Arhitectura calculatoarelor 16.10.2024

(curs 3 - S3)

1.3. Reprezentarea numerelor zecimale cu punct fix

Comparație coduri de reprezentare zecimală:

Cifra zecimală	Coduri zecimale de virgulă fixă		
	BCD8421	Exces de 3	Doi-din-cinci
0	0000	0011	11000
1	0001	0100	00011
2	0010	0101	00101
3	0011	0110	00110
4	0100	0111	01001
5	0101	1000	01010
6	0110	1001	01100
7	0111	1010	10001
8	1000	1011	10010
9	1001	1100	10100

Notă: codificarea *doi-din-cinci* nu este unică.

1.3.1. Binary Coded Decimal 8421

⟶ reprezintă o cifră zecimală pe o tetradă (grup de 4 biți)

- tetradă ≡ nibble ≡ 4 biți consecutivi
- de obicei, referit prin mai simplul Binary Coded Decimal (BCD)

ex: $297_{(10)}$ = 0010 1001 0111 (BCD)

Ponderea unui bit: $10^i \cdot 2^j$, unde | i - poziția cifrei zecimale
j - poz. bitului în tetradă

- ponderi fixe ⟶ reprezentare pozițională

1.3.2. Exces de 3

⟶ reprezintă o cifră zecimală pe o tetradă

exces = bias

⟶ valoarea adăugată la toate configurațiile codării

Excesul de 3 (E3) adaugă 3 unități la cifra zecimală BCD corespunzătoare.

- Nu este pozițional
- Facilitează adunarea

ex: $297_{(10)}$ = 0101 1100 1010 (E3)

5(2+3) 12(9+3) 10(7+3)

1.3.3. Doi-din-cinci

→ reprezintă o cifră zecimală folosind 5 biți

- 2 biți din cei 5 utilizați pentru reprezentarea unei cifre sunt 1
- 3 biți din cei 5 utilizați pentru reprezentarea unei cifre sunt 0

Doi-din-cinci (2-o-o-5) facilitează detectarea erorilor:

- folosind coduri de paritate: modificarea oricăruia dintre cei 5 biți schimbă numărul de biți de 1 și 0 în reprezentarea fiecărei cifre

ex: $297_{(10)} = 00101 \quad 10100 \quad 10001$ (2-o-o-5)

1.4. Reprezentarea numerelor de virgulă mobilă

→ Notație științifică: $X = X_M \cdot B^{X_E}$

unde X_M — mantisa - reprezentata ca un numar de virgula fixa, fractionar

X_E — exponentul -reprezentat ca un numar de virgula fixa, intreg

B — baza reprezentării - in mod obisnuit, B este 2 sau o putere a lui 2

! atat Xm cat si Xe pot fi reprezentate in SM sau C2

ex: $N = 0.55$

0.55	·2
1.10	·2
0.20	·2
0.40	·2
0.80	·2
1.60	·2
1.20	·2
0.40	·2
0.80	⋮

$\Rightarrow N = .10001100$ – partea binară fractionară

Normalizăm N:

$N = 1,0001100 \cdot 2^{-1}$ (luăm trunchiat 8 biți)

Convertim partea binară:

$1,\underset{1}{0}\underset{2}{0}\underset{3}{0}\underset{4}{1}\underset{5}{1}\underset{6}{0}\underset{7}{0} = 1 + \frac{1}{2^4} + \frac{1}{2^5} = 1,09375$

