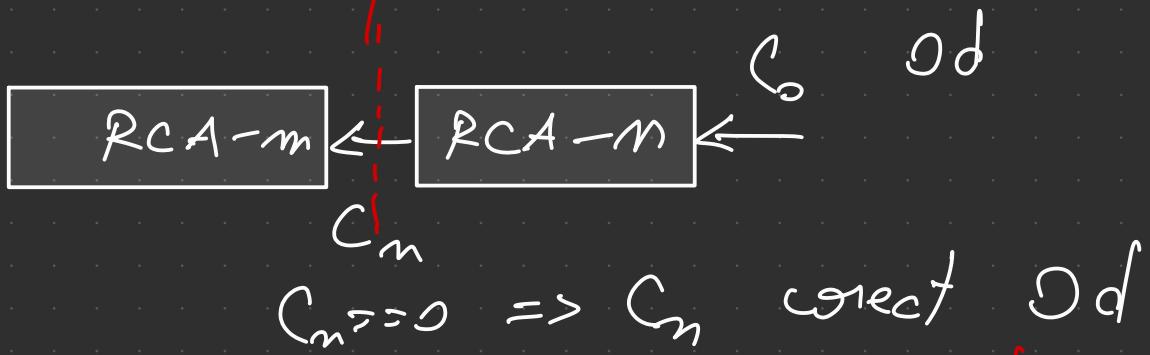


CMOS Pre-discharge  $\rightarrow$  Initialize to 0

CSKA Carry Skip Adder  
 $\rightarrow$  transportable sum pre-discharged

$n + m$  bits

parallel



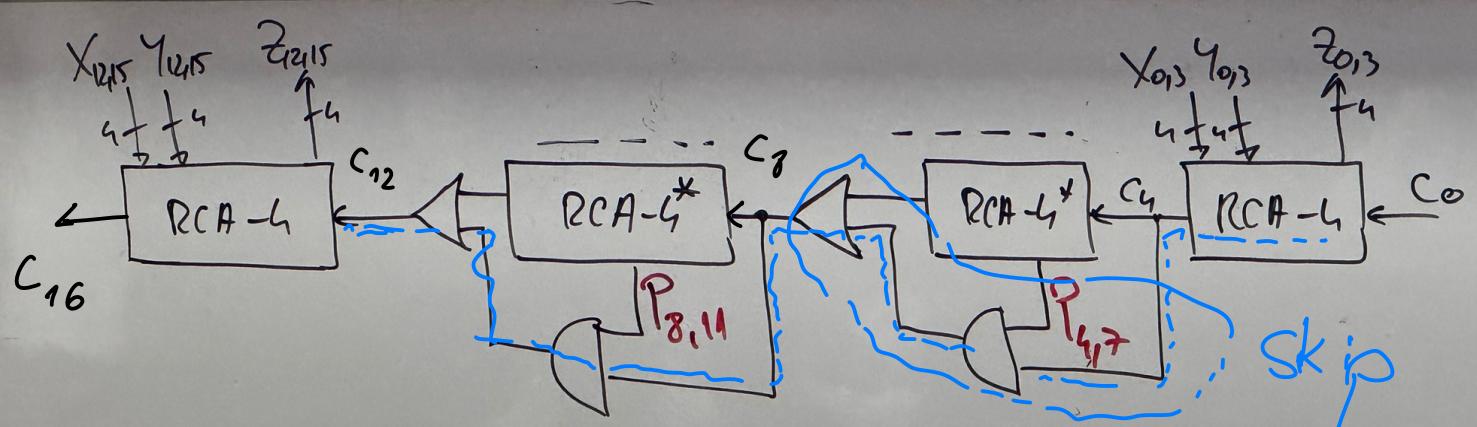
$$C_m = 0 \Rightarrow C_n \text{ correct } Cl$$

