Prof. Mariacarla Staffa
a.a. 2022/2023

Laboratorio di Architettura Degli Elaboratori

Teorema di De Morgan - Bubble Pushing

Recall: Semplificazioni di funzioni booleane

Esercizio 6

Apply T8' first when possible: W+XZ = (W+X)(W+Z)

$$Y = (A + BC)(A + DE)$$

Make: X = BC, Z = DE and rewrite equation

Y	= (A+X)(A+Z)	substitution (X=BC, Z=DE)
	= A + XZ	T8': Distributivity
	= A + BCDE	substitution

or

Y	= (A + BC)(A + DE)	T8: Distributivity
	= AA + ADE + BCA + BCDE	T3: Idempotency (AA = A)
	= A + ADE + BCA + BCDE	T9': Covering (A+AP=A → A+ADE = A)
	= A + ABC + BCDE	T7: Commutativity T9': Covering (A+AP=A → A+ABC = A)
	= A + BCDE	

This is called ***multiplying out*** an expression to get sum-of-products (SOP) form.

Multiplying Out: SOP Form

An expression is in **sum-of-products (SOP)** form when all products contain literals only:

- SOP form: Y = AB + BC’ + DE
- **NOT** SOP form: Y = DF + E(A’+B)
- SOP form: Z = A + BC + DE’F

Multiplying Out: SOP Form

Example: $Y = (A + C + D + E)(A + B)$

Apply T8' first when possible: W+XZ = (W+X)(W+Z)

Make: X = (C+D+E), Z = B and rewrite equation

Y = (A+X)(A+Z)	substitution (X=(C+D+E), Z=B)
= A + XZ	T8': Distributivity
= A + (C+D+E)B	substitution
= A + BC + BD + BE	T8: Distributivity

or

Y = AA+AB+AC+BC+AD+BD+AE+BE	T8: Distributivity
= **A**+AB+AC+AD+AE+BC+BD+BE	T3: Idempotency
= A + BC + BD + BE	T9': Covering (A+(AX) = A)

Factoring: POS Form

An expression is in **product-of-sums (POS)** form when all sums contain literals only.

- POS form: Y = (A+B)(C+D)(E'+F)
- **NOT** POS form: Y = (D+E)(F'+GH)
- POS form: Z = A(B+C)(D+E')

Factoring: POS Form

Example 1:

Y = (A + B'CDE)

Apply T8' first when possible: W+XZ = (W+X)(W+Z)

Make: X = B'C, Z = DE and rewrite equation

Y = (A+XZ)	substitution (X=B'C, Z=DE)
= (A+B'C)(A+DE)	T8': Distributivity
= (A+B')(A+C)(A+D)(A+E)	T8': Distributivity

Factoring: POS Form

Example 2: $Y = AB + C'DE + F$

Apply T8′ first when possible: W+XZ = (W+X)(W+Z)

Make: W = AB, X = C′, Z = DE and rewrite equation

Y = (W+XZ) + F	substitution W = AB, X = C′, Z = DE
= (W+X)(W+Z) + F	T8′: Distributivity
= (AB+C′)(AB+DE)+F	substitution
= (A+C′)(B+C′)(AB+D)(AB+E)+F	T8′: Distributivity
= (A+C′)(B+C′)(A+D)(B+D)(A+E)(B+E)+F	T8′: Distributivity
= (A+C′+F)(B+C′+F)(A+D+F)(B+D+F)(A+E+F)(B+E+F)	T8′: Distributivity

DeMorgan's Theorem

Number	Theorem	Name
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots} = \overline{B_0} + \overline{B_1} + \overline{B_2} \ldots$	DeMorgan's Theorem

- La negata di un prodotto è uguale alla somma delle negate
- La negata di una somma è uguale al prodotto delle negate

DeMorgan's Theorem: Dual

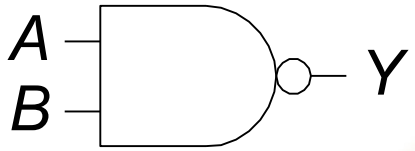
#	Theorem	Dual	Name
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots} = \overline{B}_0 + \overline{B}_1 + \overline{B}_2 \ldots$	$\overline{B_0 + B_1 + B_2 \ldots} = \overline{B}_0 \bullet \overline{B}_1 \bullet \overline{B}_2 \ldots$	DeMorgan's Theorem

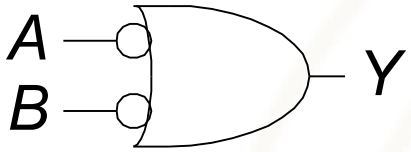
The complement of the product
is the
sum of the complements.

Dual: The **complement** of the **sum**
is the
product of the **complements**.

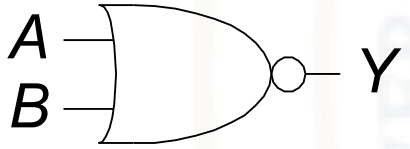
DeMorgan's Theorem: Dual

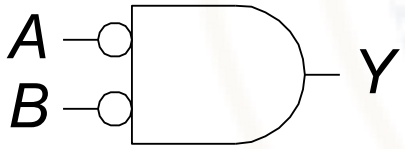
- $Y = \overline{AB} = \overline{A} + \overline{B}$





- $Y = \overline{A + B} = \overline{A} \bullet \overline{B}$





DeMorgan's Theorem: Esempio 1

$$Y = \overline{(A + \overline{BD})\overline{C}}$$

DeMorgan's Theorem: Esempio 1

$$Y = \overline{(A+\overline{BD})\overline{C}}$$

$$= \overline{(A+\overline{BD})} + \overline{\overline{C}}$$

$$= (\overline{A} \bullet \overline{\overline{BD}}) + C$$

$$= (\overline{A} \bullet (BD)) + C$$

$$= \overline{A}BD + C$$

DeMorgan's Theorem: Esempio 2

$$Y = \overline{(\overline{A}\overline{C}E + \overline{D})} + B$$

DeMorgan's Theorem: Esempio 2

$$Y = \overline{(\overline{\overline{A}C\overline{E}}+\overline{D})} + B$$

$$= \overline{(\overline{\overline{A}C\overline{E}}+\overline{D})} \bullet \overline{B}$$

$$= (\overline{\overline{A}C\overline{E}} \bullet \overline{D}) \bullet \overline{B}$$

$$= ((\overline{\overline{A}C}+\overline{E}) \bullet D) \bullet \overline{B}$$

$$= ((AC+\overline{E}) \bullet D) \bullet \overline{B}$$

$$= (ACD + D\overline{E}) \bullet \overline{B}$$

$$= A\overline{B}CD + \overline{B}D\overline{E}$$

Teoremi di De Morgan e forme SOP/POS

I teoremi di De Morgan possono essere usati per ridurre una generica espressione E in forma SOP/POS senza «passare» per la tabella di verità

- Applica esaustivamente «De Morgan» per spingere la negazione nella struttura della formula
- Applica esaustivamente la proprietà distributiva dell'AND sull'OR (SOP)
- Applica esaustivamente la proprietà distributiva dell'OR sull'AND (POS)

