



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Departament d'Arquitectura de Computadors

Tema 5

Aritmètica d'enters i coma flotant

Estructura de Computadors (EC)

2023 - 2024 Q2

Adrià Armejach (adria.armejach@upc.edu)



Link - eina d'interès

<https://www.h-schmidt.net/FloatConverter/IEEE754.html>



Suma i resta en coma flotant



Suma en coma flotant - en base 10

- $z = x + y$, amb 4 dígits de precisió a la mantissa
 - $x = 9,999 \times 10^1$
 - $y = 1,680 \times 10^{-1}$

1. Igualar els exponents (al major dels dos)

- $y = 1,680 \times 10^{-1} = 0,01680 \times 10^1$

2. Sumar les mantisses

$$\begin{array}{r} 9,999 \\ + 0,01680 \\ \hline \end{array}$$

$$10,01580$$

$$z = 10,01580 \times 10^1$$

Suma en coma flotant - en base 10

- $z = x + y$, amb 4 dígits de precisió a la mantissa
 - $x = 9,999 \times 10^1$
 - $y = 1,680 \times 10^{-1}$

3. Normalitzar el resultat (comprobar overflow/underflow)

- $z = 10,01580 \times 10^1 = 1,001580 \times 10^2$
 - Si exponent dintre del rang, no hi ha excepció

4. Arrodonir mantissa a la precisió convinguda (4 dígits)

- $z = 1,001\textcolor{red}{580} \times 10^2 = 1,002 \times 10^2$

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

1. Decodificar signe, exponent i mantissa

$x = 0\ 01111110\ 100000000000000000001101$

$E = 126 - 127 = -1$

$x = +1,100000000000000000001101 \times 2^{-1}$

$y = 1\ 10000001\ 00000000000000000000100$

$E = 129 - 127 = 2$

$y = -1,00000000000000000000100 \times 2^2$

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

2. Igualar exponents (al major)

$$x = +1,10000000000000000000001101 \times 2^{-1}$$

$$y = -1,00000000000000000000000100 \times 2^2$$

$$x = +0,0011000000000000000000001101 \times 2^2$$

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

3. Sumar o restar les mantisses

Signe x	Signe y	Operació	Signe del resultat
+	+	Suma	+
-	-	Suma	-
+	-	Al de major magnitud se li resta el de menor magnitud	Signe del de major magnitud
-	+		

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

3. Sumar o restar les mantisses

$$x = +0,0011000000000000000000001101 \times 2^2$$

$$y = -1,000000000000000000000000100 \times 2^2$$

Signe diferent: S'han de restar les mantisses

y té major magnitud: $y - x$, **signe del resultat negatiu**

$$\begin{array}{r} |y| = 1,000000000000000000000000100 \\ - |x| = 0,0011000000000000000000001101 \\ \hline = |z| = 0,11010000000000000000000010011 \end{array}$$

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

4. Normalitzar la mantissa

$$|z| = 0,1101000000000000000000010011 \times 2^2$$

$$|z| = 1,101000000000000000000010011 \times 2^1$$

Si $E < E_{\min}$: underflow

Si $E > E_{\max}$: overflow

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

5. Arrodonir la mantissa

$$|z| = 1,1010000000000000000000100\textcolor{red}{11} \times 2^1$$

$$\begin{array}{r} 1,1010000000000000000000100 \\ + 0,0000000000000000000000001 \\ \hline 1,1010000000000000000000101 \end{array}$$

$$|z| = 1,1010000000000000000000101 \times 2^1$$

A vegades és necessari tornar a normalitzar i arrodonir la mantissa

Suma/resta en coma flotant - simple precisió

- Sumar $z = x + y$; on $x = 0x3F40000D$, $y = 0xC0800004$

6. Codificar signe, exponent i fracció

$$z = -1,1010000000000000000000101 \times 2^1$$

$$E = 1 + 127 = 128 = 10000000_2$$

$$z = 1 \ 10000000 \ 1010000000000000000000101$$

$$z = 0xC0500005$$

Calcular l'error de precisió de la suma

- Diferencia en valor absolut entre el valor exacte i el representat després de l'arrodoniment - $\varepsilon = |v - v_0|$

$$\begin{aligned}\varepsilon &= (1,10100000000000000000000101 \\ &\quad - 1,101000000000000000000000100\mathbf{11}) \times 2^1 \\ &= 0,00000000000000000000000001 \times 2^1 \\ &= 0,00000000000000000000000001 \\ &= 1,0 \times 2^{-24}\end{aligned}$$

