ARCHITETTURA DEGLI ELABORATORI

A.A. 2020-2021

Università di Napoli Federico II
Corso di Laurea in Informatica

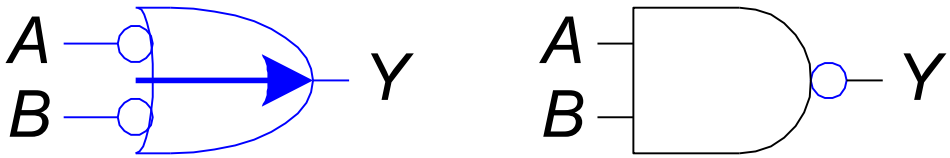
Docenti
Proff. Luigi Sauro gruppo 1 (A-G)
Silvia Rossi gruppo 2 (H-Z)

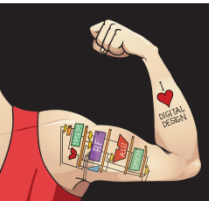
ALGEBRA DI BOOLE E RETI COMBINATORIE

Bubble Pushing

- **Backward:**
  - Body changes
  - Adds bubbles to inputs

- **Forward:**
  - Body changes
  - Adds bubble to output

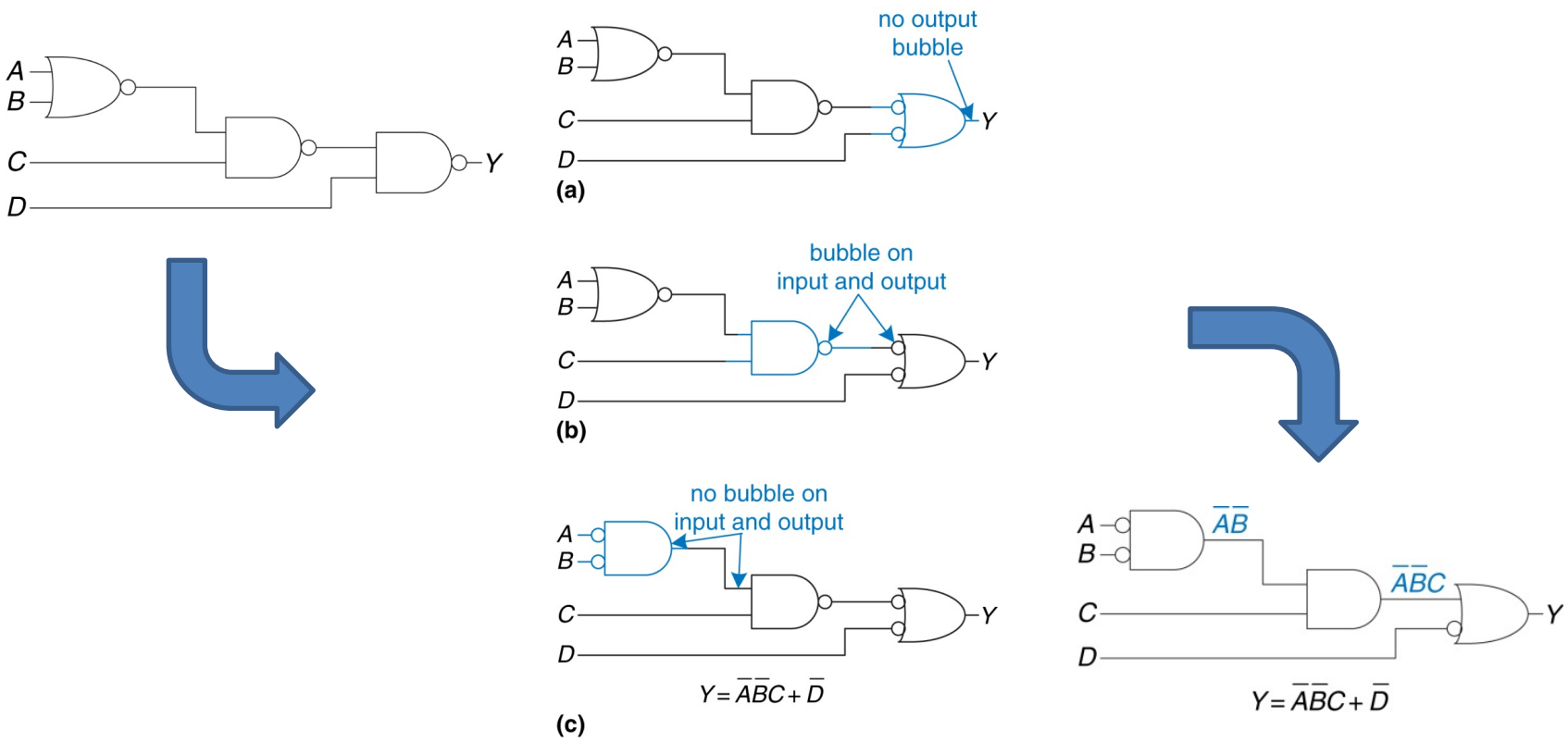




Bubble pushing

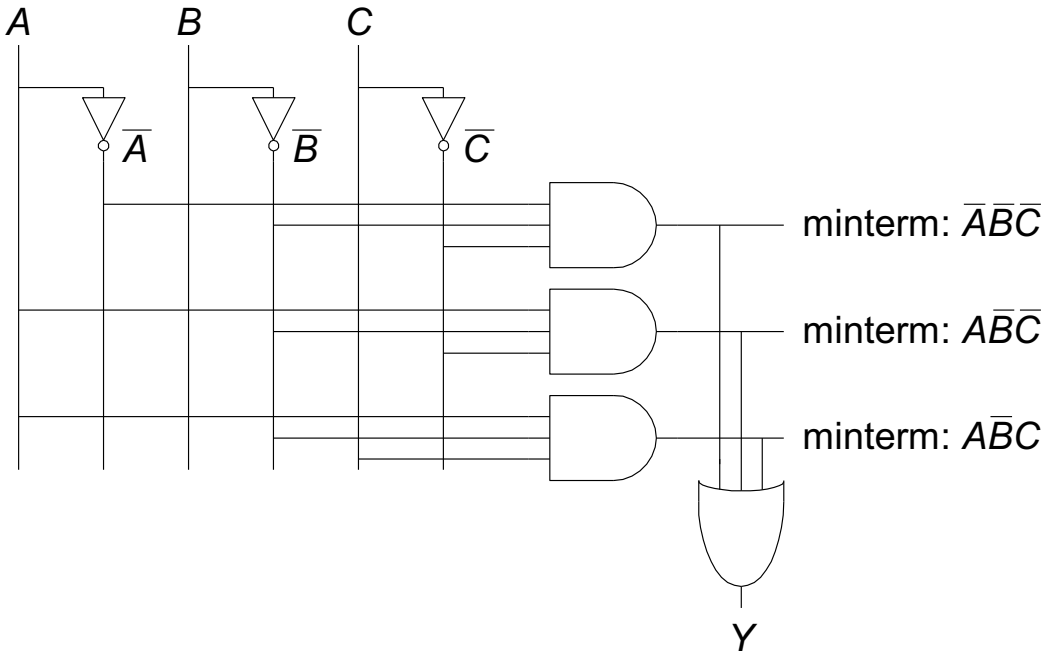
- Usando logiche multilivello e porte NAND/NOR a volte rende difficile capire quale funzione booleana un circuito realizza a causa delle molte negazioni annidate
- Per avere una espressione un po' più leggibile si possono applicare le leggi di De Morgan
- Il bubble pushing è una tecnica che applica sistematicamente le leggi di De Morgan e la legge della doppia negazione per eliminare le negazioni annidate
- Partendo dall'output Y si applicano a ritroso le leggi di De Morgan in modo che l'input e l'output di ogni nodo siano entrambi positivi o negati