Schemi circuitali SOP

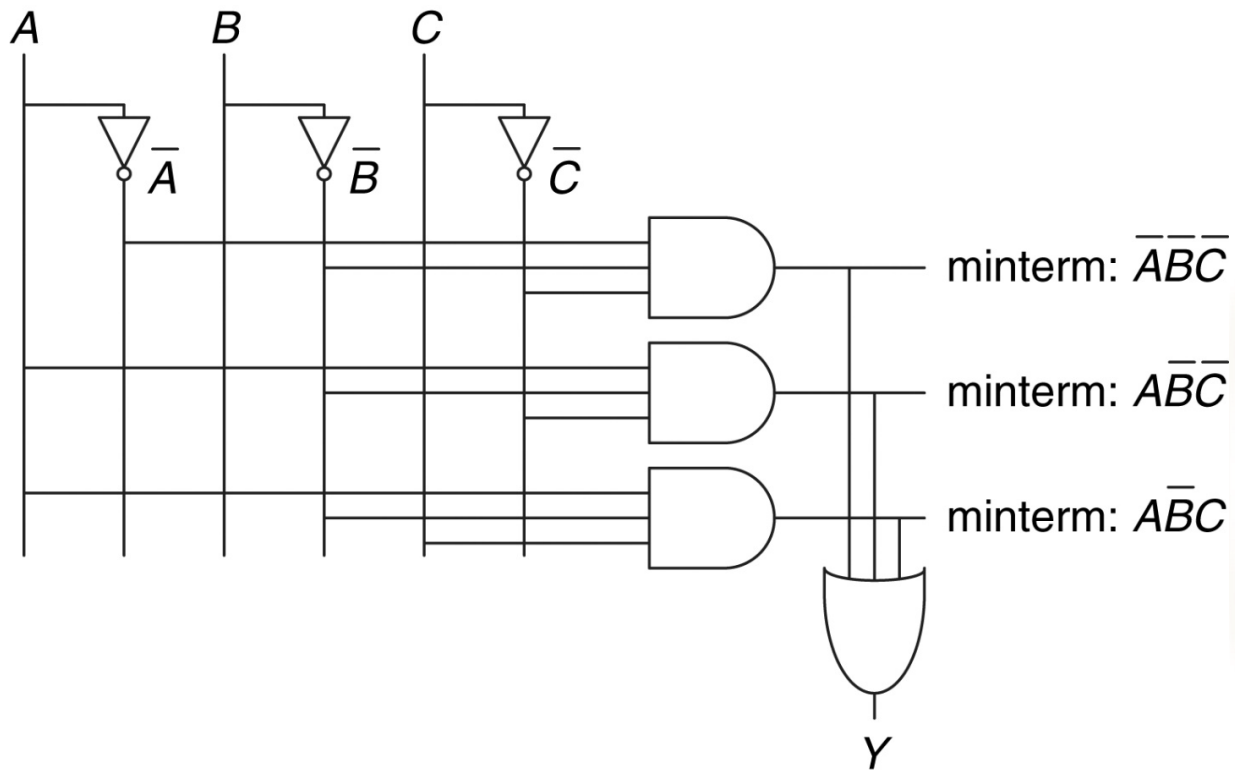
- Le formule in forma SOP hanno degli schemi circuitali molto regolari:
  - Disegna una linea di input per ogni variabile che occorre positiva
  - Aggiungi delle linee con un NOT per ogni variabile che occorre negata
  - Per ogni mintermine disegna una porta AND e aggiungi in ingresso le linee che corrispondono ai relativi litterali
  - Collega tutte le uscite delle porte AND in un unico OR
- Per questo la forma SOP viene detta una logica a due livelli AND-OR

Semplificare formule SOP

- Per T10 : $P\bar{B} + PB = P$ per ogni implicante P .
- Un implicante è detto *implicante primo* se non può essere combinato con altri implicanti della formula per ottenere un nuovo implicante con meno litterali.
- Una espressione SOP è minimale se tutti i suoi implicanti sono primi
- **Minimizzazione**: *ridurre il numero di implicanti e per ogni implicante ridurre il numero di litterali*

Schemi circuitali

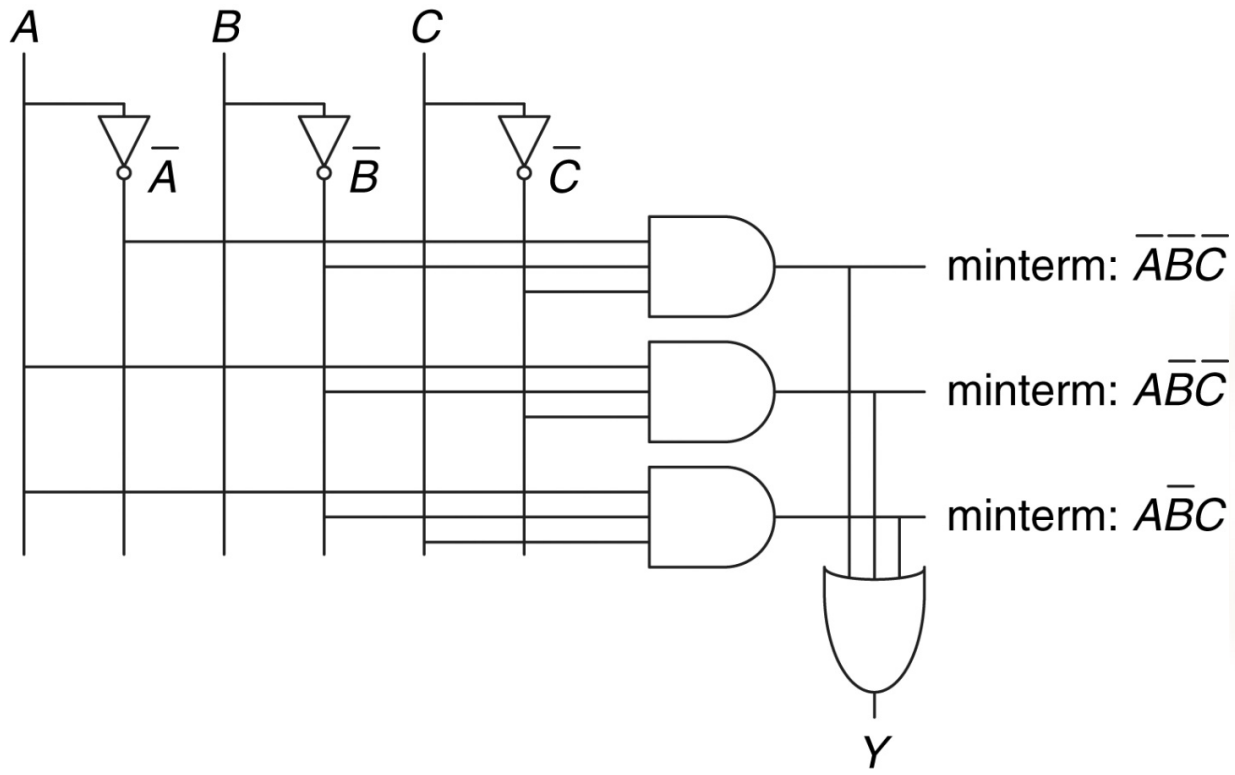
- Ad ogni espressione booleana corrisponde al seguente circuito combinatorio



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

Schemi circuitali

- Proviamo a minimizzare questa espressione booleana e di conseguenza il circuito corrispondente



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

Semplificare formule SOP

- Nel minimizzare una SOP, può essere necessario «sdoppiare» un implicante allorché questo può essere ridotto in modi differenti.