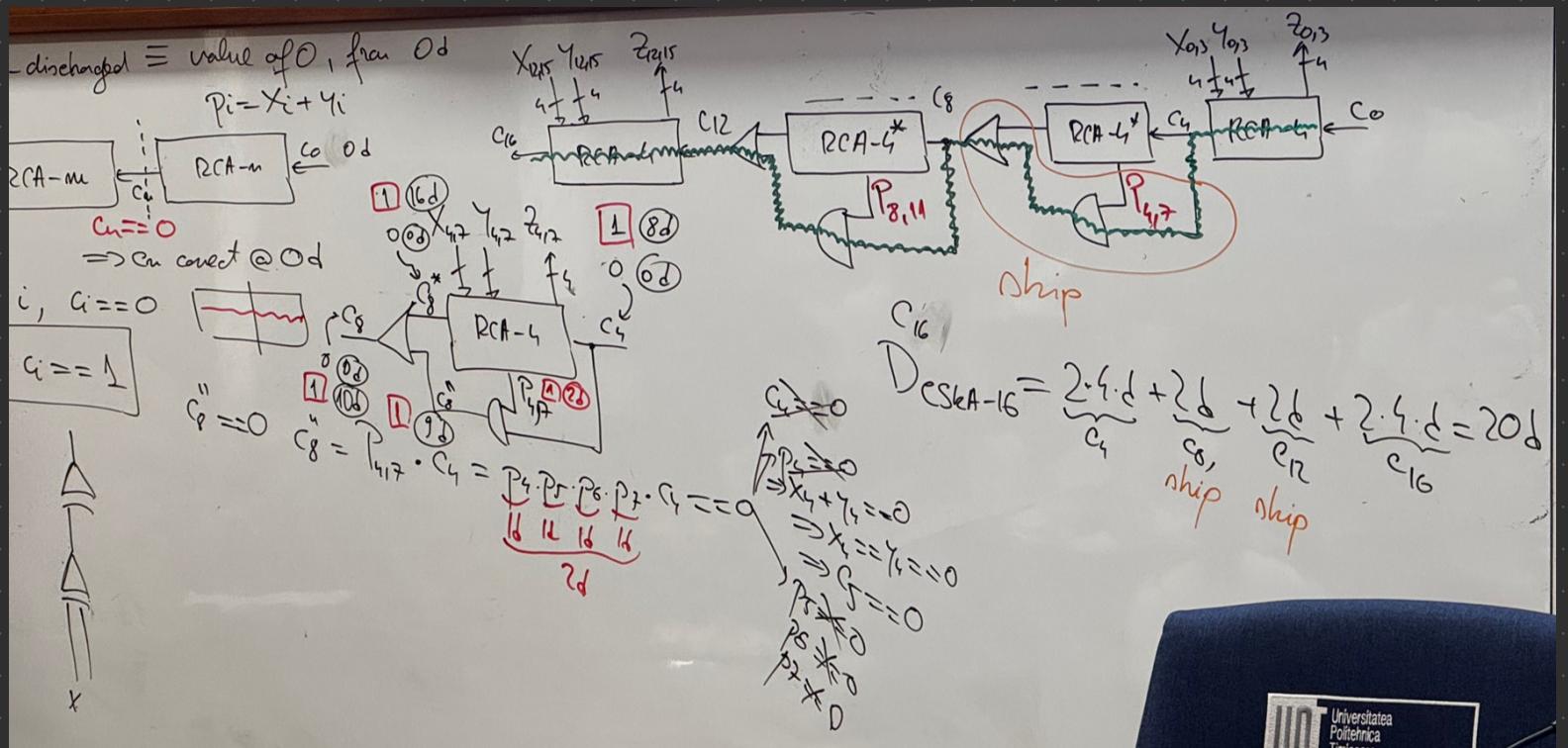
Colea critica RCA

( $\exists$ )  $i$   $C_i = 0$



( $\forall$ )  $C_i = 1$  Worst Case  
RCA  $\star$   $\rightarrow$  one regine propagate