Deoarece exponentul este −1, împărțim la 2

$\Rightarrow N' = \frac{1,09375}{2} = 0,546875$

Eroarea de aproximație $\varepsilon = N - N' = 0,55 - 0,546875 =$

$= 0,003125$

Dacă am face pe 16 biți

$N'' = .\,1000.1100.1100.1100$

Prin același procedeu obținem

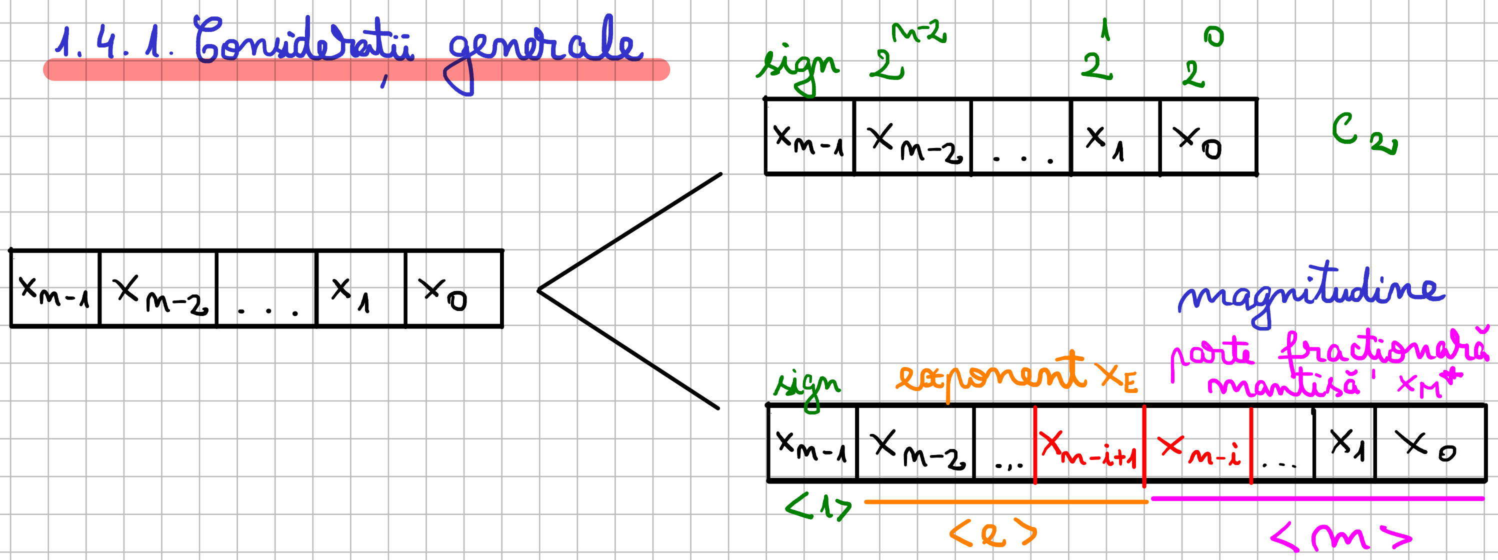
$N'' = 54998779296875$

$\varepsilon'' = N - N'' = 0,00001220703125$

$\Rightarrow \quad \varepsilon' > \varepsilon''$

Mai mulți biți duc la o precizie mai mare

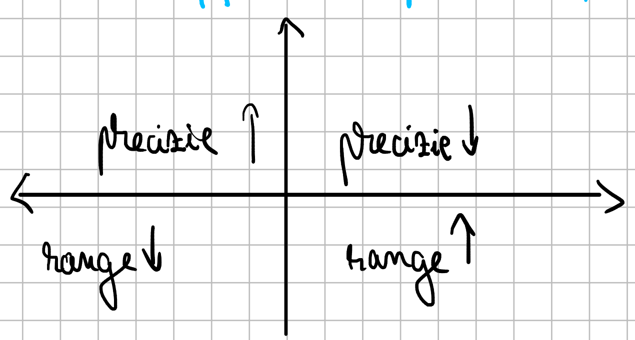
1.4.1. Considerații generale



$\Rightarrow \quad X_M = S \cdot X_M^*$, unde

- X_M - mantisă
- S - semnul nr. de virgulă mobilă
- X_M^* - partea fracționară a mantisei

Trade-off între precizie și interval de valori



Cu cat am mai multi biti, eroarea este mai mica, dar dezavantajul e ca avem mai putini biti de exponent, range ul de reprezentare este mai mic

In cealalta parte, precizia scade, dar intervalul de valori creste.

