

# Arhitectura Calculatoarelor

Oprîtoiu Flavius  
flavius.opritoiu@cs.upt.ro

20 Noiembrie 2024  
27 Noiembrie 2024

## *Cap. 3* Analiza funcțională și sinteza unităților aritmetice de virgulă mobilă

### 3.1 - Operații și arhitecturi de virgulă mobilă

În general, se consideră operanzi IEEE 754 de virgulă mobilă (floating point - FP), cu excepția cazului în care se specifică altfel. Numerele IEEE 754 pot fi:

- ↙ împachetate (sau normalizate) : pentru stocare/transmisie de date
- ↘ despachetate : utilizate în timpul calculelor

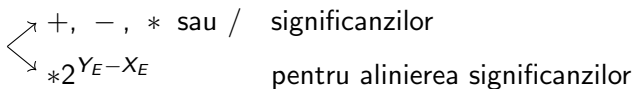
Fie  $X = X_M 2^{X_E}$  și  $Y = Y_M 2^{Y_E}$ . Cele patru operații aritmetice fundamentale sunt definite mai jos:

- ▶  $X + Y = (X_M + Y_M 2^{Y_E - X_E}) 2^{X_E}$ , dacă  $X_E \geq Y_E$
- ▶  $X - Y = (X_M - Y_M 2^{Y_E - X_E}) 2^{X_E}$ , dacă  $X_E \geq Y_E$
- ▶  $XY = X_M Y_M 2^{X_E + Y_E}$
- ▶  $\frac{X}{Y} = \frac{X_M}{Y_M} 2^{X_E - Y_E}$

### 3.1 - Operații și arhitecturi FP (contin.)

Pe baza definițiilor operațiilor fundamentale de aritmetică FP, o unitate de aritmetică FP are 2 subunități:

- ▶ calculul exponentului: efectuarea adunării sau scăderii exponenților
- ▶ calculul semnificandului: efectuarea

  $+ , - , * \text{ sau } /$       semnificanzilor  
 $*2^{Y_E - X_E}$       pentru alinierea semnificanzilor

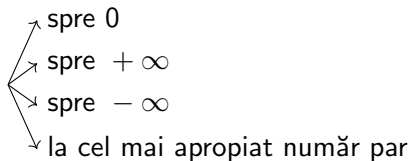
Cele două subunități operează doar cu operanzi de virgulă fixă:

- ▶ subunitatea exponentului folosește numere întregi de virgulă fixă (reprezentate în cod exces)
- ▶ subunitatea semnificandului operează cu numere fracționare de virgulă fixă

## 3.2 - Rotunjire

Rotunjire: conversia unei reprezentări de precizie ridicată la o reprezentare de precizie scăzută (pentru stocare/transmisie).

Moduri de rotunjire IEEE 754:



În IEEE 754, rotunjirea face referire la biții fracționari cu ponderi mai mici decât ponderea celui mai puțin semnificativ bit al significandului. Pentru concizie, doar în acest paragraf, rotunjirea vizează conversia unui număr cu parte întreagă și fracționară într-un număr întreg.

## 3.2 - Rotunjire (contin.)

Se consideră  $X$ , cu:

$$X = x_{n-1}x_{n-2} \dots x_1x_0.x_{-1}x_{-2} \dots x_{-m}$$

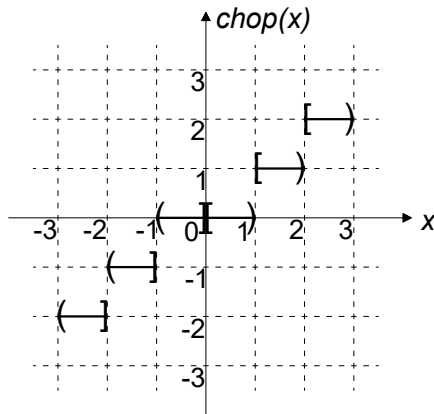
Fie  $X^*$  valoarea rotunjită a lui  $X$ , cu  $X^*$  fiind un întreg:

$$X^* = x_{n-1}^*x_{n-2}^* \dots x_1^*x_0^*$$

### 3.2 - Rotunjire (contin.)

(A) Rotunjire spre 0 (rotunjire spre interior)