Step	Equation	Justification
	$\overline{A}\,\overline{B}\,\overline{C}+A\overline{B}\,\overline{C}+A\overline{B}C$	
1	$\overline{B}\,\overline{C}(\overline{A}+A)+A\overline{B}C$	T8: Distributivity
2	$\overline{B}\,\overline{C}(1)+A\overline{B}C$	T5: Complements
3	$\overline{B}\,\overline{C}+A\overline{B}C$	T1: Identity

Non è minimale $A\bar{B}C$ e $A\bar{B}\bar{C}$ possono ridursi in $A\bar{B}$

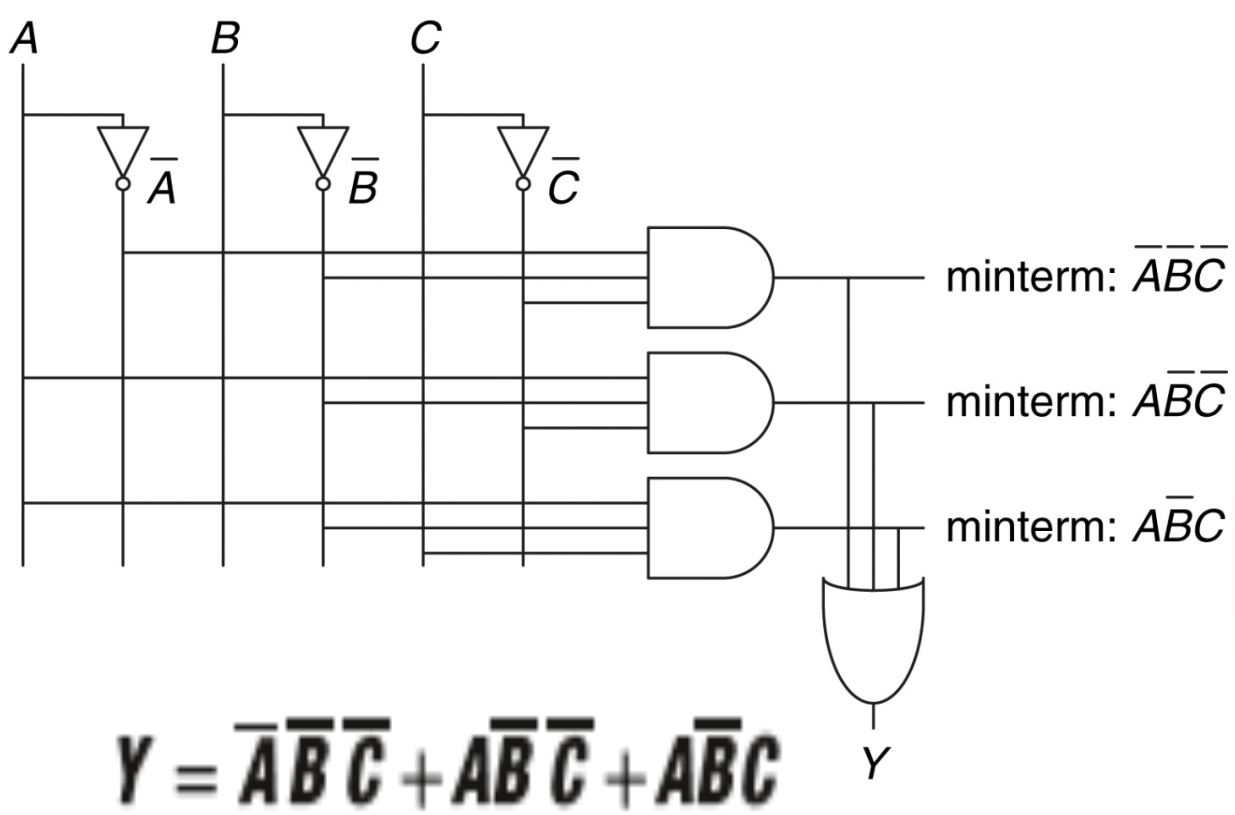
Semplificare formule SOP

Step	Equation	Justification
	$\overline{A}\,\overline{B}\,\overline{C}+A\overline{B}\,\overline{C}+A\overline{B}C$	
1	$\overline{A}\,\overline{B}\,\overline{C}+A\overline{B}\,\overline{C}+A\overline{B}\,\overline{C}+A\overline{B}C$	T3: Idempotency
2	$\overline{B}\,\overline{C}(\overline{A}+A)+A\overline{B}(\overline{C}+C)$	T8: Distributivity
3	$\overline{B}\,\overline{C}(1)+A\overline{B}(1)$	T5: Complements
4	$\overline{B}\,\overline{C}+A\overline{B}$	T1: Identity

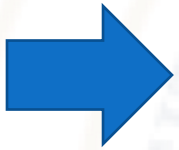
Il numero di implicanti è lo stesso ma $A\overline{B}$ ha meno letterali di $A\overline{B}C$

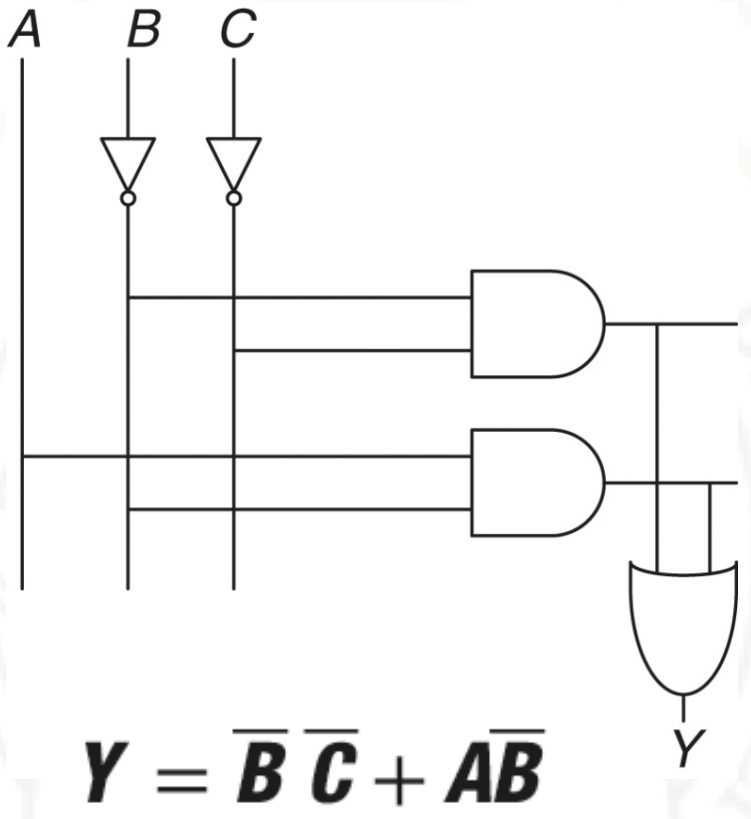
Perché semplificare una espressione?

