Arithmétique binaire

10 novembre 2024

Objectif

Comprendre comment sont effectuées
les opérations entre deux codes binaires
de nombres non entiers

Addition

- Code en virgule fixe (n,p) de q code en ca2 du nombre $q \times 2^p$
- Addition en virgule fixe $\iff$ addition entière
- Propriété utilisée par la majorité des microcontrolleurs (pas de FPU = Floating Point Unit)
- Efficacité du calcul, mais précision médiocre au voisinage de 0

Multiplication

- Code en virgule fixe (n,p) de q code en ca2 du nombre $q \times 2^p$
- Multiplication A et B donne $A \times B \times 2^{2p}$ en code (2n, 2p)
- Propriété utilisée par la majorité des microcontrolleurs (pas de FPU = Floating Point Unit)
- Efficacité du calcul, mais précision médiocre au voisinage de 0

Addition

Addition de deux nombres codés en simple précision (par exemple)

1. **bit de signe** :$(m_a < m_b).b_{n-1} + (m_a \geq m_b).a_{n-1}$ (voir diapo suivante)
2. **Mantisse** : nombre en virgule fixe si A et B même exposant
   sinon réaligner la mantisse du plus petit nombre
   pour obtenir **l'égalité des exposants**
   $\Rightarrow$ Addition en virgule fixe : addition entière
3. Si besoin, réaligner la **mantisse** du résultat et corriger **l'exposant**

Démarche algorithmique de complexité abordable

$\Rightarrow$ Mais **non combinatoire**

Calcul du signe pour l'addition de flottants

Soient X et Y deux nombres positifs sur n bits (virgule fixe ou entier)

$X = Y \iff \forall i \in [n-1, 0], x_i = y_i$ d'où $(A = B) = \prod_{i=n-1}^{0} (a_i \odot b_i)$

De manière similaire
$X < Y \iff (\overline{x_{n-1}}.y_{n-1}) + (x_{n-1} \odot y_{n-1}).(\overline{x_{n-2}}.y_{n-2}) + \ldots + (x_{n-1} \odot y_{n-1}) \ldots (x_1 \odot y_1).(\overline{x_0}.y_0)$

Il est donc "simple" de comparer deux nombres positifs.

Ici, les valeurs absolues des mantisses **réalignées** doivent être comparées.

Table de vérite et la table de karnaugh du signe de $A + B$

$m_a \geq m_b$	a_{n-1}	b_{n-1}	c_{n-1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

	00	01	11	10
0	0	1	1	0
1	0	0	1	1

D'où $c_{n-1} = (m_a < m_b).b_{n-1} + (m_a \geq m_b).a_{n-1}$
$= (m_a < m_b).b_{n-1} + (m_a > m_b).a_{n-1} + (m_a = m_b)(a_{n-1}.b_{n-1})$

Exemple d'application

$A = 1,25$ et $B = 0,75$ le résutat $C = 2,0$ Les codes sont :

- pour A : s=0, E=127, M=1.010...0
- pour B : s=0, E=126, M=1.100...0
- pour C : s=0, E=128, M=1.000...0

Le calcul se déroule ainsi :

1. **Mantisse** : réaligner $B \to m_b = 0.11$ (exposant commun $= 127$)
   Addition des mantisses :

$$\begin{array}{cccl} & A & 1 & .010...0 \\ + & B & 0 & .110...0 \\ \hline = & C & 10 & .00...0 \end{array}$$

2. **bit de signe** : 0
3. Réalignement **mantisse** $= 1.0\ldots00$ et correction exposant 127+1 =128
   D'où le code [signe exposant mantisse] =[010000000 . . . 0]

Multiplication

La multiplication consiste :

1. **bit de signe** : $a_{n-1} \oplus b_{n-1}$
2. **Exposant :** addition des exposants (attention code biaisé)
3. **Mantisse** : multiplier les deux mantisses
4. si besoin réaligner la **mantisse** et modifier **exposant**

Multiplication de complexité similaire à l'addition en virgule flottante
S'appuie sur la multiplication en virgule fixe

$\Rightarrow$ Mais **non combinatoire**

Exemple d'application (simple précision)

$A = 1,25$ et $B = 0,75$. le résultat $C = 0,9375$.

- pour A : s=0, E_a=127, M=1.0100...0
- pour B : s=0, E_b=126, M=1.1000...0
- pour C : s=0, E_c=126, M=1.1110...0

1. $a > 0$ et $B > 0 \Rightarrow c_{31} = 0$
2. **Exposant** : $Ep(A) + Ep(B) - biais = 126$ (soit -1)
3. multiplication des mantisses $$\left\| \begin{array}{cccccc} & 1 & .0 & 1 & & \\ \times & 1 & .1 & 0 & & \\ \hline & & & 0 & 0 & 0 \\ + & & .1 & 0 & 1 & \\ + & 1 & .0 & 1 & & \\ \hline = & 1 & .1 & 1 & 1 & 0 \end{array} \right.$$
4. Résultat de la forme 1.*mantisse* $\Rightarrow$ pas de modification de l'exposant

D'où le code [signe exposant mantisse] =[0011111101110 . . . 0]

Bilan connaissances en codage / arithmétique

1. Codage binaire des nombres entiers relatifs
2. Codage virgule fixe et flottant des rationnels ($\in \mathbb{Q}$)
3. Addition binaire de code d'entiers relatifs (jusqu'à réalisation)
4. Addition binaire de code en virgule fixe (jusqu'à réalisation)
5. Multiplication binaire d'entiers naturels (jusqu'à réalisation)
6. Multiplication rationnels en vigule fixe (jusqu'à réalisation)
7. Algorithmique de :
   - Addition en virgule flottante
   - Multiplication rationnels relatifs en virgule fixe
   - Multiplication rationnels relatifs en virgule flottante