$X^*$  este cel mai mare întreg pentru care  $|X^*| \leq |X|$

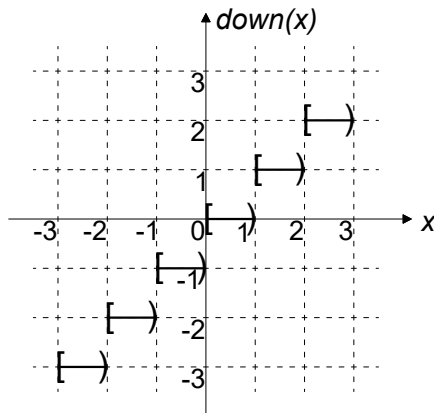


Dacă  $X$  este în cod S.-M., rotunjirea spre interior este echivalentă cu trunchierea la partea întreagă.

### 3.2 - Rotunjire (contin.)

(B) Rotunjire spre  $-\infty$  (rotunjire în jos)

$X^*$  este cel mai mare întreg pentru care  $X^* \leq X$



Dacă  $X$  este pozitiv, rotunjirea în jos este echivalenta cu rotunjirea spre 0.



## 3.2 - Rotunjire (contin.)

### ⓑ Rotunjire spre $-\infty$ (continuare)

Dacă  $X$  este în cod C2, rotunjirea în jos este echivalentă cu trunchierea la partea întreagă.

Dacă  $X$  este în cod S.-M., rotunjirea în jos este echivalentă cu:

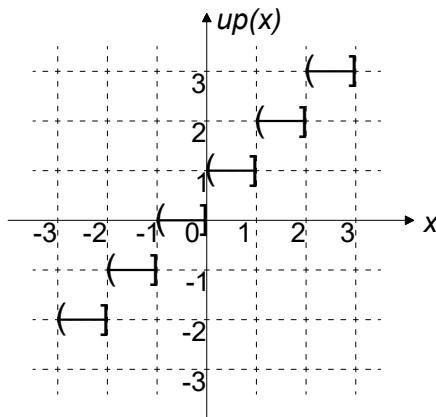
trunchiere la partea întreagă, dacă  $X \geq 0$

$$X^* = \begin{cases} x_{n-1}x_{n-2} \dots x_1x_0 - 1, & \text{if } .x_{-1}x_{-2} \dots x_{-m} \neq 0, \\ x_{n-1}x_{n-2} \dots x_1x_0, & \text{if } .x_{-1}x_{-2} \dots x_{-m} = 0 \end{cases}, \text{dacă } X < 0$$

### 3.2 - Rotunjire (contin.)

Ⓒ Rotunjire spre  $+\infty$  (rotunjire în sus)

$X^*$  este cel mai mic întreg pentru care  $X^* \geq X$



Dacă  $X$  este negativ, rotunjirea în sus este echivalentă cu rotunjirea spre 0.

## 3.2 - Rotunjire (contin.)

Caracteristicile rotunjirii în sus și în jos:

- ▶ erorile sunt în aceeași direcție  $\Rightarrow$  eroarea totală este acumulată mai repede
- ▶ furnizează o limită superioară/inferioară pentru rezultat  $\Rightarrow$  aritmetica intervalelor

## 3.2 - Rotunjire (contin.)

Ⓓ' Rotunjire la cel mai apropiat număr par

Modul de rotunjire IEEE 754 la cel mai apropiat număr par este derivat din modul de *rotunjire la cel mai apropiat număr*