- Importanza della minimizzazione: utilizzare meno porte logiche



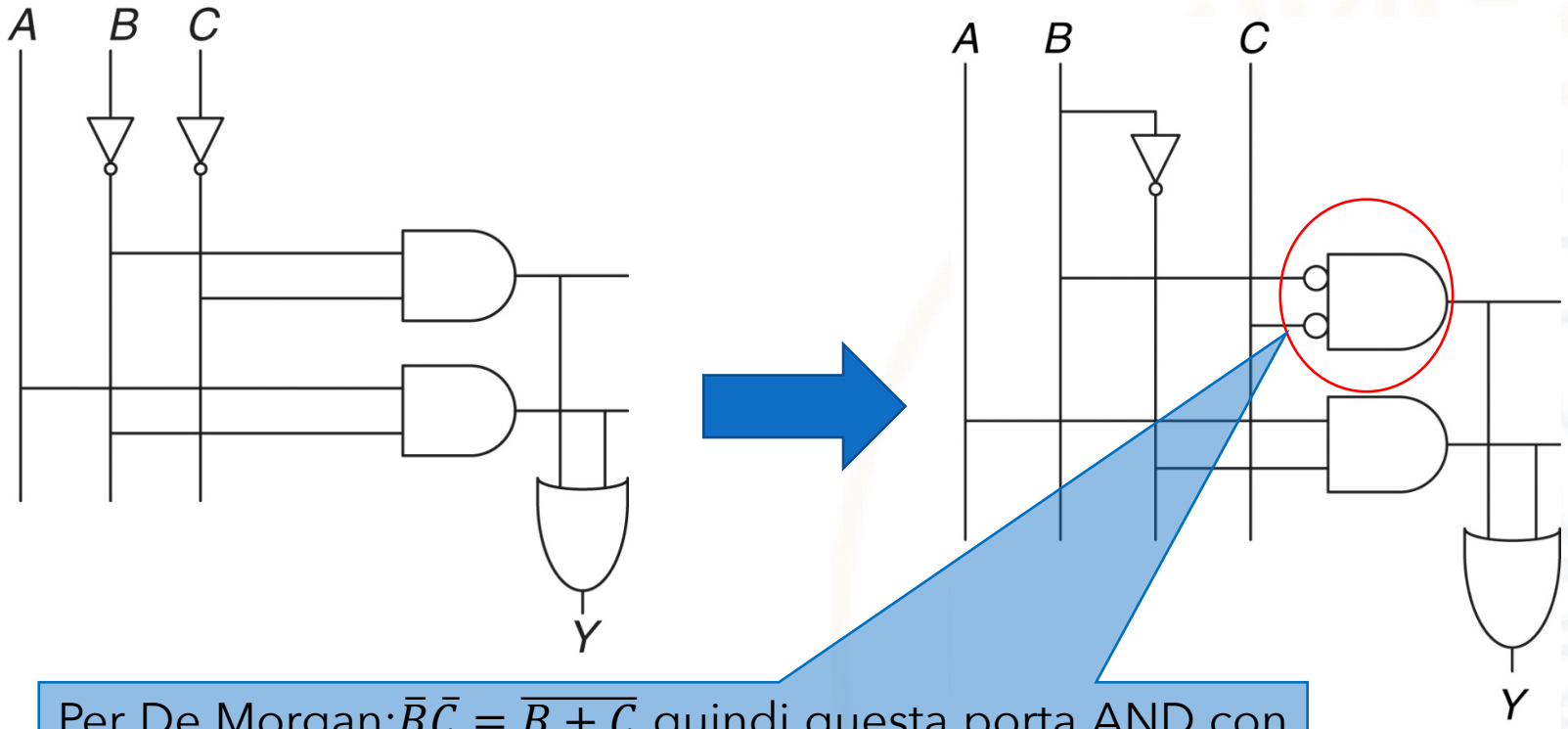
$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$





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

Schemi circuitali e minimizzazione



Per De Morgan: $\bar{B}\bar{C} = \overline{B + C}$ quindi questa porta AND con gli ingressi negati può essere sostituita da un NOR. Nella tecnologia MOSFET il NOR è più veloce di AND

Semplificare le forme POS

- Come si semplifica una forma POS?
- La proprietà principale che si usa è il combining (B+$\bar{C}$) • (B+C) = B
- Esempio: minimizza la seguente espressione boolean:

$$(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C})(\bar{A} + \bar{B} + C)(\bar{A} + \bar{B} + \bar{C}) =$$

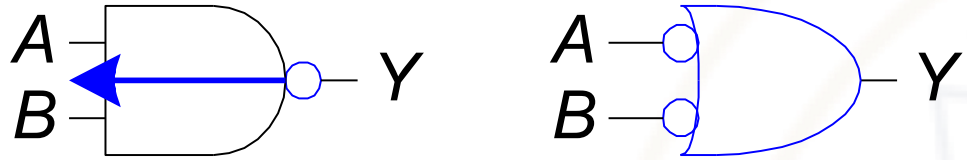
Semplificare le forme POS

- Come si semplifica una forma POS?
- La proprietà principale che si usa è il combining $(B+\bar{C}) \bullet (B+C) = B$
- Esempio:

$$(A+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C}) =$$
$$(A+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})(\bar{A}+\bar{B}+\bar{C})(\bar{A}+\bar{B}+\bar{C})=$$
$$(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})=$$
$$(B+\bar{C})(\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})=$$
$$(B+\bar{C})(\bar{A}+\bar{C})\ (\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})=$$
$$(B+\bar{C})(\bar{A}+\bar{C})\ (\bar{A}+\bar{B})$$

Bubble Pushing (from DeMorgan Theorem)

- **Backward:**
  - Cambia la forma della Porta da OR a AND o viceversa
  - Sposta le bolle in input

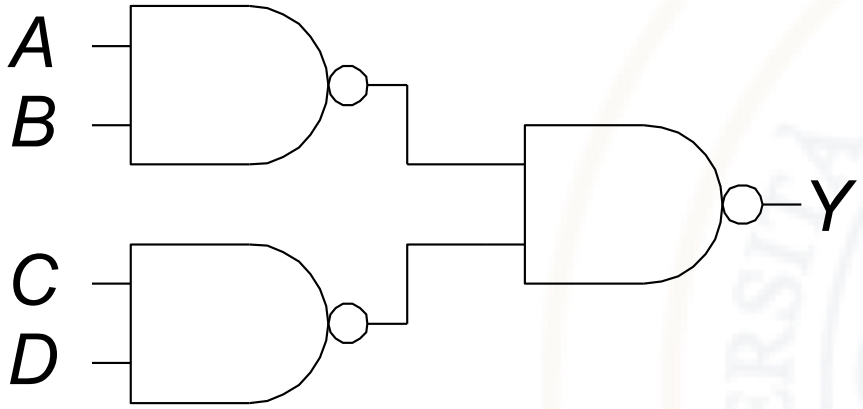


- **Forward:**
  - Cambia la forma della Porta da OR a AND o viceversa
  - Sposta le bolle in output

Bubble Pushing

- Quale è l'espressione booleana per questo circuito?

- $Y = \overline{\overline{AB} + \overline{CD}}$

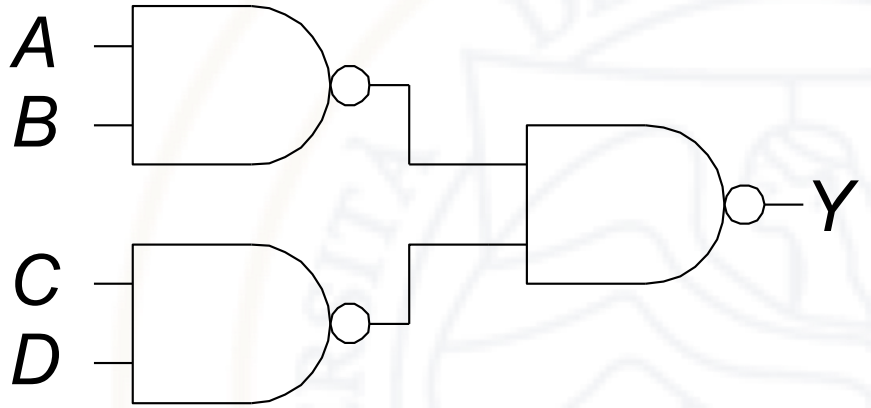


$$Y = AB + CD$$

Bubble Pushing

- Quale è l'espressione booleana per questo circuito?

