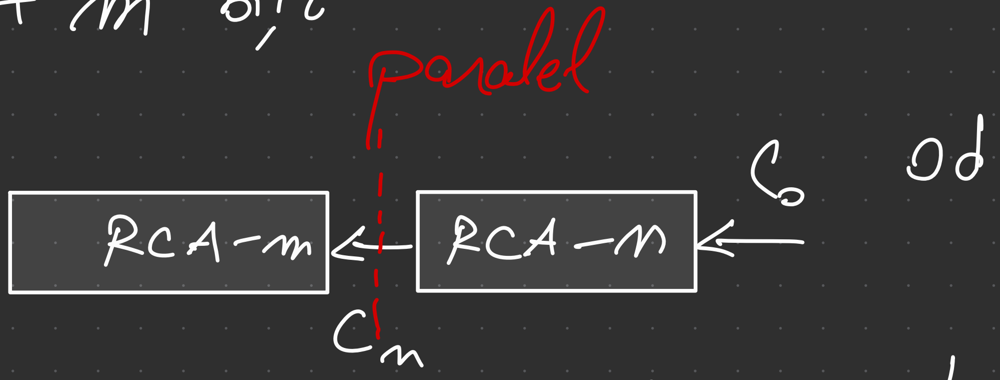
CMOS Pre-discharge -> Inițializare la 0

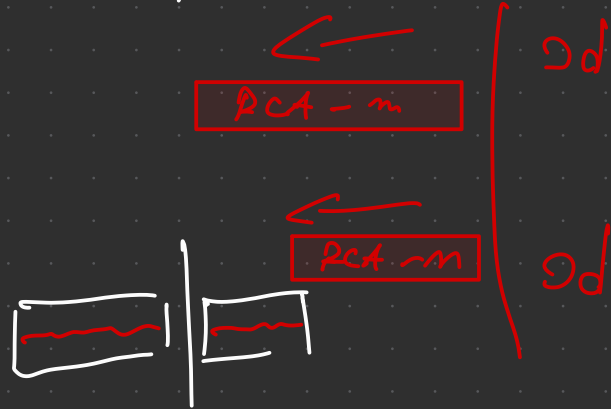
CSkA   Carry Skip Adder

-> transporturile sunt pre-discharged

$n + m$ biți



$C_n == 0 \Rightarrow C_n$ corect 0d

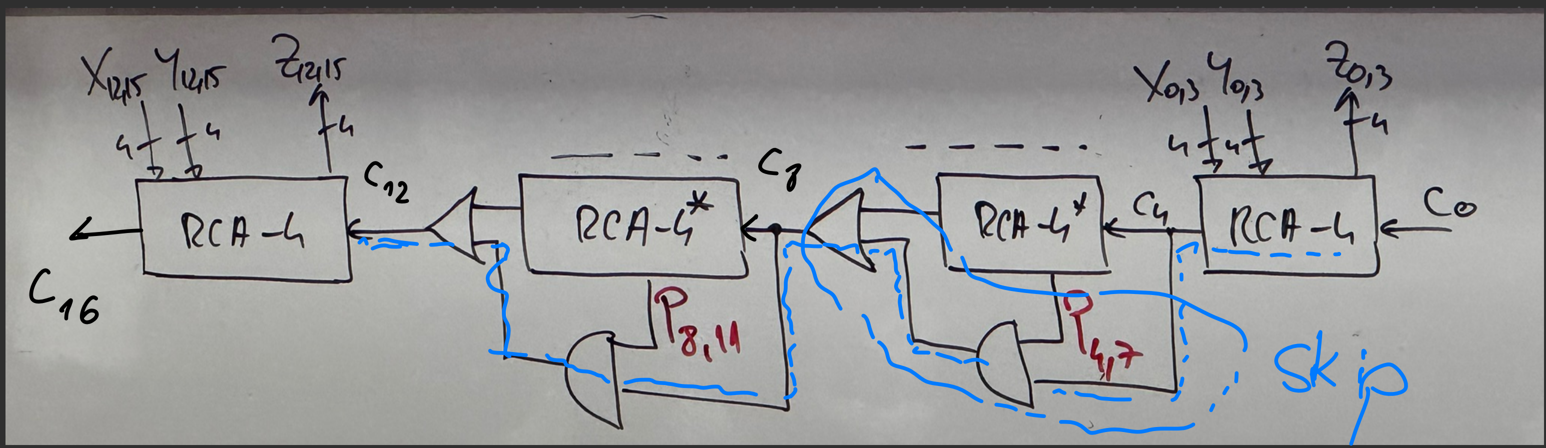


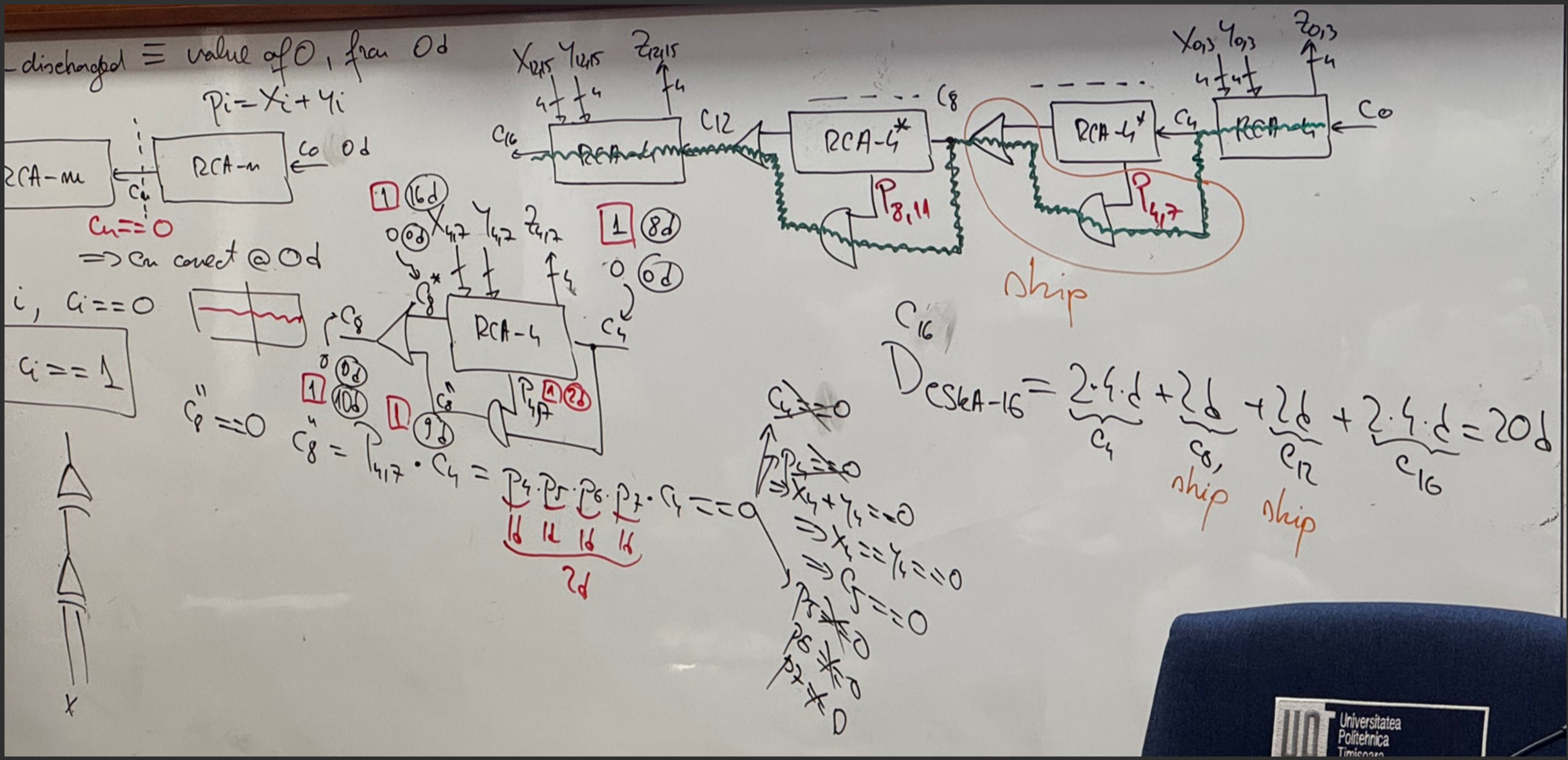
Calea critică RCA