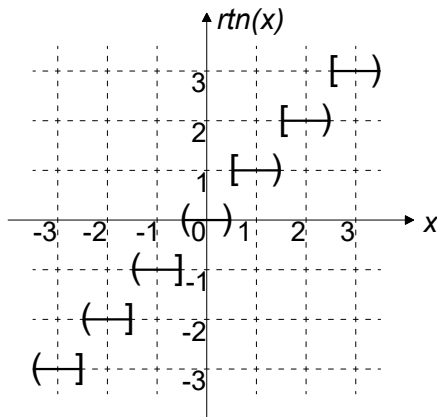
Fără a pierde din generalitate, se consideră  $X$  pozitiv.  $X^*$  rotunjit la cel mai apropiat număr este definit ca:

$$X^* = \begin{cases} x_{n-1}x_{n-2} \dots x_1x_0, & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} < \frac{1}{2} \\ x_{n-1}x_{n-2} \dots x_1x_0 + 1, & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} \geq \frac{1}{2} \end{cases}$$

În mod similar, se poate defini modul de rotunjire la cel mai apropiat număr pentru valori negative.

## 3.2 - Rotunjire (contin.)

(D') Rotunjire la cel mai apropiat număr (continuare)



### 3.2 - Rotunjire (contin.)

(D') Rotunjire la cel mai apropiat număr (continuare)

Analiza acumulării erorii: se consideră  $X$  pozitiv, având doar 2 biți de parte fracționară.

Intrări		Ieșiri	
$x_{-1}$	$x_{-2}$	$X^* = rtn(X)$	$\epsilon = X^* - X$
0	0	$x_{n-1}x_{n-2} \dots x_1x_0$	0
0	1	$x_{n-1}x_{n-2} \dots x_1x_0$	$-\frac{1}{4}$
1	0	$x_{n-1}x_{n-2} \dots x_1x_0 + 1$	$\frac{1}{2}$
1	1	$x_{n-1}x_{n-2} \dots x_1x_0 + 1$	$\frac{1}{4}$

Dacă cele 4 cazuri de mai sus sunt echiprobabile de-a lungul unei secvențe de calcule, eroarea medie se obține astfel:

$$\epsilon_{mean} = \frac{0 - \frac{1}{4} + \frac{1}{2} + \frac{1}{4}}{4} = \frac{1}{8}$$

Dacă linia 3 a tabelului de mai sus este mai probabil să apară în comparație cu celelalte,  $\epsilon_{mean}$  poate deveni mai mare decât  $\frac{1}{8}$ .

## 3.2 - Rotunjire (contin.)

ⓓ' Rotunjire la cel mai apropiat număr (continuare)

Soluția problemei acumulării erorilor: se împarte cazul

$.x_{-1}x_{-2} = .10$  în două sub-cazuri, cu probabilitate egală (sau cât mai aproape de egal), astfel încât unul dintre sub-cazuri să fie rotunjit în sus și celălalt în jos.

O abordare posibilă pentru împărțirea cazului de rotunjire a unei părți fracționare de  $\frac{1}{2}$  cu probabilități egale ar fi să fie inspectat bitul cel mai puțin semnificativ al părții întregi,  $x_0$ , diferențiind astfel între numere pare și impare. Pentru numerele pozitive, dacă  $x_0 = 0$  (numere pare), rotunjirea ar putea fi făcută în jos, în timp ce pentru  $x_0 = 1$  (numere impare), rotunjirea ar fi făcută în sus.

## 3.2 - Rotunjire (contin.)

### Ⓓ Rotunjire la cel mai apropiat număr par

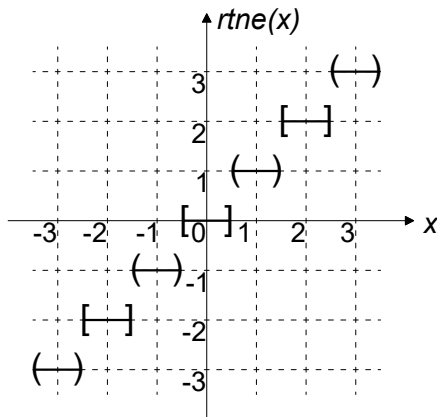
Fără a pierde din generalitate, se consideră  $X$  pozitiv.  $X^*$  rotunjit la cel mai apropiat număr par este definit ca:

$$X^* = \begin{cases} x_{n-1}x_{n-2} \dots x_1x_0, & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} < \frac{1}{2}, \text{ SAU} \\ & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} = \frac{1}{2} \text{ ŞI } x_0 = 0 \\ x_{n-1}x_{n-2} \dots x_1x_0 + 1, & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} > \frac{1}{2}, \text{ SAU} \\ & \text{dacă } .x_{-1}x_{-2} \dots x_{-m} = \frac{1}{2} \text{ ŞI } x_0 = 1 \end{cases}$$