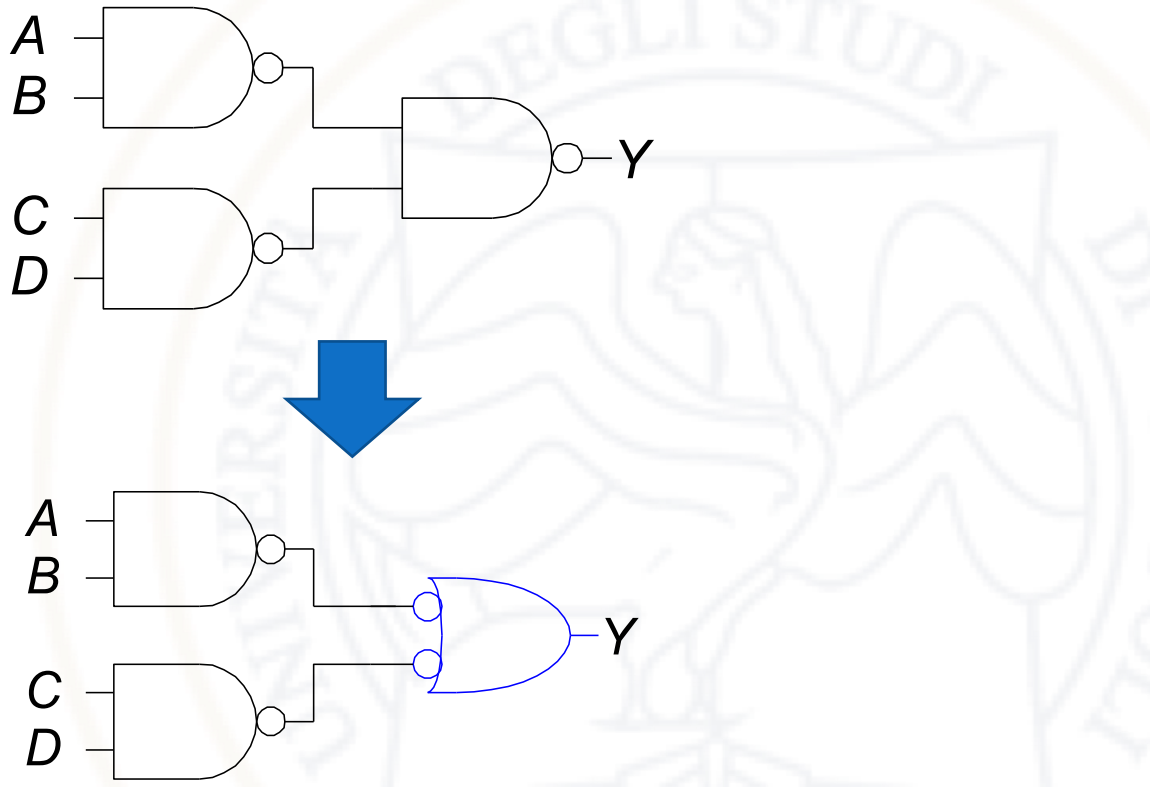
- $Y = \overline{\overline{AB} \cdot \overline{CD}}$



Bubble Pushing

- Quale è l'espressione booleana del seguente circuito?

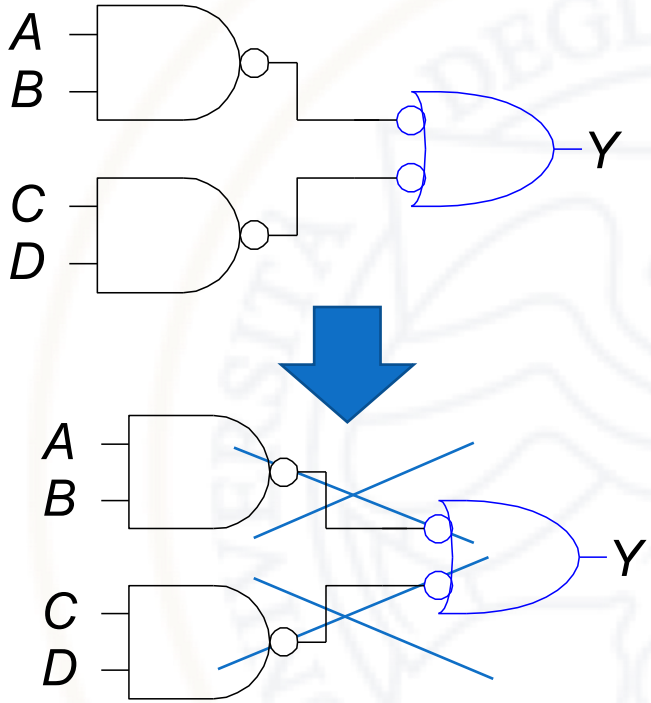
- $Y = \overline{\overline{AB} \cdot \overline{CD}} =$
- $= \overline{\overline{AB}} + \overline{\overline{CD}}$



Bubble Pushing

- Quale è l'espressione booleana del seguente circuito?

- $Y = \overline{\overline{AB} \cdot \overline{CD}} =$
- $= \overline{\overline{AB}} + \overline{\overline{CD}}$
- $= AB + CD$

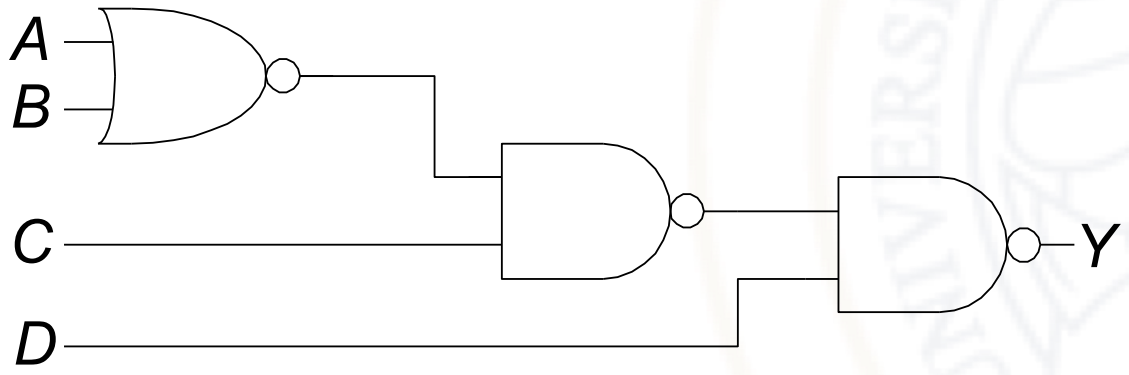


Bubble pushing

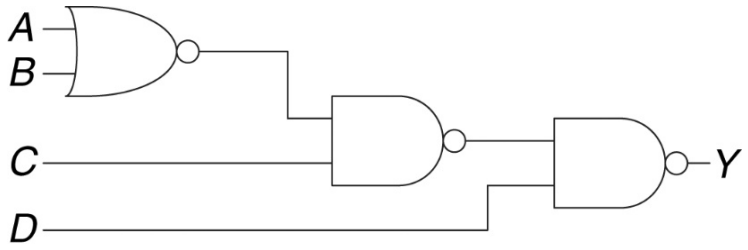
- Usando logiche multilivello e porte NAND/NOR a volte rende difficile capire quale funzione booleana un circuito realizza a causa delle molte negazioni annidate
- Per avere una espressione un po' più leggibile si possono applicare le leggi di De Morgan
- Il bubble pushing è una tecnica che applica sistematicamente 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

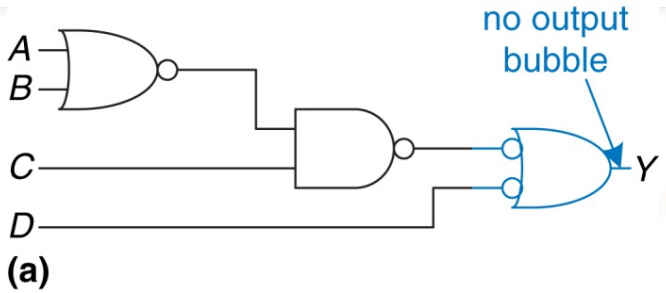
Bubble Pushing Rules

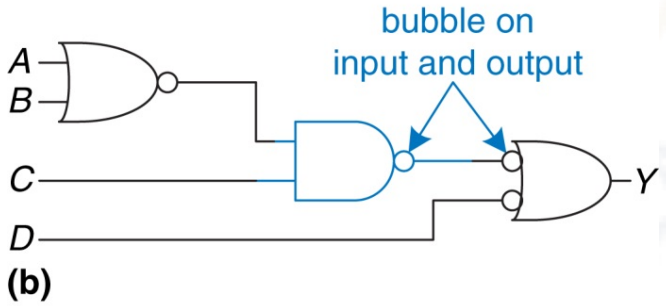
- Inizia dall'output, quindi lavora verso gli input
- Spingi indietro le bolle sull'output finale
- Cambia le porte man mano che appilichi il bubble pushing
- Annulla le bolle annidate