Constrângeri

(A) Reprezentarea valorii 0

- X_E egal cu 0: ar trebui să fie cea mai mică valoare posibilă
  - comparație rapidă cu 0
  - erori de rotunjire

cea mai mică valoare X_E:
$$X_E \begin{cases} SM: & 11\ldots1 = -2^{e-1}-1 \\ C2: & 10\ldots0 = -2^{e-1} \end{cases}$$

- X_M egal cu 0: ar trebui să fie configurația all-0, indiferent de codul de reprezentare folosit, SM sau C2

$$0 \begin{cases} SM: & S|11\ldots1|00\ldots0 \\ C2: & S|10\ldots0|00\ldots0 \end{cases}$$

- Compararea rapidă cu 0 a numerelor de virgulă mobilă
  - necesită ca X_E să fie all-0
  - $\Rightarrow$ utilizarea unui cod bias (sau exces) pentru X_E
    - fiecărei configurații binară a noului cod exces i se asociază o valoare egală cu valoarea binară a codului la care se adaugă valoarea excesului

Excesul : cea mai mare valoare reprezentabila in campul X_E

$$\text{exces} \begin{cases} SM: & 2^{e-1}-1 \\ C2: & 2^{e-1} \end{cases}$$

=> noua reprezentare a lui 0 :

$$0 \longrightarrow S|00\ldots0|00\ldots0$$

Observatie: reprezentarea lui 0 este cu semn, semnul S fiind codificat in MSB-ul formatului.

(B) Reprezentarea mantisei

- Inerent redundantă
  - În zecimal, următoarele expresii se referă la aceeași valoare: $0.237 = 0.0237 * 10^1 = 0.000237 * 10^3 = 23.7 * 10^{-2} = \cdots$
- Utilizarea *mantisei normalizate*: reprezentare unică

MSB-ul părții fracționare a mantisei : $\begin{cases} 1 & \text{dacă mantisa este în SM} \\ \overline{S} & \text{dacă mantisa este în C2} \end{cases}$

Exemplu:

$0.\underbrace{1}_{msb=1}11_{SM} = \frac{7}{2^3}$ $\quad$ $1.\underbrace{1}_{msb=1}11_{SM} = \frac{-7}{2^3}$

$\underbrace{0}_{S}.\underbrace{1}_{msb=\overline{S}}11_{C2} = \frac{7}{2^3}$ $\quad$ $\underbrace{1}_{S}.\underbrace{0}_{msb=\overline{S}}01_{C2} = \frac{-7}{2^3}$

Interval de valori: $\frac{1}{2} \le \|X_M\| < 1$

1.4.2. IEEE 754

→ format portabil

32 de biți	- precizie simpla
64 de biți	- precizie dubla
128 de biți	- precizie cvadrupla

Format de schimb de date:
- pe 16 biți (precizie înjumătățită)
  - folosit pentru accelerarea Deep Neural Networks

Formatul de precizie simplă

S	X_E	X_S^*
<1>	<8>	<23>

$$bias = 127 = 2^{8-1} - 1$$

Câmpul exponentului, X_E, reprezintă exponentul real într-un cod exces de $127 (= 2^{8-1} - 1)$.

Pentru standardul IEEE 754, mantisa consta din bitul de semn si significand.
Significandul este fara semn si are reprezentarea :

$$X_S = \underbrace{1.}_{\text{ascuns}} X_S^*$$

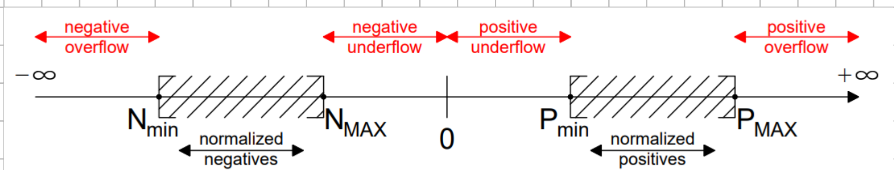
- X_S - significand
- 1 - poziția supraunitară; **ascunsă** în reprezentare
- X_S^* - partea fracționară a significandului

Interval de valori pt significand: $1 \leq X_S < 2$

Valoarea unei reprezentari IEEE 754 de precizie simpla este data de relatia :

$$X = (-1)^S \cdot 2^{X_E - \text{exces}} \cdot (1.X_S^*)$$

Limitele standardului IEEE 754 de precizie simplă:



Determinarea P_{MAX} =

S	$X_E = 254$	$X_S^* = .11...1$
0	11 11 1110	11 ... 1
<1>	<8>	<23>

$$X_E = X_{E_{MAX}} - 1 = 255 - 1 = 254$$

$$X_S^* = .\underbrace{11...1}_{23} = 1 - 2^{-23}$$

$$P_{MAX} = (-1)^0 \cdot 2^{254-127} \cdot (1.11...1) = 2^{127} \cdot (2 - 2^{-23}) =$$

$$= 3.4 \cdot 10^{38}$$

Determinarea

$$P_{MIN} = \begin{array}{c|c|c} S & X_E = 1 & X_S^* = .00\ldots0 \\ \hline 0 & 0000\ 0001 & .00\ldots0 \\ \hline \langle 1 \rangle & \langle 8 \rangle & \langle 23 \rangle \end{array}$$

$$X_E = X_{E_{min}} + 1 = 0 + 1 = 1$$

$$X_S^* = .\underbrace{00\ldots}_{23}0 = 0$$

$$P_{MIN} = (-1)^0 \cdot 2^{1-127} \cdot (1.00\ldots0) = 2^{-126}$$

$$\approx 1{,}18 \cdot 10^{-38}$$

Valori speciale

A) Not a Number (NAN)

- rezultatul unor operații precum $\infty - \infty$, $0 * \infty$, $\frac{\infty}{\infty}$, $\frac{0}{0}$, X mod 0, $\sqrt{\text{valoare negativă}}$
- câmpul exponentului, $X_E = X_{E_{max}} = 255$
- câmpul de parte fracționară a significandului, $X_S^* \neq 0$
  - valoarea părții fracționare a significandului indică operația care a produs NaN

B) Infinit

- indică un rezultat care depășește capacitatea de reprezentare
  - preferabil operației de trunchiere la P_{MAX}
  - **Q**: de ce?
- rezultatul unor operații precum $X \div 0$, $X \pm \infty$, $\sqrt{\infty}$
- câmpul exponentului, $X_E = X_{E_{max}} = 255$
- câmpul de parte fracționară a significandului, $X_S^* = 0$

$$a = b = \frac{P_{MAX}}{\sqrt{2}} \qquad \sqrt{a^2 + b^2} = \sqrt{\frac{P_{MAX}^2 + P_{MAX}^2}{2}} = P_{MAX}$$

C) Zero

- câmpul exponentului, $X_E = X_{E_{min}} = 0$
- câmpul părții fracționare a significandului, $X_S^* = 0$

⑧ Numere denormalizate (Denormals)

- indică un rezultat, în valoare absolută, mai mic decât P_{min}, dar cu $X_S^\star$ nenul
  - preferabil trunchierii la P_{min}
- câmpul exponentului, $X_E = X_{E_{min}} = 0$
- câmpul părții fracționare a significandului, $X_S^\star \neq 0$

Valoarea unui denormal este dată de:

$$X_D = (-1)^S * 2^{1-exces} * (0.X_S^\star)$$

unde

- X_D - număr denormalizat
- 0 - poziția supraunitară; **ascunsă** în reprezentare
- $X_S^\star$ - parte fracționară a significandului

Exemplu: Reprezentați $X = 612.8046875_2$ în formatul IEEE 754 de precizie simplă

$612.8046875_{10} = 1001100100.1100111_2$

$$\begin{array}{r|l}
612 - & \\
512 & (2^9) \\
\hline
100 - & \\
64 & (2^6) \\
\hline
36 - & \\
32 & (2^5) \\
\hline
4 & (2^2)
\end{array}
\qquad
\begin{array}{l|l}
.8046875_{10} & \times 2 \\
1.609375_{10} & \times 2 \\
1.21875_{10} & \times 2 \\
0.4375_{10} & \times 2 \\
0.875_{10} & \times 2 \\
1.75_{10} & \times 2 \\
1.5_{10} & \times 2 \\
1.0_{10} &
\end{array}$$

$612.8046875_{10} = 1.0011001001100111_2 \times 2^9$

$X_E = (9+bias)_{10} = (9+127)_{10} = 136_{10} = 10001000_2$

$X_S^\star = .0011001001100111_2$

	S	X_E		$X_S^\star$					
X:	0 1 0 0	0 1 0 0	0 0 0 1	1 0 0 1	0 0 1 1	0 0 1 1	1 0 0 0	0 0 0 0	
	4	4	1	9	3	3	8	0_{16}	

