L'électricité, ça compte !

Gilles GEERAERTS Naïm QACHRI

Université Libre de Bruxelles
Département d'Informatique
http://www.ulb.ac.be/di

1 Introduction

Nous sommes aujourd'hui tous familiers avec les ordinateurs et avec leurs capacités de calcul phénoménales. Un ordinateur est capable de réaliser des additions, des multiplications, *etc*, tout comme un être humain. Mais si l'humain a besoin d'une feuille de papier et d'un crayon pour réaliser ces calculs, l'ordinateur, par contre, est composé uniquement de circuits électriques. Comment peut on, à l'aide de ces circuits, réaliser ces opérations mathématiques simples ? Comment peut on utiliser l'électricité pour compter ? C'est ce que cette série d'exercices va vous faire découvrir...

2 Logique des propositions

Dans la vie quotidienne nous communiquons en utilisant des phrases qui contiennent les mots « et », « ou », « vrai » et « faux ». Ces expressions permettent d'exprimer des concepts *logiques*.

Par exemple, la phrase :

Il pleut.

peut être soit vraie, soit fausse, selon qu'il pleuve ou non. De même pour la phrase :

Je prends mon parapluie.

De telles phrases sont appelées en logique des *propositions*.

On peut combiner les propositions pour former des phrases, à l'aide de mots, de connecteurs comme « et », « ou », « soit », « ou bien », *etc*. En logique, ces connecteurs sont appelés des *opérateurs*. Le sens des phrases dépend alors des propositions et des opérateurs utilisés. Par exemple :

Il pleut **et** je prends mon parapluie.

exprime qu'à la fois il pleut et que je prends mon parapluie. Les deux propositions « il pleut » et « je prends mon parapluie » doivent être vraies en même temps pour que la

phrase soit vraie. S'il ne pleut pas et que je prends mon parapluie, ou bien si je ne prends pas mon parapluie même s'il pleut, ou encore si je ne prends pas mon parapluie et qu'il ne pleut pas, cette phrase est fausse.

En plus des connecteurs, nous pouvons également *inverser* la valeur logique d'une phrase en utilisant une *négation*. Par exemple les phrases :

Je prends mon parapluie

et

Je **ne** prends **pas** mon parapluie

sont l'inverse l'un de l'autre : la première est vraie quand la second est fausse, et vice versa.

Exercice Indiquez dans quels cas les phrases suivantes sont vraies (considérez bien tous les cas possibles !) :

1. Il pleut **et** je prends mon parapluie.
2. Il ne pleut pas **ou** je **ne** prends **pas** mon parapluie.
3. Le chat est sur la clôture **et** le chien aboie **ou** il fait calme dehors.
4. Le chat est sur la clôture **et** le chien aboie **ou** il **ne** fait **pas** calme dehors.
5. Le chien aboie **ou** le chat est sur la clôture **et** la lune est pleine.

Comme on le voit, la langue française est parfois ambiguë... L'exercice suivant met aussi cela en lumière :

Exercice Dans un restaurant, le menu indique qu'on peut prendre « un fromage **ou** un dessert ». Dans quel cas cette phrase est elle vraie ? Peut on prendre le fromage et le dessert ?

Par ailleurs, on peut s'inscrire au Master en Informatique si on a fait un Bachelier en informatique **ou** un Bachelier en math. Dans quel cas cette phrase est elle vraie ? Peut on s'inscrire au Master si on est titulaire des deux diplômes ?

Comparez les réponses des deux derniers exercices... Cela nous montre qu'il faut clarifier les choses !

Pour rendre les choses plus claires, nous allons procéder par étapes. Tout d'abord, nous allons donner des noms simples aux propositions (par exemple des lettres : p, q) et nous allons « oublier » leur « vraie » signification. Par exemple si p est la proposition « il pleut » et q est la proposition « je prends mon parapluie », on remplace :

Il pleut **et** je prends mon parapluie.

par

p **et** q

De même, nous allons introduire un opérateur **non** pour exprimer la négation. Par exemple, si p est la proposition « il pleut », alors la proposition :

Il **ne** pleut **pas**

sera en fait représentée par :

non p

car elle est la négation de la proposition p. On fait ainsi clairement apparaître que p est vraie quand **non** p est fausse et vice versa. Prenez donc garde, dans la suite, à ne pas confondre **faux** et **non** : **faux** est une valeur logique, alors que **non** est un opérateur !

Ensuite, nous allons fixer clairement les signification des connecteurs **et**, **ou** et **non**. Pour ce faire, nous allons utiliser les *tables de vérité*. Une table de vérité est un tableau qui dit, pour chaque combinaison de valeurs des propositions, quelle est la valeur de la phrase.