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

$(\forall)\ C_i == 1$ Worst Case

RCA* -> are ieșire Propagate





Lățimea optimă pt. RCA

$$n = \underbrace{b}_{\substack{\text{RCA}\\\text{segment}\\\text{width}}} * k$$

$\frac{n}{b}$ RCA: 2 no skip (first, last); $\frac{n}{b} - 2$ with skip

$$\Delta^{z/cout}_{CS_kA-n} = 2bd + 2\left(\frac{n}{b}-2\right)b + 2b\,d =$$

$$= \left(4b + \frac{2n}{b} - 4\right) d$$

$$\frac{d\,\Delta^{Z/cout}_{CSkA-n}}{db} = 4 - \frac{2n}{b^2} = 0$$

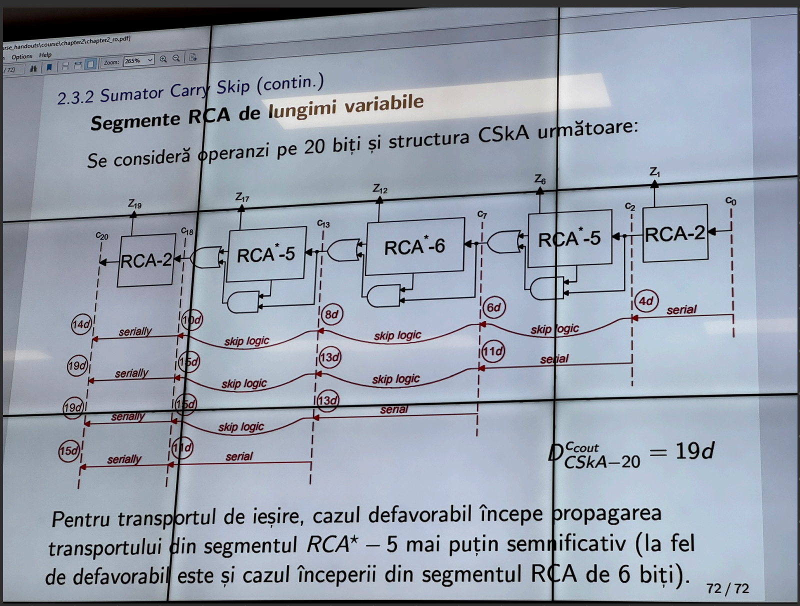
$$\boxed{b = \sqrt{\frac{n}{2}}}$$

$n = 32 \to \underline{b = 4}$ $\quad \Delta_{opt} = 28d$ $\quad \Delta_{opt} = 4\left(\sqrt{2n} - 1\right)d$