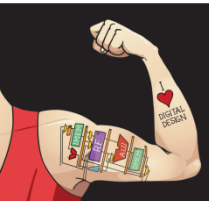
Bubble pushing



From Logic to Gates

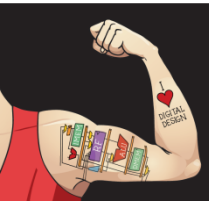
- Two-level logic: ANDs followed by ORs
- Example: $Y = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$





Circuit Schematics Rules

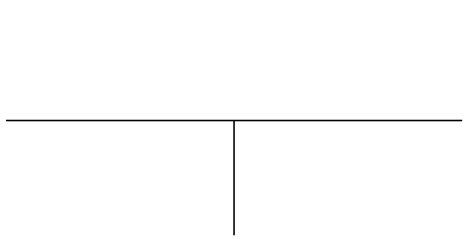
- Inputs on the left (or top)
- Outputs on right (or bottom)
- Gates flow from left to right
- Straight wires are best



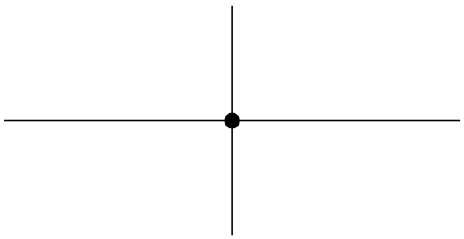
Circuit Schematic Rules (cont.)

- Wires always connect at a T junction
- A dot where wires cross indicates a connection between the wires
- Wires crossing *without* a dot make no connection

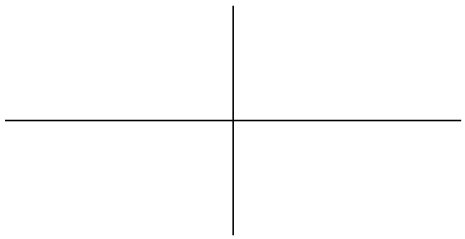
wires connect at a T junction

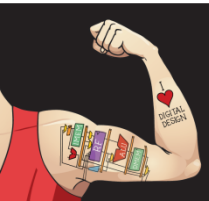


wires connect at a dot



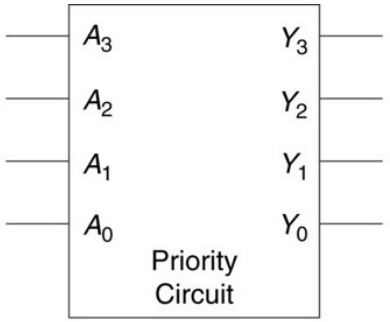
wires crossing without a dot do not connect





Circuito a priorità

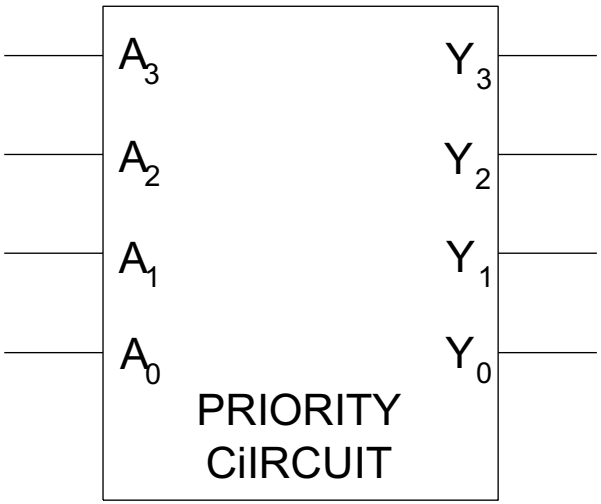
- I circuiti a priorità vengono utilizzati per assegnare una risorsa condivisa secondo un ordine di priorità fra chi ne fa richiesta
- A esempio posso avere 4 possibili richiedenti con priorità 3>2>1>0
- Gli input A_0 ,…, A_3 rappresentano le richieste della risorsa
- Gli output Y_0 ,…, Y_3 rappresentano a chi assegnata la risorsa, di volta in volta uno solo di essi sarà uguale a 1