Lă-timea optimă pf. RCA

$$M = b * k$$

RCA  
segment  
width

first  
last

$$\frac{M}{b} \quad \text{RCA} \quad \leftarrow \frac{n}{b} - 2 \quad \text{width skip}$$

$$\begin{aligned}
 & \text{Z/cout} \\
 & D_{CSkA-m} = 2bd + 2\left(\frac{n}{b} - 2\right)b + 2bd = \\
 & = \left(3b + \frac{2n}{b} - 4\right)d
 \end{aligned}$$

$$\frac{d}{d b} \Delta_{CSkA-m} = h - \frac{2^m}{b^2} = 0$$

$$b = \sqrt{\frac{m}{2}}$$

$$n = 32 \rightarrow \underline{\underline{b=4}} \quad D_{opt} = 28d$$

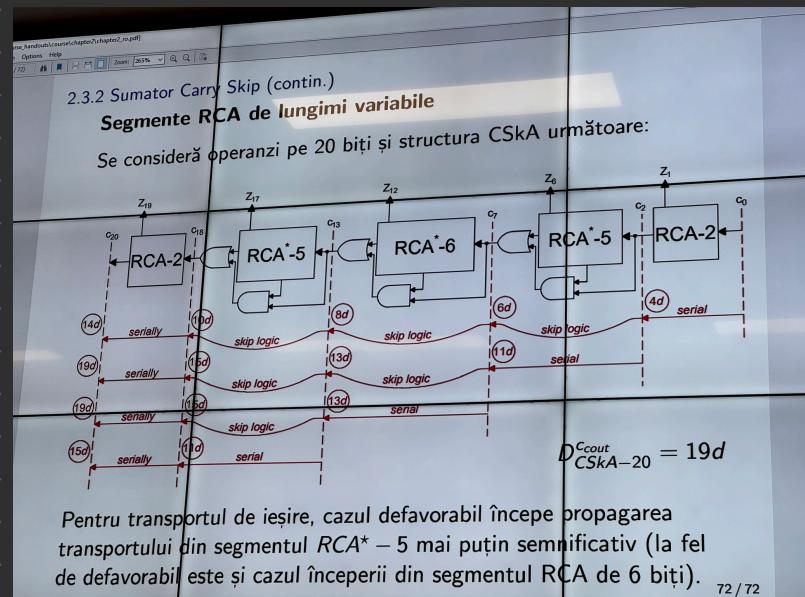
$$m = 128 \rightarrow \underline{\underline{b=8}} \quad D_{opt} = 60d$$

$$D_{opt} = 4(\sqrt{2m} - 1)d$$

$$RCA_{128} = \delta = 256d$$

Variable-sized RCA segments

Net. Patenta Cout  
EXAMEN



bitul transport

nodes ff

bitul sumă

# Carry Select Adder

$CS_e A$

$\rightarrow$  principiu sumei conditionate prin transport

$$C_i \begin{cases} 0 \Rightarrow Z_i & C_{i+1} \text{ if } C = 0 \\ 1 \Rightarrow Z_i & C_{i+1} \text{ if } C = 1 \end{cases}$$