## 3.2 - Rotunjire (contin.)

(D) Rotunjire la cel mai apropiat număr par (continuare)



### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP

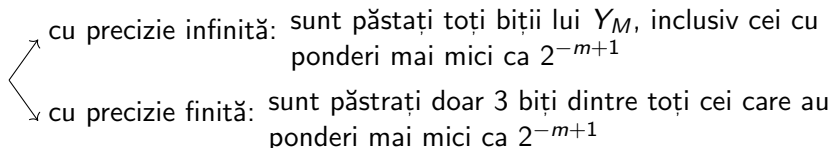
Se consideră semnificanții  $X_M$  și  $Y_M$ , pe  $m$  biți:

$$X_M = 1.x_{m-2}x_{m-3} \dots x_i \dots x_1x_0$$

$$Y_M = 1.y_{m-2}y_{m-3} \dots y_i \dots y_1y_0$$

Se consideră  $X_E \geq Y_E$  așa încât pentru adunarea semnificanzilor,  $Y_M$  trebuie aliniat prin deplasare la dreapta cu  $d = |X_E - Y_E|$  poziții.

Alinierea lui  $Y_M$  se poate face:

- 
- cu precizie infinită: sunt păstrați toți biții lui  $Y_M$ , inclusiv cei cu ponderi mai mici ca  $2^{-m+1}$
  - cu precizie finită: sunt păstrați doar 3 biți dintre toți cei care au ponderi mai mici ca  $2^{-m+1}$

### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP (contin.)

Cei 3 biți păstrați la alinierea lui  $Y_M$  cu precizie finită se numesc *biți sticky*:

- ▶  $g$ : bit de gardă, cu ponderea  $2^{-m}$ ; protejează împotriva pierderii de precizie
- ▶  $r$ : bit de rotunjire, cu ponderea  $2^{-m-1}$ ; utilizat pentru rotunjirea rezultatului
- ▶  $s$ : bitul sticky, cu ponderea  $2^{-m-2}$ ; se obține ca rezultat al unui SAU logic între toți ceilalți biți mai puțin semnificativi care au fost deplasați la dreapta din  $Y_M$ , exceptând biții  $g$  și  $r$

După aliniere,  $Y_M$  devine  $Y_{MaI}$ .

### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP (contin.)

Se consideră rezultatul FP al unei adunări  $\Rightarrow Z_M = X_M + Y_{MaI}$ , cu condiția ca  $X_E \geq Y_E$ .

Deoarece adunarea semnificanzilor poate genera un transport de ieșire, rezultă că forma lui  $Z_M$  este:

$$Z_M = z_m \ z_{m-1} \cdot z_{m-2} \ z_{m-3} \dots z_1 \ z_0 \mid g \ r \ s$$

Operația de normalizare pentru  $Z_M$  va produce  $Z_{M_n}$ , cu

$$Z_{M_n} = 1 \cdot z_{m-2_n} \ z_{m-3_n} \dots z_{1_n} \ z_{0_n} \mid R \ S$$

Cei doi biți,  $R$  și  $S$ , sunt necesari pentru efectuarea operației de rotunjire, succesive normalizării.

### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP (contin.)