Expressat en decimal,

$$\varepsilon = 6,0 \times 10^{-8}$$

Bits de guarda - simple precisió

- Els bits addicionals als propis de la mantissa s'anomenen **bits de guarda**
- Al igualar els exponents al major (pas 2)
 - Quantes posicions hem de desplaçar en el pitjor cas?
 - $E_{\min} = -126$, $E_{\max} = 127$
 - Més de 200 posicions a la dreta -> més de 200 bits de guarda!
 - Necessitaríem un sumador de més de 200 bits!
- Es pot aconseguir **el mateix resultat** amb només 3 bits de guarda

Bits de guarda - simple precisió

- Bits de guarda
 - **Guard (G)**: bit 24 de la mantissa
 - **Round (R)**: bit 25 de la mantissa
 - **Sticky (S)**: es forma substituint el tercer bit de guarda per la OR lògica d'aquest bit i tots els que té a la seva dreta


1,10100000110101010010110

0,0000000110100000110101010010110


GRS

0,00000001101000001101010101

- El resultat al operar amb 3 bits de guarda (GRS) és el mateix que si s'utilitzen bits infinits



Multiplicació i Divisió en coma flotant



Multiplicació i Divisió en coma flotant

Multiplicació	Divisió
1. Sumar els exponents	1. Restar els exponents
2. Multiplicar les mantisses	2. Dividir les mantisses
3. Normalitzar la mantissa del resultat	
4. Arrodonir la mantissa del resultat	
5. Assignar signe negatiu al resultat si els operands tenen signe diferent	

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

$x = 0 \ 01111110 \ 110000000000000000000000$

$E = 126 - 127 = -1$

$x = +1,110000000000000000000000 \times 2^{-1}$

$y = 1 \ 01111101 \ 101000000000000000000010$

$E = 125 - 127 = -2$

$y = -1,101000000000000000000010 \times 2^{-2}$

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

1. Sumar els exponents

$$E = -1 + (-2) = -3$$

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

2. Multiplicar les mantisses

$$\begin{array}{r} 1110 \\ \times 11010000000000000000000010 \\ \hline 1110 \\ 1110 \\ 1110 \\ 1110 \\ \hline 10110110000000000000000011100 \end{array}$$

- Resultat, amb la coma: $10,1101100000000000000000011100 \times 2^{-3}$

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

3. Normalitzar la mantissa

$$|z| = 10,110110000000000000000011100 \times 2^{-3}$$

$$|z| = 1,0110110000000000000000011100 \times 2^{-2}$$

Exponent dintre del rang, no hi ha overflow/underflow

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

4. Arrodonir la mantissa

$$\begin{array}{r} 1,0110110000000000000000011110 \times 2^{-2} \\ + 0,00000000000000000000000001 \\ \hline 1,0110110000000000000000010 \times 2^{-2} \end{array}$$

Multiplicació en coma flotant - precisió simple

- $z = x \times y$; on $x = 0x3F600000$, $y = 0xBED00002$

5. Codificar signe, exponent i mantissa

$$z = -1,011011000000000000000010 \times 2^{-2}$$

$$E = -2 + 127 = 125 = 01111101_2$$

$$z = 1 \quad 01111101 \quad 011011000000000000000010$$

$$z = 0xBEB60002$$

Coma flotant - no associativitat

- El format de coma flotant IEEE-754 NO és associatiu

$$x + (y + z) \neq (x + y) + z$$

- Exemple: $x = -1,1 \times 2^{127}$; $y = 1,1 \times 2^{127}$; $z = 1,0$
 - $x + (y + z)$
 - $-1,1 \times 2^{127} + (1,1 \times 2^{127} + 1,0) = -1,1 \times 2^{127} + 1,1 \times 2^{127} = 0,0$
 - $(x + y) + z$
 - $(-1,1 \times 2^{127} + 1,1 \times 2^{127}) + 1,0 = 0,0 + 1,0 = 1,0$
- S'introdueix error al aproximar el resultat (arrodoniment)