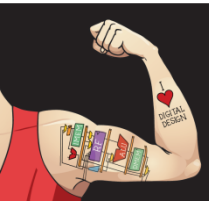
Multiple-Output Circuits

- **Example: Priority Circuit**

  Output asserted corresponding to most significant TRUE input



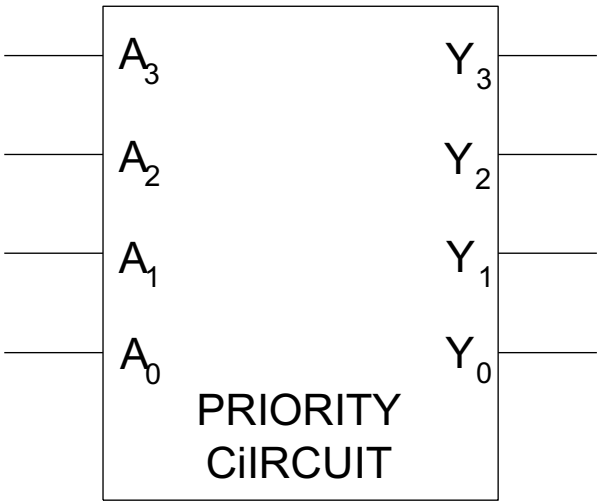
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				



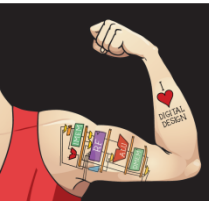
Multiple-Output Circuits

- **Example: Priority Circuit**

  Output asserted corresponding to most significant TRUE input

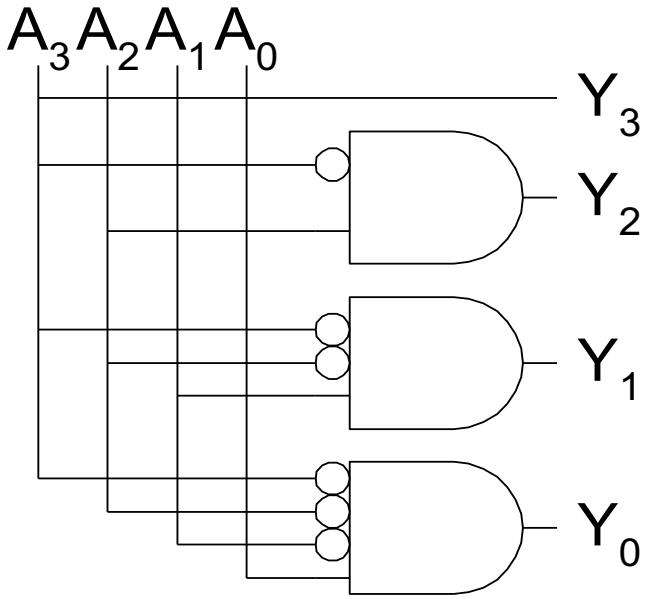


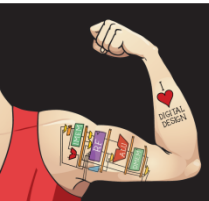
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0



Priority Circuit Hardware

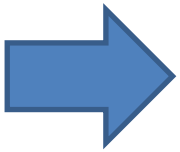
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0