$X = 44193380_{16}$

ex: $75.046875 = 1001011.000011$

$$\begin{array}{r l}
75 - & \\
64 & 2^6 \\
\hline
11 - & \\
8 & 2^3 \\
\hline
3 - & 2^1 \\
2 & \\
\hline
1 & -2^0 \\
1 & \\
\hline
0 &
\end{array}
\qquad
\begin{array}{l|l}
0.046825 & \cdot 2 \\
0.093750 & \cdot 2 \\
0.187500 & \cdot 2 \\
0.750000 & \cdot 2 \\
1.500000 & \cdot 2 \\
1.000000 &
\end{array}$$

$= 1.001011000011 \cdot 2^6$

$\Rightarrow X_E = 6 + bias = 6 + 127 = 1000\ 0101$

$X_S^\star = \underbrace{0010110000110\ldots0}_{23}$

S	X_E			$X_S^\star$			
0 1 0 0	0 0 1 0	1 0 0 1	0 1 1 0	0 0 0 1	1 0 0 0	0 0 0 0	0 0 0 0
4	2	9	6	1	8	0	0

(16)

Exemplu: Dându-se configurația $412A0000_{16}$ care reprezintă un număr de virgulă mobilă în formatul IEEE 754 de precizie simplă, determinați corespondentul său zecimal.

	4	1	2	A	0	0	0	0_{16}
X:	0 1 0 0	0 0 0 1	0 0 1 0	1 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0

S | X_E | X_S^*

$X_E=10000010_2=(128+2)_{10}=130_{10}$

$X_S^*=.010101_2$

$X=(-1)^S\times2^{X_E\text{-bias}}\times(1.X_S^*)=(-1)^0\times2^{130-127}\times(1.010101_2)=2^3*1.010101_2$

$X=1010.101_2=(10+2^{-1}+2^{-3})_{10}=(10+0.5+0.125)_{10}=10.625_{10}$

ex: $C1AE0000_{(16)}$

$1 \mid 100\ 0001\ 1 \mid 010\ 1110\ 0000\ 0000\ 0000\ 0000$

S | X_E | X_S^*

$X_E = 128 + 3 = 131$

$X_S^* = .010111$

$$X = (-1)^S \cdot 2^{X_E-127} \cdot (1.010111) =$$

$$= (-1) \cdot 2^4 \cdot 1.010111 =$$

$$= -10101.11 = 21,75$$

1.4.3. Formatul de virgulă mobilă IBM

Specific sistemelor IBM S/360, S/370

- Formate de 32, 64 și 128 de biți
- Baza 16 ($=2^4$)
- Fără bit ascuns
- Normalizarea mantisei:
  - Toți biții mantisei sunt la dreapta virgulei binare
  - Pot exista cel mult 3 leading-zeroes
- Un singur caz special - zero:
  - $X_E = 0$
  - $X_M^* = 0$

Formatul de 32 de biți:

sign	exponent	fractional part of mantissa
S	X_E	X_M^*
<1>	<7>	<24>

exponent / bias $= 64 = 2^{7-1}$

Valoarea lui X poate fi exprimata ca :

$$X = (-1)^S \cdot 16^{X_E - \text{bias}} \cdot (0.X_M^*)$$

Exemplu: Reprezentați valoarea reală $X = 612.8046875_2$ în formatul de virgulă mobilă IBM pe 32 de biți

$612.8046875_{10} = 1001100100.1100111_2$

$612.8046875_{10} = 0.0010011001001100111_2 \times 16^3$

$X_E = (3+\text{bias})_{10} = (3+64)_{10} = 67_{10} = 1000011_2$

$X_M^* = .0010011001001100111_2$

	S	X_E	X_M^*
X:	0	1 0 0 0 0 1 1	0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1 1 0 0 0 0 0

X:	0 1 0 0	0 0 1 1	0 0 1 0	0 1 1 0	0 1 0 0	1 1 0 0	1 1 1 0	0 0 0 0
	4	3	2	6	4	C	E	0_{16}

$X = 43264CE0_{16}$

Exemplu: Determinați valoarea zecimală reală care corespunde configurației hexazecimale $412A0000_{16}$, având în vedere că valoarea hexazecimală reprezintă un număr de virgulă mobilă în formatul IBM pe 32 de biți.

