Architecture des ordinateurs

TD2 : arithmétique signée et non signée

Notions abordées (polycopié, chapitre III, pages 33 à 58)

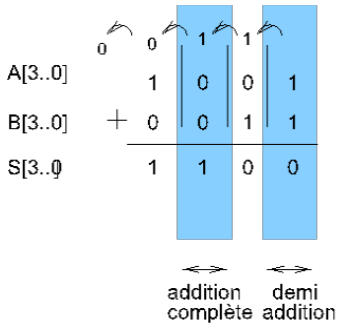
- codage des entiers naturels

$$s = \sum_{i=0}^{n-1} 2^i b_i$$

- codage des entiers relatifs en complément à 2

$$s = -2^{n-1} b_{n-1} + \sum_{i=0}^{n-2} 2^i b_i$$

- addition et interprétation signée et non signée



interprétation non signée		interprétation signée
233	11101001	-23
+ 66	+ 01000010	+ +66
299	1 00101011	+43
(débordement)		(pas de débordement)

  - la retenue indique le débordement dans une interprétation non-signée ; elle n'a pas de sens dans une interprétation signée
  - le débordement dans une addition signée se caractérise par des arguments de même signe et un résultat de signe opposé
- algorithme d'incrémentation : on inverse un bit si tous les bits qui sont à sa droite valent 1
- comparaison signée, non signée
- opérations d'extension de signe

1. Nombres entiers naturels et codage en binaire pur (non signé)

a. Quel est l'intervalle des valeurs en binaire pur sur 8 bits ? 10 bits ? 16 bits ?

b. Convertir en binaire pur les nombres décimaux suivants :

52, 102, 121, 195, 1023, 32768

c. Réaliser les additions en binaire pur sur 8 bits suivantes :

52 + 121 ; 102 + 195

Indiquer dans chaque cas s'il y a débordement.

2. Nombres entiers relatifs et codage en complément à 2

a. Quel est l'intervalle des valeurs en complément à 2 sur 8 bits ? 10 bits ? 16 bits ?
b. Convertir en complément à 2 sur 8 bits les nombres décimaux suivants :

   +52, +102, -121, +95, -1

c. Réaliser les additions en complément à 2 sur 8 bits suivantes :

   (+52) + (-121) ; (+102) + (+95)

Indiquer dans chaque cas s'il y a débordement.

3. Les bits de débordement C et V

Trouver 4 cas d'additions sur 8 bits qui correspondent aux 4 situations possibles pour C et V : C=0, V=0 ; C=1, V= 0 ; C=0 , V=1 ; C=1, V=1

4. Additionneur-soustracteur 32 bits

On suppose qu'on dispose d'un module additionneur 32 bits d'interface suivant :

```
module adder32(a[31..0], b[31..0] : s[31..0], C)
```

En réutilisant ce module adder32 , concevoir un module additionneur-soustracteur 32 bits ayant l'interface suivant :

```
module addsub32(a[31..0], b[31..0], sub : s[31..0], V, C)
```

Lorsque `sub` = 0 le module calcule la somme s[31..0] = a[31..0] + b[31..0] ; lorsque `sub` = 1, il calcule la différence. Les bits V et C ont le sens habituel des bits de débordement et de retenue-emprunt pour les opérations d'addition et de soustraction.

5. Comparateurs non signés

Concevoir modulairement des comparateurs non-signés d'interface :

```
module ucmp1(a, b : sup, eq)
module ucmp2(a[1..0], b[1..0] : sup, eq)
module ucmp4(a[3..0], b[3..0] : sup, eq)
```

`sup` indique que A est strictement supérieur à B, `eq` qu'ils sont égaux

6. Calcul de l'opposé

Le polycopié (p. 38) propose une seconde méthode pour calculer l'opposé arithmétique d'un nombre codé en complément à 2 : « le bit de poids faible est recopié tel quel ; les autres bits sont inversés seulement s'il existe un bit à 1 sur leur droite dans l'original. »

Ecrire les équations d'un circuit qui calcule l'opposé d'un nombre codé sur 4 bits.