$n = 128 \to \underline{b = 8}$ $\quad \underline{\Delta_{opt} = 60d}$

RCA_{128} : $\Delta = 256d$

Variable-sized RCA segments



Det. latența Cout

EXAMEN

↗ bitul transport

↘ bitul sumă

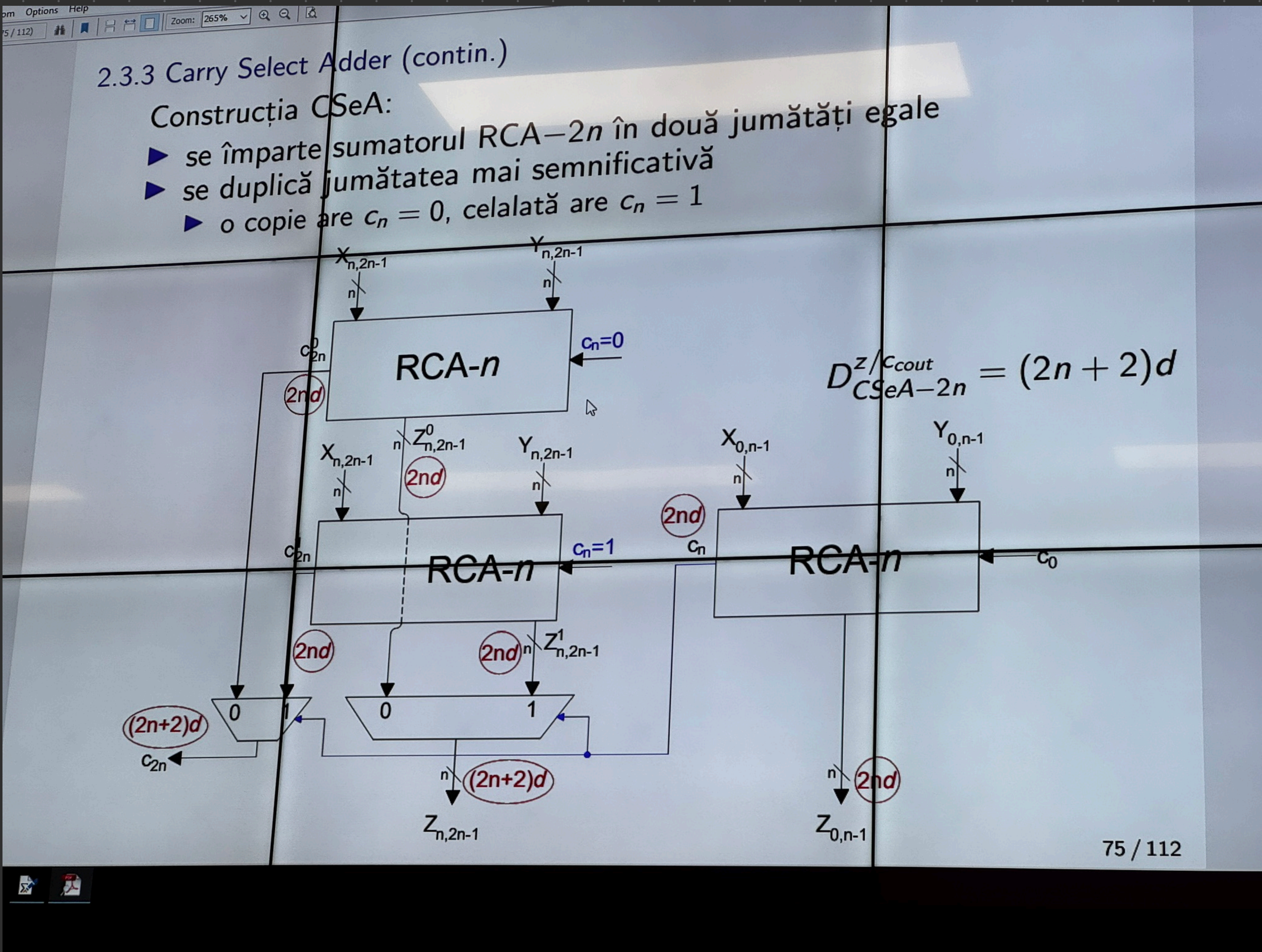
tradeoff

Carry Select Adder CS_eA

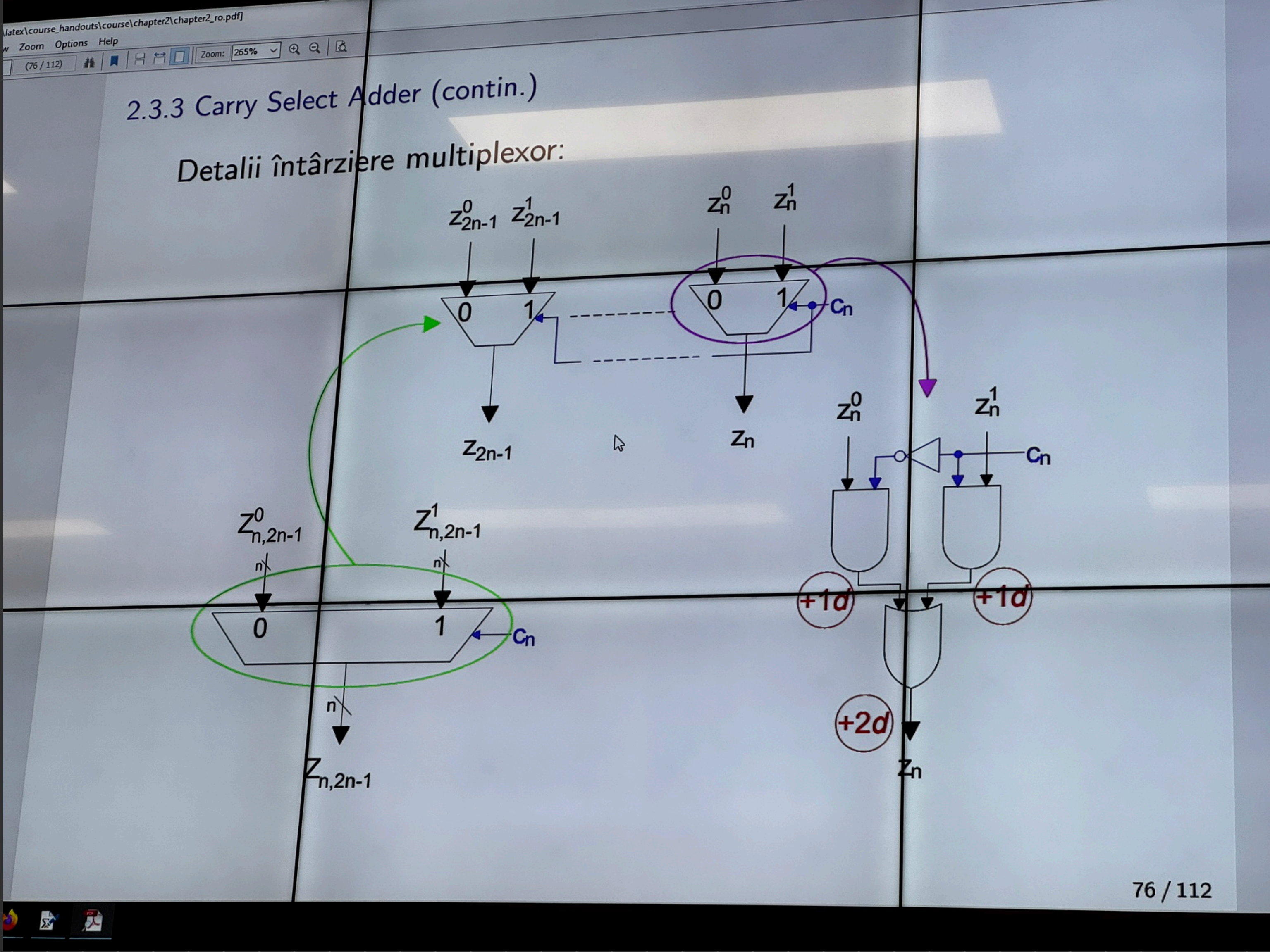
→ principiul sumei condiționate prin transport

$$C_i \begin{cases} 0 \Rightarrow z_i^0 \quad c_{i+1}^0 \quad \text{if } c==0 \\ 1 \Rightarrow z_i^1 \quad c_{i+1}^1 \quad \text{if } c==1 \end{cases}$$

RCA − 2n $\Delta_{RCA-2n} = 4n\,d$



carry 1 / carry 0 pt n;2n-1 → mux la final



$$\delta^{z/cout}_{CSeA-2n} = (2n+2)d$$

↘ duplicare 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^0_{2n} nu poate fi mai mare decât c^1_{2n}!
  - **Întrebare**: de ce?
  - $\Rightarrow$ tabelul de adevăr folosește *don't care* pentru cazurile imposibile

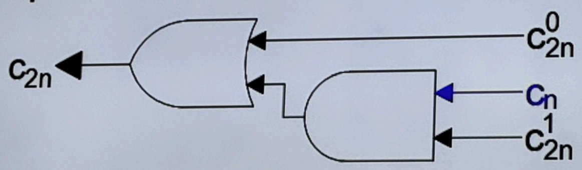
Inputs			Output
c_n	c^0_{2n}	c^1_{2n}	c_{2n}
0	0	0	0
0	0	1	0
0	1	0	*d*
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	*d*
1	1	1	1