Par exemple, dans la phrase p **ou** q, il y a deux propositions, et donc 4 combinaisons de valeurs :

1. p et q sont vraies.
2. p est vraie mais q est fausse.
3. p est fausse mais q est vraie.
4. p et q sont fausses.

On obtient alors la table de vérité suivante :

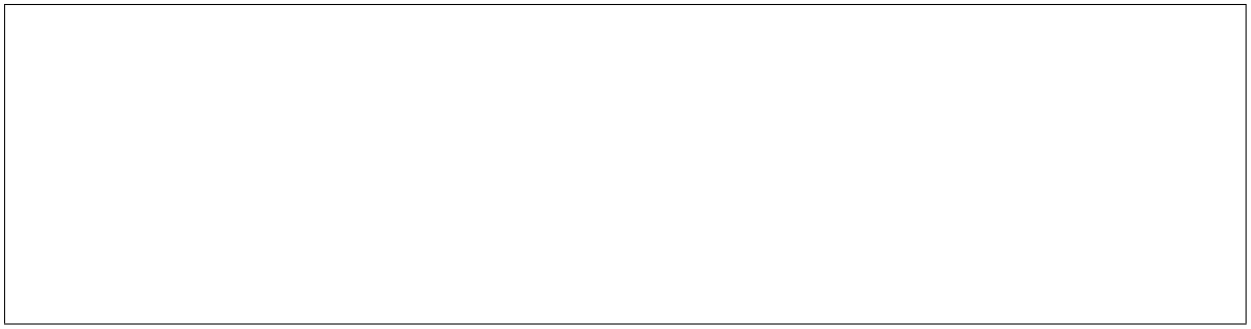
p	q	p **ou** q
vrai	**vrai**	**vrai**
vrai	**faux**	**vrai**
faux	**vrai**	**vrai**
faux	**faux**	**faux**

Exercice Donnez la table de vérité pour « p **et** q », ainsi que pour « **non** p ».

p	q	p **et** q
vrai	**vrai**	
vrai	**faux**	
faux	**vrai**	
faux	**faux**	

p	**non** p
vrai	
faux	

Exercice Si on a trois propositions p, q et r, quelles sont toutes les combinaisons de valeurs possibles ?



Enfin, nous devons expliquer comment gérer le fait qu'il y a plusieurs connecteurs dans une phrase. Cela revient, comme en mathématiques, à fixer une priorité. Par exemple, comment calcule-t-on la valeur de :

$$3 + 5 \times 8$$

Il est bien connu que $\times$ est plus prioritaire que $+$, et qu'il faut donc calculer d'abord $5 \times 8 = 40$, puis ajouter 3, et on obtient 43. Faire les choses dans le mauvais ordre donnerait $3 + 5 = 8$, puis $8 \times 8 = 64$, ce qui n'est pas la même valeur ! Autrement dit, si on devait ajouter des parenthèses, on écrirait :

$$3 + (5 \times 8)$$

et non pas :

$$(3 + 5) \times 8$$

Avec les connecteurs logiques, on décide que c'est le **non** qui est le plus prioritaire, puis le **et**, et enfin le **ou**. Par exemple :

$$p \textbf{ ou non } q \textbf{ et } r$$

doit se comprendre :

$$p \textbf{ ou } ((\textbf{non } q) \textbf{ et } r)$$

On calcule donc la valeur de **non** q, puis la valeur de (**non** q) **et** r, puis enfin la valeur finale. Cela peut se faire à nouveau à l'aide d'une table de vérité, dans laquelle on ajoute des colonnes pour se simplifier la vie. On commence par :

p	q	r	**non** q
vrai	**vrai**	**vrai**	**faux**
vrai	**vrai**	**faux**	**faux**
vrai	**faux**	**vrai**	**vrai**
vrai	**faux**	**faux**	**vrai**
faux	**vrai**	**vrai**	**faux**
faux	**vrai**	**faux**	**faux**
faux	**faux**	**vrai**	**vrai**
faux	**faux**	**faux**	**vrai**

Ensuite, on calcule (**non** q) **et** r, à partir des colonnes « r » et « **non** q ».

p	q	r	**non** q	(**non** q) **et** r
vrai	**vrai**	**vrai**	**faux**	**faux**
vrai	**vrai**	**faux**	**faux**	**faux**
vrai	**faux**	**vrai**	**vrai**	**vrai**
vrai	**faux**	**faux**	**vrai**	**faux**
faux	**vrai**	**vrai**	**faux**	**faux**
faux	**vrai**	**faux**	**faux**	**faux**
faux	**faux**	**vrai**	**vrai**	**vrai**
faux	**faux**	**faux**	**vrai**	**faux**

