- Représentation de l'info. en binaire -

shorturl.at/HINU6

1) Nombre entier:

$N_{10} = (a_{m-1} a_{m-2} \dots a_0)_2$; $a_i = 0$ ou 1.

Bits fort (HSB) : Most significant Bit

Bits faible (LSB) : least significant Bit

- Valeur décimal : $N_{10} = a_{m-1} \cdot 2^{m-1} + a_{m-2} \cdot 2^{m-2} + \dots + a_0$.

$= \sum_{i=0}^{m-1} a_i 2^i$

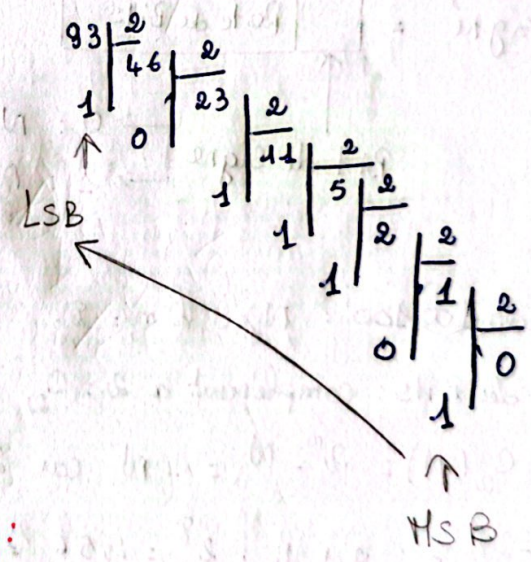
exp : $(1011101)_2 = 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^0$

$= (95)_{10}$.

- Conversion Décimal → Binaire:

$(N)_{10} = (?)_2$: division par 2 et on prend le reste comme Bit.

exp 1 : $(93)_{10} = (?)_2 = (1011101)_2$



exp 2 : $(147)_{10} = (?)_2$

$= 128 + 16 + 2 + 1$

$= 2^7 + 2^4 + 2 + 2^0$

$= (10010011)_2$.

- Valeur max d'un nbr binaire à m bits:

$(N_{max})_{10} = 2^m - 1$.

exp : $m = 4$ bits $\rightarrow (N_{max})_{10} = 2^4 - 1 = (15)_{10}$.

Base 8 : Octale

0 ⟶ 7

$8 = 2^3$ ⟶ 3 bits par chiffre.

exp : $(1101\ 0011\ 1001\ 1001)_2$

$(D\ 3\ 9\ 9)_{16}$

$(1\ 100\ 111\ 010)_2 = (1472)_8$

Base 16 : hexadécimal.

0 ⟶ 15 : 10 : A ; 13 : D
11 : B ; 14 : E
12 : C ; 15 : F.

$16 = 2^4$ ⟶ 4 bits par chiffre.

Remarque • $(AB3)_{16} = (?)_8$

$= (101010110011)_2$

$= (5\ 2\ 6\ 3)_8$

• $(517)_8 = (?)_{10} = 5 \times 8^2 + 8 + 7$

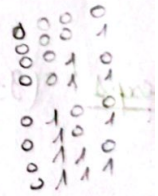
$= 320 + 15 = (335)_{10}$.

• $(1C5)_{16} = (?)_{10} = 1 \times 16^2 + 12 \times 16 + 5 \times 16^0$

$= 256 + 192 + 5$

$= (453)_{10}$.

Base 16 (mineur)



Représentation des nombres signés en Binaire :

N signé ⟷ $N \geq 0$ ou $N < 0$.

N signé = | Bit de signe | Reste de Bits |

Bit de signe → 0 : $N \geq 0$
→ 1 : $N < 0$.

$N = 00100100$: $N > 0$ si $m = 8$.

Reste des Bits : complément à 2 : C_2

$C_2(N) = 2^m - N = -N$ car $2^m = 0$ sur m bits.

Ex : sur 8 bits : $2^8 = 256 = (1\underbrace{00000000}_{0 \text{ sur } 8 \text{ bits}})_2$

calcul de $C_2(N)$: $C_2(N) = C_1(N) + 1 = -N$ avec $C_1(N)$: complément à 1 de N calculée en prenant l'inverse de chaque bits ($0 \to 1$, $1 \to 0$)

exp : sur 5 bits calculer $(-12)_{10} = (?)_2$.

$(12)_{10} = (?)_2 = (01100)_2$

Base 8 : Octale

0 ⟶ 7

$P = 2^3$ ⟶ 3 bits par chiffre.

exp : $(1101\ 0011\ 1001\ 1001)_2$

$(D\ 3\ 9\ 9)_{16}$

$(01\ 100\ 111\ 010)_2 = (1472)_8$

Base 16 : hexadécimal.

0 ⟶ 15 : 10 : A ; 13 : D
11 : B ; 14 : E
12 : C ; 15 : F.

