Projet Bloc3

Groupe B3G3D

Thème : **Représentation binaire des entiers relatifs pour permettre à la machine d'effectuer des calculs.**

problématique : expliquer aux élèves comment un ordinateur effectue des opérations numériques

Objectifs :

Les élèves doivent savoir en fin de la séance:

- Coder un nombre relatif en binaire
- Soustraire ou additionner deux nombres relatifs
- comprendre l'addition ou la soustraction en binaire à l'aide des booléens (circuits logiques)

Représentation binaire d'un entier relatif	Évaluer le nombre de bits nécessaires à l'écriture en base 2 d'un entier, de la somme ou du produit de deux nombres entiers. Utiliser le complément à 2.	Il s'agit de décrire les tailles courantes des entiers (8, 16, 32 ou 64 bits). Il est possible d'évoquer la représentation des entiers de taille arbitraire de Python.
Valeurs booléennes : 0, 1. Opérateurs booléens : and, or, not. Expressions booléennes	Dresser la table d'une expression booléenne.	Le ou exclusif (xor) est évoqué. Quelques applications directes comme l'addition binaire sont présentées.
		Caractère séquentiel de certains opérateurs booléens

Prés-requis :

- La division euclidienne.
- Le codage en binaire d'un nombre entier naturel.
- Booléens and, or, NOT

Contenu de la séance

1) Recherche de codage possible pour les entiers négatifs compatible avec l'addition des entiers positifs
2) Principe et mécanisme du complément à 2
3) Exercices proposés aux élèves
4) les transistors
5) circuits séquentiels (additionneur)
6) Synthèse (Qu'ai-je voulu faire ? Qu'ont appris mes élèves ?

1) Comment coder les négatifs ?

A. Un codage simple mais peu satisfaisant

a. Par exemple avec 4 bits, on peut coder 16 nombres et le nombre décimal 6 s'écrit ainsi $(0110)2$.
Comment coder -6 ?

b. Réponse la plus probable : On peut décider d'utiliser le bit de poids fort et de convenir que si ce bit est 0, le nombre est positif et si ce bit est 1, le nombre est négatif.
Ainsi $(0110)_2$ correspondrait à 6 et $(1110)_2$ correspondrait à –6

c. Conséquences : la plage des positifs est réduite de moitié

d. Codage de zéro ?
il y a deux zéros. C'est embêtant.

e. Résultat de l'addition $-1+1$?

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

Addition posée
C'est embêtant aussi.

B. Recherche d'un codage compatible avec l'addition des naturels

a. L'idée de la retenue qui passe par-dessus bord :

on cherche le codage adapté pour que la somme $1+(a_3a_2a_1a_0)_2=0$

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

la découverte intuitive de $(a_3a_2a_1a_0)_2$ fait apparaitre la retenue qui tombe et le zéro, comme résultat de l'addition est en réalité le codage sur 4 bits de 2^4

b. La même idée amène à la recherche, pour un entier $b=(b_3b_2b_1b_0)_2$, de l'entier $a=(a_3a_2a_1a_0)_2$ tel que $a+b=0$ qui devient $a+b=2^4$

$$a+b=2^4 \Leftrightarrow a+\left(b_3 \times 2^3+b_3 \times 2^2+b_3 \times 2^1+b_3\right)=2^4 \Leftrightarrow a=2^4-\left(b_3 \times 2^3+b_3 \times 2^2+b_3 \times 2^1+b_3\right)$$

$$\Leftrightarrow a=\left(\left(1+2+2^2+2^3\right)+1\right)-\left(b_3 \times 2^3+b_3 \times 2^2+b_3 \times 2^1+b_3\right)=\left(1-b_3\right)2^3+\left(1-b_2\right)2^2+\left(1-b_1\right)2^1+\left(1-b_0\right)+1$$

Mécaniquement : $b_k \times \left(1-b_k\right)$: inversion de bits suivie de l'addition par 1

2) Cours

Pour coder un nombre négatif en binaire, on code sa valeur absolue (entier naturel), ensuite on inverse la valeur de chaque bit et on ajoute 1 (le complément à 2)

Exemple

On veut coder le nombre –14 sur 8 octets

La valeur absolue de (- 14)	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
On inverse chaque valeur	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**1**
On ajoute 1 (complément à 2)	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**

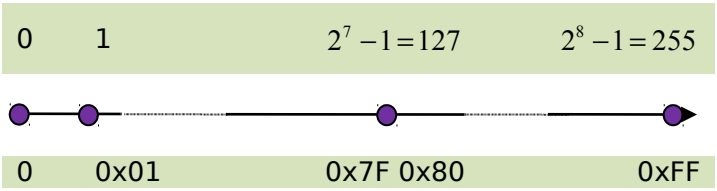
Donc – 14 se code **1 1 1 1 0 0 1 0**

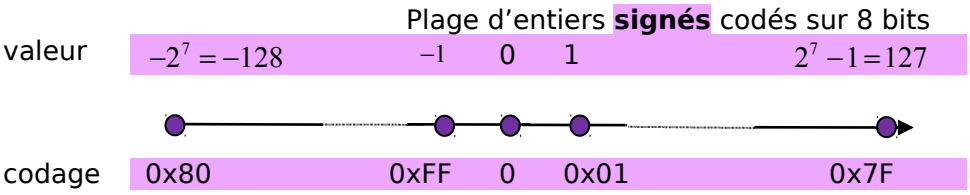
Vérification : On cherche à savoir si 14 + (- 14) = 0 (en binaire)

14	0	0	0	0	1	1	1	0
+								
-14	1	1	1	1	0	0	1	0
=	0	0	0	0	0	0	0	0

Le résultat est bien zéro, donc ce codage est correct. (en tenant compte des retenues)

RQ 1. valeur





Le plus grand entier signé codable est donc $2^7-1=127$ qui s'écrit $0x7F$ ou $(0111\,1111)_2$

Conséquence : les négatifs sont donc reconnaissables par le bit de poids fort

Exercices1 : (codage en binaire)

Calculer en binaire 13 – 10 sur 8 bits

Correction :

13	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**
+								
-10	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**0**
=								
3	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**

Exercice2 :

a) Coder - 2 en binaire sur 4 bits et vérifier le résultat obtenu.
b) Calculer en binaire (sur 4 bits) **5 – 3.**

Correction (indications)

a)

-2 :	**1**	**1**	**1**	**0**

b)

5 :	**0**	**1**	**0**	**1**
+				
-3 :	**1**	**1**	**0**	**1**
=	**0**	**0**	**1**	**0**

Le résultat en binaire correspond à 2 en décimal

Exercice3 : (décodage vers le décimal)

On travaille **sur 1 octet**.
On s'intéresse aux entiers **relatifs** codés en binaire sur 1 octet, entiers **dits signés**

Donner la valeur en décimal des entiers $(00101011)_2$ et $(11010001)_2$.