

# Corso di Calcolatori Elettronici I

---

## Minimizzazione di circuiti combinatori

**ing. Alessandro Cilaro**

Corso di Laurea in Ingegneria Biomedica

# Funzioni logiche elementari

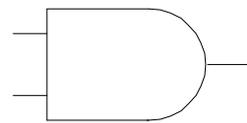
Operazioni fondamentali sui valori booleani (bit)

Corrispondono ai *circuiti elementari* che possiamo usare come “mattoncini” per costruire circuiti più complessi

<b>a</b>	<b>b</b>	<b>y</b>
0	0	0
0	1	0
1	0	0
1	1	1

**AND**

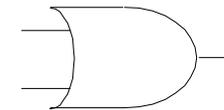
Congiunzione (o prodotto)  
(vale '1' solo se entrambe *a* e *b* sono '1')  
*a* **AND** *b* si indica anche con  $a \cdot b$



<b>a</b>	<b>b</b>	<b>y</b>
0	0	0
0	1	1
1	0	1
1	1	1

**OR**

Disgiunzione (o somma)  
(vale '0' solo se entrambe *a* e *b* sono '0')  
*a* **OR** *b* si indica anche con  $a + b$

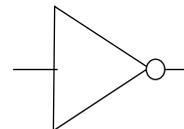


<b>a</b>	<b>y</b>
0	1
1	0

**NOT**

Negazione

**NOT**(*a*) si indica anche con  $\bar{a}$



# Funzioni logiche elementari

## AND e OR generalizzate (a più di due ingressi)

<i>a</i>	<i>b</i>	<i>c</i>	<i>y</i>
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

### AND

Congiunzione (o prodotto)  
(vale '1' solo se **tutte** le  
variabili di ingresso sono  
'1')

$a$  **AND**  $b$  **AND**  $c$  si indica  
anche con  $a \cdot b \cdot c$

<i>a</i>	<i>b</i>	<i>c</i>	<i>y</i>
0	0	0	0
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	1

### OR

Disgiunzione (o  
somma)

(vale '0' solo se **tutte**  
le variabili di ingresso  
sono '0')

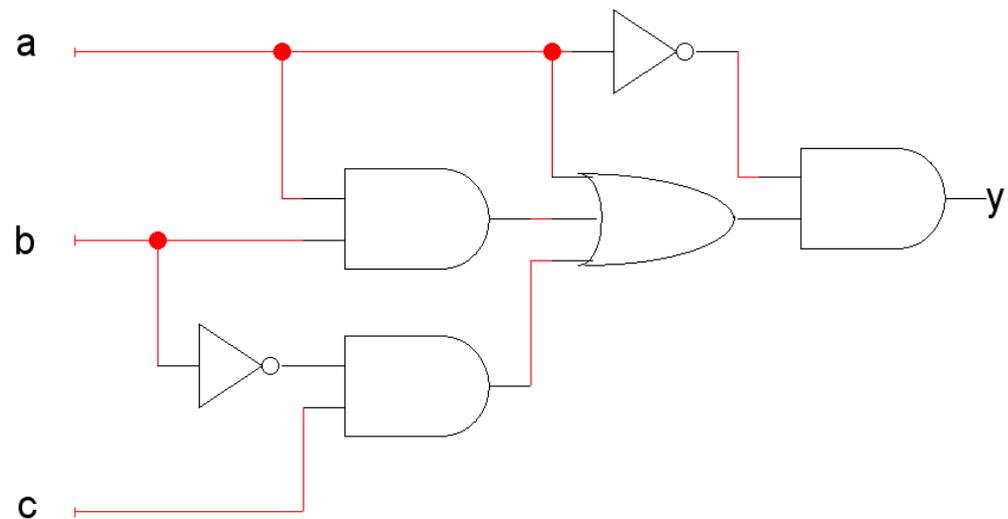
$a$  **OR**  $b$  **OR**  $c$  si indica  
anche con  $a + b + c$

# Progetto di reti logiche

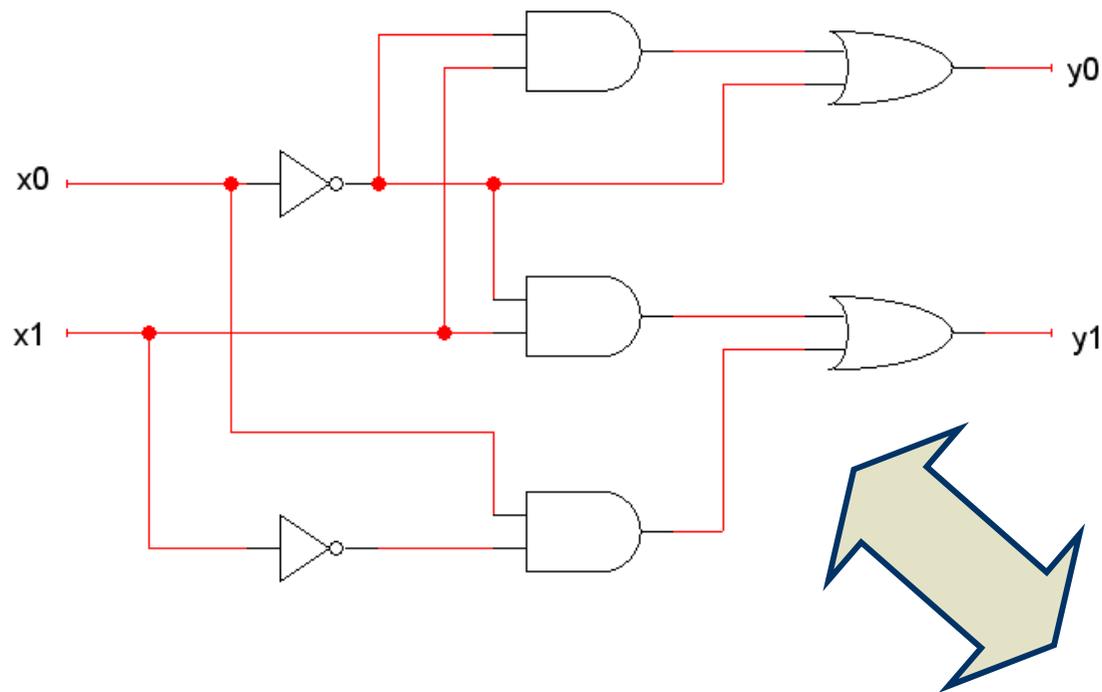
Una funzione logica generica è esprimibile attraverso la *tabella di verità*, che elenca i valori che le uscite devono avere per ciascuna possibile combinazione degli ingressi.

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
0	0	0	0
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

Una tabella di verità può essere realizzata dalla combinazione di circuiti (o porte) elementari

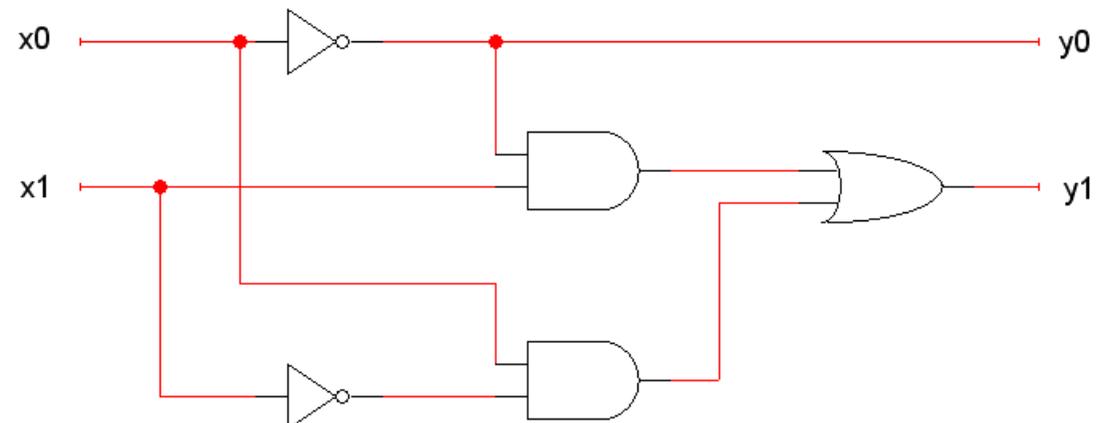


# Equivalenza tra reti logiche



Una stessa funzione logica può essere realizzata da **diversi circuiti**

I due circuiti realizzano la stessa **funzione**, ma hanno **costi differenti** !

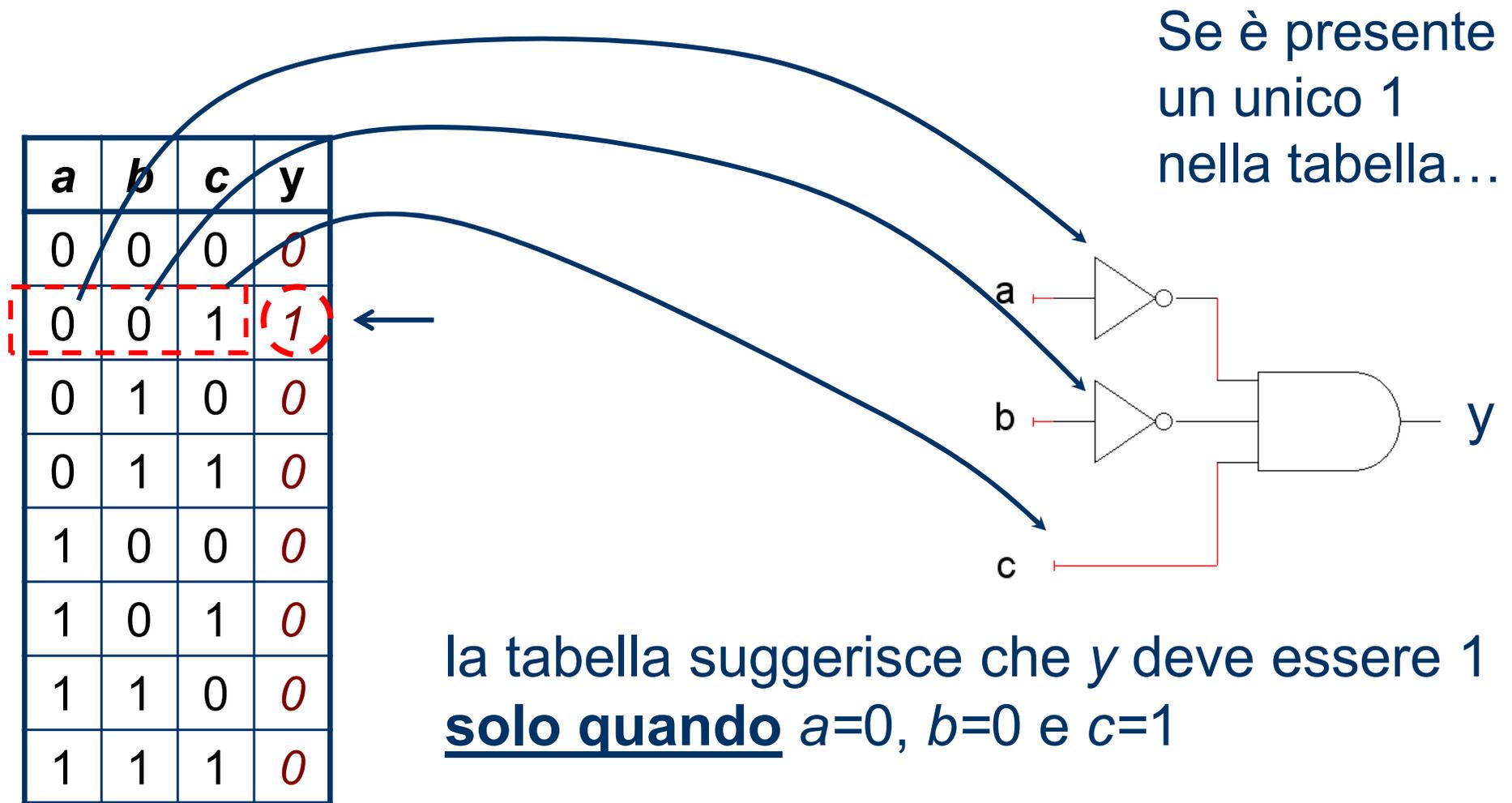


# Progetto di reti logiche

Come realizzare la corrispondenza tra una generica tabella e una sua possibile realizzazione circuitale?