$16 = 2^4$ ⟶ 4 bits par chiffre.

Remarque : $(AB3)_{16} = (?)_8$

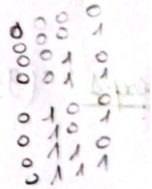
$= (101010110011)_2$

$= (5\ 2\ 6\ 3)_8$

- $(517)_8 = (?)_{10} = 5 \times 8^2 + 8 + 7$
  $= 320 + 15 = (335)_{10}$.

- $(1C5)_{16} = (?)_{10} = 1 \times 16^2 + 12 \times 16 + 5 \times 16^0$
  $= 256 + 192 + 5$
  $= (453)_{10}$.

Base 16 (mnémo)



2/ Représentation des nombres signés en Binaire :

N signé ⟷ $N \geq 0$ ou $N < 0$.

N signé = | Bit de signe | Reste de Bits |

Bit de signe : 0 : $N \geq 0$; 1 : $N < 0$.

$N = 00100100$: $N > 0$ si $m = 8$.

Reste des Bits : complément à 2 : C_2

$C_2(N) = 2^m - N = -N$ car $2^m = 0$ sur m bits.

Ex : sur 8 bits : $2^8 = 256 = (1\underbrace{00000000}_{0 \text{ sur 8 bits}})_2$

calcul de $C_2(N)$: $C_2(N) = C_1(N) + 1 = -N$ avec $C_1(N)$: complément à 1 de N calculée en prenant l'inverse de chaque bits ($0 \to 1$, $1 \to 0$)

exp : sur 5 bits calculer $(-12)_{10} = (?)_2$.

$(12)_{10} = (?)_2 = (01100)_2$

↑ S.S > 0

• Nombres spéciaux :

- $(0)_{10} \Rightarrow E = 0$ et $M = 0$
- Infini $(N/0) \Rightarrow E = 255$ (Simple) et $E = 2047$ (Double) et $M = 0$
- NaN = (not a number) 0/0 ; $E = 255$ ou $E = 2047$ et $M \neq 0$

2/ Arithmétique Binaire :

a/ Addition Binaire :

- Additionneur bit/bit en tenant compte de la retenue
- Cas général : addition de 2 nbrs à n bits chaque ⟼ résultat sur (n+1) bits au maximum.

Ex :

```
   1 0 0 1 1
 +   0 1 0 1
 -----------
   1 1 0 0 0

     1 1 1 1 1
   + 1 0 1 0 1
 -------------
   1 1 0 1 0 0
   ↑
 Carry
```

b/ soustraction :

- soustraction bit par bit :

$0 - 0 = 0$; $1 - 0 = 1$; $1 - 1 = 0$; $0 - 1 = 11$

Ex :

```
   1 0 1 0
 - 0 1 0 1
 ---------
   0 1 0 1

   1 1 0 1
 - 0 1 1 1
 ---------
   0 1 1 0
```

- Soustraction par C_2 :

$a - b = a + C_2(b) = a + C_1(b) + 1$.

Ex : sur 5 bits

$9 - 5 = 01001 - 00101 = 00100$

```
     1 1 0 1 0
   +         1
   -----------
     1 1 0 1 1
   + 0 1 0 0 1
   -----------
   (1) 0 0 1 0 0
```

bit signe → B.S

$(5)_{10} - (9)_{10} = (11100)_2 = (-4)$

$(-9)_{10} + (-9)_{10} = (\cancel{1}01110)_2$

Résultat incorrecte : on a un dépassement de capacité (overflow)

$(9)_{10} + (9)_{10} = 10010 < 0$ (B.S)

c/ Addition en hexa / soustraction :

- Addition chiffre par chiffre
- Retenue si le résultat $> (F)_{16} \Rightarrow$ Ajouter au chiffre suivant
- soustraction : Emprunte si $b > a$ lorsqu'on fait $a - b$.

```
   A 3 9
 + 1 A B
 -------
 (B E 4)16
```

$(20)_{10} = (14)_{16}$

```
    F 5 3 C
  + A 2 C 4
  ---------
 (1 9 8 0 0)16
```

```
   F 5 3 C
 - A 2 4 B
 ---------
   5 2 F 1
```

$(13)_{16} = 16 + 3 = 19$

Ex : $(0,75)_{10} = (?)_{float}$

$(0,75)_{10} = (0,11)_2 = (-1)^0 \times 1,1 \times 2^{-1}$

$\rightarrow S=0,\ M=1,\ e=-1.$

avec $E = -1+127 = 126 = (01111110)_2$

- $(0,75)_{10} = (0\ 01111110\ \underbrace{10\ldots0}_{23\text{ bits}})_{float}$

$= (3F\ 4\ 00000)_{16}$: float

- $(-2345,125)_{10} = (?)_{float}$

$= -(100100101001,001)_2 = (-1)^1 \times 1,00100100100100100 \times 2^{11}$

$e=11 \longrightarrow E = 11+127 = 138 = (1000\,1010)_2$

$(-2345,125)_{10} = (1\ \underbrace{1000\,1010}_{8\text{ bits}}\ \underbrace{00100100100100100\ldots0}_{23\text{ bits}})_{float}$

$= (C5\ 12\ 9200)_{16}$: float.