Finalement, on complète avec une colonne p **ou** ((**non** q) **et** r), calculée à partir des colonnes p et (**non** q) **et** r :

p	q	r	**non** q	(**non** q) **et** r	p **ou** ((**non** q) **et** r)
vrai	**vrai**	**vrai**	**faux**	**faux**	**vrai**
vrai	**vrai**	**faux**	**faux**	**faux**	**vrai**
vrai	**faux**	**vrai**	**vrai**	**vrai**	**vrai**
vrai	**faux**	**faux**	**vrai**	**faux**	**vrai**
faux	**vrai**	**vrai**	**faux**	**faux**	**faux**
faux	**vrai**	**faux**	**faux**	**faux**	**faux**
faux	**faux**	**vrai**	**vrai**	**vrai**	**vrai**
faux	**faux**	**faux**	**vrai**	**faux**	**faux**

Exercice Ajoutez les parenthèses aux expressions suivantes de manière à ce que la priorité soit correcte :

1. **non** p **ou non** q
2. p **ou** q **et** p **ou non** q
3. p **et** q **ou** p **et non** q **ou non** p **et** q **ou non** p **et non** q

Exercice Calculez les tables de vérité pour les expressions suivantes :

1. (**non** p) **ou** (**non** q)
2. **non** (p **et** q)
3. p **ou non** p

p	q	**non** p	**non** q	(**non** p) **ou** (**non** q)
vrai	**vrai**			
vrai	**faux**			
faux	**vrai**			
faux	**faux**			

p	q	p **et** q	**non** (p **et** q)
vrai	**vrai**		
vrai	**faux**		
faux	**vrai**		
faux	**faux**		

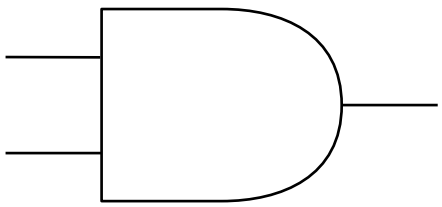
p	q	**non** p	p **ou** (**non** p)
vrai	**vrai**		
vrai	**faux**		
faux	**vrai**		
faux	**faux**		

3 Logique et électronique

3.1 Portes logiques

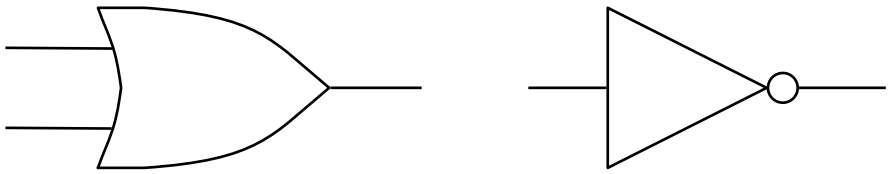
Jusqu'à présent nous avons manipulé des « phrases logiques » écrites en français. Il est également possible de représenter cela de façon graphique, à l'aide de *portes logiques*. Une porte logique est une forme qui représente soit un **et**, soit un **ou**, soit un **non**, et qui possède des « pattes » représentant ses entrées et ses sorties.

Par exemple, la formule p **et** q possède deux entrées : p et q ; et une sortie : la valeur calculée à l'aide du **et**. On représente l'opérateur **et** ainsi :

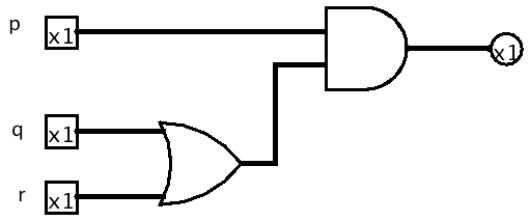


Les deux traits à gauche représentent les entrées (p et q, par exemple), et le trait à droite représente la sortie. Ainsi, si on fait « entrer » les valeurs **vrai** et **faux** sur les deux pattes d'entrée, c'est la valeur **faux** qui se retrouvera à la sortie.

Voici les portes logiques pour **ou** et **non** :



On combine ces portes logiques, en reliant les entrées et les sorties les unes aux autres. On identifie aussi clairement les entrées et les sorties du circuit. Par exemple, si on considère p **et** (q **ou** r), on a trois entrées p, q et r, et une seule sortie. On dessine alors quelque chose comme :



Remarquez que nous avons ici ajouté des carrés pour les entrées et un rond pour la sortie. C'est la convention que nous utiliserons par la suite.