	4	1	2	A	0	0	0	0_{16}
X:	0 1 0 0	0 0 0 1	0 0 1 0	1 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0

	S	X_E	X_M^*
X:	0	1 0 0 0 0 0 1	0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

$X_E = 1000001_2 = (64+1)_{10} = 65_{10}$

$X_M^* = .0010101_2$

$X = (-1)^S \times 16^{X_E - \text{bias}} \times (0.X_M^*) = (-1)^0 \times 16^{65-64} \times (0.0010101_2) = 2^4 \times 0.0010101$

$X = 10.101_2 = (2 + 2^{-1} + 2^{-3})_{10} = (2 + 0.5 + 0.125)_{10} = 2.625_{10}$

II. Analiza funcțională și sinteza dispozitivelor de adunare și scădere, binară și zecimală

2.1. Sumatoare seriale

→ adună o pereche de biți ai celor 2 operanzi, în fiecare ciclu de ceas

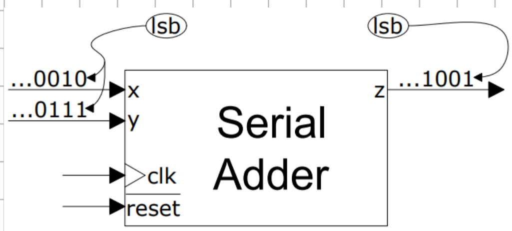
- Avantaje:
  - Suprafața ⇓; consum de energie ⇓; frecvența de operare ⇑;
- Dezavantaj:
  - Latența rezultatului final ⇑;

Tipuri de sumatoare seriale :
- LSDF (Least significant digit first)
- MSDF (Most significant digit first)

LSDF

$$
\begin{array}{lcccccl}
X: & x_{n-1} & x_{n-2} & \cdots & x_1 & x_0 & \\
Y: & y_{n-1} & y_{n-2} & \cdots & y_1 & y_0 & + \\
\hline
Z: & z_{n-1} & z_{n-2} & \cdots & z_1 & z_0 &
\end{array}
$$

Simbolul sumatorului serial:

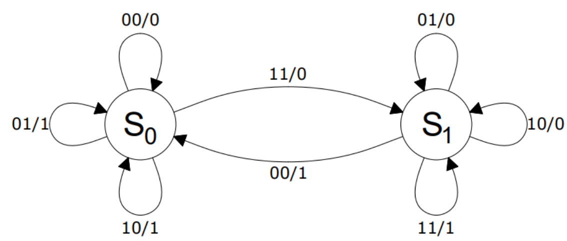


Propagarea transportului -> utilizeaza starea interna a adunatorului

Are 2 stări interne:
- S0: fara propagare de carry din rangul anterior
- S1: cu propagare de carry din rangul anterior

Etapele sintezei unui sumator serial

A) Diagrama de tranziție



B) Tabelul de stări

Stare \ Cfg. intrare	(x,y) 00	01	11	10
S_0	S_0 / 0	S_0 / 1	S_1 / 0	S_0 / 1
S_1	S_0 / 1	S_1 / 0	S_1 / 1	S_1 / 0

→ se iau de pe diagramă

C) Codificarea stărilor

- numărul minim de variabile de stare care pot codifica stările
  - pentru s stări, numărul minim de variabile de stare este $\lceil log_2 s \rceil$
  - $\Rightarrow$ pentru sumatorul serial (având $s = 2$), este necesară doar o variabilă de stare ($\lceil log_2 s \rceil = \lceil log_2 2 \rceil = 1$)

w → 0 : codifică S_0
w → 1 : codifică S_1

D) Tabel de tranziție

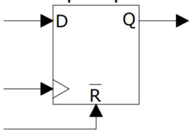
Var. stare, w \ Cfg. intrare	(x,y) 00	01	11	10
0	0 / 0	0 / 1	1 / 0	0 / 1
1	0 / 1	1 / 0	1 / 1	1 / 0

→ practic la fel cu cel de sus (B)

E) Tabel de excitație

→ dependent de tipul elementelor de stocare utilizate

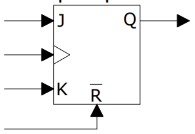
Flip-flop D



Q(t+1)=D

Intrări			Ieșiri	
w	x	y	D	z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Flip-flop J-K → la curs



$Q(t+1)=J \cdot \overline{Q(t)} + \overline{K} \cdot Q(t)$

Intrări			Ieșiri		
w	x	y	J	K	z
0	0	0	0	*	0
0	0	1	0	*	1
0	1	0	0	*	1
0	1	1	1	*	0
1	0	0	*	1	1
1	0	1	*	0	0
1	1	0	*	0	0
1	1	1	*	0	1

