30.10.2024

Arhitectura calculatoarelor
(curs 5 - S5)

2.2.3. Scăzătoare bazate pe propagarea serială a transportului / împrumutului

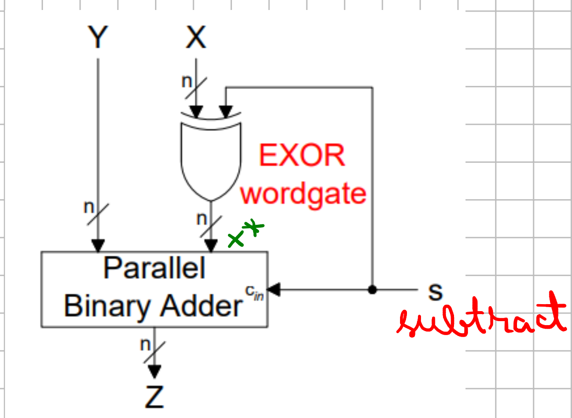
Operația de scădere

X	scăzător	$x_{n-1}x_{n-2}\dots x_1x_0$
Y	descăzut	$y_{n-1}y_{n-2}\dots y_1y_0$
Z	diferența	$Y-X$

Modalități de realizare a operației de scădere

(A) Utilizând sumatoare binare

$$Y-X = Y+(-X)$$



$$X \longrightarrow (-1)\cdot X$$
(EXOR wordgate)

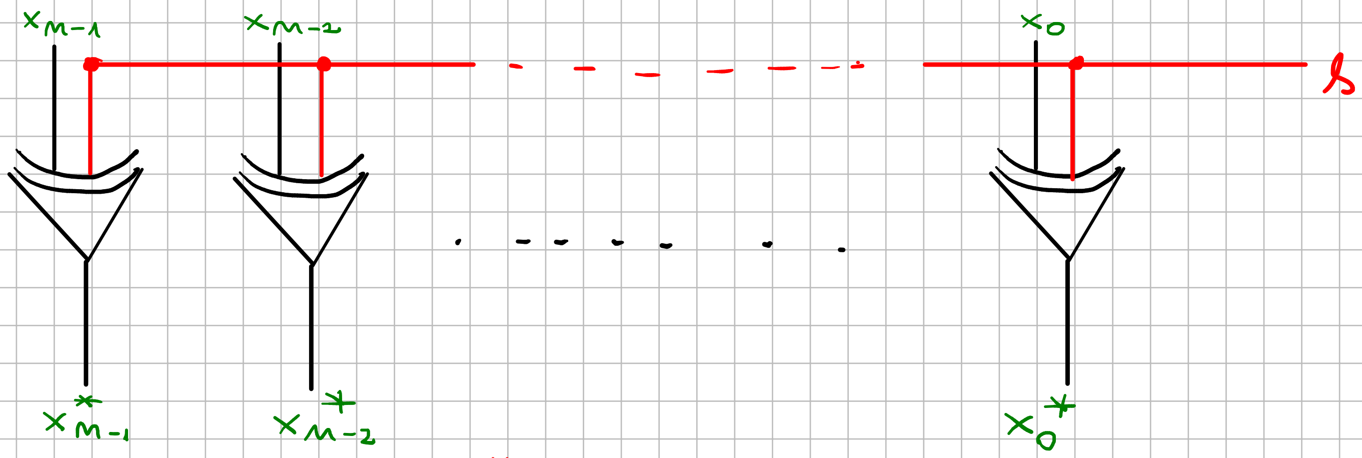
$\Rightarrow$ sumator / scăzător binar datorită s

s:

1: $Z = Y + \overline{X} + 1 = Y - X$ (x^*, c_1; $c_2 \Rightarrow$ −)

0: $Z = Y + X + 0 = Y + X$ (s)

Un wordgate se refera la o serie de porti EXOR (in cazul nostru) in care toate portile au o intrare comuna s in cazul nostru. Practic se aplica XOR intre toti bitii nostrii si s



$$s \begin{cases} 0: & x^* = x \\ 1: & x^* = \overline{x} \end{cases}$$

Este modalitatea de scadere preferata. Accentul cade pe sumatorul paralel binar, il pot transforma usor in scazator

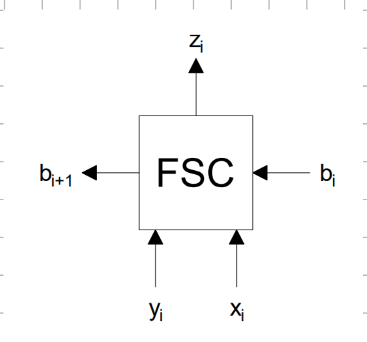
B. Scăzătoare dedicate

Utilizarea celulelor Full Subtracter Cells (FSCs):

- transportul este înlocuit de împrumut
- operație implementată:

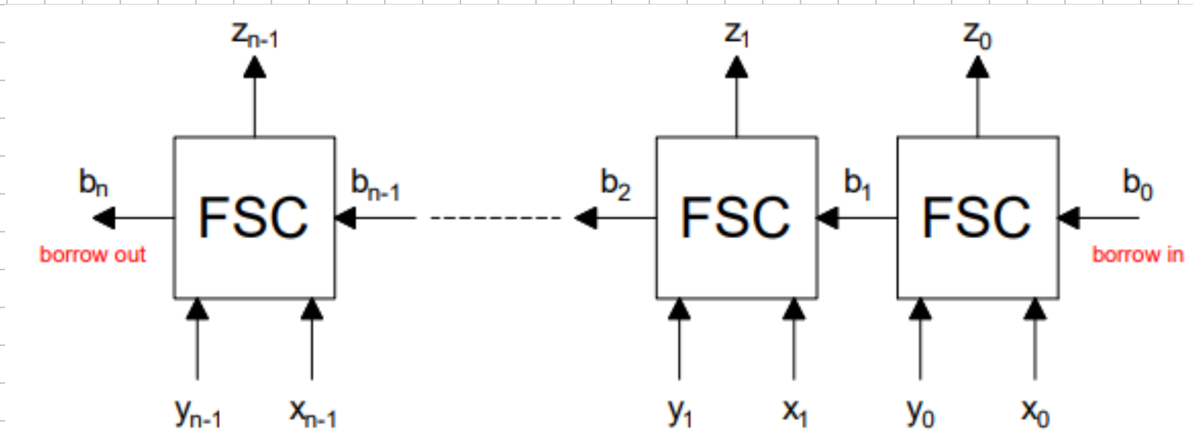
$$y_i - x_i - b_i \begin{cases} z_i \\ b_{i+1} \end{cases}$$

Nu e preferat in mod particular. Isi justifica investitia, doar daca operatia de scadere este atat de frecventa, incat plusul (avantajul) adaugarii unui scazator este mai mare decat costul. In principiu, la procesoarele de uz general nu se justifica.



b - borrow (împrumut)

Arhitectură scăzător pe n biți



Sinteză

⟶ tabel de adevăr

$y_i - x_i - b_i$

borrow_{i+1}

$$\begin{array}{r} 0 - \\ 1 \\ \hline 1 \end{array}$$

	Inputs			Outputs	
	y_i	x_i	b_i	z_i	b_{i+1}
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1