Don’t cares

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

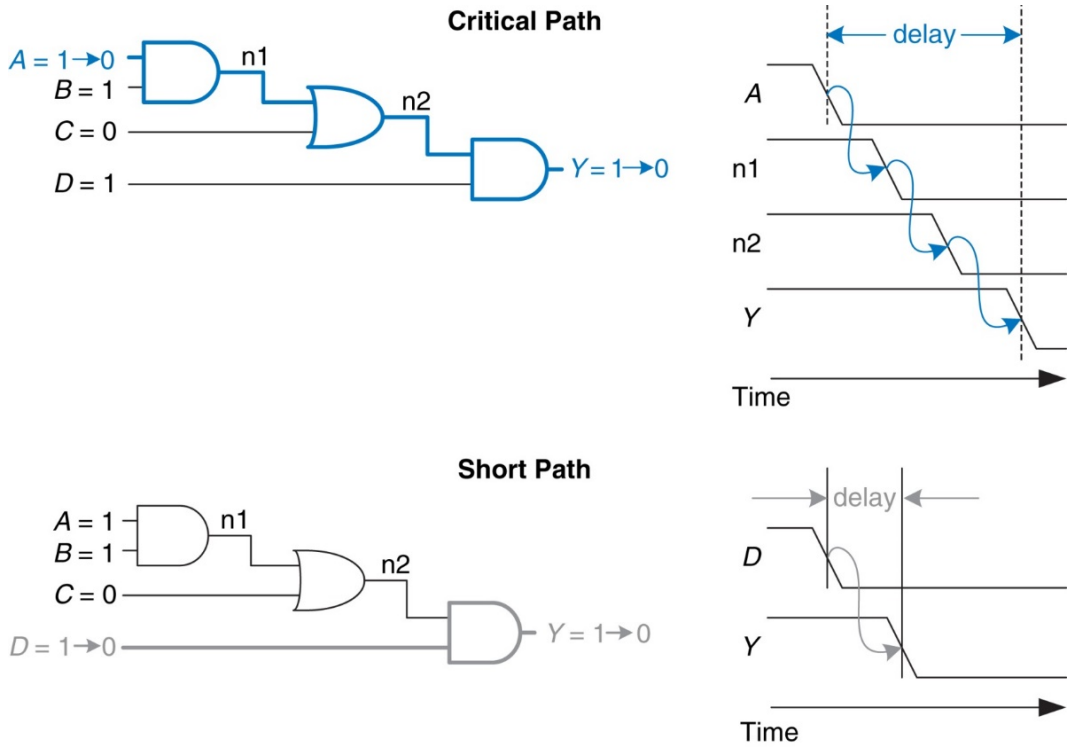


A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	X	0	0	1	0
0	1	X	X	0	1	0	0
1	X	X	X	1	0	0	0

don’t cares

Vantaggi delle SOP/POS

- La logica a 2 livelli della forma SOP presenta dei vantaggi, ad esempio, nei tempi di propagazione



Limiti delle forme SOP/POS

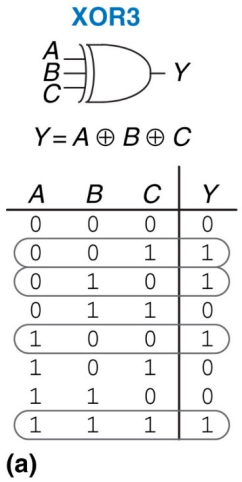
- Tuttavia alcune funzioni booleane, poste in forma SOP, sono estremamente poco succinte e quindi richiedono un numero considerevole di porte
- Presa la tabella di verità di funzione booleana di n variabili:
  - Se il numero di 1 nella colonna di output è piccolo allora la forma SOP è succinta
  - Se il numero di 0 nella colonna di output è piccolo allora la forma POS è succinta
  - Se il numero di 0 e 1 è più o meno lo stesso? Problema: avrò circa 2^{n-1} mintermini/maxtermini

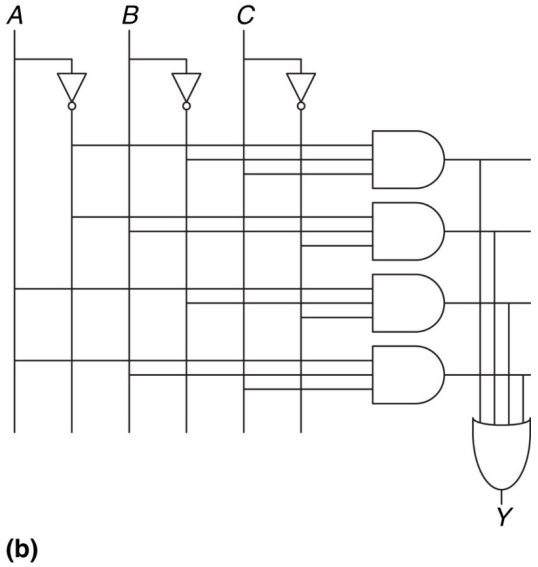
Limiti delle forme SOP/POS

- Considerimo ad esempio uno XOR a più variabili
- Y=1 sse il numero di input uguali a 1 è dispari
- XOR3 in forma SOP

