Architecture des ordinateurs

Opérations en binaire

Addition en base 2

- Comme en décimal, sauf que $1 + 1 = 10$!
- Avec une retenue : $1 + 1 + 1 = 11$
- Table d'addition :

r_i	+	a_i	+	b_i	=	c_i
		0	+	0	=	0
		0	+	1	=	1
		1	+	0	=	1
		1	+	1	=	10
1	+	1	+	1	=	11

			r			
	1	0	0	1	1	0
+	0	1	0	1	0	0
=	1	1	1	0	1	0

r	r	r		r	r	r	
	1	0	1	0	1	0	1
+	1	1	1	0	1	1	1
=1	1	0	0	1	1	0	0

Le décalage à gauche et à droite

- Multiplication par 10 en base 10 : on ajoute un zéro
- Multiplication par 2 en base 2 : idem !
- Division : on enlève un chiffre à droite

	140_{10}	10001100_2
$\leftarrow$ **décalage à gauche**	$140\mathbf{0}_{10}$	$10001100\mathbf{0}_2$

	140_{10}	10001100_2
$\rightarrow$ **décalage à droite**	14_{10}	1000110_2

Soit un nombre $a_n \cdots a_0 = \sum_{0 \leq i \leq n} a_i b^i$, on a

$$\begin{aligned} b \times (a_n \cdots a_0) &= b \times \sum_{0 \leq i \leq n} a_i b^i \\ &= \sum_{0 \leq i \leq n} a_i b^{i+1} \\ &= a_n \cdots a_0 0 \end{aligned}$$

Multiplication sans table en base 10 (1/2)

- exemple : 27×43 ?
- on met 1 et 43 dans la première ligne d'une table, on additionne chaque ligne à elle-même, jusqu'à ce que la première case dépasse le multiplicateur :

1	43
2	86
4	172
8	344
16	688
32	...

- On ne conserve que les lignes qui nous servent à obtenir 27 :

Multiplication sans table en base 10 (1/2)

- exemple : 27×43 ?
- on met 1 et 43 dans la première ligne d'une table, on additionne chaque ligne à elle-même, jusqu'à ce que la première case dépasse le multiplicateur
- On ne conserve que les lignes qui nous servent à obtenir 27 :

$$
\begin{array}{crr}
 & 1 & 43 \\
+ & 2 & 86 \\
+ & \cancel{4} & \cancel{172} \\
+ & 8 & 344 \\
+ & 16 & 688 \\
\hline
= & & 1161
\end{array}
$$

Multiplication sans table en base 10 (2/2)

- Cette approche permet d'utiliser que des additions

$$\begin{aligned} 27 \times 43 &= (1 + 2 + 8 + 16) \times 43 \\ &= 1 \times 43 + 2 \times 43 + 8 \times 43 + 16 \times 43 \\ &= 43 + 86 + 344 + 688 \\ &= 1\,161 \end{aligned}$$

- Le choix des lignes est en quelque sorte une conversion en base 2...

$$\begin{aligned} 27_{10} &= (16 + 8 + 2 + 1)_{10} \\ &= 2^4 + 2^3 + 2^1 + 2^0 \\ &= 11011_2 \end{aligned}$$

Multiplication en binaire (1/2)

- Même approche qu'en base 10 sans table
- Le multiplicateur est déjà en base 2

Algorithme pour multiplier a et b :

```
r <- 0
tant que a est différent de 0
   si le bit de droite de a vaut 1
     r ← r + b
   décaler a à droite
   décaler b à gauche
```

- Testons cet algorithme sur $27_{10} \times 43_{10}$ ...

Multiplication en binaire (2/2)

- Table de multiplication :

a_i	$\times$	b_i	$=$	c_i
0	$\times$	0	$=$	0
0	$\times$	1	$=$	0
1	$\times$	0	$=$	0
1	$\times$	1	$=$	1

```
     101011
    x 11011
 ----------
     101011
    101011.
   000000..
  101011...
 101011....
 ----------
10010001001
```

Exercices

- Convertir 127_{10} et 93_{10} en binaire (en passant pas la base 8 + conversion rapide), les additionner dans cette représentation. Convertir le résultat en décimal pour vérifier qu'on n'a pas fait d'erreur.
- Multiplier 35_{10} et 211_{10} avec l'algorithme de multiplication sans table.
- Convertir 35_{10} et 211_{10} en binaire et faire tourner à la main l'algorithme de multiplication (convertir le résultat en décimal pour le vérifier).
- Poser la multiplication de 11001_2 et 101110_2