<b><i>a</i></b>	<b><i>b</i></b>	<b><i>c</i></b>	<b><i>y</i></b>
0	0	0	<i>0</i>
0	0	1	<i>1</i>
0	1	0	<i>0</i>
0	1	1	<i>0</i>
1	0	0	<i>0</i>
1	0	1	<i>0</i>
1	1	0	<i>0</i>
1	1	1	<i>0</i>

# Progetto di reti logiche



→ usiamo la porta AND con delle NOT laddove gli ingressi devono essere 0

# Progetto di reti logiche

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

come realizzare una rete corrispondente alla tabella data se sono presenti diversi 1, come accade qui?

osservazione:

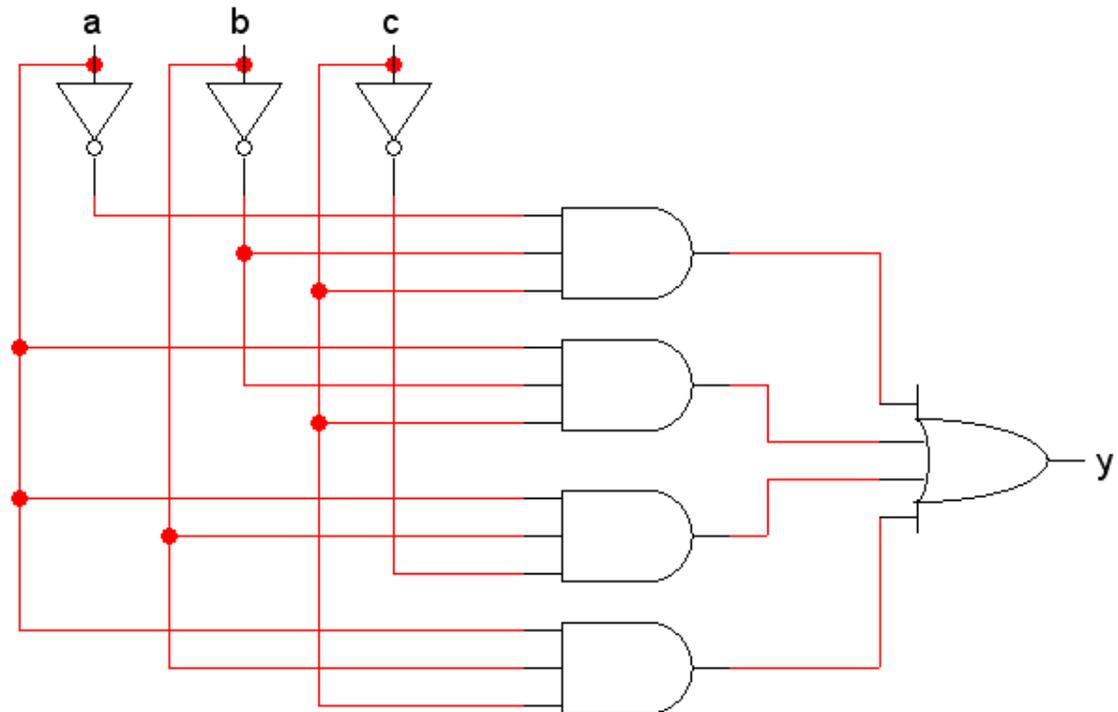
y deve essere alta quando  
 $a=0$  e  $b=0$  e  $c=1$ , **oppure**  
 $a=1$  e  $b=0$  e  $c=1$ , **oppure**  
 $a=1$  e  $b=1$  e  $c=0$ , **oppure**  
 $a=1$  e  $b=1$  e  $c=1$

usiamo tante AND per ogni 1 nella tabella ed una porta OR (che realizza l'**oppure**) con le AND messe in ingresso

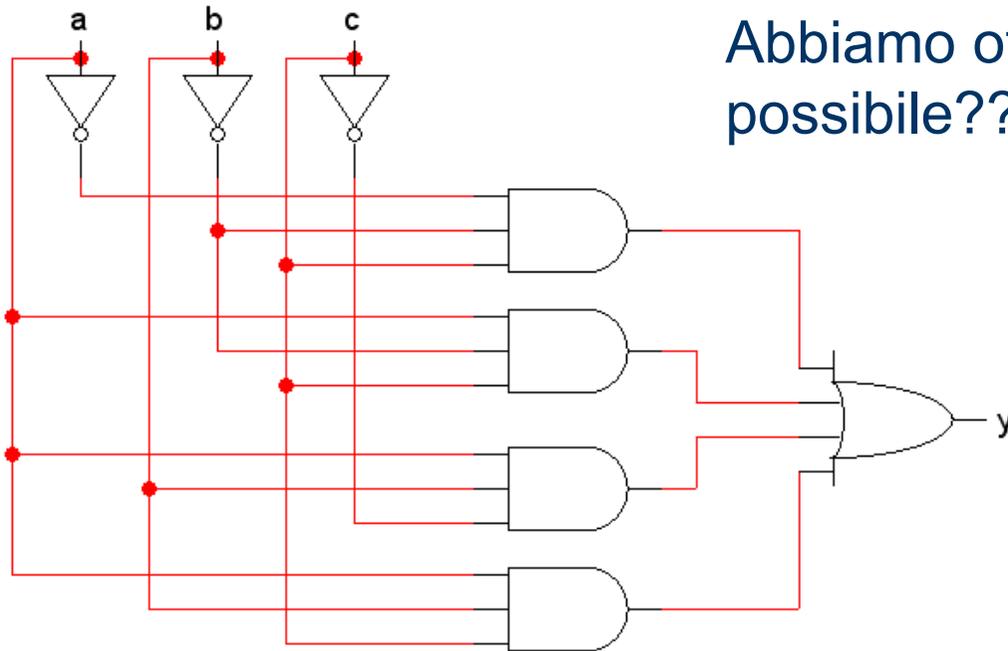
# Progetto di reti logiche

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

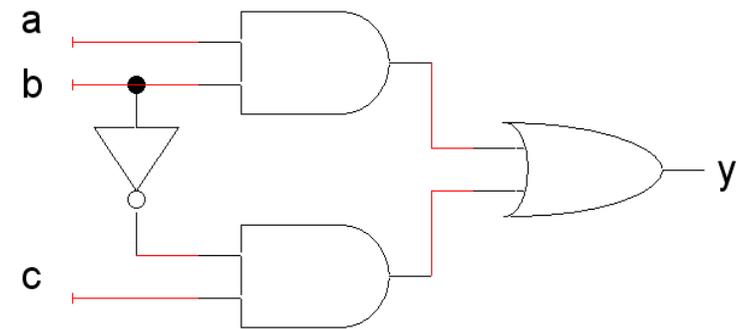
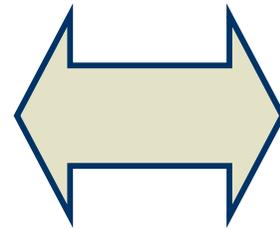
Otteniamo la rete come struttura di tipo NOT-AND-OR (detta anche *somma di prodotti*)



# Progetto di reti logiche



Abbiamo ottenuto la rete migliore possibile??

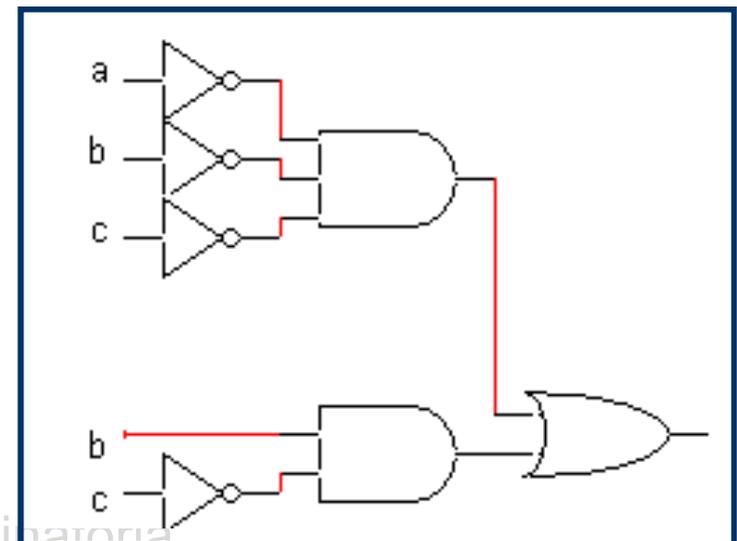
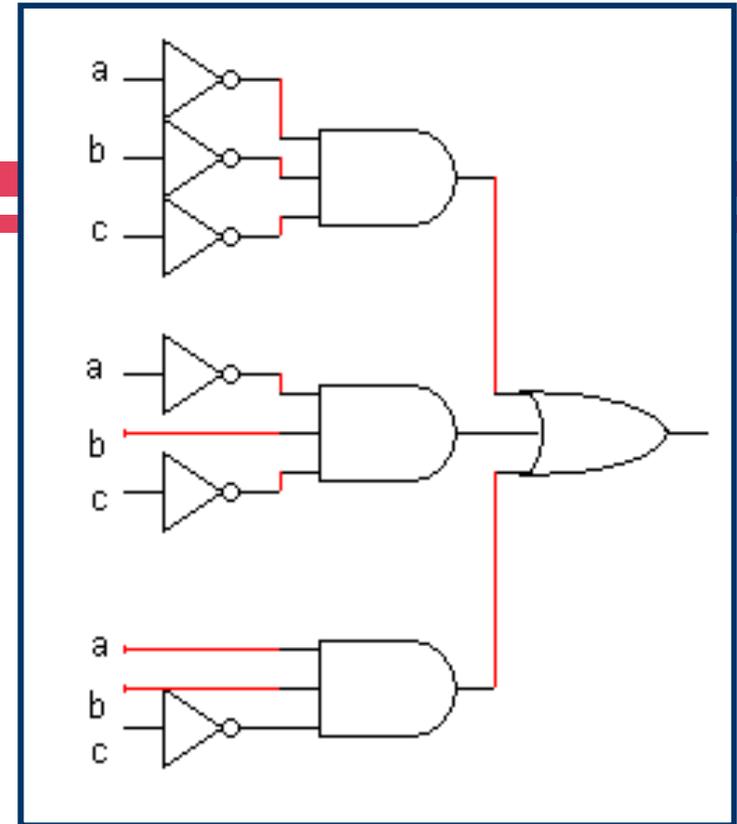
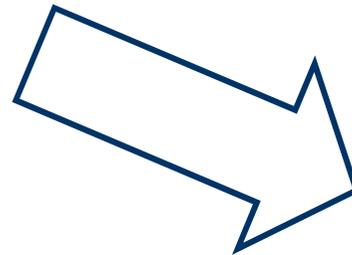
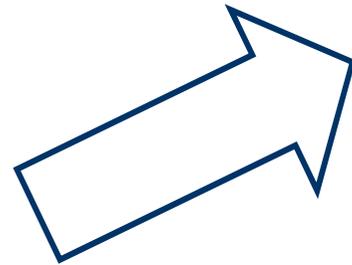


**Osservazione:** la rete sulla destra risponde agli ingressi nella stessa maniera di quella che abbiamo trovato (basta verificare il valore generato per ciascuna combinazione degli ingressi), ma è sicuramente più piccola e meno costosa!!

# Sintesi di Reti Logiche

Altro esempio:

a	b	c	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0



I due circuiti realizzano la stessa **funzione**,  
ma hanno **costi differenti** !

# Teoria delle Reti Logiche

---

- ◆ Come identificare circuiti, o reti, minimi ?
- ◆ risulta chiaro come sia necessario uno strumento formale per l'analisi ed il progetto delle reti logiche...
- ◆ ... usiamo l'Algebra di Boole!

# Alcune definizioni...

$$y = a\bar{b} + b\bar{d}x$$

**SOMMA DI PRODOTTI**

$b, x, \neg d$  sono *letterali*

$$y = a \cdot (b + \bar{c}) \cdot (\bar{a} + x)$$

**PRODOTTO DI SOMME**

termine elementare (*clausola*)

$$f(a, b, c) = \dots + a\bar{b}\bar{c} + \dots$$

fattore elementare

$$f(a, b, c) = \dots (\dots) \cdot (a + \bar{b} + \bar{c}) \cdot (\dots) \dots$$

*mintermine* ( $P_i$ )

*maxtermine* ( $S_i$ )

# Mintermini e Maxtermini

**mintermine** di una funzione di  $n$  ingressi: indica uno qualsiasi dei  $2^n$  prodotti in cui figurano tutte le  $n$  le variabili

**maxtermine** di una funzione di  $n$  ingressi: indica una qualsiasi delle  $2^n$  somme in cui figurano tutte le  $n$  le variabili

$$P_0 = \overline{\overline{a}}\overline{b}\overline{c} \quad P_5 = a\overline{b}\overline{c}$$
$$S_0 = a + b + c \quad S_5 = \overline{a} + \overline{b} + \overline{c}$$

**osservazione:**

0 in binario su tre bit è **000**, 5 è **101**.