Exercice Dessinez, à l'aide de portes logiques, les formules suivantes :

1. **non** p **ou non** q
2. p **ou** ((**non** q) **et** r)

3.2 Le passage de la logique à l'électronique

Vous avez vu les rudiments de la logique à la section précédente. Maintenant, nous allons essayer de comprendre quel est le rapport à l'électricité et arriver aux fondements de l'électronique numérique, qui est exploitée dans les ordinateurs.

Nous devons répondre essentiellement à deux questions :

1. Comment représenter les valeurs **vrai** et **faux** à l'aide d'électricité ?
2. Comment calculer les **et**, **ou** et **non** sur base de signaux électriques ?

Pour représenter le **vrai** et le **faux** à l'aide d'électricité, on choisit simplement une tension (un certain nombre de volts) qui permet de différencier entre ces deux valeurs. Par exemple, on supposera qu'une tension de plus de 5 volts représente **vrai** et qu'une tension de moins de 5 volts représente **faux**.

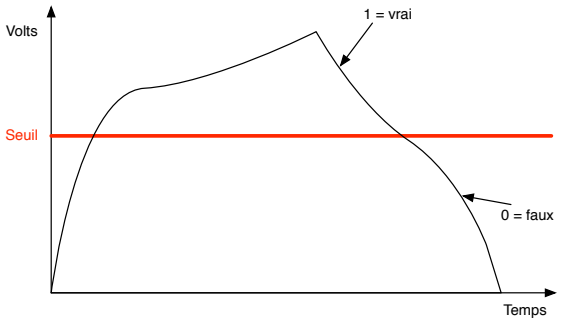


FIG. 1 – Exemple de représentation par l'électricité des notions de vrai et faux

On construit ensuite des circuits intégrés ou *puces* (voir Fig. 2) qui ont des « pattes ». Ces pattes correspondent à celles des portes logiques et sont donc, soit des entrées soit des sorties. On peut y connecter des fils électriques qui transporteront du courant. La puce interprétera le courant comme un **vrai** ou un **faux** sur ses pattes d'entrée et répondra par la valeur demandée sur ses pattes de sortie.

On peut donc entièrement réaliser un circuit numérique à l'aide de portes logiques... sans devoir sortir son fer à souder ! Naturellement, tout cela est fait à l'aide d'un logiciel

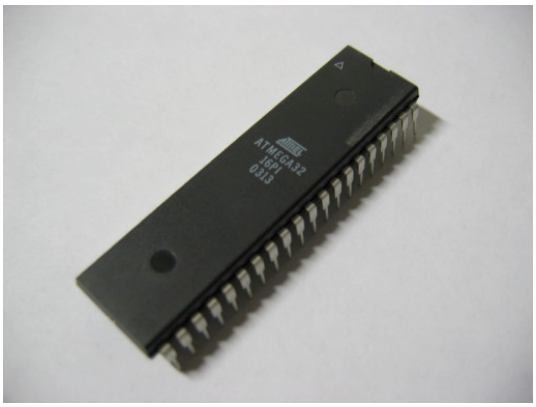


FIG. 2 – Une puce électronique qui peut calculer des **et**, des **ou** ...

qui permet de tester les circuits très facilement. C'est avec un de ces logiciels que vous allez concevoir votre premier circuit aujourd'hui.

3.3 L'outil `Logisim`

Nous allons à présent mettre en pratique sur les machines ce que vous venez d'apprendre. Vous avez devant vous un programme du nom de `Logisim`. Ce programme permet nous seulement de dessiner des portes logiques et de les connecter entre elles pour obtenir un circuit, mais il permet également d'observer le comportement de ce circuit. En effet, l'utilisateur peut décider de donner des valeurs (**vrai** ou **faux**, représentées par 1 et 0) aux entrées. `Logisim` calcule et affiche alors les valeurs des sorties, comme si on avait fait circuler du courant électrique dans le circuit.

Aperçu de l'interface du logiciel La Fig. 3 montre la fenêtre principale de `Logisim`. En 1, vous voyez la zone dans laquelle vous dessinerez votre circuit logique. En 2, vous retrouvez la barre d'outils qui possède tout ce dont vous avez besoin pour faire vos circuits et reprend, dans l'ordre, l'outil de sélection, l'outil d'édition, les fils de liaisons pour relier vos différents composants, l'outil d'annotation, la composante d'entrée, la composante de sortie, le **non**, le **et**, et le **ou**.