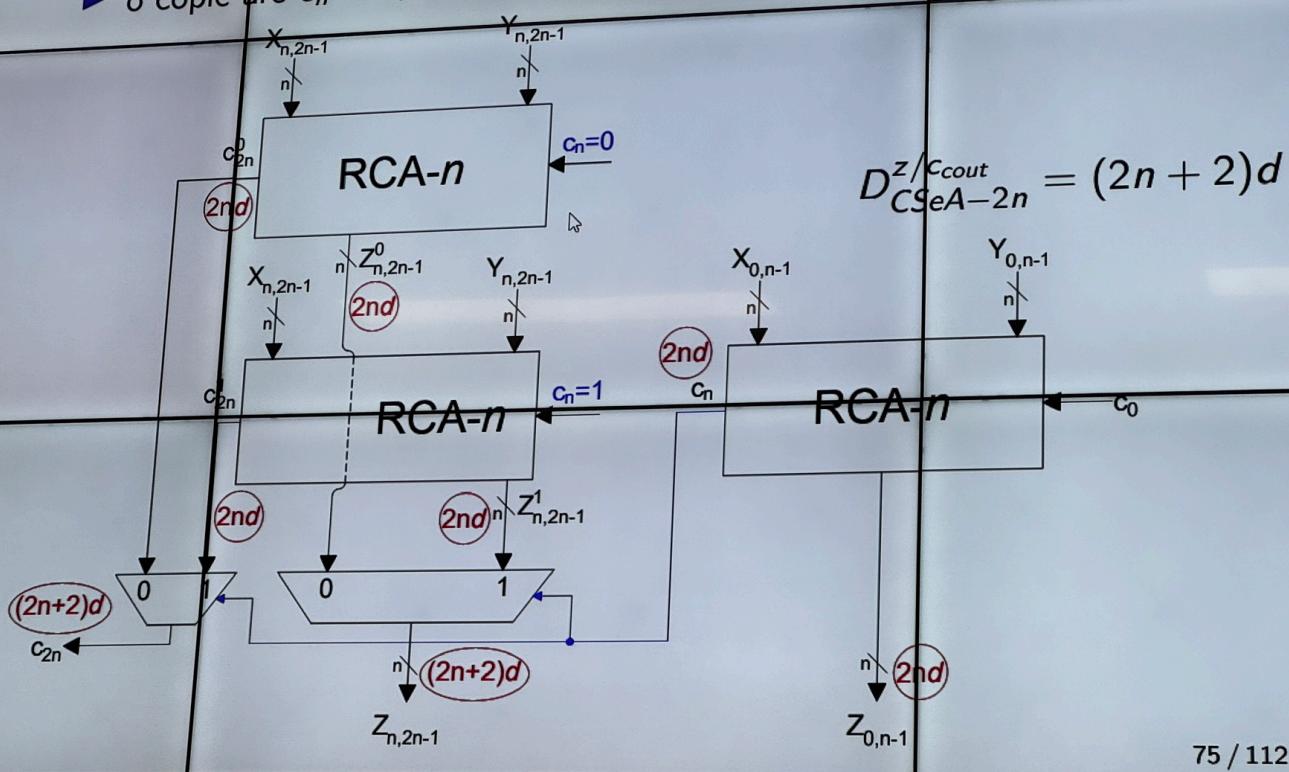
$RCA - 2n$

$$S_{RCA-2n} = 4n \text{ d}$$

### 2.3.3 Carry Select Adder (contin.)

Construcția CSeA:

- se împarte sumatorul  $RCA-2n$  în două jumătăți egale
- se duplică jumătatea mai semnificativă
- o copie are  $c_n = 0$ , celalată are  $c_n = 1$