**In generale:** Nei mintermini  $P_i$  si nega in corrispondenza dello zero, nei maxtermini  $S_i$  si nega in corrispondenza degli uno

$$\overline{P_i} = S_i \quad (\text{dai teoremi di de Morgan})$$

$$\forall i \neq j \quad P_i \cdot P_j = 0, \quad S_i + S_j = 1$$

$$\sum P_i = 1, \quad \prod S_i = 0$$

# forma normale di tipo P

$$f(x_1, \dots, x_n) = \bar{x}_1 f(0, x_2, \dots, x_n) + x_1 f(1, x_2, \dots, x_n) =$$

$$= \bar{x}_1 [\bar{x}_2 f(0, 0, x_3, \dots, x_n) + x_2 f(0, 1, x_3, \dots, x_n)] +$$

$$x_1 [\bar{x}_2 f(1, 0, x_3, \dots, x_n) + x_2 f(1, 1, x_3, \dots, x_n)]$$

.....

$$= f(0, 0, \dots, 0) \bar{x}_1 \bar{x}_2 \dots \bar{x}_n + \dots + f(1, 1, \dots, 1) x_1 x_2 \dots x_n =$$

$$\sum_{i=0}^{2^n-1} \alpha_i P_i$$

Forma canonica: definisce univocamente la funzione

mintermini

dove

$$\alpha_0 = f(0, 0, \dots, 0), \alpha_1 = f(0, 0, \dots, 1), \dots, \alpha_{2^n-1} = f(1, 1, \dots, 1),$$

“uscite” della funzione: sono gli ‘1’ e ‘0’ della tabella di verità, non sono variabili!

# forma normale di tipo P

---

- ◆ una funzione di  $n$  variabili, assegnata mediante una tabella di verità, può essere espressa algebricamente da una somma di prodotti.
- ◆ Ciascun termine della somma è associato ad un "1" presente nella colonna della tabella ed è un prodotto delle  $n$  variabili, ciascuna delle quali nella forma negata o non a seconda che nelle colonne corrispondenti sia presente uno "0" o un "1".
- ◆ Qualsiasi funzione è pertanto **algebraica**

# forma normale di tipo $P$

- ◆ Esempio:

a	b	c	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

nella forma algebrica si scrive un *mintermine* per ogni **1** presente nella tabella di verità

$$y = \overline{a}\overline{b}\overline{c} + \overline{a}\overline{b}c + \overline{a}b\overline{c} + \overline{a}bc + abc$$

# forma normale di tipo $P$

- ◆ Viceversa, qualsiasi funzione algebrica può essere posta in forma normale  $P$  “aggiungendo” i letterali mancanti
- ◆ Basta sviluppare tutte le operazioni fino ad ottenere una somma di prodotti
- ◆ Le clausole che non siano mintermini (ovvero che non contengano tutti le variabili della funzione) possono essere moltiplicate per la somma di tutte le possibili clausole ottenibili con le variabili assenti
- ◆ *Esempio....*

# forma normale di tipo P

- ◆ Partendo da:

$$y(a,b,c,d) = \bar{a}b + bc + bd$$

- ◆ si può ricavare la forma P moltiplicando il primo termine per  $(c + \bar{c})(d + \bar{d})$   
il secondo per  $(a + \bar{a})(d + \bar{d})$   
ed il terzo per  $(a + \bar{a})(c + \bar{c})$
- ◆ Si otterrà una forma normale di tipo P con 12 mintermini

# forma normale di tipo $S$

---

$$f(x_1 \dots x_n) = \prod_{i=0}^{2^n - 1} (\alpha_i + S_i)$$

- ◆ si può ottenere con il procedimento duale di quello usato per la forma di tipo  $P$
- ◆ In alternativa, si può negare la forma di tipo  $P$  e poi applicare de Morgan

# forma normale di tipo $S$

---

- ◆ una funzione di  $n$  variabili può essere espressa algebricamente da un prodotto di somme;
- ◆ ciascun fattore del prodotto è associato ad uno  $0$  presente nella colonna della tabella ed è una somma delle  $n$  variabili, ciascuna delle quali nella forma negata o non a seconda che nelle colonne corrispondenti sia presente un  $1$  o uno  $0$ .

# forma normale di tipo S

## ◆ Esempio

a	b	c	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

nella forma algebrica si scrive  
un ***maxtermine*** per ogni **0**  
presente nella tabella di verità



$$y = (\bar{a} + b + c) \cdot (\bar{a} + b + \bar{c}) \cdot (\bar{a} + \bar{b} + c)$$



# Implicanti di una funzione

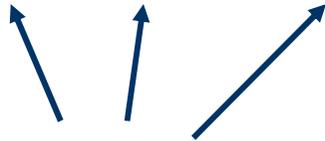
- ◆ Un implicante di  $f$  è una funzione  $h$  tale che

$$\bar{h} + f = 1$$

In altri termini: se  $h$  è 1, anche  $f$  **deve** necessariamente essere 1

- ◆ Esempi:

$$f = ab + abc + cd$$



tutte le clausole in una forma SOP (somma di prodotti) sono implicanti!

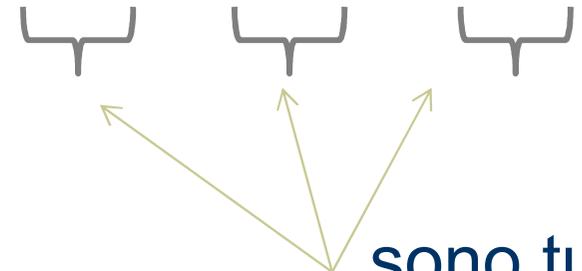
$$f = a + ((a \oplus d) \uparrow b) \equiv (cd)$$



$a$  è sicuramente un implicante

# Implicanti di una funzione

- ◆ in una funzione scritta come *somma di prodotti*, ciascun prodotto rappresenta un implicante
  - infatti in una somma, se è alto anche solo un termine, è alta l'intera somma

$$y = \overline{a}\overline{b}\overline{c} + \overline{a}bc + abc$$


sono tutti implicanti. Ad esempio, se è alto  $abc$ , è alta anche  $y$

# Implicanti primi

- ◆ Un implicante di una funzione  $f$  si dice **primo** se non implica a sua volta nessun altro implicante della funzione

$$f = ab + abc + cd$$

$abc$  è un implicante **non primo**, poiché  
implica a sua volta  $ab = abc + abc$

$ab$  è un implicante **primo**

- ◆ Una funzione scritta come somma solo di implicanti primi si dice scritta in **forma PI**

# Implicanti primi

- ◆ I termini di una somma sono sempre implicanti, ma non è detto che siano implicanti primi:

$$y = \bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}\bar{c} = \bar{a}\bar{b}\bar{c} + bc(\bar{a} + a) = \bar{a}\bar{b}\bar{c} + bc$$

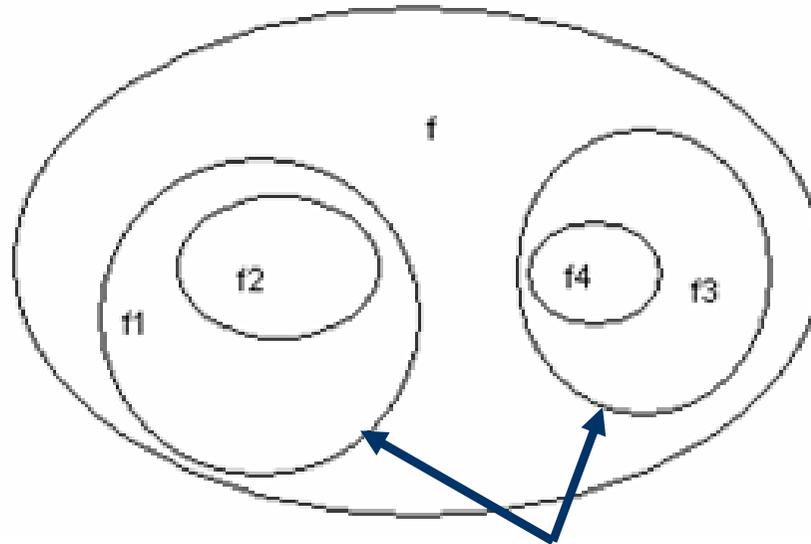
implicazioni:

$$abc \rightarrow bc \rightarrow y$$

Esempio:  $abc$ , è un implicante di  $y$ , ma anche di  $bc$ , quindi non è primo.  $bc$ , invece, non implica nient'altro a parte la funzione  $y$  stessa, quindi è *primo*

# Implicanti primi

- ◆ Un'interpretazione insiemistica:
- ◆ Tutti i sottoinsiemi di  $f$  sono suoi implicanti



- ◆ Solo  $f_1$  ed  $f_3$  sono implicanti primi

# Proprietà degli implicanti

1. Una clausola  $Y$  ne implica un'altra  $X$  se e solo se  $Y$  contiene tutti i letterali di  $X$

$$bc = bc(\bar{a} + a) = \bar{a}bc + abc$$

Esempio:

$abc$  implica  $bc$  poiché ne contiene tutti i letterali

2. La somma di due clausole di ordine  $n$  che contengono  $n-1$  letterali uguali ed in cui un letterale dell'una sia il complemento di quello dell'altra è la clausola di ordine  $n-1$  formata dai letterali comuni (detta **consenso**)  
*nell'esempio al punto 1 c'è un consenso che genera  $bc$*

# Funzioni di costo

$$f = bc(a\bar{d} + \bar{b} + c) + \bar{c}(d + \bar{a})(b + c)$$

<b>Costo di letterali (CL)</b>	<b>11</b>
<b>Costo di funzioni o di porte (CP)</b>	<b>7</b>
<b>Costo di ingressi (CI)</b>	<b>17</b>

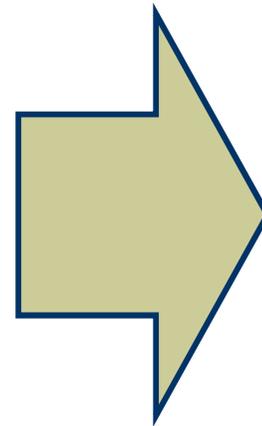
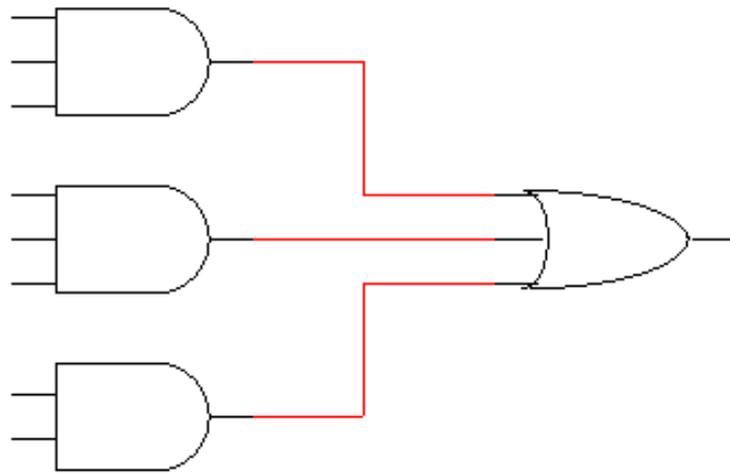
# Implicanti e funzioni di costo

---

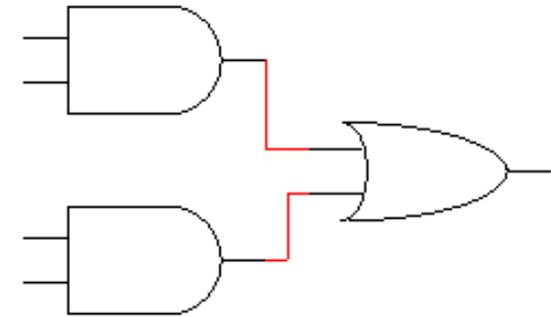
- ◆ una forma elementare che minimizzi i valori dei costi  $CL$  e  $CI$  è una forma  $PI$
- ◆ fra le forme minime a 2 livelli che minimizzano  $CP$  ne esiste almeno una  $PI$
- ◆ sotto il vincolo di rete a 2 livelli, la forma minima va allora cercata tra le forme  $PI$

