abord la représentation naïve des entiers négatifs : `-21 = 10010101` et `-73 = 11001001`.

$$\begin{array}{rrr} & 0001\,0101 & 21 \\ + & 0100\,1001 & 73 \\ \hline & 0101\,1110 & 94 \end{array} \qquad \begin{array}{rrr} & 0100\,1001 & 73 \\ \times & 0100\,1001 & 73 \\ \hline & 1001\,0010 & -18 \end{array} \qquad \begin{array}{rrr} & 0001\,0101 & 21 \\ + & 1100\,1001 & -73 \\ \hline & 1101\,1110 & -94 \end{array}$$

$$\begin{array}{rrr} & 0100\,1001 & 73 \\ + & 1001\,0101 & -21 \\ \hline & 1101\,1110 & -94 \end{array} \qquad \begin{array}{rrr} & 0100\,1001 & 73 \\ + & 1100\,1001 & -73 \\ \hline & 0001\,0010 & 18 \end{array}$$

Exercice 11. Des petits entiers.

On considère l'ensemble E entiers signés représentables sur 4 bits. Listez les éléments de E en décimal, en binaire puis en hexadécimal.

Correction

Correction.

Entier	Binaire	Hexa
0	0000	`0x0`
1	0001	`0x1`
⋮	⋮	⋮
6	0110	`0x6`
7	0111	`0x7`

Entier	Binaire	Hexa
-8	1000	`0x8`
-7	1001	`0x9`
⋮	⋮	⋮
-2	1110	`0xE`
-1	1111	`0xF`

Exercice 12. Opposés.

Considérons les entiers signés représentables sur 8 bits. Calculez les opposés des nombres suivants exprimés en hexadécimal. Vérifiez ensuite vos calculs sur machine.

1) `01` 2) `FF` 3) `42` 4) `80`

Exercice 13. Formule de l'opposé.

Soit n un entier naturel non nul. Considérons un entier a dans $]\!\!]-2^{n-1}, 2^{n-1}[\![$ et notons c son codage sur n bits.

1) Montrez que dans $\mathbb{N}$, on a $c + (\sim c) = \overline{1\dots1}$.

Correction. Notons $a = \sum_{i=0}^{n-1} a_i 2^i$. L'entier correspondant au codage $\sim c$ est donc $\sim a = \sum_{i=0}^{n-1}(1 - a_i)2^i$. Ainsi

$$a + \sim a = \left(\sum_{i=0}^{n-1} a_i 2^i\right) + \left(\sum_{i=0}^{n-1}(1 - a_i)2^i\right) = \sum_{i=0}^{n-1}(a_i + 1 - a_i)2^i = \sum_{i=0}^{n-1} 2^i = \overline{1\dots1}\,.$$

2) Déduisez-en, toujours dans $\mathbb{N}$ la valeur exacte de $c + (\sim c) + 1$.

Correction. Comme $\overline{1\dots1} = 2^n - 1$. On en déduit que $c + (\sim c) + 1$ représente le codage binaire de l'entier 2^n.

3) Montrez que $(\sim c) + 1 \equiv -c \;\; [2^n]$.

Correction. On a $c + (\sim c) + 1 = 2^n \equiv 0 \;\; [2^n]$. Donc $(\sim c) + 1 \equiv -c \;\; [2^n]$. Autrement dit, $(\sim c) + 1$ représente le codage binaire de l'opposé de c modulo 2^n.

Exercice 14. Entiers signés.

Considérons les entiers signés représentables sur 8 bits. Effectuez les calculs suivants en binaire puis exprimez le résultat en hexadécimal et en décimal (en faisant apparaitre le signe). Vérifiez ensuite vos calculs sur machine en utilisant des `unsigned char`. Toutes les expressions suivantes sont exprimées en décimal.

1) $43 - 38$ 2) $38 - 43$ 3) $(-2) \times 42$ 4) $(-4) \times (-4) \times (-8)$

Correction.

$$
\begin{array}{rl}
38 & \texttt{0010 0110} \\
\sim 38 & \underline{\texttt{1101 1001}} \\
-\,38 & \texttt{1101 1010}
\end{array}
\quad
\begin{array}{rl}
43 & \texttt{0010 1011} \\
\sim 43 & \underline{\texttt{1101 0100}} \\
-\,43 & \texttt{1101 0101}
\end{array}
\quad
\begin{array}{rl}
2 & \texttt{0000 0010} \\
\sim 2 & \underline{\texttt{1111 1101}} \\
-\,2 & \texttt{1111 1110}
\end{array}
\quad
\begin{array}{rl}
4 & \texttt{0000 0100} \\
\sim 4 & \underline{\texttt{1111 1011}} \\
-\,4 & \texttt{1111 1100}
\end{array}
\quad
\begin{array}{rl}
8 & \texttt{0000 1000} \\
\sim 8 & \underline{\texttt{1111 0111}} \\
-\,8 & \texttt{1111 1000}
\end{array}
$$

$$
\begin{array}{rl}
43 & \texttt{0010 1011} \\
-\,38 & \underline{\texttt{1101 1010}} \\
5 & \texttt{0000 0101}
\end{array}
\qquad
\begin{array}{rl}
38 & \texttt{0010 0110} \\
-\,43 & \underline{\texttt{1101 0101}} \\
-\,5 & \texttt{1111 1011}
\end{array}
\qquad
\begin{array}{rl}
(-2) & \texttt{1111 1110} \\
\times\,42 & \underline{\texttt{0010 1010}} \\
 & \texttt{1111 110.} \\
 & \texttt{1111 0...} \\
+ & \underline{\texttt{110.\ ....}} \\
-\,84 & \texttt{1010 1100}
\end{array}
$$

Le dernier calcul n'est volontairement pas corrigé. Pensez bien, lorsque vous obtenez un résultat négatif, à calculer son opposé pour retrouver sa valeur binaire.

Correction