7) Ecuațiile de ieșire și feedback (Minimizări)

pt J-K

Intrări			Ieșiri		
w	x	y	J	K	z
0	0	0	0	*	0
0	0	1	0	*	1
0	1	0	0	*	1
0	1	1	1	*	0
1	0	0	*	1	1
1	0	1	*	0	0
1	1	0	*	0	0
1	1	1	*	0	1

Ecuațiile de ieșire:

$$z = wxy + w\bar{x}\bar{y} + \bar{w}x\bar{y} + \bar{w}\bar{x}y =$$

$$= w(xy + \bar{x}\bar{y}) + \bar{w}(x\bar{y} + \bar{x}y) =$$

$$= w\left(\overline{x \oplus y}\right) + \bar{w}(x \oplus y) =$$

$$= w \oplus x \oplus y$$

Ecuațiile feedback

J

$w \backslash x,y$	00	01	11	10
0	0	0	1	0
1	d	d	d	d

$J = x \cdot y$

K

$w \backslash x,y$	00	01	11	10
0	d	d	d	d
1	1	0	0	0

$K = \bar{x} \cdot \bar{y} = \overline{x + y}$

Pentru D

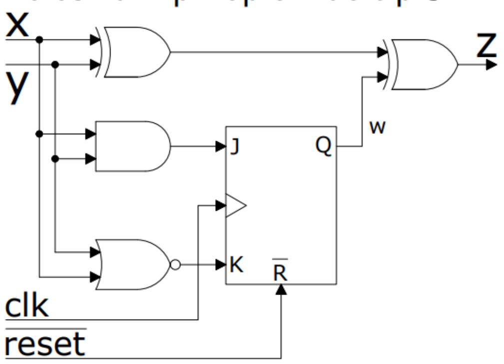
D

$w \backslash x,y$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

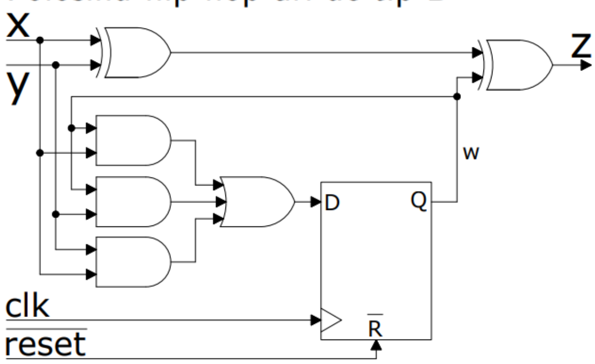
$D = w \cdot x + w \cdot y + x \cdot y$

G) Sinteza sumatorului serial

Folosind flip-flop-uri de tip J-K



Folosind flip-flop-uri de tip D



2.2. Sumatoare și scăzătoare paralele

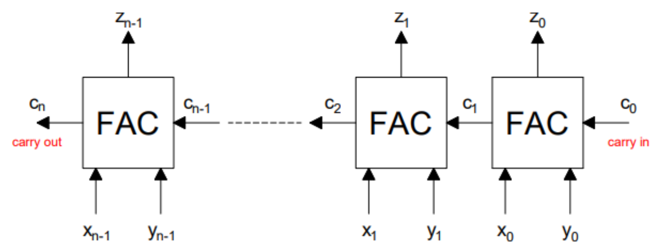
2.2.1. Sumatoare paralele bazate pe propagarea serială a transportului

Ripple Carry Adder (RCA)

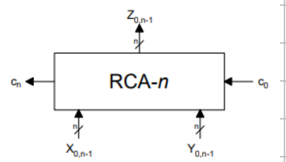
→ utilizeaza celule dedicate de insumare pentru fiecare rang binar

Propagarea carry - ului : catre pozitia mai semnificativa(la stanga)

Arhitectură RCA pe *n* biți:

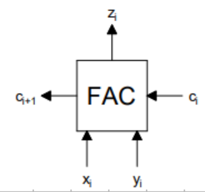


Simbolul unui sumator RCA pe *n* biți:



Full Adder Cell (FAC)

Simbol:



→ adună 3 biți

→ $x_i + y_i$ + carry ⇒ suma + carry urm.