c_{2n}

c_n \ c^0_{2n}, c^1_{2n}	00	01	11	10
0	0	0	1	d
1	0	1	1	d

$$c_{2n} = c^0_{2n} + c^1_{2n} \cdot c_n$$

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



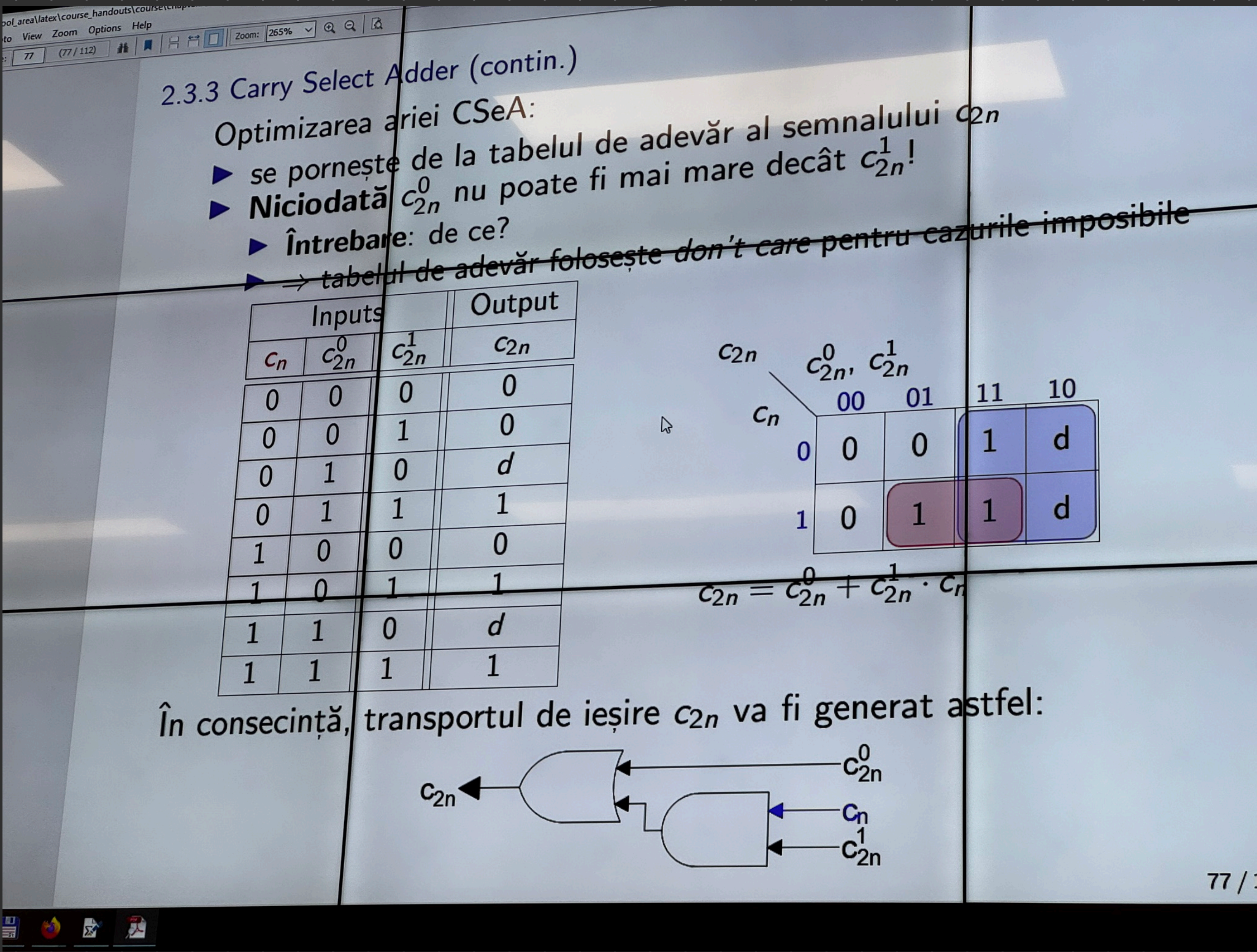
15 bit $\rightarrow$ $2k = 15$ ×

$k + k + k + 1 = 15$ ×

$3k + 1 = 15$

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

$4k = 12$ $k = 3$



Subiect → nr. de biți → distribuire a segmentelor eficientă

$CS_{k/e}A$ Multilevel

Conditional Sum Adder

CSuA

→ primul etaj calculează $\begin{cases} z_i^0,\ c_{i+1}^0 - c_i = 0 \\ z_i^1,\ c_{i+1}^1 - c_i = 1 \end{cases}$

→ celelalte nivele folosesc biți transport care leagă segmentele

$$\boxed{c_{i+1} = \underbrace{x_i y_i}_{g_i} \overline{c_i} + \underbrace{(x_i + y_i)}_{p_i} c_i}$$