$0 - 0 - 1 = 0 - 1$

$0 - 1 - 1$

$$\begin{array}{r} 1000 - \\ 0001 \\ \hline 0111 \end{array}$$

2: $\begin{array}{r} 100 - \\ 001 \\ \hline 1 \end{array}$ z_i $\quad b_{i+1} = 1$

3. $\begin{array}{r} 100 - \\ 011 \\ \hline 01 \end{array}$ $z_i \; b_{i+1}$

⟶ Ecuațiile ieșirilor

z_i — Identică cu FAC

y_i \ x_i, b_i	00	01	11	10
0	0	1	0	1
1	1	0	1	0

checker-board

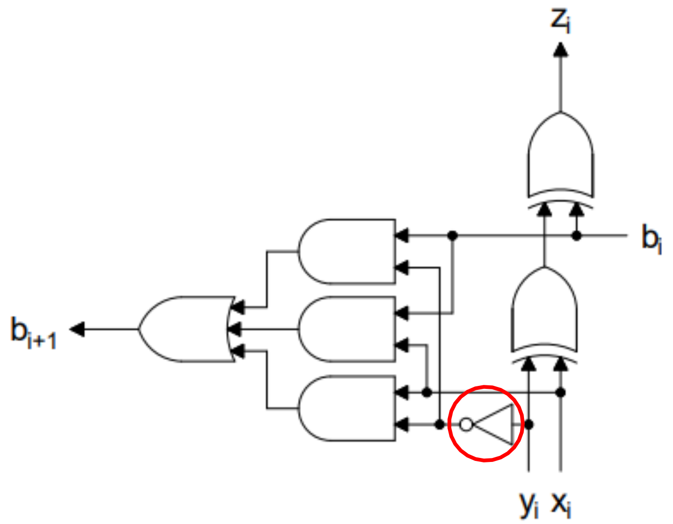
$\Rightarrow \quad z_i = y_i \oplus x_i \oplus b_i$

b_{i+1}

y_i \ x_i, b_i	00	01	11	10
0	0	1	1	1
1	0	0	1	0

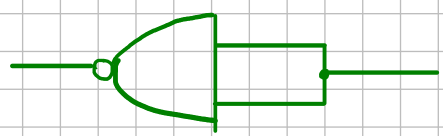
$b_{i+1} = \overline{y_i}\, b_i + \overline{y_i}\, x_i + b_i x_i$

Sinteza FSC, utilizând porți de tip EXOR, AND, OR, INV:



În afară de inversor <=> FAC

(B) INV cu NAND



(C) se neagă y_i la c_{i+1} MUX stânga

! DE FĂCUT

Ⓒ Scăzătoare BCD (se aplică și pt E3)

⟶ Se consideră:

$Y^{(k)}, X^{(k)}$ — 2 nr. BCD pe k cifre

$Y^{(k)} = y_{k-1}\, y_{k-2} \dots y_1\, y_0$

$X^{(k)} = x_{k-1}\, x_{k-2} \dots x_1\, x_0$

y_j și x_j – cifre BCD, $\forall\, j \in [0, k-1]$

$$Z^{(k)} = Y^{(k)} - X^{(k)}$$

Se definește complementul de 9 al unei cifre BCD, x_i, ca fiind:

$\overline{x_i^*} = 9 - x_i$ nu vom avea borrow

$\overline{X^*}^{(k)} = \overline{x_{k-1}^*}\ \overline{x_{k-2}^*} \dots \overline{x_0^*}$ Complementam fiecare cifra

$\underbrace{\qquad\qquad\qquad}_{<k> \text{ cifre}}$ ⟶ 4·k biți

$$= \boxed{\begin{array}{cccc} 9 & 9 & \dots & 9 \\ x_{k-1} & x_{k-2} & \dots & x_0 \end{array} \ -}$$

nu avem borrow

$$\Rightarrow \overline{X^*}^{(k)} = \underbrace{10^k - 1}_{(99\dots9)} - X^{(k)}$$

Diferenta poate fi rescrisa (fara a ma interesa daca trebuie sa fac imprumut din cel mai semnificativ rang - in principiu nici nu am de unde).

$$Z^{(k)} = (Y^{(k)} - X^{(k)}) \bmod 10^k =$$

ex: $k = 2 \longrightarrow 10^2 = 100$

$$\begin{array}{r} ab- \\ cd \\ \hline ef \end{array} \bmod 100 = ef$$

$a \bmod b = (a+b) \bmod b = (a-b) \bmod b$

$= (y^{(k)} + 10^k - x^{(k)}) \bmod 10^k =$

$= (y^{(k)} + \underline{10^k - 1 - x^{(k)}} + 1) \bmod 10^k =$

$\overline{x^{*}}^{(k)} =$ complementul de 9 al lui $x^{(k)}$

$\Rightarrow z^{(k)} = y^{(k)} + \overline{x^{*}}^{(k)} + 1$

Pot renunta la mod 10^k deoarece ne va pastra rezultatul asa cum am aratat mai sus

$\longrightarrow$ Proiectarea unui modul pentru determinarea complementului de 9 a unei cifre zecimale

$x_i = x_3 x_2 x_1 x_0 \longrightarrow$ cifră BCD de convertit

$\overline{x_i^*} = x_3^* x_2^* x_1^* x_0^* \longrightarrow$ complementul de 9 a lui x_i

unde $\overline{x_i^*} = 9 - x_i$

Tabel de adevăr al unitatii pentru calcularea complementului de 9

	Inputs				Outputs				
	x_3	x_2	x_1	x_0	x_3^*	x_2^*	x_1^*	x_0^*	
0	0	0	0	0	1	0	0	1	$= 9 - 0 = 9$
1	0	0	0	1	1	0	0	0	
2	0	0	1	0	0	1	1	1	
3	0	0	1	1	0	1	1	0	
4	0	1	0	0	0	1	0	1	
5	0	1	0	1	0	1	0	0	$= 9 - 5 = 4$
6	0	1	1	0	0	0	1	1	
7	0	1	1	1	0	0	1	0	
8	1	0	0	0	0	0	0	1	
9	1	0	0	1	0	0	0	0	
10					d	d	d	d	
⋮					-	-	-	-	
15					d	d	d	d	

Minimizare

x_3^* → Ne uităm pe coloana lui x_3^*

x_3x_2 \ x_1x_0	00	01	11	10
00	1 (0)	1 (1)	0 (3)	0 (2)
01	0 (4)	0 (5)	0 (7)	0 (6)
11	d (12)	d (13)	d (15)	d (14)
10	0 (8)	0 (9)	d (11)	d (10)

$$x_3^* = \overline{x_3} \cdot \overline{x_2} \cdot \overline{x_1} =$$
$$= \overline{x_3 + x_2 + x_1}$$

x_2^*

x_3x_2 \ x_1x_0	00	01	11	10
00	(0)	(1)	1 (3)	1 (2)
01	1 (4)	1 (5)	(7)	(6)
11	d (12)	d (13)	d (15)	d (14)
10	(8)	(9)	d (11)	d (10)

$$x_2^* = \overline{x_2}\, x_1 + x_2\, \overline{x_1} =$$
$$= x_2 \oplus x_1$$

Nu avem tipar checker-board, deoarece nu avem sau exclusiv intre toate variabilele de intrare

x_1^*

x_3x_2 \ x_1x_0	00	01	11	10
00	(0)	(1)	1 (3)	1 (2)
01	(4)	(5)	1 (7)	1 (6)
11	d (12)	d (13)	d (15)	d (14)
10	(8)	(9)	d (11)	d (10)

