

Codage binaire de l'information

- L'architecture actuelle des ordinateurs nécessite une représentation en binaire de toute information :
 - {0, 1}, {faux, vrai}, { éteint, allumé}, {noir, blanc}, . . .
- En Chine (premier millénaire av. JC), système binaire lié au {Yin, Yang} ou {actif, passif}
- Leibniz en 1703 : représentation des nombres en binaire.
- \bullet Boole (1815-1864) crée une algèbre avec 2 valeurs numériques : 0 et 1 C'est l'algèbre de Boole ou calcul booléen.

Codage des nombres entiers en machine?

- Représenter les entiers relatifs
- Les opérations arithmétiques +, -, x et / doivent être faciles à effectuer.
- Contrainte : architecture du matériel, la taille des mots mémoire est limitée : 16, 32, 64,. . . bits.
- Objectif : représenter l'information de la façon compacte Ne pas produire de dépassements de capacité (overflow)

3

Mots mémoire de taille suffisamment grande

Problème des lanceurs Ariane 4 et 5



Codage des entiers en binaire

- On va présenter quatre codages :
 - Codage binaire naturel signé
 - · Décimal codé binaire
 - Codage "complément à deux"
 - Codage "complément à un"

Sciences
Université de Paris

Codage binaire naturel signé

- On fixe la longueur des mots
- Le bit de poids fort est réservé au signe

Exemple: sur 4 bits, +6 est codé par 0110 -6 est codé par 1110

Avantages

5

- + Codage/décodage très facile
- + Représentation des entiers négatifs
- Inconvénients
- Il y a deux représentations de 0Les opérations arithmétiques
- ne sont pas faciles



Décimal codé binaire

Ancien système, binary coded decimal (BCD) chaque chiffre est codé en binaire sur 4 bits Exemple: L'entier positif 19032 010 = 0000; 110 = 0001; ...;

810 = 1000 ; 910 = 1001 codé : 0001 1001 0000 0011 0010

Avantages inconvénients

+ Codage/décodage facile - On gâche de l'espace : 6 combinaisons

+ Pas de limitation des sont non utilisées

- Opérations arithmétiques compliquées

Représentations utilisées dans des systèmes de bases de données (SGBD)

grandeurs représentées

Utilisées dans des systèmes basiques avec affichage digital

Complément à 2 : CA2

- On fixe le nombre d'octets pour stocker des nombres et faire des opérations : 1.2.3.4.... octets
- Quantité d'informations que l'on peut coder dans ce système

 - 8 bits (1 octet): On peut coder 2⁸ = 256 nombres
 16 bits (2 octets): On peut coder 2¹⁶ = 65536 nombres
- Codage des nombres positifs et négatifs :

 - 1 octet : 256 nombres → de -128 à +127
 2 octets : 512 nombres → de -32768 à +32767



Codage des entiers relatifs par complément à 2

- Les entiers positifs : séquence de bits commençant par 0 et écriture binaire signée
- Les entiers négatifs : séquence commençant par 1
 - -n est codé par le codage binaire de 2k-n
- Le 1^{er} bit est naturellement appelé "bit de signe"

00000111

0 1 1 1 1 1 1 1 → 127



Codage des entiers relatifs par complément à 2

• Les entiers négatifs : séquence commençant par 1

-n est codé par le codage binaire de 2k - n

Sur 8 bits $\begin{array}{l} -1: \{2^8 - 1\}_{10} = \{100000000\}_2 - 1 = \{111111111\}_2 \\ -2: \{2^8 - 2\}_{10} = \{111111110\}_2 \end{array}$

000000000000 $-127: (2^8 - 127)_{10} = (2^8 - 2^7 + 1) = (10000001)_2$

 $0.0000001 \rightarrow 1$

1 1 1 1 1 1 0 → -2 **1** 1 1 1 1 1 1 1 → -1



Codage des entiers relatifs par complément à 2

• Les entiers négatifs : séquence commençant par 1

-n est codé par le codage binaire de 2k-n

· Réciproquement sur 4 bits

si le code CA2 m = (s3s2s1s0)2 avec s3 = 1 m représente un nombre négatif m=-n

m est le code de $(2^4 - n)_{10} = (2^4 + nb)_{10} = (m)_{10}$

(nb)10 = (m)10 - 24



Complément à deux sur k bits

- * Sur k bits, on code les entiers de $\{-2^{k\cdot 1},...,2^{k\cdot 1}-1\}$
- Le complément à deux du complément à deux d'un entier $n \le 2^{k \cdot 1}$ est l'entier lui même $2^k - (2^k - n) = n$
- · Le complément à deux de zéro est zéro

11

 2^k - 0 = 2^k , restriction à la taille k bits fixée k=8 1 0000 0000

• Si l'entier n est codé sur k bits en complément à deux : $n = s_{k-1} s_{k-2} \dots s_3 s_2 s_1 s_0$