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

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

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

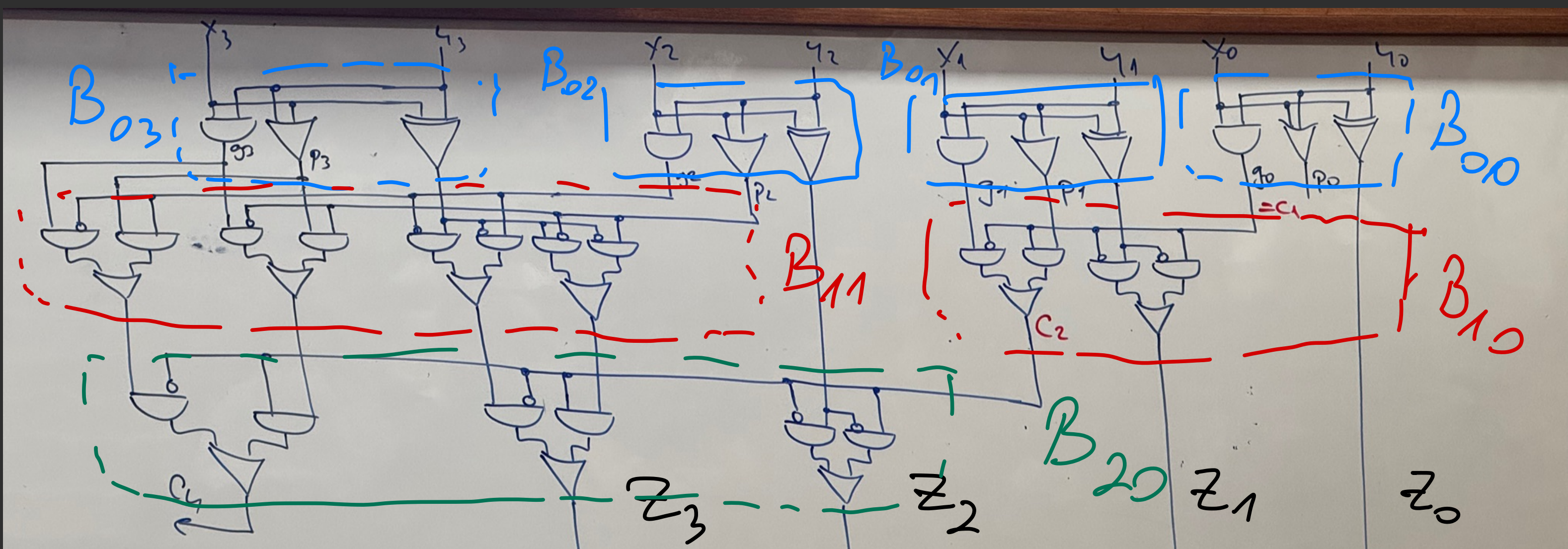
$$= (\overline{g_i} + c_i)(\overline{p_i} + \overline{c_i})$$

$$= \overline{g_i}\,\overline{p_i} + \overline{g_i}\,\overline{c_i} + \overline{p_i} c_i + \cancel{c_i \overline{c_i}}$$

$$\boxed{\overline{c_{i+1}} = \overline{g_i}\,\overline{c_i} + \overline{p_i}\,\overline{c_i}}$$

Sum bits	Carry bits
$z_0 = x_0 \oplus y_0 \oplus \cancel{0}$	$c_1 = g_0 \cdot \underbrace{\overline{c_0}}_{=1} + p_0 \underbrace{c_0}_{=0} = g_0$
$z_1 = x_1 \oplus y_1 \oplus c_1 = \overline{x_1 \oplus y_1}\, c_1 + (x_1 \oplus y_1)\,\overline{c_1}$	$c_2 = g_1 \overline{c_1} + p_1 c_1$
$z_2 = \overline{x_2 \oplus y_2}\, c_2 + (x_2 \oplus y_2)\,\overline{c_2}$	$c_3 = g_2 \overline{c_2} + p_2 c_2$ $\overline{c_3} = \overline{g_2}\,\overline{c_2} + \overline{p_2}\, c_2$
$z_3 = \overline{x_2 \oplus y_2}\, c_3 + (x_2 \oplus y_2)\,\overline{c_3}$ $= (\overline{x_2 \oplus y_2})(g_2 \overline{c_2} + p_2 c_2) +$ $+ (x_2 \oplus y_2)(\overline{g_2}\,\overline{c_2} + \overline{p_2}\, c_2)$	

→ Înlocuire Carry cu formula lor



2.3.4 Conditional Sum Adder (contin.)

Arhitectura Conditional Sum Adder (CSuA):

$$D^{z}_{CSuA-4} = 6d$$

$$D^{c_{cout}}_{CSuA-4} = 5d$$