L'outil de sélection : Par défaut, toutes les entrées du circuit sont à 0 ou **faux**. L'outil de sélection permet, en cliquant sur une entrée, de change cette valeur. À chaque clic, la valeur change.

L'outil d'édition : il permet principalement de sélectionner les composants et de les déplacer dans l'espace de travail.

Les fils liaisons : les fils de liaisons permettent de relier les composants entre eux. Un exemple est donné à la Fig. 4 où lon a lié une entrée à un **non**, et la sortie de ce **non** à la

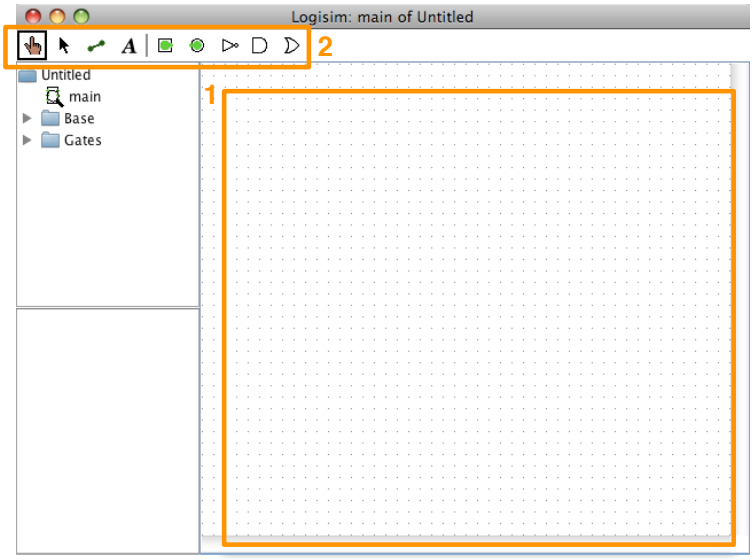


FIG. 3 – Un aperçu de la fenêtre de travail du logiciel *Logisim*

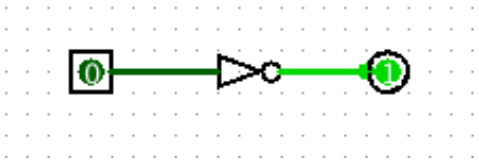


FIG. 4 – Exemple de liaisons.

sortie globale du circuit.

Les composantes d'entrée et de sortie : ils permettent de fournir les entrées logiques à notre circuit d'une part, mais également de récupérer la valeur de sortie produite par un circuit.

Les composants et, ou et non : ils représentent les opérations logiques correspondantes. Les composants **et** et **ou** prennent en entrée entre 2 et 5 entrées comme dans la figure suivante et une sortie. **non** ne prend évidemment qu'une seule entrée et sortie. C'est à vous de jouer avec les composantes d'entrées, de liaisons et de sorties pour effectuer les opérations logiques de votre choix.

Exercice Recopiez dans `Logisim` le deuxième circuit de l'exercice précédent. Choisissez des valeurs pour les entrées et vérifiez que le sortie est bien correcte (référez vous aux tables de vérité que vous avez faites en début de séance).

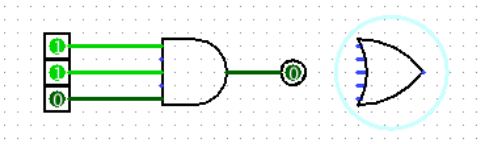


FIG. 5 – composant logique **et** et **ou**

Exercice Tout le monde a déjà joué à ces questionnaires de magazines qui se proposent, à l'aide de quelques questions bien ciblées, de déterminer la personnalité de celui qui répond. Si les questionnaires ne sont composés que de questions où vous avez le choix entre « oui » et « non », il devient intéressant de créer un petit circuit logique qui permet de calculer le résultat du quizz.

Voici un exemple de questionnaire :

- Question 1 : aimez vous les roses ?
  - oui
  - non
- Question 2 : avez vous apprécié le film « Titanic » ?
  - oui
  - non
- Question 3 : appréciez vous les soirées en tête à tête ?
  - oui
  - non