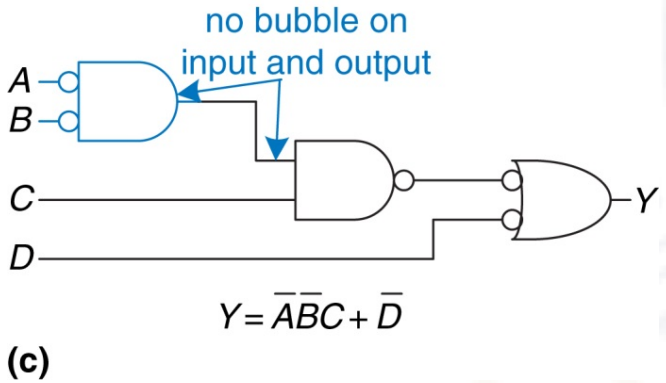


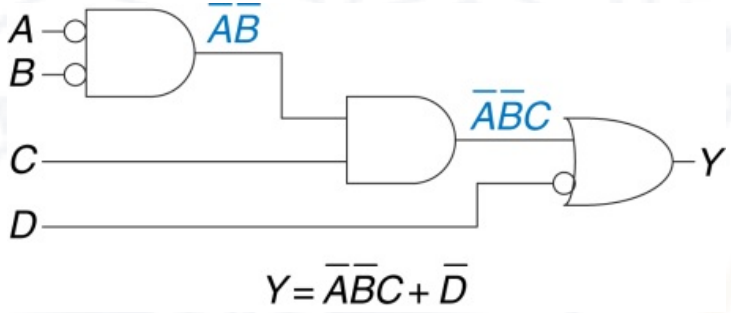
Bubble pushing





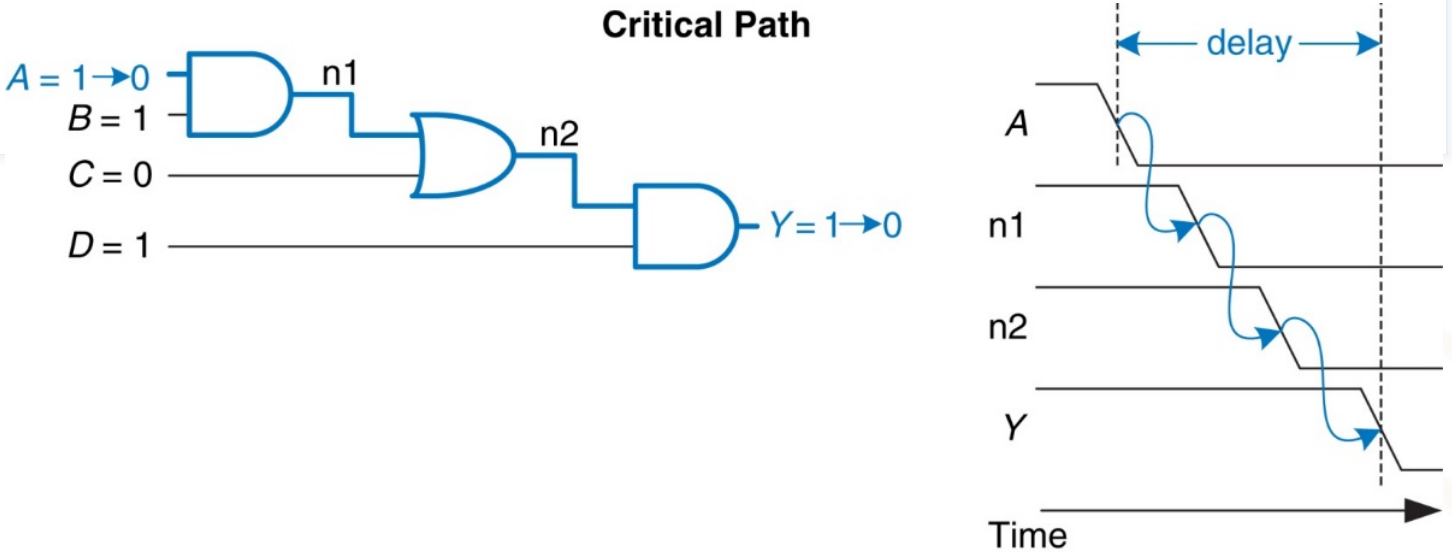


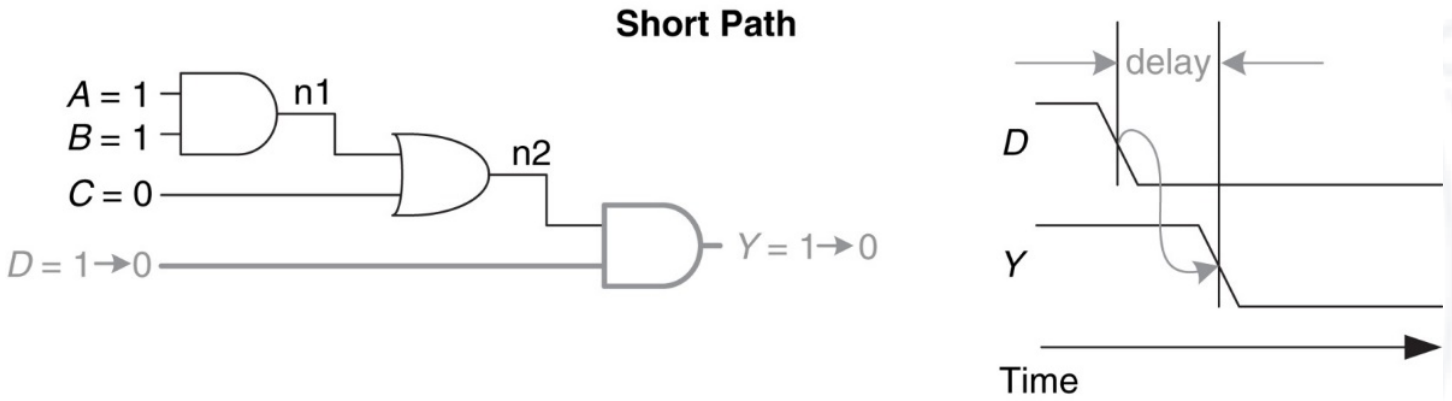




Vantaggi delle SOP/POS

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





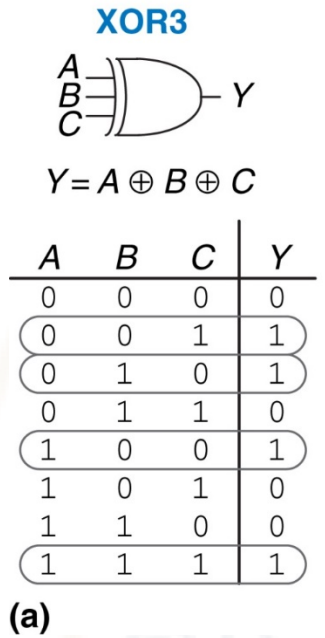
Limiti delle forme SOP/POS

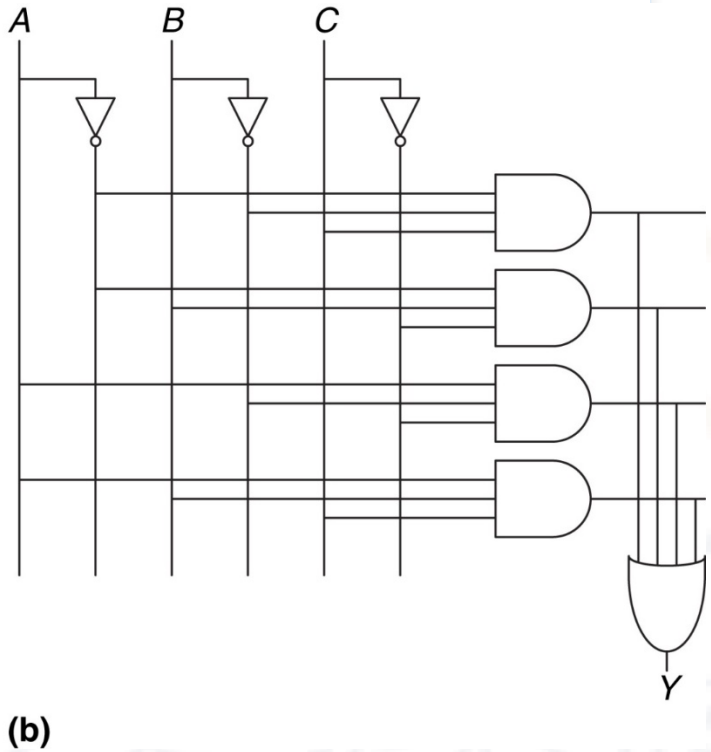
- Tuttavia alcune funzioni booleane, poste in forma SOP, sono estremamente poco succinte e quindi richiedono un numero considerevole di porte
- Presa la tabella di verità di funzione booleana di n variabili:
  - Se il numero di 1 nella colonna di output è piccolo allora la forma SOP è succinta