Tabel de adevăr

Inputs			Outputs	
x_i	y_i	c_i	z_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

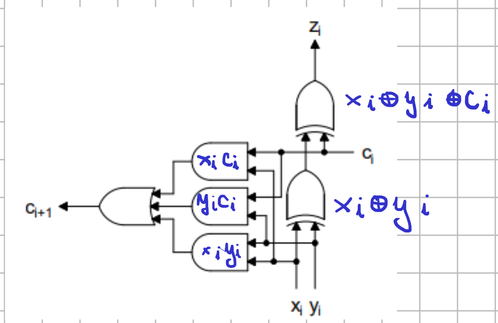
Ecuațiile ieșirilor

$$z_i = x_i \oplus y_i \oplus c_i$$
$$c_{i+1} = x_i \cdot y_i + x_i \cdot c_i + y_i \cdot c_i$$

se fac minimizări de sus

Sinteza FAC

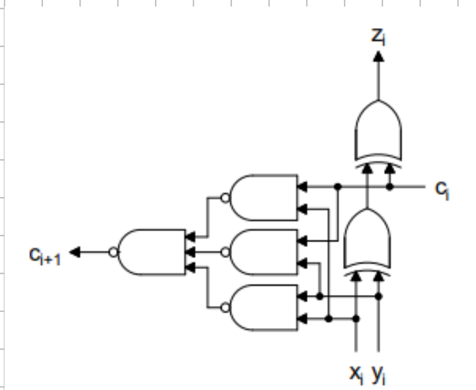
A) EXOR, AND, OR (cea mai folosită)



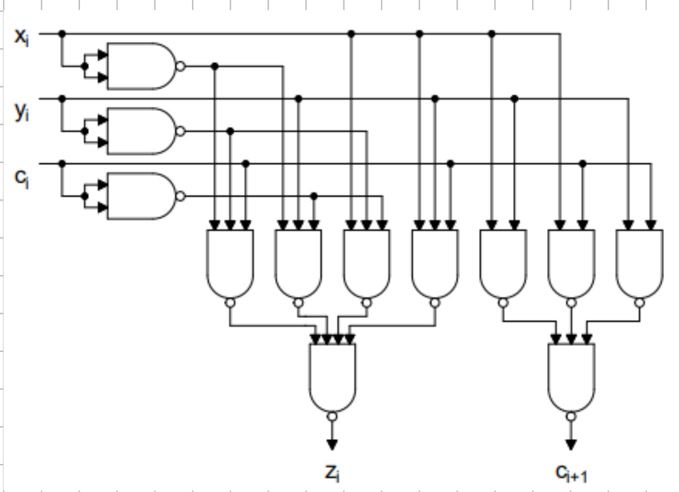
$$z_i = x_i \oplus y_i \oplus c_i$$
$$c_{i+1} = x_i \cdot y_i + x_i \cdot c_i + y_i \cdot c_i$$

B) EXOR, NAND

$\overline{\overline{c_{i+1}}}$ și așa obțin



C) NAND

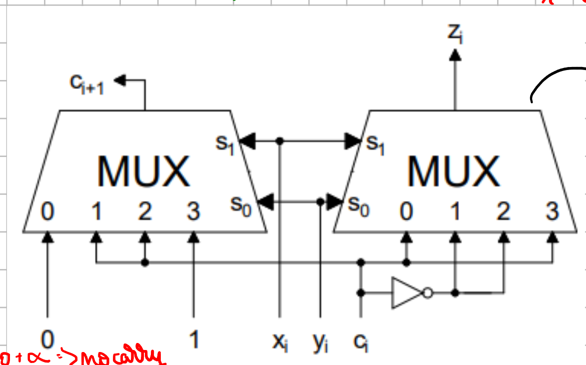


pt z_i → de la tabel val de „1" ca sumă de mintermi

D) Multiplexoare

$0 \oplus 1 = 1$
$1 \oplus \alpha = \bar{\alpha}$

$$z_i = x_i \oplus y_i \oplus c_i$$



$0+0+\alpha \Rightarrow$ no carry
$1+1+\alpha \Rightarrow$ carry

	x_i	y_i	z_i
0	0	0	c_i
1	0	1	$\overline{c_i}$
2	1	0	$\overline{c_i}$
3	1	1	c_i

$a \oplus b = a\bar{b} + \bar{a}b$