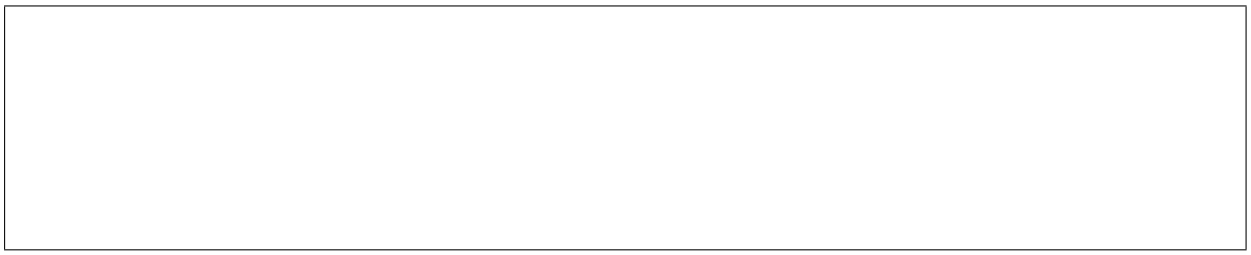
Les résultats sont les suivants : si vous avez répondu 0 ou 1 fois par « oui » à une question, alors vous êtes plutôt terre à terre. Si vous avez répondu deux fois « oui », alors vous êtes sentimental, mais sans abandonner votre réalisme. Si vous avez répondu « oui » à toutes les questions, vous êtes un grand romantique.

On vous demande de réaliser un circuit électronique qui possède trois entrées appelées r_1, r_2 et r_3, une pour chaque réponse. Les entrées seront mise à **vrai** si vous répondez oui, et à **faux** si vous répondez non à la question correspondante. Le circuit aura une seule sortie, qui sera à vrai si et seulement si la personne qui répond se classe dans la première catégorie (c'est-à-dire si elle a répondu 0 ou 1 fois « oui »). Attention, votre circuit doit pouvoir accepter n'importe quelles valeurs pour les entrées, et pas seulement les réponses que vous auriez données ! Une autre personne doit pouvoir utiliser le circuit après vous et obtenir une réponse correcte elle aussi.

Clairement, il n'y a aucune réponse « oui » si et seulement si les trois entrées sont à **faux**, c'est-à-dire si :

non e_1 **et non** e_2 **et non** e_3

vaut **vrai**. Mais dans quelle conditions a-t-on exactement une seule réponse « oui » ? Donnez la formule correspondante.



Réalisez ensuite le circuit dans `Logisim`

4 Changements de base

Nous avons vu que nous avons des outils puissants et simples pour manipuler les concepts logiques. Nous avons vu qu'il était simple de transposer ces manipulations au niveau électrique. Mais jusqu'à présent nous n'avons manipulé que des valeurs **vrai** et **faux.** Peut on se servir des mêmes idées pour manipuler d'autres types de valeurs, par exemple des nombres entiers ?

La réponse est oui ! mais il faut d'abord exprimer ces nombres de façon différente de ce dont nous avons l'habitude. C'est ici qu'intervient la notion de base.

Dans la vie courante, nous utilisons, sans le savoir, la base 10. Considérons, par exemple, le nombre 1245. Il est représenté par 4 symboles : 1, 2, 4 et 5. Tous ces symboles sont compris entre 0 et 9. Que signifie cette représentation ? Il est facile de voir que :

$$
\begin{gathered}
1245 \\
= \\
1000 + 200 + 40 + 5 \\
= \\
1 \times 1000 + 2 \times 100 + 4 \times 10 + 5 \times 1 \\
= \\
1 \times 10^3 + 2 \times 10^2 + 4 \times 10^1 + 5 \times 10^0
\end{gathered}
$$

Nous avons maintenant le « secret » de cette représentation, car on a réussi à ramener le nombre 1245 à des puissances de 10, qu'on appelle la *base*.

Que se passe-t-il si on considère une autre base, par exemple 5 ? Supposons que le nombre 231 est maintenant en base 5 et non plus en base 10. Pour faire la différence on écrira 231_5 pour bien faire comprendre la base utilisée. À quel nombre en base 10 le nombre 231_5 correspond il ?

$$
\begin{gathered}
231_5 \\
= \\
2 \times 5^2 + 3 \times 5^1 + 1 \times 5^0 \\
=
\end{gathered}
$$

$$2 \times 25 + 3 \times 5 + 1 \times 1$$
$$=$$
$$50 + 15 + 1$$
$$=$$
$$66_{10}$$

Il s'agissait en fait de 66 !

Exercice À quel nombre en base 10 correspond le nombre 1012_3 ?

4.1 Base 2

Dans les circuits électroniques, les nombres sont représentés en base 2 (on dit aussi « en binaire »). Pourquoi ? Tout simplement parce qu'en base 2, il n'y a que deux symboles : le 0 et le 1, et que ceux-ci peuvent facilement être représentés à l'aide des valeurs logiques **faux** et **vrai**. Chacun de ces symboles est appelé un *bit* (pour *binary digit*, soit chiffre binaire). Nous y reviendrons...