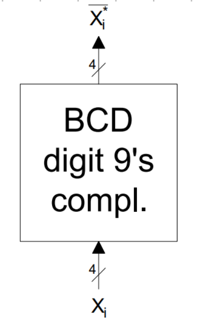
$$x_1^* = x_1$$

x_0^*

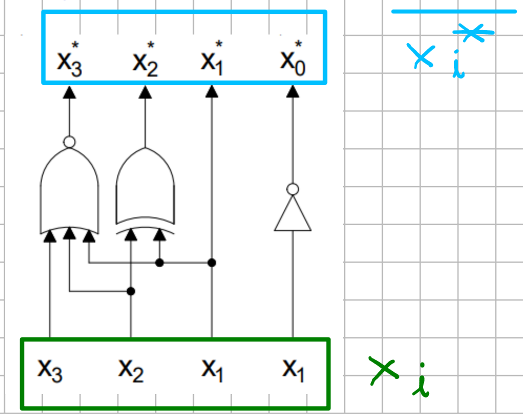
x_3x_2 \ x_1x_0	00	01	11	10
00	1 (0)	(1)	(3)	1 (2)
01	1 (4)	(5)	(7)	1 (6)
11	d (12)	d (13)	d (15)	d (14)
10	1 (8)	(9)	d (11)	d (10)

$$x_0^* = \overline{x_0}$$

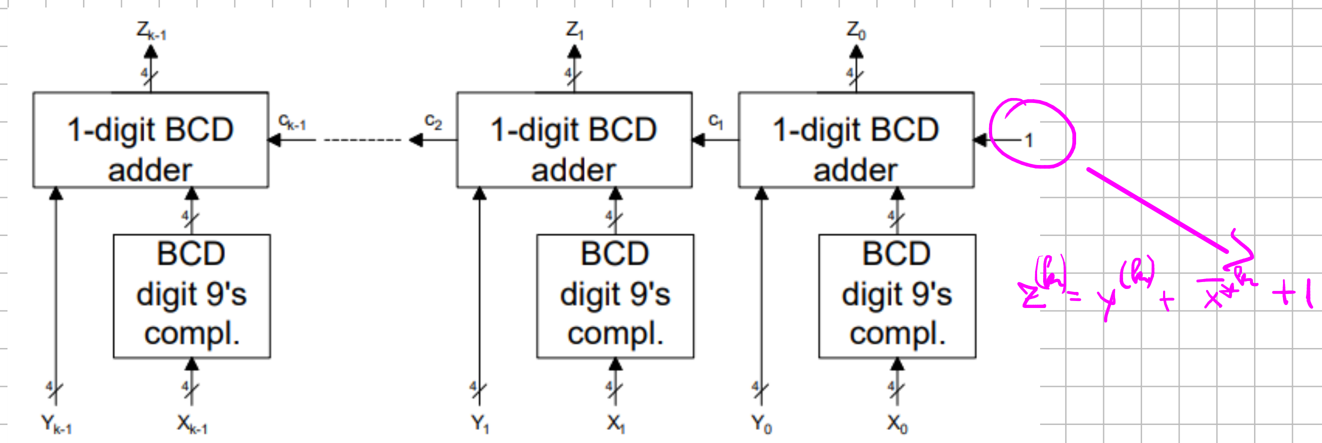
Simbol :



Arhitectura



Arhitectura unui scăzător pentru numere BCD pe k-cifre:



Avem nevoie de un sumator BCD, deoarece pt fiecare complement de 9 de la cifrele lui x adun cifra corespunzatoare a lui y. Transportul este ignorat datorita mod 10^k.

Ex:

$$\begin{array}{r} 803\ - \\ 279 \\ \hline 524 \end{array}$$

9's complement

$$\begin{array}{r} 999\ - \\ 279 \\ \hline 720 \end{array} \Rightarrow \begin{array}{r} 803\ + \\ 720 \\ \hline (1)\,523\ + \\ 1 \\ \hline 524 \end{array}$$

nu ne interesează

$Y^{(k)}$
$\overline{X}^{*(k)}$
1
$\oplus$

2.3. Calculul paralel al sumei

2.3.1. Sumator Carry Lookahead

→ Un sumator Carry Lookahead complet (F-CLA) este caracterizat de ecuatia : (F → Full)

$c_{i+1} = x_i y_i + c_i (x_i + y_i)$

$g_i = x_i \cdot y_i$ → variabilă generate (de generare)

$p_i = x_i + y_i$ → variabilă propagate (de propagare)

provenită de la FAC, doar s-a dat factor comun

$\Rightarrow c_{i+1} = g_i + p_i \cdot c_i$

1, dacă $g_i == 1$ $(x_i = y_i = 1)$

1, dacă $c_i \cdot p_i == 1$ ($c_i == 1$ și $p_i == 1$ $\Rightarrow x_i$ sau $y_i = 1$)

Pe baza recursivității :

$$\begin{aligned} c_{i+1} &= g_i + p_i \cdot c_i \qquad c_i = g_{i-1} + p_{i-1} \cdot c_{i-1} \\ &= g_i + p_i \cdot g_{i-1} + p_i \cdot p_{i-1} \cdot c_{i-1} \\ &= \dots \\ &\boxed{= g_i + p_i \cdot g_{i-1} + \dots + p_i \cdot p_{i-1} \cdots p_1 \cdot g_0 + p_i \cdot p_{i-1} \cdots p_0 \cdot c_0} \end{aligned}$$

→ are 2 nivele logice (AND, OR)

Dezavantaje:
- fan-out ridicat: p_i este utilizat de $i+1$ termeni
- fan-in ridicat: c_{i+1} are $i+2$ termeni