  - Se 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

- XOR8 richiede 128 AND8 e un OR128

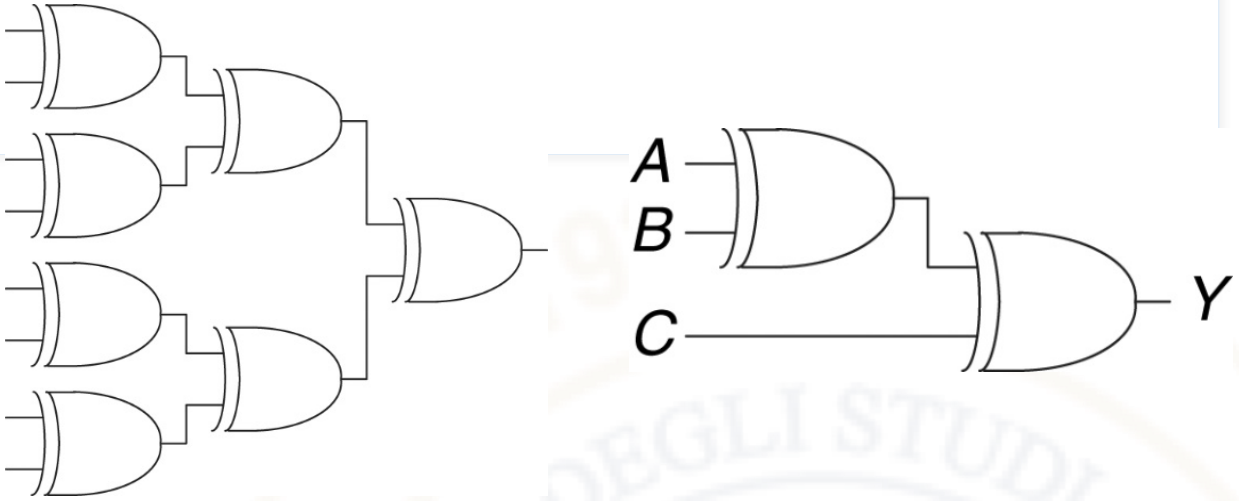




$$Y = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$$

Logiche multilivello

- Per ridurre il numero di porte logiche a volte è necessario ricorrere ad una logica multilivello
- Ad esempio è facile verificare che

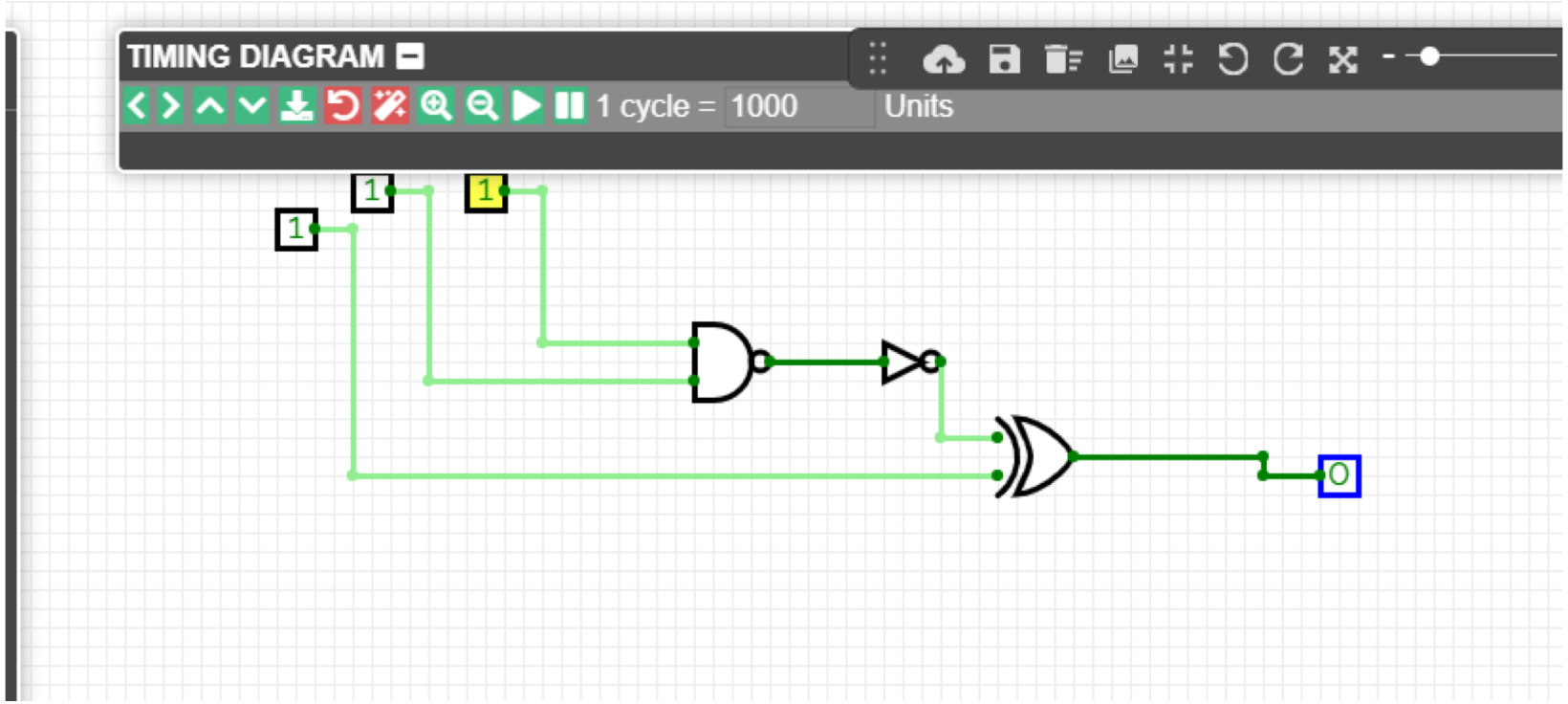


- Analogamente per XOR8:

$$A \oplus B \oplus C = (A \oplus B) \oplus C$$

Simulatore online di Circuiti

https://circuitverse.org/simulator



Regole per gli schemi dei circuiti

Ingressi a sinistra (o in alto)