Regardons d'abord comment on peut convertir un nombre de la base 10 vers la base 2.

La méthode la plus simple consiste à diviser le nombre à convertir par deux, à noter le résultat et le reste, et à recommencer jusqu'à obtenir 0 comme résultat de la division. Nous devons à chaque étape récupérer le reste de la division, et c'est la suite de ces restes, lue « à l'envers » qui forme le nombre binaire.

Par exemple, si on considère le nombre 24_{10}, on procède ainsi pour le convertir en binaire (voir Fig. 6).

- On divise 24 par 2, on obtient 12, reste 0.
- On divise 12 par 2, on obtient 6, reste 0.
- On divise 6 par 2, on obtient 3, reste 0.
- On divise 3 par 2, on obtient 1, reste 1.
- On divise 1 par 2, on obtient 0, reste 1.

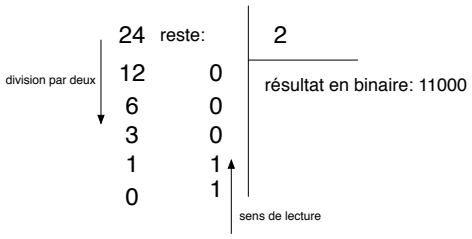


FIG. 6 – Exemple de conversion en binaire

Si on lit les reste à l'envers, on obtient : 11000, ce qui est bien la représentation de 24 en base 2 :

$$11000_2$$
$$=$$
$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$$
$$=$$
$$16 + 8 + 0 + 0 + 0$$
$$=$$
$$24_{10}$$

Exercice Convertissez les nombres 235_{10} et 562_{10} en binaire.

La manipulation des nombres en base 2 ne diffère pas de la manipulation de nombres en base 10. Une addition se fait de la même manière que vous l'avez vu dans le calcul

écrit. Les deux nombres à additionner sont écrits l'un au dessus de l'autre. On parcourt ensuite les colonnes de droit à gauche en additionnant les chiffres qui s'y trouvent, ainsi que l'éventuel report de la colonne précédente. Attention, maintenant le report se fait dès que la somme dépasse 1, et non pas 9 !

Voici un exemple :

```
report:   1  1
          011010
        + 010010
        --------
          101100
```

Exercice Convertissez 45_{10} et 23_{10} en base 2 et effectuez leur addition en base 2. Combien faut il de *bits* pour représenter le résultat ?

5 Binaire et logique

Récapitulons : nous avons un moyen de représenter des nombres entiers en base 2, nous pouvons manipuler des éléments binaires à l'aide de la logique et nous pouvons le faire

grâce à des circuits électriques. Nous avons donc tout ce qu'il nous faut pour manipuler des nombres avec des circuits électroniques.

Nous allons voir comment réaliser un circuit électronique qui réalise une somme. Pour simplifier, nous allons commencer par la somme de deux *bits* appelés b_1 et b_2. Ce seront les deux entrées de notre circuit. En sortie, nous devrons calculer deux valeurs : la somme s et le report r. La valeur s est ce que nous inscririons dans la colonne quand nous faisons la somme de b_1 et b_2, dans une addition euclidienne. Par exemple, si $b_1 = 1$ et $b_2 = 0$, la somme s vaut 1 et le report vaut 0. Si $b_1 = 1$ et $b_2 = 1$, la somme s vaut 0 et le report vaut 1.

Exercice Pour quelles valeurs de b_1 et b_2 la valeur s doit-elle valoir 1 ? et 0 ? N'oubliez pas que 1 correspond à **vrai** et 0 à **faux**.

b_1	b_2	s
vrai	**vrai**	
vrai	**faux**	
faux	**vrai**	
faux	**faux**	

Faites le même exercice pour le report r.

b_1	b_2	r
vrai	**vrai**	
vrai	**faux**	
faux	**vrai**	
faux	**faux**	

Exercice Écrivez la formule logique qui donne la valeur de s en fonction de b_1 et b_2. Faites la même chose pour le report r.

Exercice Réalisez, sous `Logisim`, un circuit qui prend deux entrées b_1 et b_2, et a deux sorties s et r, qui correspondent à la somme et au reste de $b_1 + b_2$.

Voilà ! Vous venez de réaliser un circuit électronique qui fait des additions !