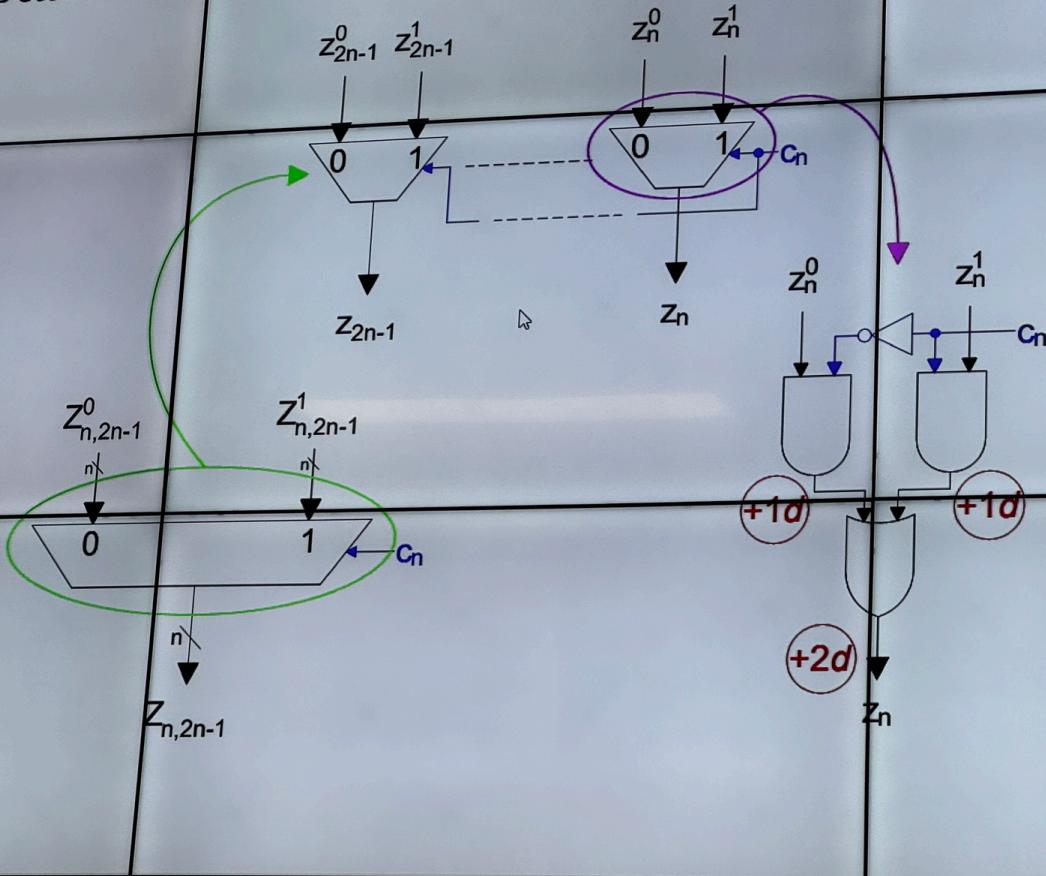


75 / 112

$\leftarrow$  carry 1 pt  $n; 2n-1 \rightarrow$  mux la final  
 carry 0

### 2.3.3 Carry Select Adder (contin.)

Detalii întârziere multiplexor:



76 / 112

$$f^{2 \text{ fent}}_{CS_e A - 2n} = (2n + 2)d$$

↙ duplicate hardware + cost energie

### 2.3.3 Carry Select Adder (contin.)

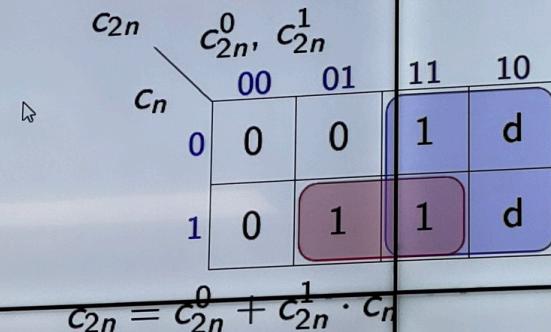
Optimizarea ariei CSeA:

- se pornește de la tabelul de adevăr al semnalului  $c_{2n}$
- Niciodată  $c_{2n}^0$  nu poate fi mai mare decât  $c_{2n}^1$ !

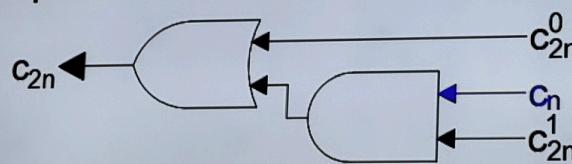
Întrebare: de ce?