Cazuri de normalizare:

$Z_{M_n} =$	1.	$z_{m-2_n}$	$z_{m-3_n}$	$\dots$	$z_{1_n}$	$z_{0_n}$		$R$	$S$
Caz 1) $z_m = 1$	1.	$z_{m-1}$	$z_{m-2}$	$\dots$	$z_2$	$z_1$		$z_0$	$(g \text{ OR } r \text{ OR } s)$
$\Rightarrow$ depl. dreapta 1 bit, $Z_E++$									
Caz 2) $z_m = 0,$ $z_{m-1} = 1$	1.	$z_{m-2}$	$z_{m-3}$	$\dots$	$z_1$	$z_0$		$g$	$(r \text{ OR } s)$
$\Rightarrow Z_M$ este deja normalizat									
Caz 3) $z_m = 0,$ $z_{m-1} = 0,$ $z_{m-2} = 1$	1.	$z_{m-3}$	$z_{m-4}$	$\dots$	$z_0$	$g$		$r$	$s$
$\Rightarrow$ depl. stânga 1 bit, $Z_E--$									
Caz 4) $z_m = 0,$ $z_{m-1} = 0,$ $z_{m-2} = 0,$ $z_{m-3} = 1$	1.	$z_{m-4}$	$z_{m-5}$	$\dots$	$g$	$0$		$0$	$0$
$\Rightarrow$ depl. stânga 1 bit, $Z_E- = 2$									

### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP (contin.)

Dacă normalizarea lui  $Z_M$  necesită o operație de deplasare la stânga cu 2 sau mai mulți biți:

- ▶ se atașază bitul  $g$  lui  $Z_M$ , după noua poziție a bitului  $z_0$
- ▶ se completează pozițiile libere mai puțin semnificative din  $Z_M$  cu 0
- ▶ se alege  $R = S = 0$

Rotunjirea lui  $Z_{M_n}$  utilizează biții  $R$  și  $S$  determinați anterior:

- ▶ sunt folosiți  $R$  și  $S$  pentru implementarea tuturor celor 4 moduri de rotunjire

### 3.3 Reguli de normalizarea și rotunjirea unui rezultat FP (contin.)

#### Reguli de rotunjire

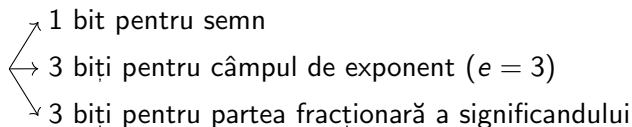
Mod de rotunjire	$Z_{M_n} > 0$	$Z_{M_n} < 0$
spre 0	_____ (sunt neglijăți $R$ și $S$ )	_____
spre $-\infty$	_____	<i>if</i> ( $R$ <u>or</u> $S$ ) <i>then</i> $Z_{M_n} - 1$
spre $+\infty$	<i>if</i> ( $R$ <u>or</u> $S$ ) <i>then</i> $Z_{M_n} + 1$	_____
la cel mai apropiat nr. par	<i>if</i> ( $R$ <u>and</u> ( $S$ <u>or</u> $z_{0_n}$ )) <i>then</i> $Z_{M_n} + 1$	<i>if</i> ( $R$ <u>and</u> ( $S$ <u>or</u> $z_{0_n}$ )) <i>then</i> $Z_{M_n} - 1$

### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP

Eroarea de rotunjire nu este corelată cu diferența exponenților celor 2 operanzi (vezi Fig. 5.12 din [?]).

Se va utiliza un format FP simplificat și redus:

- ▶ inspirat de IEEE 754, cu câmpuri mai înguste



- ▶ aceeași relație de calcul al bias-ului ca în IEEE 754  
$$bias = 2^{e-1} - 1 = 3$$
- ▶ aceleași excepții ca în IEEE 754



### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Se consideră următorii operanzi:

$$X = 0.5625_{(10)} = 0.1001_{(2)} = 1.\underbrace{001}_{\text{mantisa}} \cdot 2^{-1}$$

$$Y = -3.75_{(10)} = -11.11_{(2)} = -1.\underbrace{111}_{\text{mantisa}} \cdot 2^1$$

Formatul operanzilor împachetați:

$X$ : 