⇒ Sumatoarele F-CLA operează numere de lățime redusă

$z_{i+1} = x_{i+1} \oplus y_{i+1} \oplus c_{i+1}$ sunt biti de la intrare

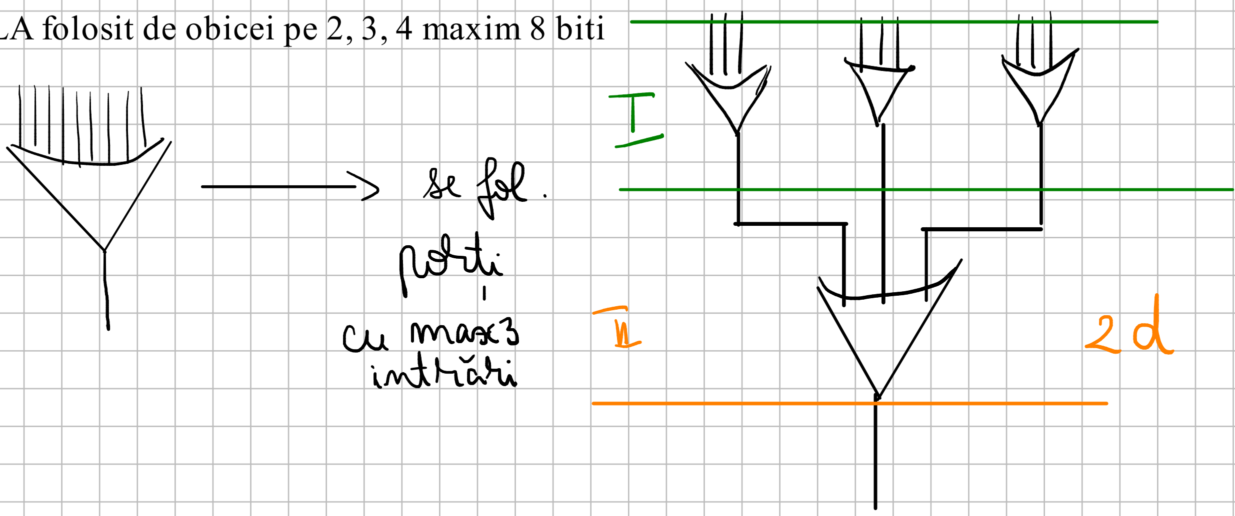
c_{i+1} → are latență de $+2d$

fan out ridicat : Cu cat o linie este folosita de mai multe componente, cu atat este mai incarcata si va fi afectata de o putere consumata crescuta, care are efect asupra intarzierii

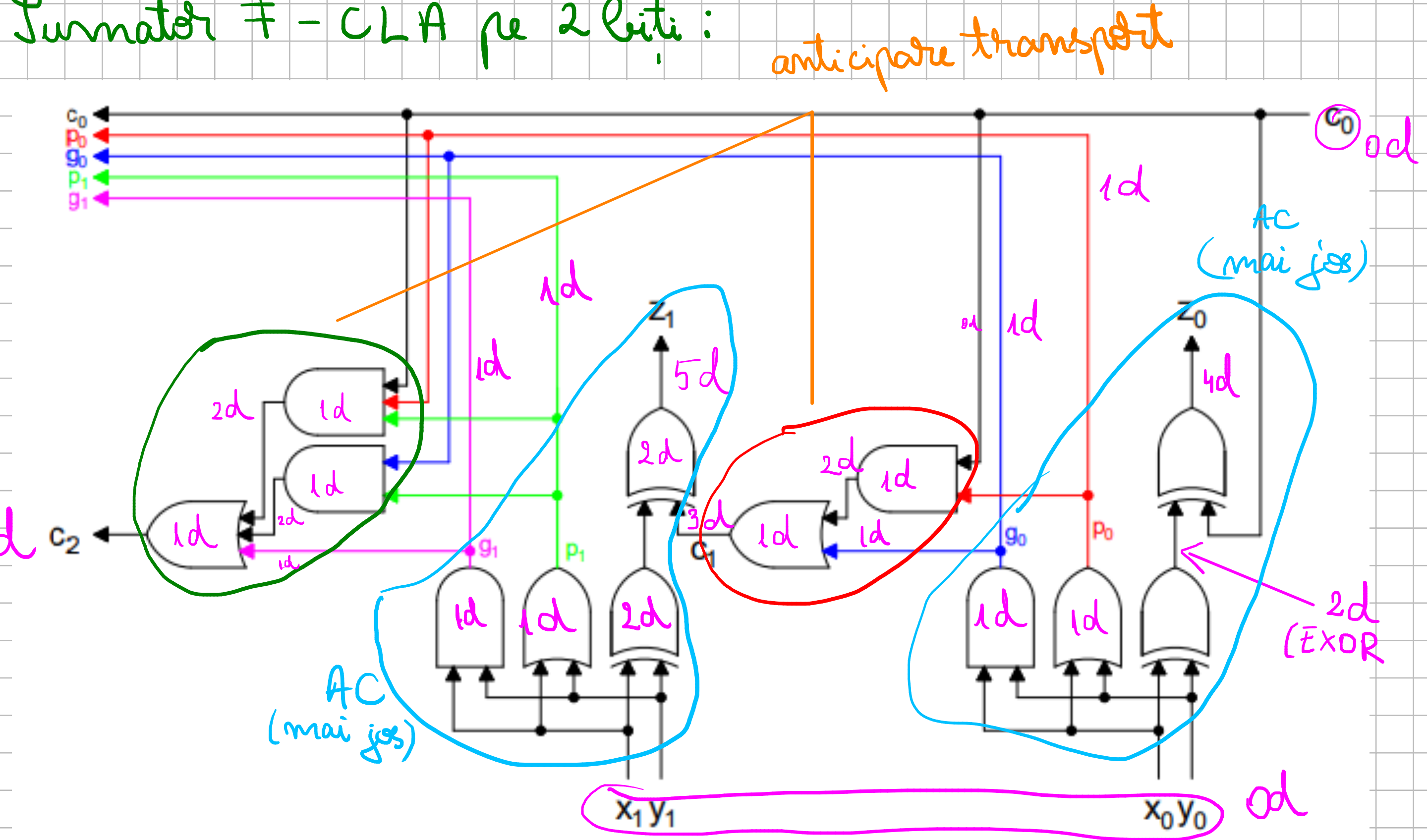
=> caracteristica la nivel electronic

fan in ridicat : Se refera la nr de intrari ale unei porti, ultimul termen are nr maxim de factori
c(i + 1) are i + 2 termeni

F-CLA folosit de obicei pe 2, 3, 4 maxim 8 biti



Sumator F-CLA pe 2 biți :



Pornește de la scrierea lui $c_{i+1} = g_i + p_i c_i$

$p_i \longrightarrow x_i + y_i = 1 \quad (x_i \oplus y_i = 1)$

x_i	y_i
0	1
1	0

în cazul $x_i = y_i = 1 \Rightarrow p_i = 1$

mai imp. $\Rightarrow g_i = 1$

$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

Întârzierea F-CLA pe n biți: $n \in [0,4]$ 8 (caz extrem)

$$\delta^{C_{out}}_{F-CLA} = 3d$$

$$\delta^{Z}_{F-CLA} = 5d$$

$$z_i = \underbrace{(x_i \oplus y_i)}_{2d} \oplus c_i \Rightarrow \delta_{z_i} = \max(2d, \underbrace{\delta_{c_i}}_{\text{mereu } 3d}) + 2d = 3d + 2d = 5d$$

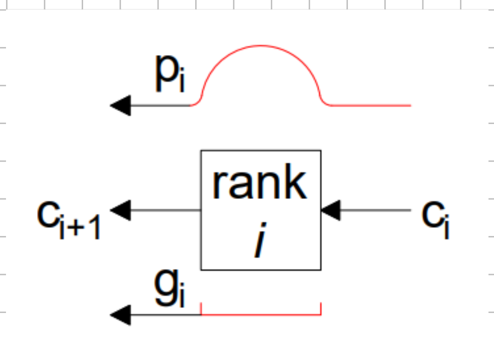
Acestea sunt rezultate teoretice, dar fizic nu putem suporta o astfel de implementare.