~~tabelul de adevăr folosește don't care pentru cazurile imposibile~~

Inputs		Output	
$c_n$	$c_{2n}^0$	$c_{2n}^1$	$c_{2n}$
0	0	0	0
0	0	1	0
0	1	0	$d$
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	$d$
1	1	1	1



În consecință, transportul de ieșire  $c_{2n}$  va fi generat astfel:



$$1 \text{ s bit} \rightarrow 2k = 15 \quad \times$$

$$k+k+k+1 = 15 \quad \times$$

$$3k+1 = 15$$

$$k+k+k+1 + k+2 = 15$$

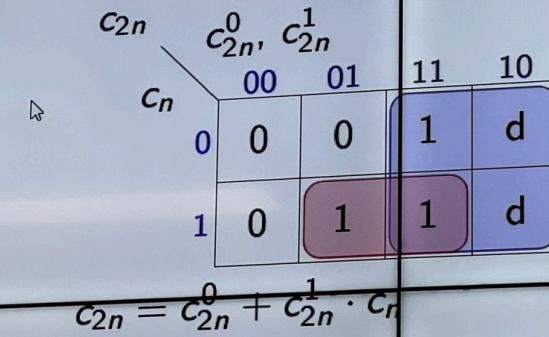
$$4k = 12 \quad k=3$$

### 2.3.3 Carry Select Adder (contin.)

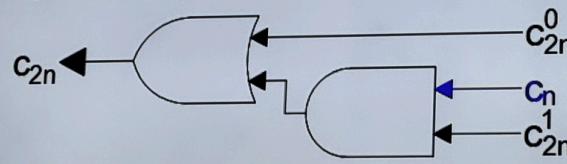
Optimizarea ariei CSeA:

- se pornește de la tabelul de adevăr al semnalului  $c_{2n}$
- **Niciodată**  $c_{2n}^0$  nu poate fi mai mare decât  $c_{2n}^1$ !
- Întrebare: de ce?
- → tabelul de adevăr folosește *don't care* pentru cazurile imposibile

Inputs		Output	
$c_n$	$c_{2n}^0$	$c_{2n}^1$	$c_{2n}$
0	0	0	0
0	0	1	0
0	1	0	$d$
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	$d$
1	1	1	1



În consecință, transportul de ieșire  $c_{2n}$  va fi generat astfel:



Subiect  $\rightarrow$  m. de biți  $\rightarrow$  distribuție  
a segmentelor eficiente

$CS_{k/e}$  Multilevel

# Conditional Sum Adder

CSUA

$\rightarrow$  primul etaj calculează  $\begin{cases} \bar{z}_i, c_{i+1} - c_i = 0 \\ z_i, c_{i+1} - c_i = 1 \end{cases}$

$\rightarrow$  celelalte nivele folosesc bitii transport care legă segmentele

$$c_{i+1} = \underbrace{x_i y_i}_{g_i} \bar{c}_i + \underbrace{(x_i + y_i)}_{p_i} c_i$$

$$c_i = 0 \rightarrow c_{i+1} = g_i$$

$$c_i = 1 \rightarrow c_{i+1} = p_i$$

$$\begin{aligned} \bar{c}_{i+1} &= \bar{g}_i \bar{c}_i + \bar{p}_i c_i = \bar{g}_i \bar{c}_i \cdot \bar{p}_i c_i = \\ &= (\bar{g}_i + c_i) \cdot (\bar{p}_i + \bar{c}_i) \\ &= \bar{g}_i \bar{p}_i + \underbrace{\bar{g}_i \bar{c}_i + \bar{p}_i c_i}_{\text{cancel}} + c_i \bar{c}_i \end{aligned}$$

$$\bar{c}_{i+1} = \bar{g}_i \bar{c}_i + \bar{p}_i \bar{c}_i$$

Sum bits

$$Z_0 = X_0 \oplus Y_0 \oplus 0$$

Carry bits

$$C_1 = g_0 \cdot \overline{C_0} + p_0 \cdot C_0 \quad C_0 = g_0$$

$$Z_1 = X_1 \oplus Y_1 \oplus C_1 = \overline{X_1 \oplus Y_1} C_1 + (X_1 \oplus Y_1) \overline{C_1}$$