 $(n)_{10} = -s_{k-1}2^{k-1} + \sum_{i=0}^{k-2} s_i 2^{i}$

En complément à deux, le bit de signe s_{k-1} a comme poids -2^{k-1}



Algorithme de codage

Soit un nombre $m\in\mathbb{Z}\quad m\in\{\text{-}2^{k\text{-}1}\text{ , ... , }2^{k\text{-}1}-1\}\text{,}$

- * Si m $\in \{0, \, ... \, , \, 2^{k \cdot 1} \cdot 1\} \,$ positif, écriture binaire et on rajoute des 0 devant pour former k bits.
- Si m = -n avec n $\in \{0,\,...\,,\,2^{k\cdot 1}$ $1\}$, il faut "calculer" 2^k n en binaire

Remarque : [11111111111] = $\sum_{i=0}^{k-1} 2^i \ et \sum_{i=0}^{k-1} 2^i + 1 = 2^k$

n + x = [111111111]; x s'obtient en inversant les bits $n + x + 1 = 2^k$

Le code est 2^k - n Soit x + 1

- Ecrire n en binaire sur k bits
- Inverser bit à bit Ajouter 1 en binaire



Sciences Université de Pari

Sciences
Université de Pari

Avantages et inconvénients + Codage/décodage facile + Représentation unique de zéro + Opérations arithmétiques faciles - Taille mémoire fixée.

15

Qu'est-ce qu'un bon système d'addition?

• Facilement compatible avec les opérations, addition et soustraction

• Taille fixée une addition de nombres peut sortir de la portée

110 + 80 = 190, ou -110 - 80 = -190,
l'addition de deux nombres en CA2s peut sortir de l'intervalle {-127, ..., 128}

• Addition de deux nombres de signes opposés codés en CA2k est codable en CA2k

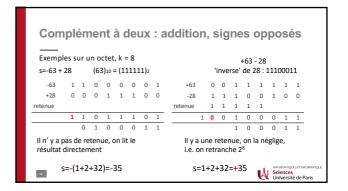
110 + (-80) = 30 et -110 + 80 = -30

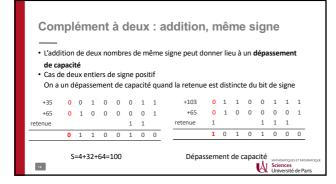
• Addition de deux nombres de même signe, le système doit

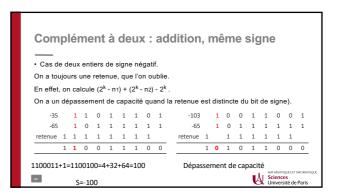
- donner le résultat si il est dans le bon intervalle

110 + 24 = 154; -110 - 24 = -154

- indiquer le problème







Résumé pour l'addition en "complément à deux"

- 2 nombres de signes opposés
 - Le résultat est représentable avec le nombre de bits fixés, pas de dépassement de capacité
 - S'il y a une retenue, on l'oublie
 - On lit directement le résultat codé en CA2
- 2 nombres de même signe
 - Il y a dépassement de capacité si la retenue est distincte du bit de signe
 - S'il y a une retenue on l'oublie si on a vérifié qu'il n'y avait pas overflow
 - On lit directement le résultat codé en CA2
- Ce codage est donc souvent choisi en pratique



Complément à 1 CA1_k

- Entiers positifs : commencent par 0 ⇒ code = écriture binaire
- Entiers négatifs : commencent par 1

pour les nombres négatifs Inversion bit à bit

 $011111111 \rightarrow 127$ $10000000 \rightarrow -127$

11111110 → -1

Sciences
Université de Paris

Complément à 1

- Avantages
- + Plus facile de coder/décoder les nombres négatifs : On inverse simplement bit à bit
 - -27 : \rightarrow 27 = (11011)₂ \rightarrow [11100100] est le codage de -27
- Inconvénients
 - Opérations arithmétiques plus compliquées
 - Détection de l'overflow plus compliquée
 - 2 manières de coder le "0", donc 2 tests pour voir si un nombre est nul
 - On code un nombre de moins : -127 \Rightarrow 127 au lieu de -128 \Rightarrow 127



Conclusion

22

- La représentation des nombres entiers est limitée par la taille du mot mémoire qui leur est affectée
- Le code le plus souvent utilisé est le code complément à deux. Il n'a qu'une seule représentation du zéro, les opérations arithmétiques et la détection de dépassement de capacité sont faciles à effectuer.
- Dans tous les cas et en fonction des architectures d'ordinateurs il y aura toujours des opérations dont le résultat n'est pas représentable.
 Sans précautions, elles engendrent des résultats aberrants ou empêchent la poursuite des calculs.