0	0	1	0	.	0	0	1
---	---	---	---	---	---	---	---

$-1 + \text{bias} = 2_{10} = 010_2$

$Y$ : 

1	1	0	0	.	1	1	1
---	---	---	---	---	---	---	---

$1 + \text{bias} = 4_{10} = 100_2$

### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Algoritmul de adunare cu rotunjire a numerelor de virgulă mobilă:

Pasul ①: Despachetare operanzi

- ▶ atașare bit ascuns
- ▶ verificare excepții: unul din operanzi este 0,  $\pm\infty$ , NaN

X : 

0	0	1	0	1	.	0	0	1
---	---	---	---	---	---	---	---	---

Y : 

1	1	0	0	1	.	1	1	1
---	---	---	---	---	---	---	---	---

### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Pasul ②: Calcularea diferenței exponenților,  $d = X_E - Y_E$

- ▶ dacă  $d < 0$  rezultă că  $|X| < |Y|$ 
  - ▶ se interschimbă operanzii
  - ▶ se alege exponentul rezultatului,  $Z_E = Y_E$
- ▶ dacă  $d \geq 0$ 
  - ▶ se alege exponentul rezultatului,  $Z_E = X_E$

Interschimbarea operanzilor permite reducerea ariei unității FP (păstrează facilitatea de deplasare la dreapta pentru aliniere doar pentru  $Y$ ).

Calculează  $d = X_E - Y_E = -2$ . Pentru că  $d$  este negativ  $\Rightarrow$  INTERSCHIMBĂ operanzii și alege  $Z_E = Y_E = 4$ .

$X :$ 

1	1	0	0	1	.	1	1	1
---	---	---	---	---	---	---	---	---

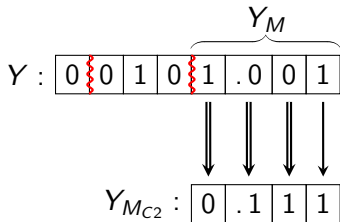
$Y :$ 

0	0	1	0	1	.	0	0	1
---	---	---	---	---	---	---	---	---

### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Pasul ③: Dacă  $sign(X) \neq sign(Y) \Rightarrow$  se complementează de doi  $Y_M$

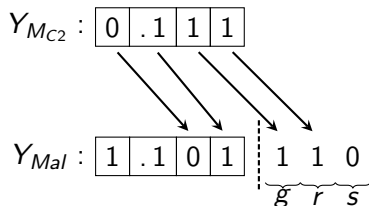
- ▶ semne diferite implică operația de scădere
  - ▶ complementarea de doi permite utilizarea unui sumator binar pentru scăderea semnificanzilor
- ▶ se complementează de doi doar significandul  $Y_M$  (reduce aria unității FP evitând includerea aceleași facilități și pentru  $X_M$ )



### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Pasul ④: Se aliniază  $Y_M$  prin deplasare la dreapta cu  $|d|$  poziții,  $Y_M$  după aliniere este referit prin  $Y_{Mal}$

- ▶ dacă în Pasul 3  $Y_M$  a fost complementat de doi, la deplasarea la dreapta, se introduc biți de 1 în pozițiile mai semnificative ale lui  $Y_M$ , în loc de 0
- ▶ sunt păstrați biții sticky:  $g$ ,  $r$  și  $s$



### 3.4 Adunarea/scăderea cu rotunjire a numerelor FP (contin.)

Pasul ⑤: Se adună cei 2 significanți:  $Z_M = X_M + Y_{Mal}$

- ▶ dacă  $sign(X) = sign(Y)$  potențialul bit de transport de ieșire generat este păstrat (este parte din rezultat)
- ▶ dacă  $sign(X) \neq sign(Y)$  și nu este generat bitul de transport de ieșire  $\Rightarrow Z_M$  este negativ și trebuie complementat de doi
- ▶ dacă  $sign(X) \neq sign(Y)$  și este generat bitul de transport de ieșire  $\Rightarrow Z_M$  este pozitiv și bitul transport este neglijat