$$C_2 = g_1 \cdot \overline{C_1} + p_1 \cdot C_1$$

$$Z_2 = \overline{X_2 \oplus Y_2} C_2 + (X_2 \oplus Y_2) \overline{C_2}$$

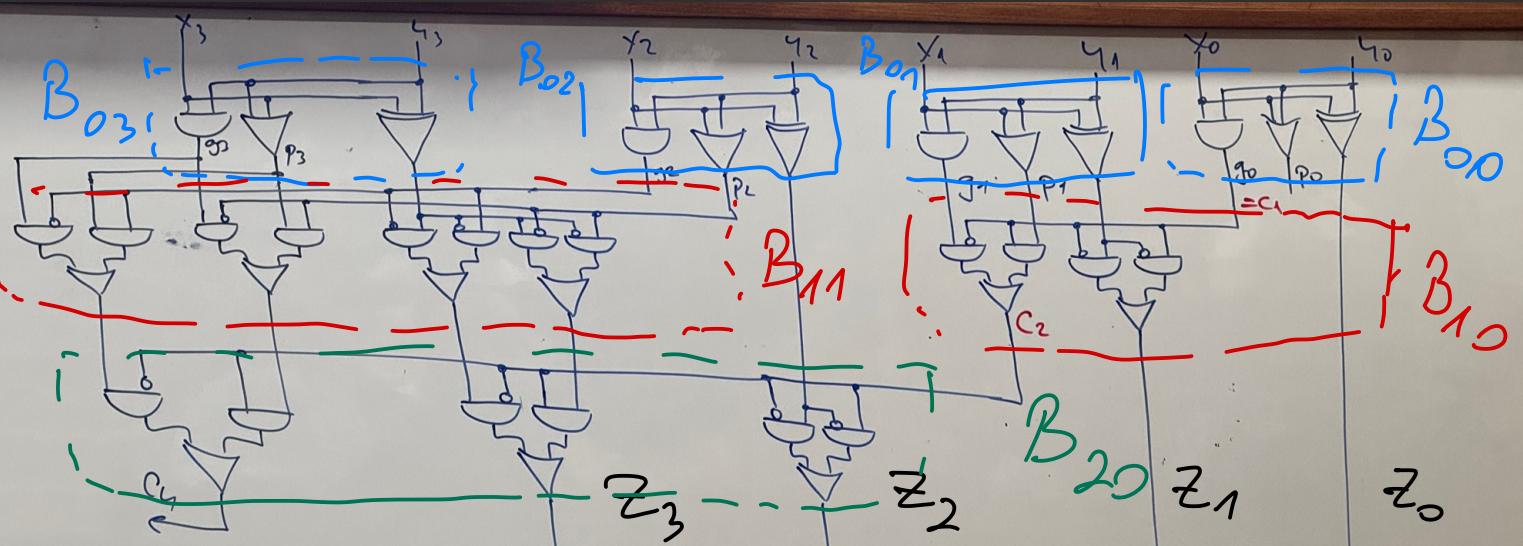
$$C_3 = \frac{g_2}{g_2} \overline{C_2} + \frac{p_2}{p_2} C_2$$

$$Z_3 = \overline{X_2 \oplus Y_2} C_3 + (X_2 \oplus Y_2) \overline{C_3}$$

$$= (\overline{X_2 \oplus Y_2}) (g_2 \overline{C_2} + p_2 C_2) +$$

$$+ (X_2 \oplus Y_2) (\overline{g_2} \overline{C_2} + \overline{p_2} C_2)$$

→ Implementing Carry in formula for



## 2.3.4 Conditional Sum Adder (contin.)

## Arhitectura Conditional Sum Adder (CSuA):

$$D_{CSuA-4}^z = 6d$$

$$D_{CSuA-4}^{Cout} = 5d$$