# Implicanti e funzioni di costo

$$a\bar{b}\bar{c} + abc + cd$$



$$ab + cd$$



Forma **non-PI**

$$CL = 8$$

$$CI = 11$$

$$CP = 4$$

Forma **PI**

$$CL = 4$$

$$CI = 6$$

$$CP = 3$$

# Ricerca di forme minime

---

funzione  $f$  da minimizzare



riscrittura della funzione  $f$  in forma PI



scelta del sottoinsieme ottimo di PI la cui somma dà ancora  $f$

# Alcuni definizioni

- ◆ Un implicante primo  $E_i$  di una funzione  $f$  è detto *essenziale* se è l'unico ad essere implicato da una mintermine di  $f$
- ◆ In altri termini,  $E_i$  è l'unico a “coprire” un determinato mintermine della funzione
- ◆ Il *nucleo*  $N$  della funzione è la somma dei suoi implicanti primi essenziali

$$N = \sum_{i=1}^k E_i$$

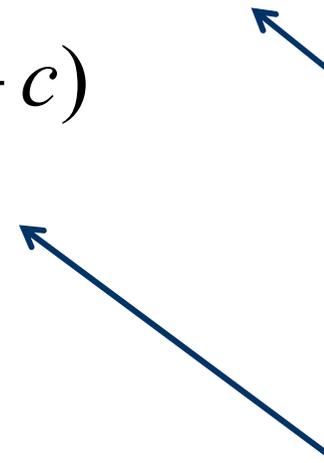
- ◆ Ogni forma minima di  $f$  è del tipo  $f = N + R$  con  $N$  e  $R$  (insieme degli implicanti primi non essenziali) eventualmente nulli

# Ricerca di Implicanti Primi (PI)

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$$\begin{aligned}f(a, b, c) &= \bar{a}bc + a\bar{b}\bar{c} + a\bar{b}c + abc \\ &= bc(\bar{a} + a) + a\bar{b}(\bar{c} + c) \\ &= bc + a\bar{b}\end{aligned}$$

scriviamo la funzione come somma di *mintermini*, ciascuno corrispondente ad un '1' nella tabella di verità. Cerchiamo poi di semplificare l'espressione con passaggi algebrici



# Ricerca di Implicanti Primi (PI)

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$$\begin{aligned} f(a,b,c) &= \bar{a}bc + \dots + abc \\ &= bc(\bar{a} + a) + \dots \\ &= bc + \dots \end{aligned}$$

scriviamo la funzione come somma di *mintermini*, ciascuno corrispondente ad un '1' nella tabella di verità

abbiamo potuto raccogliere due termini e semplificare l'espressione poiché tra le due clausole è presente un **consenso**

$$\begin{array}{l} \bar{a}bc \\ abc \end{array}$$

# Rappresentare le funzioni booleane

- ◆ Consenso:
  - presenza di due clausole (non necessariamente mintermini) aventi gli stessi letterali, tranne uno presente negato in una clausola e non negato nell'altra
- ◆ Esempi...

$$\bar{a}bc + \dots + abc = \dots + bc + \dots \quad \dots + a\bar{b}\bar{c} + \dots + abc = \dots + ac$$

# Ricerca di PI: metodo agebrico (Quine)

---

- ◆ Scrivere la funzione in forma P (somma di mintermini)
- ◆ cercare tutti i possibili consensi tra coppie qualsiasi di clausole (aventi lo stesso numero di letterali)
- ◆ al passaggio successivo, eliminare clausole che sono state “assorbite” in clausole più piccole (poiché corrispondono ad implicanti non primi)
- ◆ iterare i raccoglimenti per clausole di dimensione via via minore
- ◆ quando non sarà più possibile individuare consensi, la funzione sarà scritta in forma PI

# Metodo agebrico: esempio

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

Forma P

$$\bar{a}b + \bar{a}c + bc + \bar{b}c + ac =$$

$$\bar{a}b + c + c = \bar{a}b + c$$

# Mappe di Karnaugh

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
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

bc	00	01	11	10
a				
0			1	1
1			1	



E' semplicemente un altro modo di disporre la tabella di verità



a differenza della tabella di verità, tutte le celle adiacenti corrispondono ad un consenso

(è il motivo per cui si sceglie la sequenza di valori 00, 01, 11, 10)

# Mappe di Karnaugh

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
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

bc	00	01	11	10
a				
0			1	1
1			1	

$$\bar{a}b = \bar{a}bc + \bar{a}b\bar{c}$$

$$bc = \bar{a}bc + abc$$



a differenza della tabella di verità, tutte le celle adiacenti corrispondono ad un consenso

(è il motivo per cui si sceglie la sequenza di valori 00, 01, 11, 10)

# Mappe di Karnaugh

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
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

bc	00	01	11	10
a			1	1
0			1	
1			1	

$\bar{a}bc + abc = bc$

$\bar{a}bc + abc = bc$

Sulla mappa di Karnaugh tutte le celle adiacenti corrispondono ad un consenso. Ciò non accade sempre nella tabella di verità!

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}\bar{b}c + \bar{a}\bar{b}\bar{c} + abc + a\bar{b}c =$$

$$\bar{a}b + \bar{a}c + bc + \bar{b}c + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}\bar{b}c + \bar{a}\bar{b}\bar{c} + abc + a\bar{b}c =$$

$$\bar{a}b + \bar{a}c + bc + \bar{b}c + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \boxed{\bar{a}c} + bc + \bar{b}c + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \bar{a}c + \boxed{bc} + \bar{b}c + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \bar{a}c + bc + \boxed{\bar{b}c} + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \bar{a}c + bc + \bar{b}c - \boxed{ac} =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \bar{a}c + \boxed{bc} + \boxed{\bar{b}c} + ac =$$

$$\bar{a}b + c =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

# Mappe di Karnaugh

$$f(a, b, c) = \bar{a}(b + c) + ac$$

$$\bar{a}b + \bar{a}c + ac =$$

$$\bar{a}b(c + \bar{c}) + \bar{a}(b + \bar{b})c + a(b + \bar{b})c =$$

$$\bar{a}bc + \bar{a}b\bar{c} + \bar{a}\bar{b}c + a\bar{b}c + a\bar{b}\bar{c} =$$

$$\bar{a}b + \bar{a}c + bc + \bar{b}c + ac =$$

$$\boxed{\bar{a}b} + \boxed{c} =$$

		bc			
		00	01	11	10
a	0		1	1	1
	1		1	1	

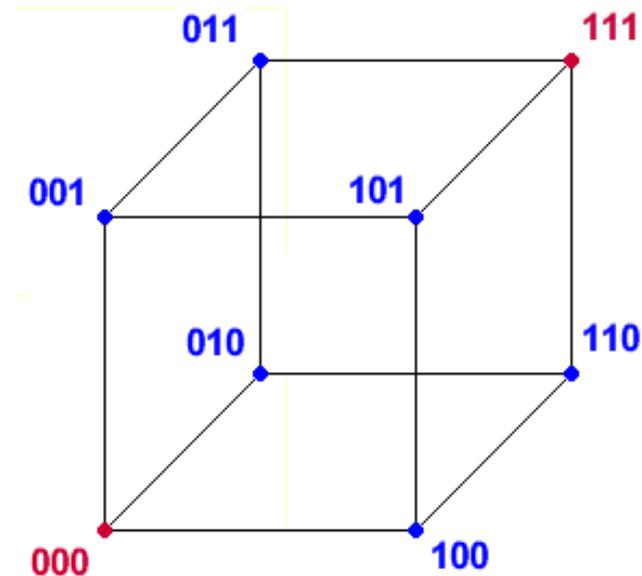
Gruppi di celle sulle mappe di Karnaugh che corrispondono ad una clausola (ad es.  $\bar{a}b$  oppure  $c$ ) sono detti **cubi**

# Mappe di Karnaugh

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
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

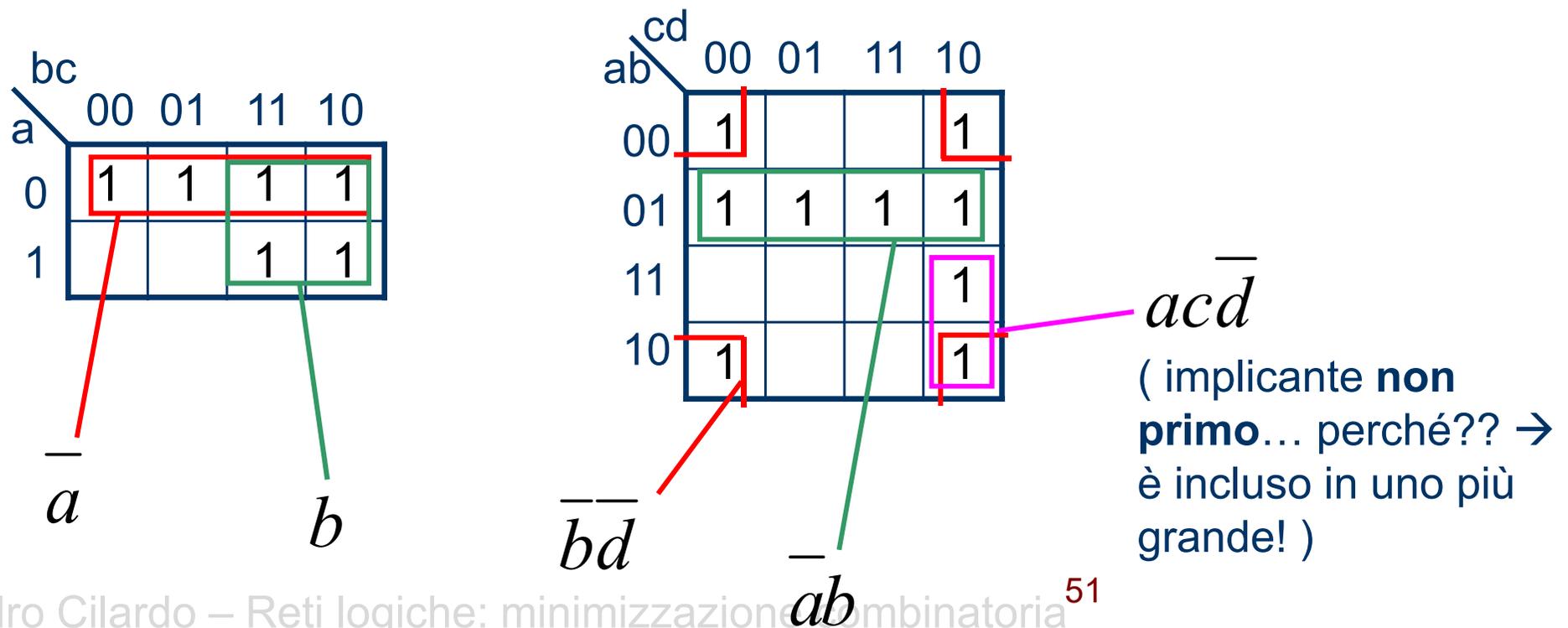
bc	00	01	11	10
a				
0			1	1
1			1	

tutte le celle adiacenti  
corrispondono ad un  
consenso



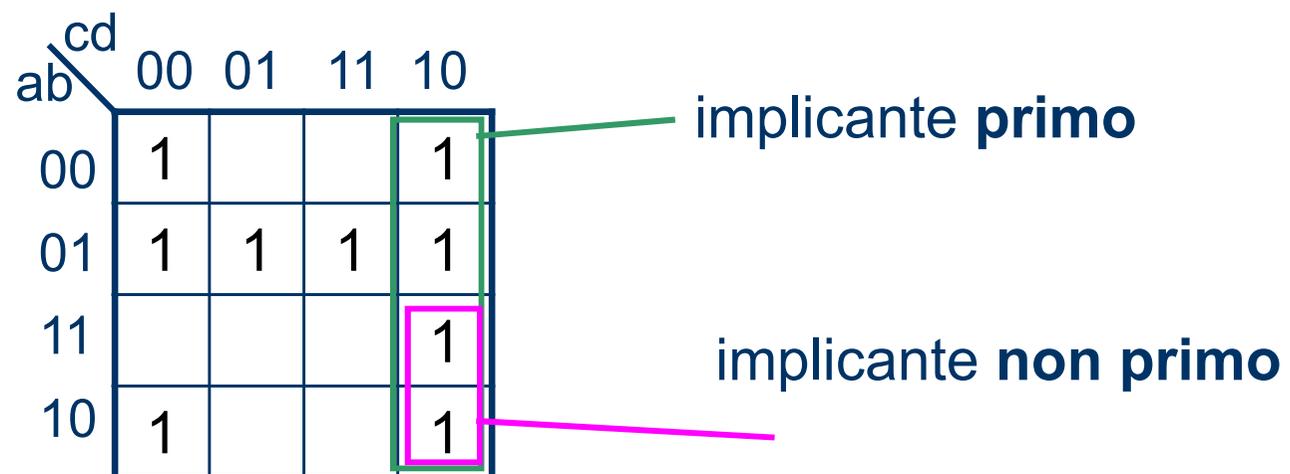
# Mappe di Karnaugh

Vari modi per identificare **cubi** su una mappa di Karnaugh.  
 La caratteristica essenziale di un cubo è quella di contenere alcune variabili sempre con lo stesso segno, mentre le rimanenti assumo tutte le possibili combinazioni di valori negati o non negati (un cubo conterrà quindi sempre un numero di celle pari ad una potenza di 2)