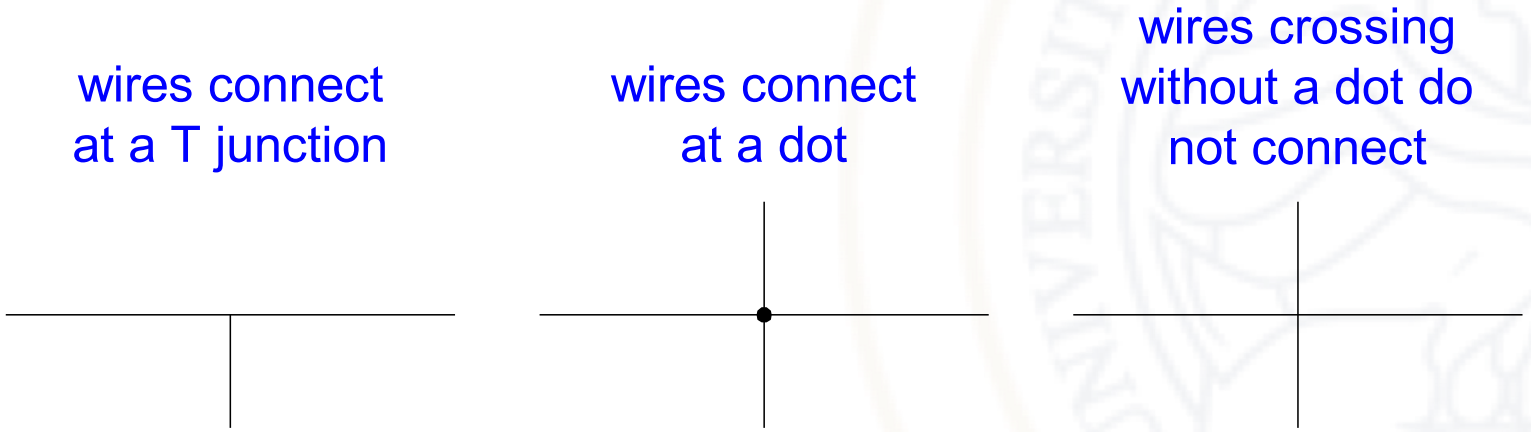
Uscite a destra (o in basso)

I cancelli scorrono da sinistra a destra

Le linee di connessione (wires) dirite sono i migliori

Circuit Schematics Rules

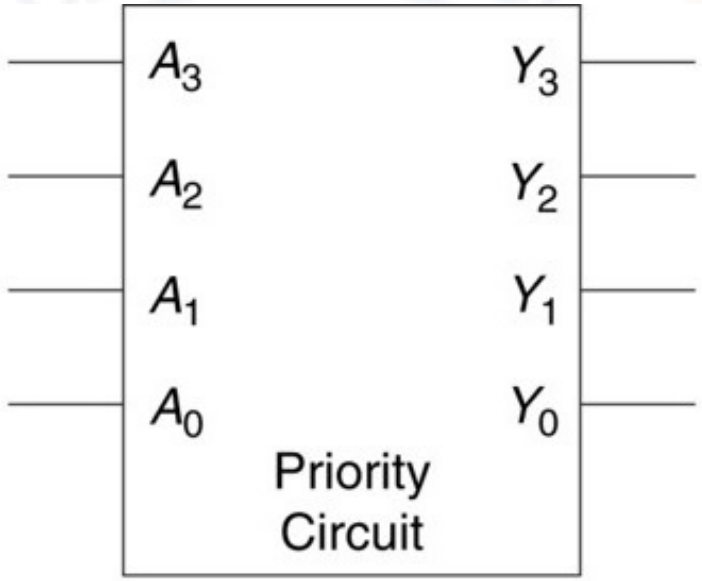
- Le line di trsmissione (fili o wires) si collegano sempre a una giunzione a T
- Un punto in cui le connessioni si incrociano indica una connessione tra i fili
- I fili che si incrociano senza un punto non creano alcuna connessione



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 viene assegnata

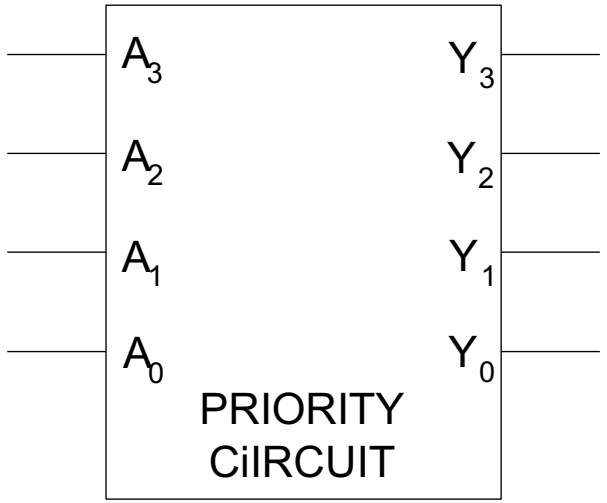
la risorsa, di volta in volta uno solo di essi sarà uguale a 1



Multiple-Output Circuits

Esempio: circuito prioritario

L'uscita prioritaria corrisponde all'input VERO più significativo

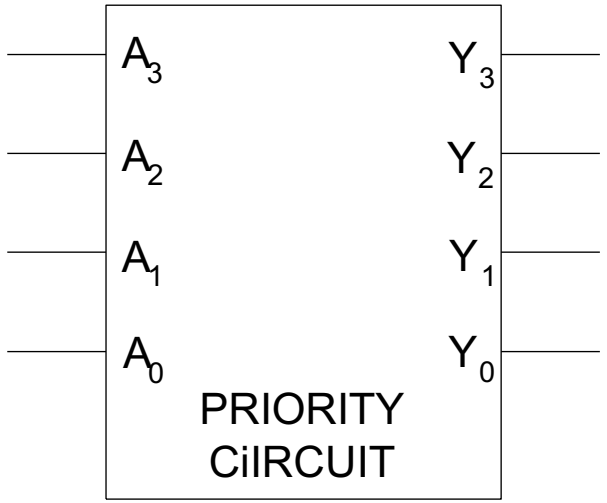


A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

Multiple-Output Circuits

Esempio: circuito prioritario

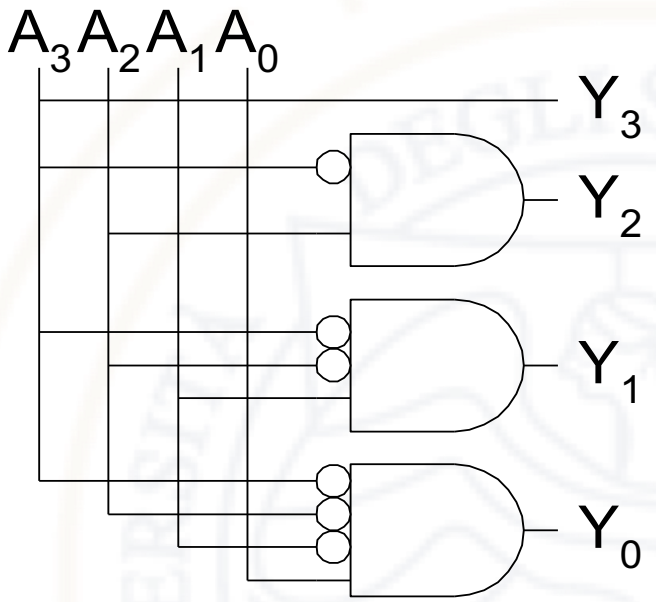
L'uscita prioritaria corrisponde all'input VERO più significativo



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

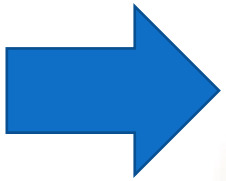
Hardware del circuito prioritario

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



Valori cosiddetti: 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