Datorita constrangerii valorilor lui n, se pune problema a cum se poate pastra acest principiu de pastrare a transportului

Ecuația $c_{i+1} = g_i + p_i \cdot c_i$ poate fi interpretată prin prisma celor 2 variabile, g_i și p_i astfel:

- g_i : transportul este generat "în" rangul i
- p_i : transportul este propagat "peste" rangul i

Simbolizare grafică a celor 2 ecuații



gi ne asigura ca in rangul acesta, independent de ce se intampla inainte, in rangul i se genereaza un transport de iesire

p_i ne arata ca daca exista transport de intrare, acesta va fi propagat peste rangul i direct la (i + 1)

Ex:

Se consideră transportul c_4 și modalitatea recursivă de exprimare a acestuia

Folosim $c_{i+1} = g_i + p_i \cdot c_i$

$$c_4 = g_3 + p_3 c_3 =$$

$$= g_3 + p_3 (g_2 + p_2 c_2) = g_3 + p_3 g_2 + p_3 p_2 c_2 =$$

$$= g_3 + p_3 g_2 + p_3 p_2 (g_1 + p_1 c_1) =$$

$$= g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 c_1 =$$

$$= g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 (g_0 + p_0 c_0) =$$

$$= \underbrace{g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0}_{G(0,3)} + \underbrace{p_3 p_2 p_1 p_0}_{P(0,3)} c_0$$

$\cancel{c_0}$

$P(0,3)$ – variabila de propagare la nivel de bloc pt. rangurile între 0 și 3

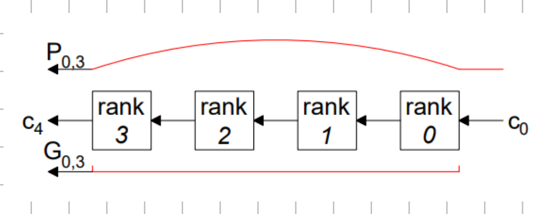
$G(0,3)$ – variabila de generare la nivel de bloc pt. rangurile între 0 și 3

$\Rightarrow c_4 = G_{0,3} + P_{0,3} \cdot c_0$

$G_{0,3}$ și $P_{0,3}$ se numesc variabile de generare, respectiv propagare, la nivel de bloc, având semnificațiile următoare:

- $G_{0,3}$ indică faptul că transportul este generat în blocul de ranguri de la 0 până la 3, inclusiv
- $P_{0,3}$ indică faptul că transportul este propagat peste blocul de ranguri de la 0 până la 3, inclusiv

Simbolizare grafică



Pe de altă parte, expresia extinsă a lui c_4 poate fi grupată ca în ecuațiile de mai jos:

$$\begin{aligned} c_4 &= g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0 + p_3 \cdot p_2 \cdot p_1 \cdot p_0 \cdot c_0 \\ &= \underbrace{\underbrace{g_3 + p_3 \cdot g_2}_{G_{2,3}} + \underbrace{p_3 \cdot p_2}_{P_{2,3}} \cdot \underbrace{(g_1 + p_1 \cdot g_0)}_{G_{0,1}}}_{G_{0,3}} + \underbrace{\underbrace{p_3 \cdot p_2}_{P_{2,3}} \cdot \underbrace{p_1 \cdot p_0}_{P_{0,1}}}_{P_{0,3}} \cdot c_0 \end{aligned}$$

Variabile de generare/propagare la nivelul unui bloc de ranguri pot fi exprimate in termenii variabilelor de propagare/generare la nivel de sub-bloc, ca in expresiile de mai jos:

I $G_{0,3} = G_{2,3} + P_{2,3} \cdot G_{0,1}$

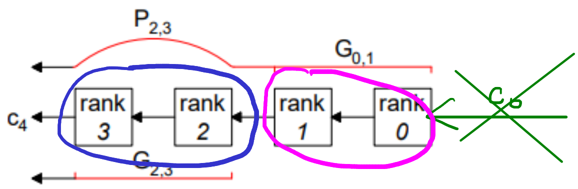
II. $P_{0,3} = P_{2,3} \cdot P_{0,1}$

I. indică faptul că pentru a fi generat în blocul de ranguri de la 0 până la 3, transportul fie:

- este generat în blocul de ranguri de la 2 până la 3,
- fie este generat în blocul de ranguri de la 0 până la 1 ȘI este propagat peste rangurile de la 2 până la 3

Generarea transportului în blocul de ranguri de la 0 până la 3 este simbolizată grafic astfel:

nu contează c_0

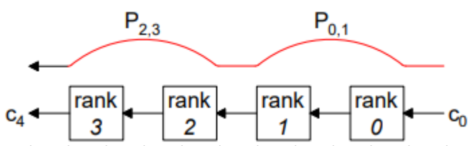


fie 0 fie 0 (mai puțin semnificativ, dar propagat peste $G_{2,3}$)

II. indică faptul că pentru a fi propagat peste blocul de ranguri de la 0 până la 3, transportul :

- trebuie să fie propagat peste rangurile de la 2 până la 3, ȘI
- trebuie să fie propagat peste ranguri de la 0 până la 1

Propagarea transportului peste blocul de ranguri de la 0 până la 3 este simbolizată grafic astfel:



Dacă nu am oricare dintre condiții de la subbloc, nu pot propaga peste toate cele 4 ranguri.

Notații

Până acum: $G_{i,j}$ $j > i$ ex: $G_{0,1}$; $G_{2,3}$; $G_{0,3}$

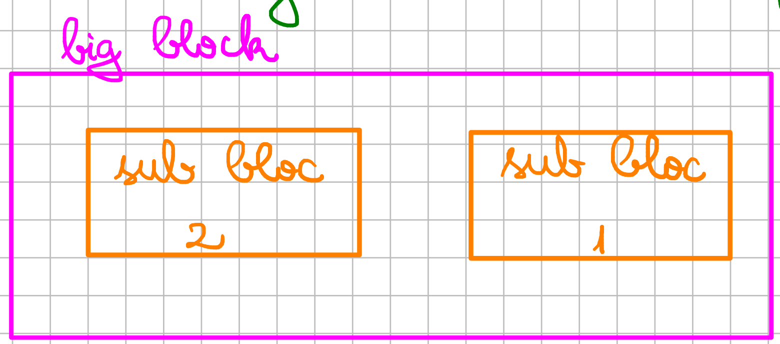
$\boxed{\text{rank } i} \Rightarrow G_{i,i} = g_i$

$$\Rightarrow \left| \begin{aligned} G_{i,i} &= g_i = x_i \cdot y_i \\ P_{i,i} &= p_i = x_i + y_i \end{aligned} \right. \quad , \forall \text{ rang } i$$