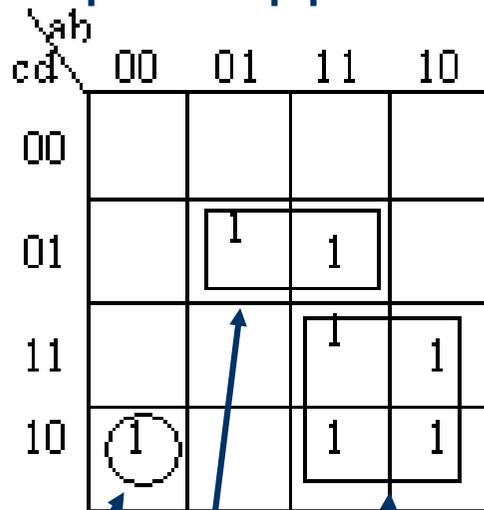
# PI su mappe di Karnaugh

- ◆ La ricerca di Implicanti Primi sulle mappe di Karnaugh è effettuata semplicemente cercando tutti i cubi più larghi (corrispondenti a clausole più piccole) che coprano celle con '1'

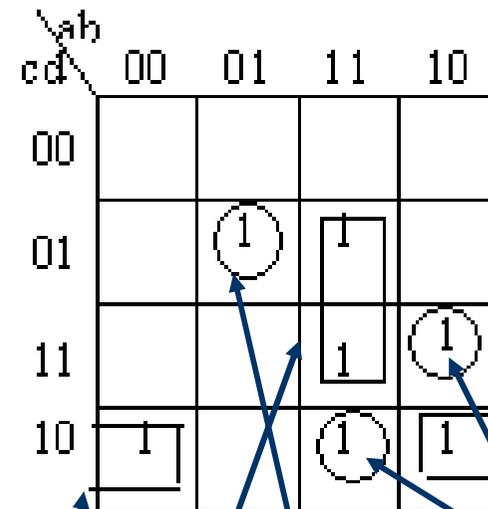


# Mappe di Karnaugh

Due modi per rappresentare la stessa funzione:



a)



b)

$$a) y_1 = \overline{a}\overline{b}c\overline{d} + \overline{b}c\overline{d} + ac \quad b) y_2 = \overline{b}c\overline{d} + abd + \overline{a}b\overline{c}d + \overline{a}b\overline{c}d + \overline{a}b\overline{c}d$$

La forma a sinistra contiene più implicant primari ed è quindi migliore in termini di costo. Non è tuttavia ancora minima. L'implicante *non primo*  $\overline{a}\overline{b}c\overline{d}$  si potrebbe ulteriormente sostituire con l'implicante *primo*  $\overline{b}c\overline{d}$

# Rappresentare le funzioni booleane

<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>y</i>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

ab \ cd	00	01	11	10
00		1		
01	1		1	1
11			1	1
10			1	

**Esercizio:** individuare la forma PI (somma di implicantii primi) con metodo grafico

# Esercizio

---

- ◆ Scrivere la seguente funzione in forma PI usando il metodo algebrico:

$$f(a, b, c) = (a \equiv c) \cdot b + \bar{a}b$$

# Esercizio

- ◆ Scrivere la seguente funzione in forma PI usando il metodo algebrico:

$$\begin{aligned} f(a,b,c) &= (a \equiv c) \cdot b + \bar{a}b = \\ &= (ac + \bar{a}\bar{c}) \cdot b + \bar{a}b(c + \bar{c}) = \\ &= abc + \bar{a}b\bar{c} + \bar{a}bc + \bar{a}b\bar{c} = \\ &= bc + \bar{a}b \end{aligned}$$

# Esercizio

- ◆ Scrivere la seguente funzione in forma PI usando il metodo algebrico:

$$\begin{aligned} f(a,b,c) &= (a \equiv c) \cdot b + \bar{a}b = \\ &= (ac + \bar{a}\bar{c}) \cdot b + \bar{a}b(c + \bar{c}) = \\ &= abc + \bar{a}b\bar{c} + \bar{a}bc + \bar{a}b\bar{c} = \\ &= bc + \bar{a}b \end{aligned}$$

bc	00	01	11	10
a				
0			1	1
1			1	

# Metodo di McCluskey

---

- ◆ ciascuna clausola della funzione viene indicata da una stringa di  $1$ ,  $0$  e  $'-'$  :
  - $1$  per le variabili in forma affermata,
  - $0$  per quelle in forma negata
  - $-$  per le variabili che non compaiono nel prodotto

# Esempi di clausole (McCluskey)

- ◆ Convenzioni usate per indicare clausole nel metodo di McCluskey:

$$\bar{a}bc \longrightarrow 011 \quad \bar{a}bd \longrightarrow 01-1$$

$$\bar{a}\bar{b}cd \longrightarrow 0101 \quad \bar{a}bc \longrightarrow 011-$$

} in questo caso, stiamo considerando una funzione avente quattro variabili  $a$ ,  $b$ ,  $c$ ,  $d$ , e non tre come nel primo esempio  $\rightarrow$  i simboli rappresentati sono quattro

# Perché questa convenzione?

- ◆ stessa idea delle mappe di Karnaugh:
  - identificare facilmente consensi tra clausole

$\bar{a}bc \longrightarrow 011$	$\bar{a}bc\bar{d} \longrightarrow 0110$	$\bar{a}bc \longrightarrow 011$	$\bar{a}bd \longrightarrow 01-1$
$abc \longrightarrow 111$	$\bar{a}\bar{b}\bar{c}\bar{d} \longrightarrow 0100$	$abc \longrightarrow 110$	$abd \longrightarrow 11-1$



in questi casi c'è consenso



c'è consenso

in questo caso non c'è consenso

# Perché questa convenzione?

$$\begin{array}{lclclcl} \bar{a}bc + abc = bc & 011 & + & 111 & = & -11 \\ \bar{a}bc\bar{d} + \bar{a}bc\bar{d} = \bar{a}bc & 0110 & + & 0100 & = & 01-0 \\ \bar{a}bc + abc & 011 & + & 110 & & \\ \bar{a}bd + abd = bd & 01-1 & + & 11-1 & = & -1-1 \end{array}$$

- ◆ Osservazione: coppie di clausole che differiscono per un solo simbolo (**incluso il trattino -**) generano un consenso
- ◆ ogni coppia si può sostituire con un'unica clausola avente un trattino al posto del simbolo discordante

# Metodo di McCluskey

- ◆ Si scrive la funzione in forma P (somma di mintermini)
- ◆ ciascun mintermine (in quanto particolare clausola) viene denotato con la convenzione vista prima (1, 0 o '-' per ciascuna variabile della funzione)
- ◆ Le clausole individuate vengono suddivise in classi contenenti elementi con equal numero di 1
  - questa scelta facilita l'individuazione di eventuali consensi, riducendo il numero di confronti

# Metodo di McCluskey

<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>y</b>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

esempio di inizializzazione della tabella di McCluskey

Implicanti del 4° ordine	
1) 0001	← mintermini contenenti un solo '1'
-----	
2) 0011	← mintermini contenenti due '1'
3) 1100	← mintermini contenenti due '1'
-----	
4) 0111	
5) 1101	← mintermini contenenti tre '1'
6) 1110	
-----	
7) 1111	← mintermine contenente quattro '1'

# Metodo di McCluskey

- ◆ Per ogni  $i$  da 0 a  $k-1$ : si generano i consensi di ordine  $k-1$  accoppiando le clausole della "classe  $i$ " con quelle della "classe  $i+1$ "; si marcano le clausole che generano consenso
- ◆ Le clausole non marcate sono **implicanti primi** di ordine  $k$ . Nei consensi generati si eliminano gli eventuali doppi; i consensi stessi sono ordinati e riorganizzati da "classe 0" a "classe  $k-1$ ".
- ◆ Si pone  $k=k-1$  e si ripetono i precedenti passi finché non sia  $k \geq 0$ .

Implicanti del 4° ordine		
1)	0001	✓
-----		
2)	0011	✓
3)	1100	✓
-----		
4)	0111	✓
5)	1101	✓
6)	1110	✓
-----		
7)	1111	✓

# Metodo di McCluskey: esempio

Implicanti del 4 <sup>o</sup> ordine		
1)	0001	✓
-----		
2)	0011	✓
3)	1100	✓
-----		
4)	0111	✓
5)	1101	✓
6)	1110	✓
-----		
7)	1111	✓

# Metodo di McCluskey: esempio

Implicanti del 4° ordine	Implicanti del 3° ordine
1) 0001 ✓ -----	1-2) 00-1 -----
2) 0011 ✓	2-4) 0-11
3) 1100 ✓ -----	3-5) 110- ✓
4) 0111 ✓	3-6) 11-0 ✓ -----
5) 1101 ✓	4-7) -111
6) 1110 ✓ -----	5-7) 11-1 ✓
7) 1111 ✓	6-7) 111- ✓

# Metodo di McCluskey: esempio

Implicanti del 4° ordne	Implicanti del 3° ordne	Implicanti del 2° ordne
1) 0001 ✓ -----	1-2) 00-1 -----	----- 3-5-6-7) 11--
2) 0011 ✓ -----	2-4) 0-11 -----	3-6-5-7) 11--
3) 1100 ✓ -----	3-5) 110- ✓ -----	
4) 0111 ✓ -----	3-6) 11-0 ✓ -----	
5) 1101 ✓ -----	4-7) -111 -----	
6) 1110 ✓ -----	5-7) 11-1 ✓ -----	
7) 1111 ✓ -----	6-7) 111- ✓ -----	

# Metodo di McCluskey: esempio

Implicanti del 4° ordne	Implicanti del 3° ordne	Implicanti del 2° ordne
1) 0001 ✓	1-2) 00-1	-----
-----	-----	3-5-6-7) 11--
2) 0011 ✓	2-4) 0-11	3-6-5-7) 11--
3) 1100 ✓	3-5) 110- ✓	
-----	3-6) 11-0 ✓	
4) 0111 ✓	-----	
5) 1101 ✓	4-7) -111	
6) 1110 ✓	5-7) 11-1 ✓	
-----	6-7) 111- ✓	
7) 1111 ✓		

$\overline{\overline{abd}}$

$\overline{acd}$

le clausole non  
marcate sono  
implicanti primi

$ab$

$bcd$

# L'esempio sulla mappa di Karnaugh

- ◆ Il metodo di McCluskey ci ha consentito di trovare **tutti** gli implicant primi della funzione:

$$\overline{\overline{a}}bd, \overline{a}cd, bcd, ab$$

- ◆ Sappiamo che una forma che minimizzi le diverse funzioni di costo dovrà essere una somma di tali PI
- ◆ **non necessariamente, però, li includerà tutti**
- ◆ Esempio:

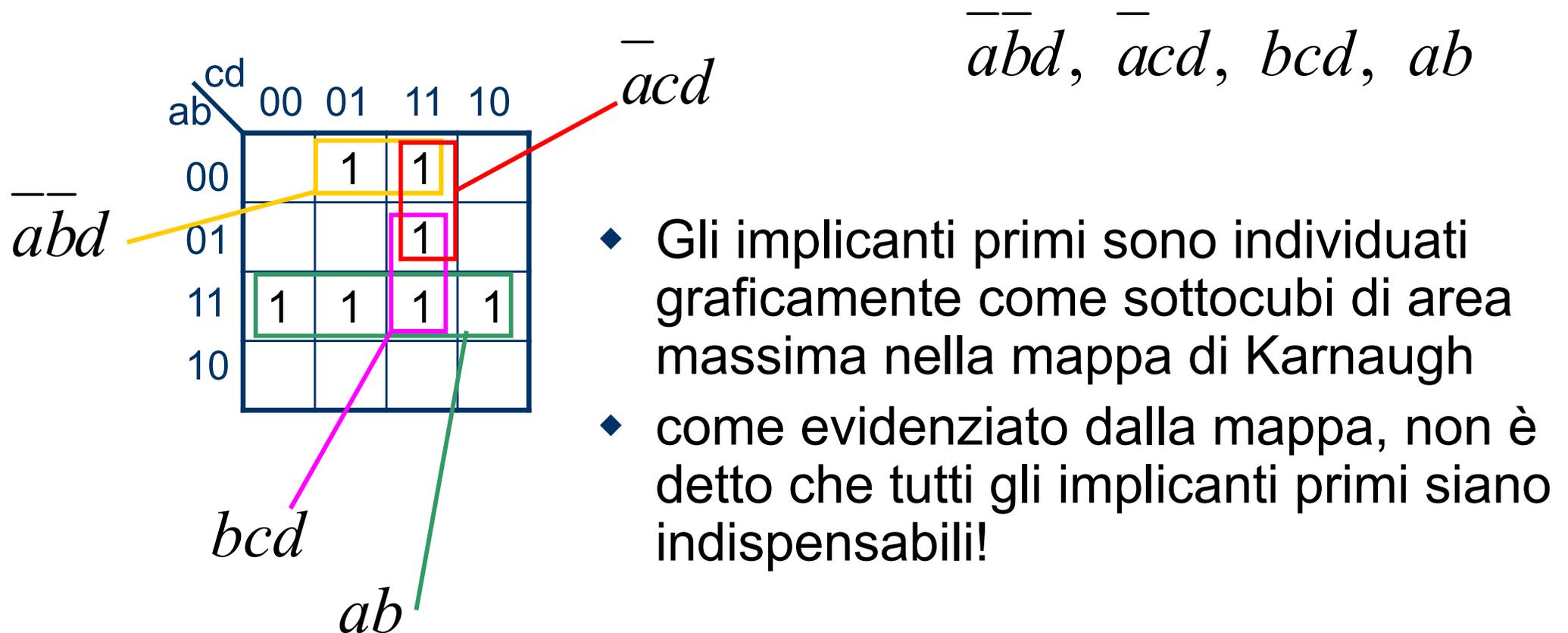
$$f = \overline{\overline{a}}bd + \overline{a}cd + bcd + ab =$$

← certamente questa forma non è minima!

$$\overline{\overline{a}}bd + \overline{\overline{a}}bcd + \overline{a}bcd + bcd + ab = \overline{\overline{a}}bd + bcd + ab$$

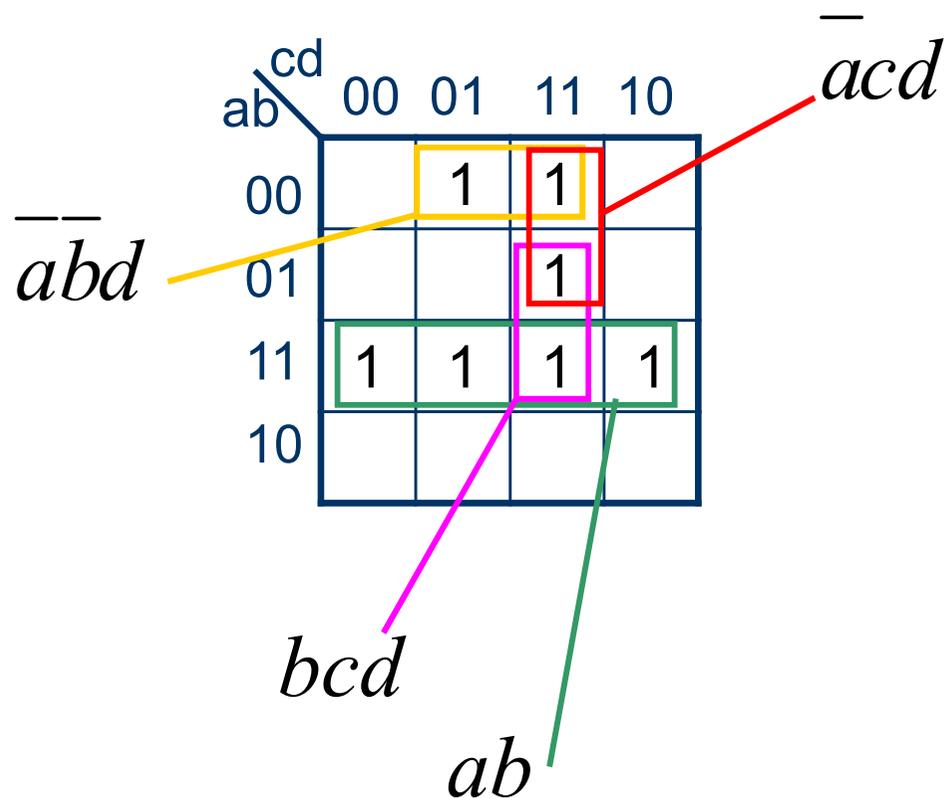
# L'esempio sulla mappa di Karnaugh

- ◆ Il metodo di McCluskey ci ha consentito di trovare **tutti** gli implicant primari della funzione:



# L'esempio sulla mappa di Karnaugh

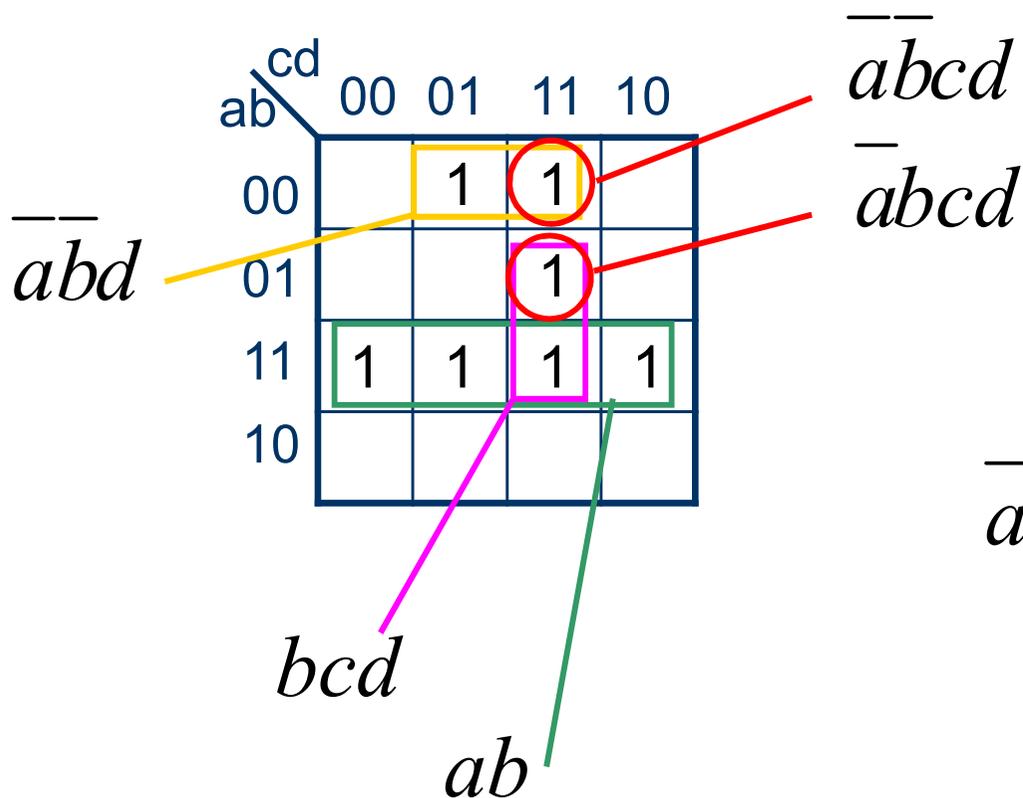
- ◆ Il metodo di McCluskey ci ha consentito di trovare **tutti** gli implicant primi della funzione:



$$f = \overline{\overline{a}}bd + \overline{a}cd + bcd + ab$$

# L'esempio sulla mappa di Karnaugh

- Il metodo di McCluskey ci ha consentito di trovare **tutti** gli implicant primi della funzione:

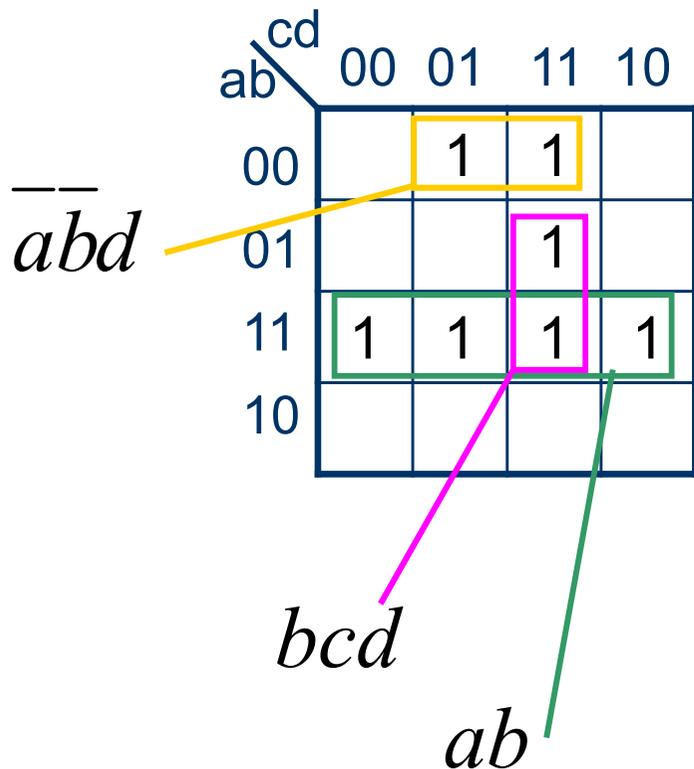


$$f = \overline{\overline{a}}bd + \overline{\overline{a}}bcd + \overline{a}bcd + bcd + ab =$$

$$\overline{\overline{a}}bd + \overline{\overline{a}}bcd + \overline{a}bcd + bcd + ab$$

# L'esempio sulla mappa di Karnaugh

- Il metodo di McCluskey ci ha consentito di trovare **tutti** gli implicant primi della funzione:



$$f = \overline{ab}d + \overline{acd} + bcd + ab =$$

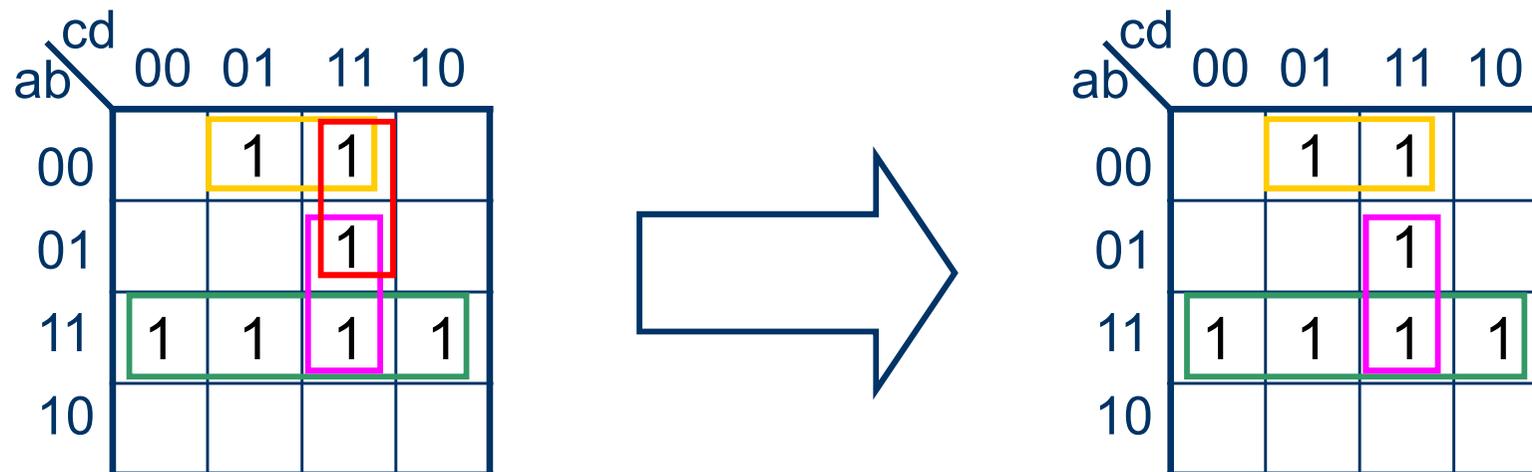
$$\underbrace{\overline{abd} + \overline{abcd}}_{\overline{abd}} + \underbrace{\overline{abcd} + bcd}_{bcd} + ab =$$

$$\overline{abd} + bcd + ab$$

# Copertura minima

- ◆ Individuati gli implicanti primi, occorre scegliere tra di essi **un insieme minimo** che consenta di “coprire” ancora la funzione

esempio:



# Problema della copertura minima

- ◆ Il problema precedente si può ricondurre alla seguente formulazione:
- ◆ Data una matrice di  $N$  righe e  $M$  colonne, i cui elementi siano  $a_{ij} = 1$  oppure  $a_{ij} = 0$ , si dice che una riga  $i$  copre una colonna  $j$  se  $a_{ij} = 1$ .
- ◆ Problema della *copertura minima*:
  - selezionare il numero minimo di righe che coprano tutte le colonne.
- ◆ Il problema della copertura è di interesse generale in molti settori differenti (ad esempio, in applicazioni di testing)

# Copertura minima

	$C_1$	$C_2$	$C_3$	$C_4$	$C_5$	$C_6$
$R_1$		X				
$R_2$	X		X	X		
$R_3$			X			X
$R_4$					X	

- ◆ la riga  $R_1$  **copre** la colonna  $C_2$
- ◆ la riga  $R_2$  **copre** le colonne  $C_1$ ,  $C_3$  e  $C_4$
- ◆ etc..
- ◆ ...
- ◆ la colonna  $C_3$  è **coperta** dalle righe  $R_2$  e  $R_3$
- ◆ etc..

# Copertura minima

colonne: mintermini, ovvero '1' della funzione

	$C_1$	$C_2$	$C_3$	$C_4$	$C_5$	$C_6$
$R_1$		X				
$R_2$	X		X	X		
$R_3$			X			X
$R_4$					X	

righe: implicanti primi precedentemente trovati con metodo di McCluskey o con mappe di Karnaugh

## ◆ nel nostro caso:

- mettiamo sulle righe gli implicanti primi
- e sulle colonne i mintermini che implicano gli implicanti primi:

“la riga  $X$  copre la colonna  $i$ ”  
significa quindi:

*l'implicante primo  $X$  è implicato dal mintermine  $i$*

# Copertura minima

Ad esempio:

	$C_1$	$C_2$	$C_3$	$C_4$	$C_5$	$C_6$
$R_1$		X				
$R_2$	X		X	X		
$R_3$			X			X
$R_4$					X	

$\overline{\overline{abcd}}$        $\overline{\overline{abcd}}$

$\overline{\overline{abd}}$

$$f = \dots + \overline{\overline{abcd}} + \overline{\overline{abcd}} + \dots = \dots + \overline{\overline{abd}} + \dots$$

$a$	$b$	$c$	$d$	$y$
0	0	0	0	0
<b>0</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>1</b>
0	0	1	0	0
<b>0</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>1</b>
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0

*mintermini*

$\overline{\overline{abcd}}$

$\overline{\overline{abcd}}$

*implicante primo*



# Copertura minima

	$C_1$	$C_2$	$C_3$	$C_4$	$C_5$	$C_6$
$R_1$		X				
$R_2$	X		X	X		
$R_3$			X			X
$R_4$					X	

$\overline{\overline{abcd}}$        $\overline{\overline{abcd}}$

$\overline{\overline{abd}}$

$$f = \dots + \overline{\overline{abcd}} + \overline{\overline{abcd}} + \dots = \dots + \overline{\overline{abd}} + \dots$$

$a$	$b$	$c$	$d$	$y$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0

la tabella ci dice che la presenza dell'implicante (**riga  $R_3$** ) copre quei due **1** nella tabella di verità

# Costruzione della matrice

<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>y</b>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

*Ricordate che i mintermini si indicano con  $P_i$  dove  $i$  è il valore decimale corrispondente alla configurazione degli ingressi relativi a quel mintermine*

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$R_1$							
$R_2$							
$R_3$							
$R_4$							

# Costruzione della matrice

- ◆ Mettiamo sulle righe tutti gli implicant primari precedentemente individuati tramite il metodo di McCluskey

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$\bar{a}cd$ →	A = 0-11						
$\bar{a}\bar{b}d$ →	B = 00-1						
$bcd$ →	C = -111						
$ab$ →	D = 11--						

# Costruzione della matrice

- ◆ Mettiamo sulle righe tutti gli implicant primi precedentemente individuati tramite il metodo di McCluskey

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$\bar{a}cd$ →		X	X				
$\bar{\bar{a}}bd$ →							
$bcd$ →							
$ab$ →							

# Costruzione della matrice

- ◆ Mettiamo sulle righe tutti gli implicant primi precedentemente individuati tramite il metodo di McCluskey

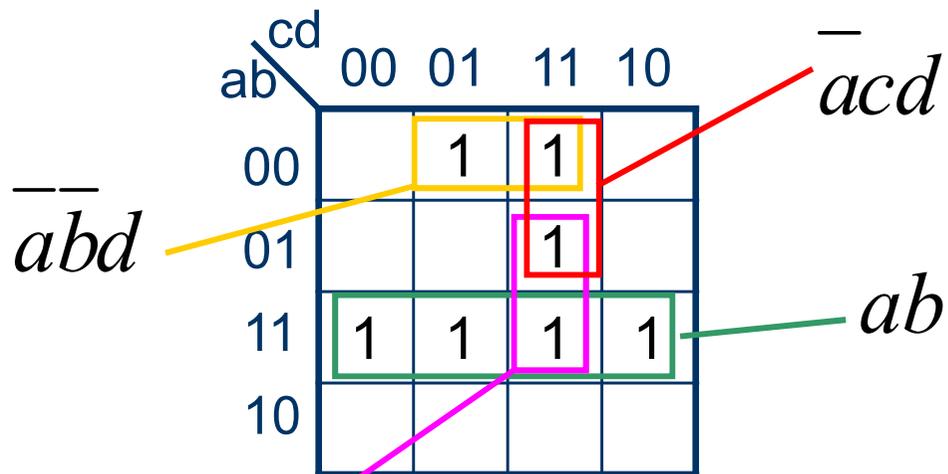
	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$\bar{a}cd$ → A = 0-11		X	X				
$\bar{a}\bar{b}d$ → B = 00-1	X	X					
$bcd$ → C = -111							
$ab$ → D = 11--							

# Costruzione della matrice

- ◆ Mettiamo sulle righe tutti gli implicant primi precedentemente individuati tramite il metodo di McCluskey

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$\overline{acd}$ →		X	X				
$\overline{abd}$ →	X	X					
$bcd$ →			X				X
$ab$ →				X	X	X	X

# Costruzione della matrice



confrontate la matrice di copertura con la Mappa di Karnaugh

$bcd$

$\overline{acd}$

$\overline{abd}$

$bcd$

$ab$

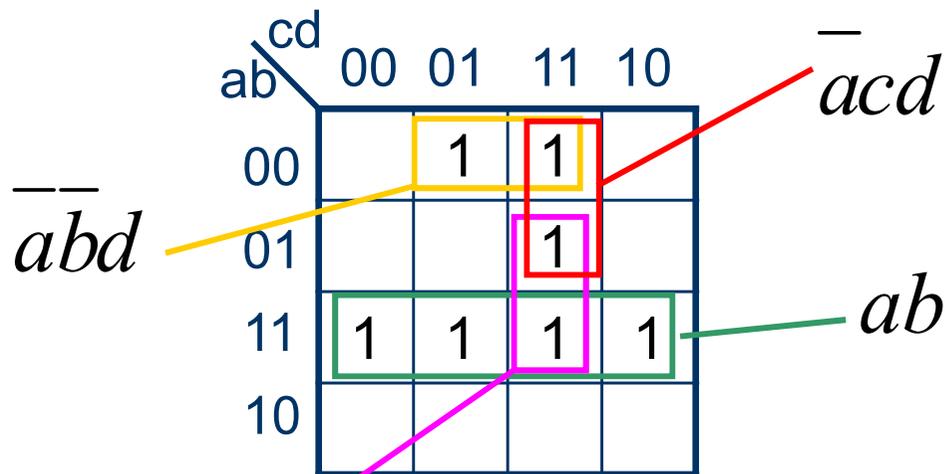
	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
A = 0-11		X	X				
B = 00-1	X	X					
C = -111			X				X
D = 11--				X	X	X	X

# Individuazione del nucleo

---

- ◆ Per quanto visto prima, il primo passo da fare per individuare la copertura è selezionare l'insieme degli implicanti primi **essenziali** (il *nucleo*).
- ◆ Tali implicanti devono infatti essere sempre presenti e la loro selezione è univoca
- ◆ Nella matrice di copertura vengono individuati in corrispondenza di colonne con un unico '1'

# Costruzione della matrice



l'implicante  $ab$  è sicuramente **essenziale** (è infatti l'unico a "coprire" degli 1 della funzione). Lo stesso vale per  $\bar{a}\bar{b}d$

$bcd$

$\bar{a}\bar{c}d$

$\bar{a}b\bar{d}$

$bcd$

$ab$

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
A = 0-11		X	X				
B = 00-1	X	X					
C = -111			X				X
D = 11--				X	X	X	X

# Semplificazione della matrice

	$P_1$	$P_3$	$P_7$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$\overline{acd}$ →		X	X				
$\overline{\overline{abd}}$ →	X	X					
$bcd$ →			X				X
$ab$ →				X	X	X	X

Il nucleo  $N$  è costituito dagli implicant  $B$  e  $D$ :  $N=\{B, D\}$ . In altri termini, la funzione minima, scritta come somma di implicant primi, conterrà certamente  $B$  ed  $D$ :

$$f = \dots + \overline{\overline{abd}} + ab + \dots$$

Avendo incluso  $D$  nella funzione, i mintermini  $P_{12}$ ,  $P_{13}$ ,  $P_{14}$  e  $P_{15}$  risultano coperti. Il problema si riduce alla copertura di  $P_1$ ,  $P_3$  e  $P_7$ .

# Semplificazione della matrice

	$P_1$	$P_3$	$P_7$
$\overline{acd}$ → A = 0-11		X	X
$\overline{abd}$ → B = 00-1	X	X	
$bcd$ → C = -111			X

Il nucleo  $N$  è costituito dagli implicanti  $B$  e  $D$ :  $N = \{B, D\}$ . In altri termini, la funzione minima, scritta come somma di implicanti primi, conterrà certamente  $B$  ed  $D$ :

$$f = \dots + \overline{\overline{abd}} + ab + \dots$$

Avendo incluso  $B$  nella funzione, i mintermini  $P_1$  e  $P_3$  risultano coperti. Il problema si riduce alla copertura di  $P_7$ .

E' indifferente a questo punto scegliere  $A$  oppure  $C$

# Forma minima della funzione

La  $f$  può quindi esser scritta come:  $f = ab + \overline{a}\overline{b}d + \overline{a}cd$

oppure  $f = ab + \overline{a}\overline{b}d + bcd$

...risultato a cui si può giungere intuitivamente sulla Mappa di Karnaugh:

ab \ cd	00	01	11	10
00		1	1	
01			1	
11	1	1	1	1
10				

oppure

ab \ cd	00	01	11	10
00		1	1	
01			1	
11	1	1	1	1
10				

# Metodi di Copertura minima

- ◆ La selezione degli implicanti primi non essenziali è **arbitraria**, a differenza di quelli essenziali.