$$Y = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$$

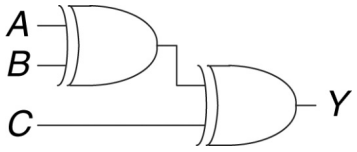
- XOR8 richiede 128 AND8 e un OR128

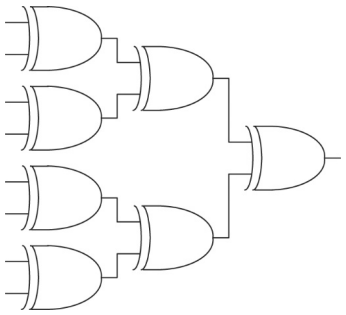




Logiche multilivello

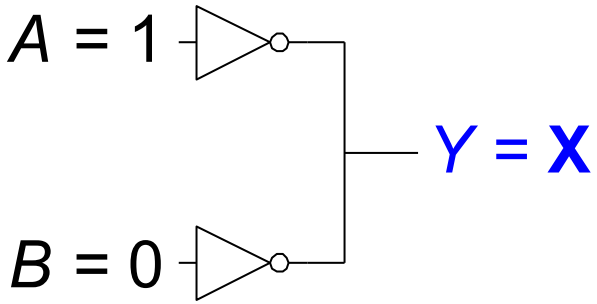
- Per ridurre il numero di porte logiche a volte è necessario ricorrere ad una logica multilivello
- Ad esempio è facile verificare che $A \oplus B \oplus C = (A \oplus B) \oplus C$
- Analogamente per XOR8:



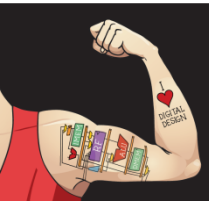


Valore Illegale: X

- **Contention:** circuit tries to drive output to 1 **and** 0
  - Actual value somewhere in between
  - Could be 0, 1, or in forbidden zone
  - Might change with voltage, temperature, time, noise
  - Often causes excessive power dissipation



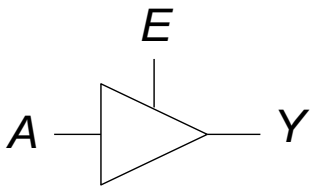
- **Warnings:**
  - Contention usually indicates a **bug**.
  - **X** is used for **"don't care"** and **contention** - look at the context to tell them apart.

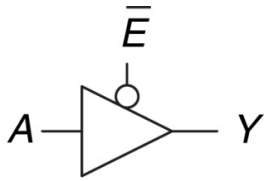


Floating: Z

- Floating, high impedance, open, high Z
- Floating output might be 0, 1, or somewhere in between
  - Gli stati di alta impedenza vengono utilizzati per disconnettere una parte di un circuito dal resto. Per questo si utilizzano i tristate buffer