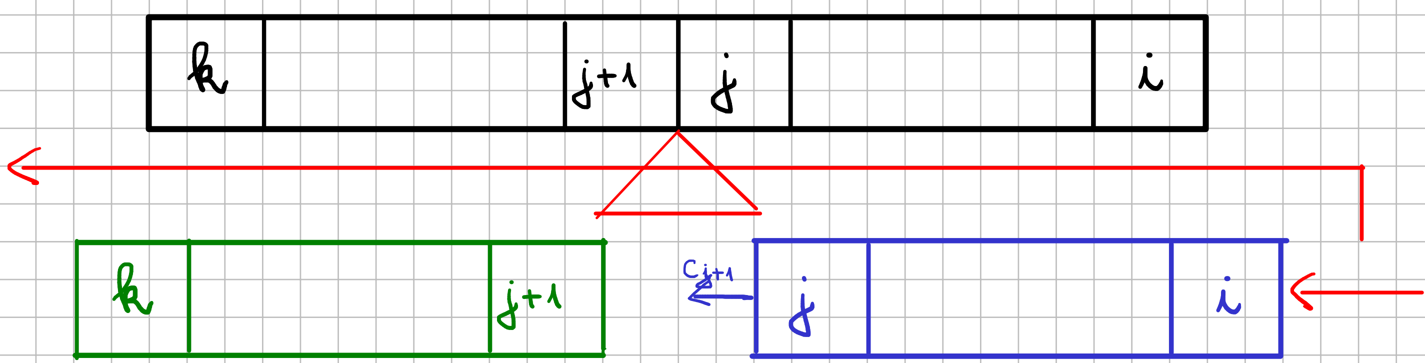
In general, transportul ımpreuna cu variabilele de generare/propagare la nivel de bloc pot fi exprimate astfel:

$$\left| \begin{aligned} c_{j+1} &= G_{i,j} + P_{i,j} \cdot c_i && , \forall\, i \le j \\ G_{i,k} &= G_{j+1,k} + P_{j+1,k} \cdot G_{i,j} && , \forall\, i \le j < k \\ P_{i,k} &= P_{j+1,k} \cdot P_{i,j} && , \forall\, i \le j < k \end{aligned} \right.$$

Ultimele 2 asigură modul în care pot compune 2 subblocuri.



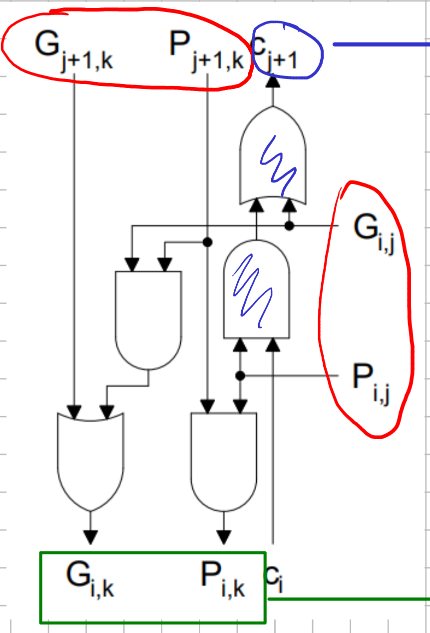
Ecuatia 3 : nu pot propaga peste intreg blocul daca nu pot propaga peste sub blocuri:



Fie generez in cel mai semnificativ sub bloc c(i + 1), fie in celalalt si il propag direct la iesire

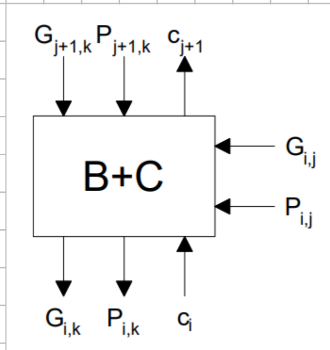
Implementare de tip B+C

intrări $\rightarrow 2d$

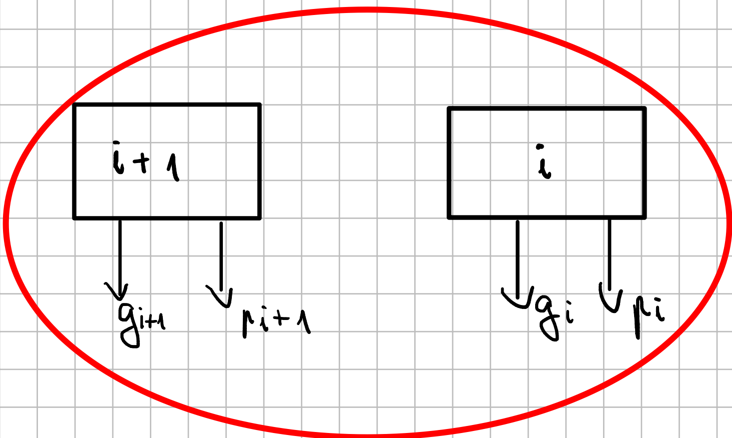


$\rightarrow +2d$ le voi considera mereu

Simbol:

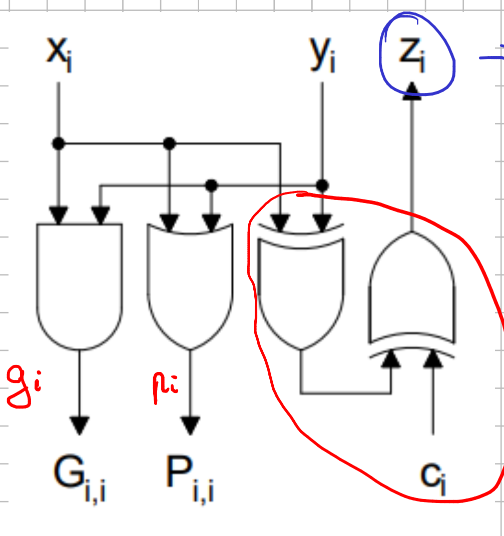


Semnalele $G_{i,i}$ și $P_{i,i}$ (pe baza cărora se vor construi semnalele de generare/propagare la nivel de bloc) împreună cu bitul sumei z_i, vor fi generați de celule de tip AC de mai jos:



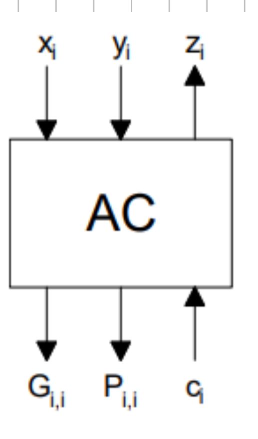
$\Rightarrow$ blocuri mai mari

Adder Cell

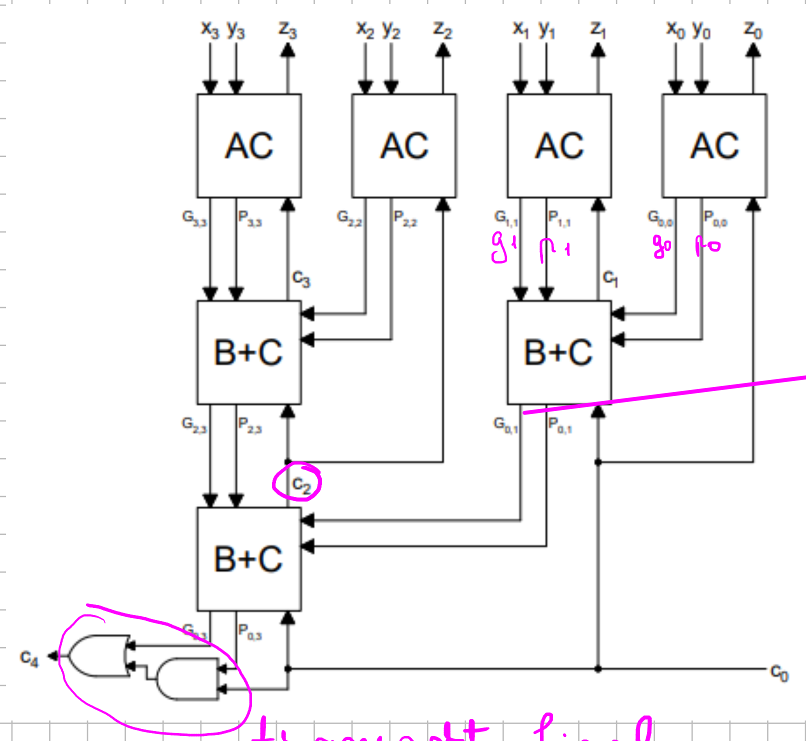


$\rightarrow +2$

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



Arhitectura CLA multinivel pe 4 biți



$\rightarrow G_{0,1} = G_{1,1} + P_{1,1} \cdot G_{0,0}$

$= g_1 + p_1 \cdot g_0$

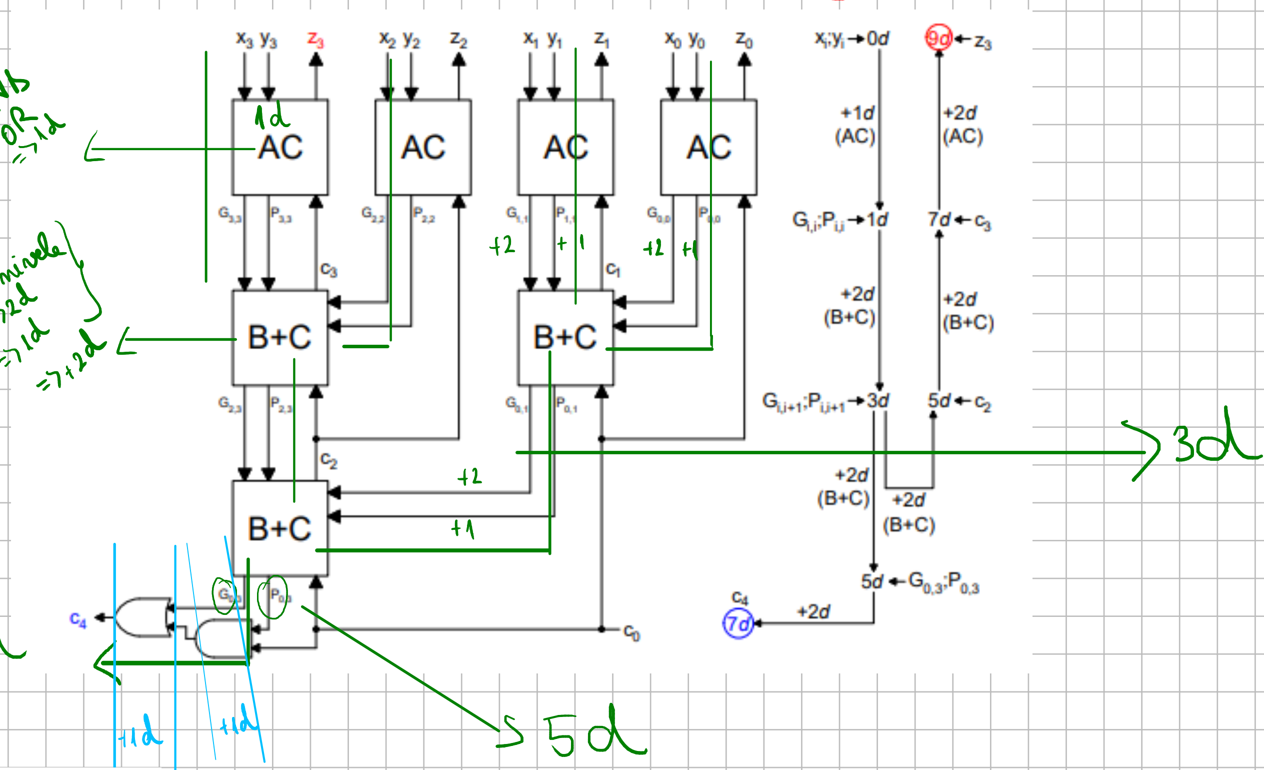
$c_2 = G_{0,1} + P_{0,1} \cdot c_0 \Rightarrow z_2$

$c_3 = G_{2,2} + P_{2,2} \cdot c_2 \Rightarrow z_3$

transport final

c_0 sare până la rangul cel mai semnificativ

4 nivele: 1 nivel AC; 2 nivele B+C ($\log_2 4$)



porți AND + OR => 1d

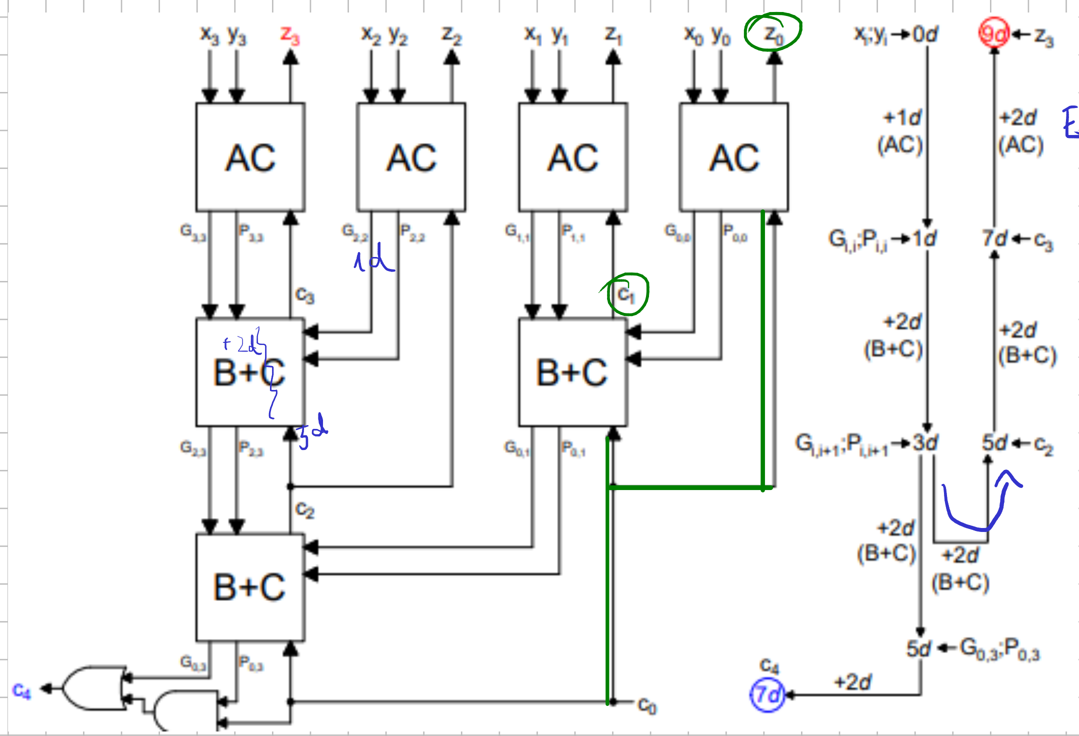
$G_{i,k}$ - 2 nivele => 2d; $P_{i,k}$ => 1d => +2d