Coma flotant en MIPS



Coma flotant en MIPS

- Coprocessador CP1
 - La unitat de coma flotant és un coprocessador amb el seu propi banc de registres
 - Joc d'instruccions específic per coma flotant
- 32 registres de 32 bits: \$f0 - \$f31
 - \$f12 i \$f14: Pas de paràmetres a subrutina
 - \$f0: Retorn del resultat de la subrutina
 - \$f20-\$f31: Registres segurs
 - Per doble precisió només s'utilitzen registres parells (\$f0, \$f2, ...)
- Registre de control (FCR)
 - Excepcions
 - Overflow/underflow
 - Bit de Condició (CC): resultat de instruccions de comparació

Coma flotant en MIPS

- Variables globals en C:

```
float var1, var2[2] = {1.0, 3.14}; //precisió simple
double var3 = -37.55;             //precisió doble
```

- Variables globals en MIPS:

```
        .data
var1:    .float 0.0
var2:    .float 1.0, 3.14
var3:    .double -37.55
```

- `.float` i `.double` alineen automàticament a adreces múltiples de 4 i de 8 respectivament

Coma flotant en MIPS

- Còpia entre registres

mfcl rt, fs	rt = fs	còpia CPU \leftarrow CP1
mtcl rt, fs	fs = rt	còpia CP1 \leftarrow CPU
mov.s fd, fs	fd = fs	còpia CP1 \leftarrow CP1

Coma flotant en MIPS

- Accés a memòria

lwc1/lwc1/swc1/sdc1		
lwc1 ft, off16(rs)	$ft = M_w[rs + \text{SignExt}(\text{off16})]$	load float
ldc1 ft, off16(rs)	$ft = M_d[rs + \text{SignExt}(\text{off16})]$	load double
swc1 ft, off16(rs)	$M_w[rs + \text{SignExt}(\text{off16})] = ft$	store float
sdc1 ft, off16(rs)	$M_d[rs + \text{SignExt}(\text{off16})] = ft$	store double

- rs és un registre de propòsit general de la CPU

Coma flotant en MIPS

- Aritmètiques

add.s fd, fs, ft	$fd = fs + ft$	suma floats
add.d fd, fs, ft	$fd = fs + ft$	suma doubles
sub.s fd, fs, ft	$fd = fs - ft$	resta floats
sub.d fd, fs, ft	$fd = fs - ft$	resta doubles
mul.s fd, fs, ft	$fd = fs \times ft$	multiplica floats
mul.d fd, fs, ft	$fd = fs \times ft$	multiplica doubles
div.s fd, fs, ft	$fd = fs / ft$	divideix floats
div.d fd, fs, ft	$fd = fs / ft$	divideix doubles

Coma flotant en MIPS

- Comparacions i salts condicionals
 - En comparacions, l'operand destinació implícit és el Bit de Condició (CC)

c.eq.s/c.eq.d/c.lt.s/c.lt.d/c.le.s/c.le.d			
c.eq.s	fs, ft	bit de condició cc=1 si fs==ft, sino cc=0	igual que float
c.eq.d	fs, ft	bit de condició cc=1 si fs==ft, sino cc=0	igual que double
c.lt.s	fs, ft	bit de condició cc=1 si fs < ft, sino cc=0	menor que float
c.lt.d	fs, ft	bit de condició cc=1 si fs < ft, sino cc=0	menor que double
c.le.s	fs, ft	bit de condició cc=1 si fs ≤ ft, sino cc=0	menor o igual que float
c.le.d	fs, ft	bit de condició cc=1 si fs ≤ ft, sino cc=0	menor o igual que double

bc1t/bc1f			
bc1t	etiqueta	Salta si cc=1 (true)	
bc1f	etiqueta	Salta si cc=0 (false)	

Conversió a coma flotant - C vs MIPS

- En C:

```
int a = 2;  
float b = (float)a;    // b = 2.0
```

- En MIPS:

```
li      $t0, 2  
mtc1    $t0, $f0 # $f0 NO val 2.0  
# $f0 = 0 00000000 00000000000000000000000010
```

```
li      $t0, 0x40000000  
mtc1    $t0, $f0 # $f0 = 2.0
```


Exemple de traducció

```
float func(float x)
{
    if (x < 1.0)
        return x*x;
    else
        return 2.0 - x;
}
```

```
        .data
const1: .float 1.0

        .text

func:
    la      $t0, const1
    lwc1    $f16, 0($t0)
    c.lt.s  $f12, $f16
    bc1f    sino
    mul.s   $f0, $f12, $f12
    b       fisi
sino:
    add.s   $f16, $f16, $f16
    sub.s   $f0, $f16, $f12
fisi:
    jr      $ra
```