- **IEEE 754 : Double précision (Double)**

S: 1 bits	E : 11 bits	M : 52 bits

avec $E = e + 1023$: Exposant avec offset

$|P_{min}| \simeq 2,23 \times 10^{-308}$

$|P_{max}| \simeq 1,8 \times 10^{308}$.

Ex : $(0,75)_{10} = (?)_{Double}$

$(0,75)_{10} = (0,11)_2 = (-1)^0 \times 1,1 \times 2^{-1}$

$\rightarrow S=0,\ M=1$ et $e=-1$

$\rightarrow E = e + 1023 = 1022 = (01111111110)_2$.

$(0,75)_{10} = (0\ 01111111110\ \underbrace{10\ldots0}_{52\text{ bits}})$

$= 3FE\ 8\ 0000\,0000\ 0000)_{16}$: double.

d) Addition avec virgule flottante

$A = (-1)^{S_a} \times 1,M_a \times 2^{E_a}$

$B = (-1)^{S_b} \times 1,M_b \times 2^{E_b}$

→ $A+B = (-1)^{S_a} \cdot 1,M_a \times 2^{E_a} + (-1)^{S_b} \times 1,M_b \times 2^{E_b}$

opération effectuée par FPU (floating point Processor)

Étape de Calcul :

1) Aligner les mantisses (rendre même exposant)
2) Additionner les mantisse alignés (bit par bit)
3) Normaliser le résultat selon : $(-1)^S \times 1,M \times 2^e$.

*) $1,1 \times 2^2 + 1,001 \times 2^3$

$\Rightarrow 0,11 \times 2^3 + 1,001 \times 2^3$ ①

$= (0,11 + 1,001)2^3$

② $= 1,111 \times 2^3$

③

Multiplication avec virgule flottante :

$A \times B = (-1)^{S_a} \times 1,M_a \times 2^{E_a} \times (-1)^{S_b} \times 1,M_b \times 2^{E_b} = (-1)^{S_a + S_b} \times 1,M_a \times 1,M_b \times 2^{E_a + E_b}$

opération effectué par FPU.

Étape de Calcul

1) Calculer le signe de résultat (selon S_a et S_b)
2) Calculer $1,M_a \times 1,M_b$
3) Calculer $E_a + E_b$
4) Normaliser le résultat

Ex :

① $A = 1,1 \times 2^2$; $B = 1,001 \times 2^3$.

② $1,M_a \times 1,M_b = 1,1 \times 1,001$

$1,1 \times 1,001$

$1,001 \times 1,1$

1001

+ 1001

$1,M_a \times 1,M_b = 1,1011$

③ $E_a + E_b = 2 + 3 \Rightarrow 010 + 011 = 101 = (5$

④ $A \cdot B = 1,1011 \times 2^5$

5) Unité en binaire pour le stockage de l'info :

1 bit → 0 ou 1
8 bit → 1 octet = 1byte
2^{10} octets → 1024 octet = 1ko = 1 kB (kilo)
1ko x 1ko = 2^{10} octets x 2^{10} octets = 1 Mo = 1 MB (Méga)
1ko x 1ko x 1ko = $(2^{10} \times 2^{10} \times 2^{10})$ octets = 2^{30} octets = 1Go = 1GB (Géga)
1ko x 1ko x 1ko x 1ko = 2^{40} octets = 1 To = 1 TB (Téta)

Rq
$1\text{kBits} = \frac{1}{8}\text{ koctets}$

exp Liaison série de 9600 bits = $\frac{9600}{8}$ octets = 1200 octets/s.

6) Codage binaire :

a) Codage BCD :

BCD = Binary code Decimal : chaque chiffre est codé sur 4 bits.

exp :

$(0)_{10} = (0000)_{BCD}$
:
$(9)_{10} = (1001)_{BCD}$

$(10)_{10} = (0001\ 0000)_{BCD}$
$(15)_{10} = (0001\ 0101)_{BCD}$

$(3729)_{10}$ = $(0011\ 0111\ 0010\ 1001)_{BCD}$

b) Codage ASCII :

ASCII = American standard code for information Interchange.
utilise 8 bits et contient des codes pour representer :
- commande (surtout pour imprimante).

exp :
- (Form feed, Line feed ...) ≡ $(12)_{10}$, $(10)_{10}$
- chiffres — A = 65 ; B = 66 ; a = 97 ; b = 98.
- lettres — '1' = 49
- symboles — '{' = 123