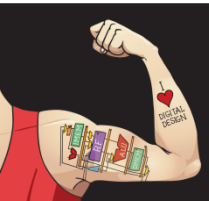
Tristate Buffer





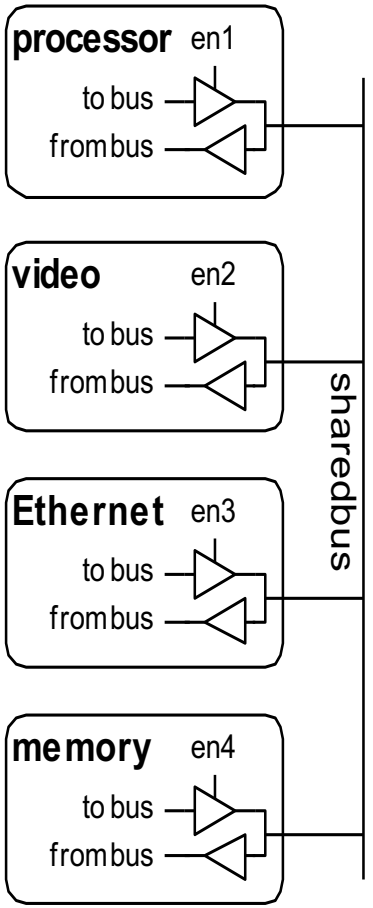
E	A	Y
0	0	Z
0	1	Z
1	0	0
1	1	1

$\bar{E}$	A	Y
0	0	0
0	1	1
1	0	Z
1	1	Z



Bus condiviso

- I tristate buffers sono usati in bus che connettono diversi chip
- Nell'esempio processore, scheda video e controller ethernet devono poter comunicare con la memoria centrale.
- Tuttavia, per evitare stati illegali, solo un componente alla volta può «immettere» segnali sul bus
- Gli altri componenti quindi devono essere temporaneamente disconnessi tramite un tristate buffer



Esercizi

- Esercizi 2.13 – 2.15 – 2.16 – 2.17 – 2.25 – 2.26 – 2.27

Mappe di Karnaugh

- Le mappe di Karnaugh sono un metodo per semplificare espressioni booleane in forma SOP
- In realtà non introducono tecniche di semplificazione nuove, sono semplicemente un espediente grafico che consente di rilevare più facilmente implicati che possono essere semplificati
- Quindi alla base delle mappe di Karnaugh c'è il solito principio:

$$PA + P\overline{A} = P$$

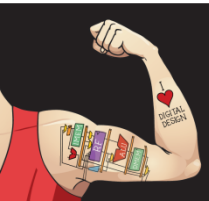
Karnaugh Maps (K-Maps)

- Boolean expressions can be minimized by combining terms
- K-maps minimize equations graphically
- $PA + P\overline{A} = P$

A	B	C	Y
0	0	0	**1**
0	0	1	**1**
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Y \ AB / C	00	01	11	10
0	**1**	0	0	0
1	**1**	0	0	0

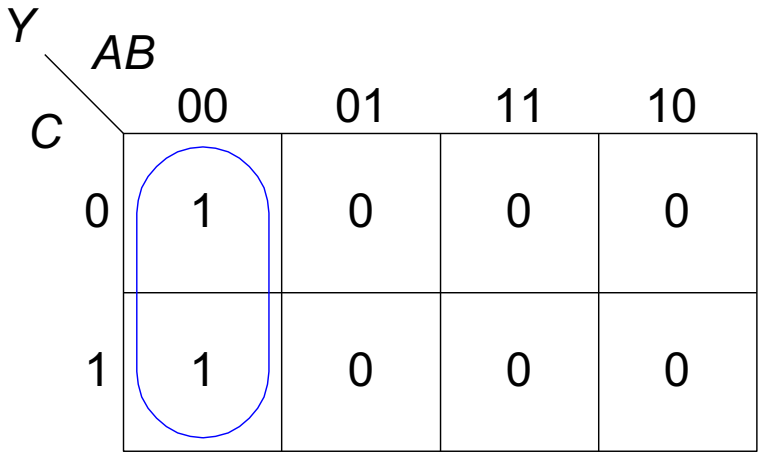
Y \ AB / C	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$



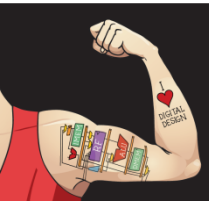
K-Map

- Circle 1's in adjacent squares
- In Boolean expression, include only literals whose true and complement form are ***not*** in the circle

A	B	C	Y
0	0	0	**1**
0	0	1	**1**
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0



$$Y = \overline{A}\overline{B}$$



3-Input K-Map

Y: C \ AB	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

Truth Table

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	**1**
0	1	1	**1**
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	**1**

K-Map

Y: C \ AB	00	01	11	10
0				
1