6 Pour aller plus loin

Dans l'exercice précédent, nous avons réalisé un circuit qui additionne 2 bits. Quand on manipule des nombres, la situation est plus compliquée, car chaque nombre est composé de plusieurs bits. Reprenons un exemple :

$$\begin{array}{cccc} & 1 & 0 & 1 \\ + & 1 & 1 & 1 \\ \hline \end{array}$$

Pour calculer le bit qui doit aller dans la colonne de droite, c'est simple : $1 + 1 = 10$ (en binaire !), et donc on met un 0 dans la colonne de droite du résultat et on reporte le 1. On peut clairement effectuer ce calcul à l'aide du circuit que l'on a défini avant, car il s'agissait de sommer 2 bits :

$$\begin{array}{cccc} & & 1 & \\ & 1 & 0 & 1 \\ + & 1 & 1 & 1 \\ \hline & & & 0 \end{array}$$

Attaquons-nous maintenant à la seconde colonne. Contrairement à la colonne précédente, nous devons maintenant calculer la somme de 3 bits : les deux bits des nombres à additionner, et le report de la colonne précédente ! Malheureusement, le circuit simple que nous avons défini dans la section précédente ne permet pas cela : il ne reçoit que deux entrées.

De manière générale, on voit que pour pouvoir additionner des nombres en binaire, on va avoir besoin de circuits qui reçoivent trois entrées : b_r, b_1 et b_2 (b_r est le bit qui correspond au report de la colonne précédente) et poduit en sortie deux bits : s (la somme) et r (le report).

Exercice Pour quelles valeurs de b_1 et b_2 et b_r la valeur s doit-elle valoir 1 ? et 0 ?

b_1	b_2	b_r	s
vrai	**vrai**	**vrai**	
vrai	**vrai**	**faux**	
vrai	**faux**	**vrai**	
vrai	**faux**	**faux**	
faux	**vrai**	**vrai**	
faux	**vrai**	**faux**	
faux	**faux**	**vrai**	
faux	**faux**	**faux**	

Faites le même exercice pour le report r.

b_1	b_2	b_r	r
vrai	**vrai**	**vrai**	
vrai	**vrai**	**faux**	
vrai	**faux**	**vrai**	
vrai	**faux**	**faux**	
faux	**vrai**	**vrai**	
faux	**vrai**	**faux**	
faux	**faux**	**vrai**	
faux	**faux**	**faux**	

Exercice Écrivez la formule logique qui donne la valeur de s en fonction de b_1, b_2 et b_r. Faites la même chose pour le report r.

Exercice Réalisez, sous `Logisim`, un circuit qui prend trois entrées b_1, b_2 et b_r, et a deux sorties s et r, qui correspondent à la somme et au report de $b_1 + b_2 + b_r$.

Exercice Réalisez, sous `Logisim`, un circuit qui réalise la somme de deux nombres binaires sur 2 bits. Votre circuit aura donc en entrée quatre valeurs : $b_{1,1}$ et $b_{1,2}$, les 2 bits du premier nombre, ainsi que $b_{2,1}$ et $b_{2,2}$, les 2 bits du second nombre. Il y aura trois sorties : s_1, s_2 et s_3, les trois bits de la somme. Pensez à réutiliser ce qui a été fait dans l'exercice précédent !

7 Conclusion

Cet atelier vous a permis de réaliser, à l'aide d'un logiciel simple, des circuits électroniques qui permettent d'additionner des nombres représentés en binaire. Vous savez donc, du moins en partie, comment font les ordinateurs pour compter.

Néanmoins, il reste encore beaucoup à apprendre et à comprendre. Par exemple, vous pourriez réfléchir aux question suivantes :

1. Comment représenter et manipuler, en binaire, des nombres négatifs ?
2. Comment représenter et manipuler, en binaire, des nombres décimaux ?
3. Comment fait on une multiplication, une division ?
4. Comment peut on comparer une valeur à une autre ?

Des réponses à ce questions sont nécessaires pour bien comprendre comment fonctionnent les ordinateurs...

La démarche qui a été illustrée ici est typique de l'informatique : on commence par réfléchir à des problèmes simples, mais définis de façon claire, comme en mathématiques. On les analyse de façon rigoureuse, à l'aide d'outils empruntés eux aussi aux mathématiques (comme la logique), et on formule une solution conceptuelle. Ensuite, on passe à la mise en œuvre. Cette démarche est valable tant pour la réalisation de matériel (comme ici) que pour la réalisation de logiciel.

Si vous désirez continuer à expérimenter à l'aide de `Logisim`, n'hésitez pas à le télécharger à l'adresse suivante : `http://ozark.hendrix.edu/~burch/logisim/`.

Bon amusement et merci de votre visite !