$D^{Cout}_{ML-CLA-4} = 7d$
$D^{z}_{ML-CLA-4} = 9d$

Se consideră un sumator CLA multinivel pe n biți. Pentru o astfel de arhitectură, există $\lceil log_2 n \rceil$ nivele de celule $B + C$.

n biți => 2^n

În general:
$D^{Cout}_{ML-CLA-n} = (2\lceil log_2 n \rceil + 3)d$
$D^{z}_{ML-CLA-n} = (4\lceil log_2 n \rceil + 1)d$



EXOR pt z_3

față de c_0

→ cel mai rpd se calculează z_0

→ pt z_1 avem nev de c_1 + întârziere B+C (2d)

→ pt. z_2 avem nev de c_2 + întârziere B+C (2d)

c_1 și c_2 nu se obțin simultan, deoarece c_1 fol. $G_{0,0}$, în timp ce c_2 utilizează $G_{0,1}$ (provenit din $G_{0,0}$)

→ pt z_3 avem nev de c_3, care se obține cu ajutorul lui c_2 + întârzieri

⇒ z_3 se obține cel mai lent

La nivel general pe n biți

$n = 2^k$ ⇒ k nivele B+C

Pentru C_{out}: AC —— 1 nivel

B+C —— k nivele

La final am: $G_{0,n-1}$; $P_{0,n-1}$ + încă 2 nivele logice

$$\Rightarrow D^{C_{out}}_{ML-CLA-N} = \underbrace{1d}_{(AC)} + \underbrace{k \cdot 2d}_{(B+C)} + \underbrace{2d}_{(\text{ultimele 2 nivele})} = 2k+3d =$$

$= 2\log_2 n + 3d$

Dacă n nu e o putere a lui 2 $\longrightarrow$ ceiling (aprox $\nearrow$)

$\lceil \log_2 7 \rceil = 3$

$\lceil \log_2 5 \rceil = 3$

ex:

7 : AC — 1 nivel

$G_{0,0}$; $P_{0,0}$

B+C — $(k-1)$ nivele

propag în sus

B+C — k nivele

AC — 1 nivel EXOR

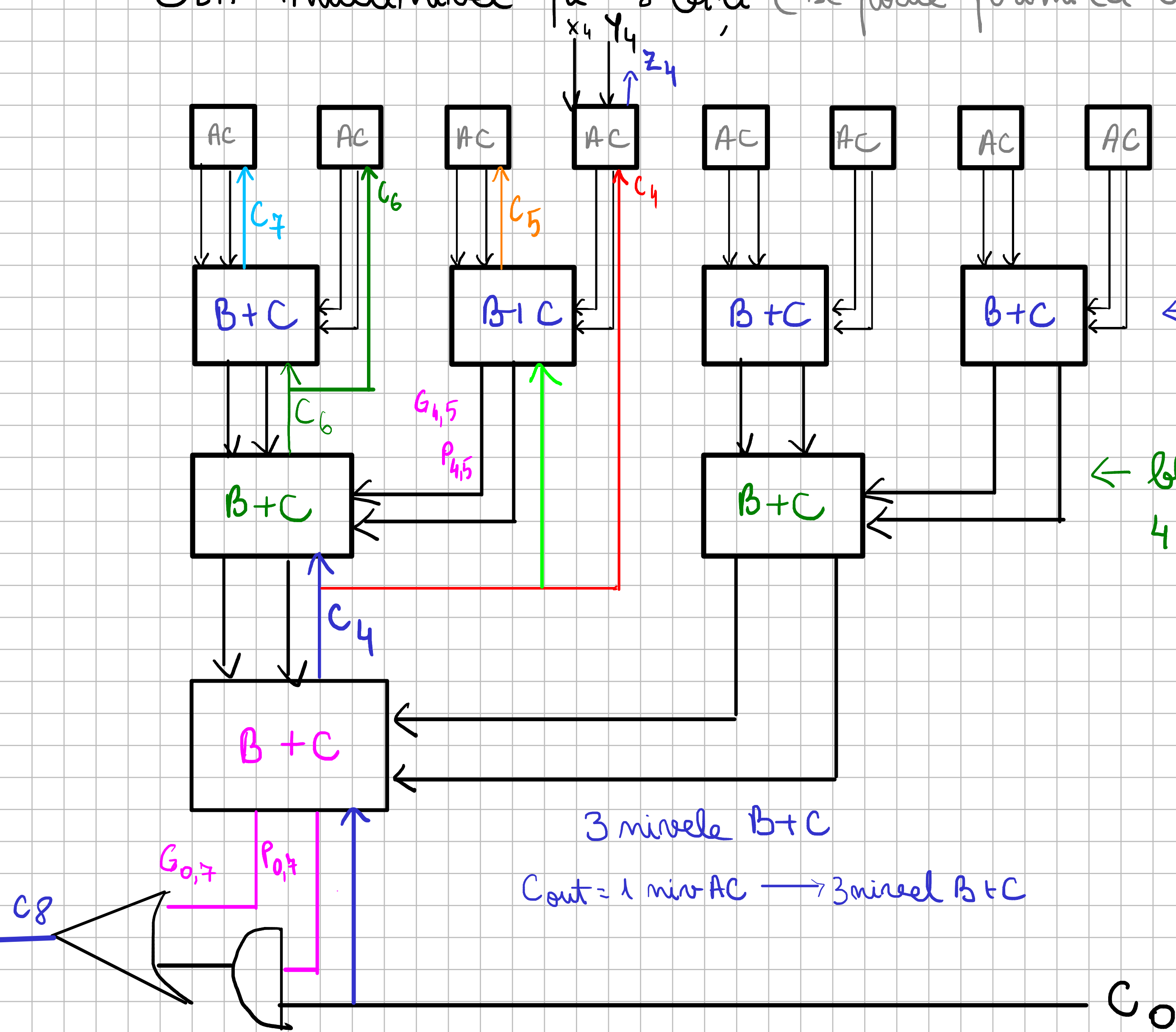
$$\delta^2_{ML-CLA} = 1d + 2(k-1)d + k \cdot 2d + 2d =$$

$$= 3d + 2kd - 2d + 2kd =$$

$$= 4kd + 1d =$$

$$= (4\log_2 4 + 1)\, d$$

CLA multinivel pe 8 biți (se poate primi la examen)



⟶ Din ultimul bloc B+C ne vom determina, cu ajutorul lui c_0 C_8

⟶ Tot cu ajutorul lui c_0, care intră în B+C, vom determina C_4

⟶ C_4 va fi folosit și în calcularea lui z_4

⟶ C_6 va fi obținut din blocul în care intră $G_{4,5}$, $P_{4,5}$, respectiv G , P

⟶ C_7 va fi obținut din blocul în care intră $G_{7,7}$; $P_{7,7}$; $G_{6,6}$; $P_{6,6}$ + C_6

ETC (de continuat) ! nivele B+C : $\log_2 8 = 3$

Dacă era pe 7 biți scăpam de prima celulă AC și de prima celulă B+C de la nivelul 1 B+C

Celula 1 B+C de la nivelul 2 B+C preia ieșirea de la noua primă celulă AC.

Dacă era sumator pe 6 biți ⇒ dispar primele 2 celule AC, precum și primele 2 celule B+C de sub el.

Ex curs : Sumator pe 5 biți