	$P_1$	$P_8$	$P_9$	$P_{24}$	$P_{27}$
A= -100-		X	X	X	
B= --001	X		X	X	
C= 0-00-	X	X	X		
D= 11-11					X
F= 110-1				X	X

# Metodi di Copertura minima

---

- ◆ La selezione degli implicant primari non essenziali è **arbitraria**, a differenza di quelli essenziali.
- ◆ La scelta deve essere:
  - Ottima secondo le prefissate funzioni di costo
  - ottenibile in maniera computazionalmente efficiente
- ◆ Due fondamentali alternative
  - Tabellare (righe/colonne dominanti)
  - Algebrica (Petrick)

- ◆ Esprime la condizione algebrica secondo la quale **TUTTE** le colonne devono essere coperte da **ALMENO UN** implicante.
- ◆ Può allora essere espressa come una AND di OR.
- ◆ Tale forma può essere trasformata in una OR di AND che esprime tutte le possibili alternative per la soluzione del problema di copertura

# Metodo di Petrick: esempio

Materiale facoltativo

	M <sub>1</sub>	M <sub>2</sub>	M <sub>3</sub>	M <sub>4</sub>	M <sub>5</sub>	M <sub>6</sub>	M <sub>7</sub>	M <sub>8</sub>	M <sub>9</sub>
A	1		1		1		1	1	1
B	1	1		1			1	1	
C		1				1			1
D			1				1		
E				1		1			
F					1			1	
G						1			1

$$(A+B)(B+C)(A+D)(B+E)(A+F)(C+E+G)(A+B+D)(A+B+F)(A+C+G)=$$
$$=ACE + ABC + ABG + ABE + BCDF + BDFG$$

per risolvere il problema di copertura possono essere usati {A,C,E}, oppure {A,B,C}, oppure...

# Righe/Colonne dominanti

- ◆ Chiamiamo “linea” indifferentemente una riga o una colonna
- ◆ Una linea  $L$  domina la linea  $K$  se la “include”, ovvero se contiene tutti i suoi 1

La colonna 3 domina la 1

	1	2	3	4	5	6
A		X				
B	X		X	X		
C	X		X		X	X
D			X		X	

La riga C  
domina la D

# Righe/Colonne dominanti

---

- ◆ Se si eliminano le righe ***dominate*** e le colonne ***dominanti*** da una matrice di copertura, se ne trae una equivalente
  - una matrice che rappresenta, cioè, il medesimo problema di copertura
- ◆ Perché?

# Righe/Colonne dominanti

- ◆ Una **riga dominata** (ad es. D) corrisponde ad un implicante che copre *meno mintermini* della riga dominante (ad es. C)
  - Il corrispondente implicante (ad es. D) sicuramente non è utile nella forma minima della funzione e si può eliminare dalla ricerca

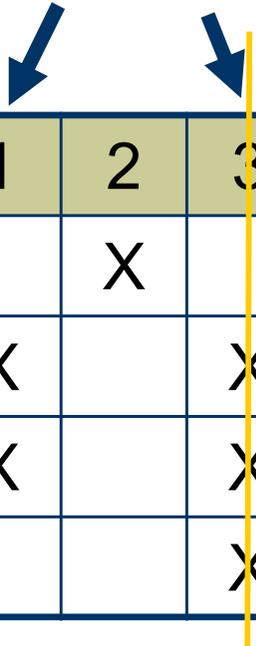
	1	2	3	4	5	6
A		X				
B	X		X	X		
C	X		X		X	X
D			X		X	

La riga C domina la D

# Righe/Colonne dominanti

- ◆ Una **colonna dominante** (ad es. 3) richiede la presenza di un sovrainsieme (ad es. B, C, D) degli implicant richiesti dalla colonna dominata (ad es. la colonna 1, che richiede solo B e C), e può quindi essere eliminata

La colonna 3 domina la 1



	1	2	3	4	5	6
A		X				
B	X		X	X		
C	X		X		X	X
D			X		X	

Nell'esempio, il mintermine (colonna) 1 *richiede* che nella forma minima **siano presenti B o C**. Di conseguenza, il mintermine nella colonna 3 sarà *automaticamente coperto* e lo si può escludere dal problema di copertura.

# Righe/Colonne dominanti

	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$
	$P_1$	$P_8$	$P_9$	$P_{24}$	$P_{27}$
$\overline{bcd}$	A= -100-	X	X	X	
$\overline{cde}$	B= --001	X	X	X	
$\overline{acd}$	C= 0-00-	X	X		
$\overline{abde}$	D= 11-11				X
$\overline{abce}$	F= 110-1			X	X

Ulteriore esempio con eliminazione delle **colonne dominanti**:

se si trova un sottoinsieme di  $\{A,B,C,D,F\}$  che copre  $P_8$ , (necessariamente dovrà includere A e/o C) lo stesso sottoinsieme coprirà anche  $P_9$

# Righe/Colonne dominanti

	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$	$\overline{abcde}$
	$P_1$	$P_8$	$P_9$	$P_{24}$	$P_{27}$
$\overline{bcd}$ A= -100-		X	X	X	
$\overline{cde}$ B= --001	X		X	X	
$\overline{acd}$ C= 0-00-	X	X	X		
$abde$ D= 11-11					X
$abce$ F= 110-1				X	X

Ulteriore esempio con eliminazione delle **righe dominate**:

il sottoinsieme {A,B,C,F} copre gli stessi mintermini di {A,B,C,D,F} ma con un numero minore di implicant: D si può eliminare dalla ricerca

# Righe/Colonne dominanti: riepilogo

- ◆ Il metodo tabellare per righe/colonne dominanti procede allora come segue:
  - Si ricercano gli implicanti primi ( $PI$ ) e si individuano quelli essenziali;
  - Si includono nella forma minima i  $PI$  essenziali, eliminandoli dalla matrice, unitamente con i mintermini ricoperti;
  - Si eliminano le righe dominate e le colonne dominanti;
  - Si individuano i  $PI$  essenziali “secondari” della matrice così ridotta;
  - Si ripetono i passi 2, 3, 4 finché è possibile.

# Esempio

---

$$\begin{aligned} f(x_1, x_2, x_3, x_4, x_5) &= \sum (0,1,2,8,9,15,17,21,24,25,27,28,31) = \\ &= \overline{x_1 x_2 x_3 x_4 x_5} + \overline{x_1 x_2 x_3 x_4} x_5 + \overline{x_1 x_2 x_3} x_4 \overline{x_5} + \dots \end{aligned}$$

# Esempio

Implicanti del 5° ordne	Implicanti del 4° ordne	Implicanti del 3° ordne
1) 00000 ✓ -----	1-2 0000- 1-3 000-0 ... ..	
2) 00001 ✓ 3) 00010 ✓ 4) 01000 -----		
5) 01001 6) 10001 7) 11000 -----		
8) 10101 9) 11001 10) 11100 -----		
11) 01111 12) 11011 -----		
13) 11111		

# Esempio

Implicanti del 5° ordne	Implicanti del 4° ordne	Implicanti del 3° ordne
1) 00000 √	1-2 0000-	
-----	1-3 000-0	
2) 00001 √	1-4 0-000	
3) 00010 √	-----	
4) 01000 √	2-5 0-001	
-----	2-6 -0001	
5) 01001 √	4-5 0100-	
6) 10001 √	4-7 -1000	
7) 11000 √	-----	
-----	5-9 -1001	
8) 10101 √	6-8 10-01	
9) 11001 √	6-9 1-001	
10) 11100 √	7-9 1100-	
-----	7-10 11-00	
11) 01111 √	9-12 110-1	
12) 11011 √	-----	
-----	11-13 -1111	
13) 11111 √	12-13 11-11	

# Esempio

Implicanti del 5° ordne	Implicanti del 4° ordne	Implicanti del 3° ordne
1) 00000 √	1-2 0000- √	1-2/4-5 0-00-
-----	1-3 000-0	1-4/2-5 0-00-
2) 00001 √	1-4 0-000 √	-----
3) 00010 √	-----	2-5/6-9 --001
4) 01000 √	2-5 0-001 √	2-6/5-9 --001
-----	2-6 -0001 √	4-5/7-9 -100-
5) 01001 √	4-5 0100- √	4-7/5-9 -100-
6) 10001 √	4-7 -1000 √	
7) 11000 √	-----	
-----	5-9 -1001 √	
8) 10101 √	6-8 10-01	
9) 11001 √	6-9 1-001 √	
10) 11100 √	7-9 1100- √	
-----	7-10 11-00	
11) 01111 √	9-12 110-1	
12) 11011 √	-----	
-----	11-13 -1111	
13) 11111 √	12-13 11-11	

# Esempio

Implicanti del 5° ordne	Implicanti del 4° ordne	Implicanti del 3° ordne
1) 00000 ✓ -----	1-2 0000- ✓ 1-3 000-0 <=== (J)	1-2/4-5 0-00- <=== (C) 1-4/2-5 0-00-
2) 00001 ✓ 3) 00010 ✓ 4) 01000 ✓ -----	1-4 0-000 ✓ ----- 2-5 0-001 ✓ 2-6 -0001 ✓ 4-5 0100- ✓ 4-7 -1000 ✓ -----	----- 2-5/6-9 --001 <=== (B) 2-6/5-9 --001 4-5/7-9 -100- <=== (A) 4-7/5-9 -100-
5) 01001 ✓ 6) 10001 ✓ 7) 11000 ✓ -----	5-9 -1001 ✓ 6-8 10-01 <=== (G) 6-9 1-001 ✓ 7-9 1100- ✓ 7-10 11-00 <=== (H) 9-12 110-1 <=== (F) -----	
8) 10101 ✓ 9) 11001 ✓ 10) 11100 ✓ -----	11-13 -1111 <=== (E) 12-13 11-11 <=== (D)	
11) 01111 ✓ 12) 11011 ✓ -----		
13) 11111 ✓		

# Esempio

Implicanti	Mintermini coperti
<b>A = -100-</b>	<b>8, 9, 24, 25</b>
<b>B = --001</b>	<b>1, 9, 17, 25</b>
<b>C = 0-00-</b>	<b>0, 1, 8, 9</b>
<b>D = 11-11</b>	<b>27, 31</b>
<b>E = -1111</b>	<b>15, 31</b>
<b>F = 110-1</b>	<b>25, 27</b>
<b>G = 10-01</b>	<b>17, 21</b>
<b>H = 11-00</b>	<b>24, 28</b>
<b>J = 000-0</b>	<b>0, 2</b>

# Esempio

	P <sub>0</sub>	P <sub>1</sub>	P <sub>2</sub>	P <sub>8</sub>	P <sub>9</sub>	P <sub>15</sub>	P <sub>17</sub>	P <sub>21</sub>	P <sub>24</sub>	P <sub>25</sub>	P <sub>27</sub>	P <sub>28</sub>	P <sub>31</sub>
A				1	1				1	1			
B		1			1		1			1			
C	1	1		1	1								
D											1		1
E						1							1
F										1	1		
G							1	1					
H									1			1	
J	1		1										

Primi implicanti essenziali: J, E, G, H

# Esempio

	P <sub>0</sub>	P <sub>1</sub>	P <sub>2</sub>	P <sub>8</sub>	P <sub>9</sub>	P <sub>15</sub>	P <sub>17</sub>	P <sub>21</sub>	P <sub>24</sub>	P <sub>25</sub>	P <sub>27</sub>	P <sub>28</sub>	P <sub>31</sub>
A				1	1				1	1			
B		1			1		1			1			
C	1	1		1	1								
D											1		1
E						1							1
F										1	1		
G							1						
H									1			1	
I	1												
J			1										

Gli implicanti primi essenziali J,E,G,H coprono i mintermini: 0, 2, 15, 17, 21, 24, 28, 31

# Esempio

	P <sub>1</sub>	P <sub>8</sub>	P <sub>9</sub>	P <sub>25</sub>	P <sub>27</sub>
A		1	1	1	
B	1		1	1	
C	1	1	1		
D					1
F				1	1

↑  
riga D dominata dalla F

	P <sub>1</sub>	P <sub>8</sub>	P <sub>25</sub>	P <sub>27</sub>
A		1	1	
B	1		1	
C	1	1		
F			1	1

	P <sub>1</sub>	P <sub>8</sub>
A		1
B	1	
C	1	1

**F** implicante primo essenziale secondario:  
copre P<sub>25</sub> e P<sub>27</sub>

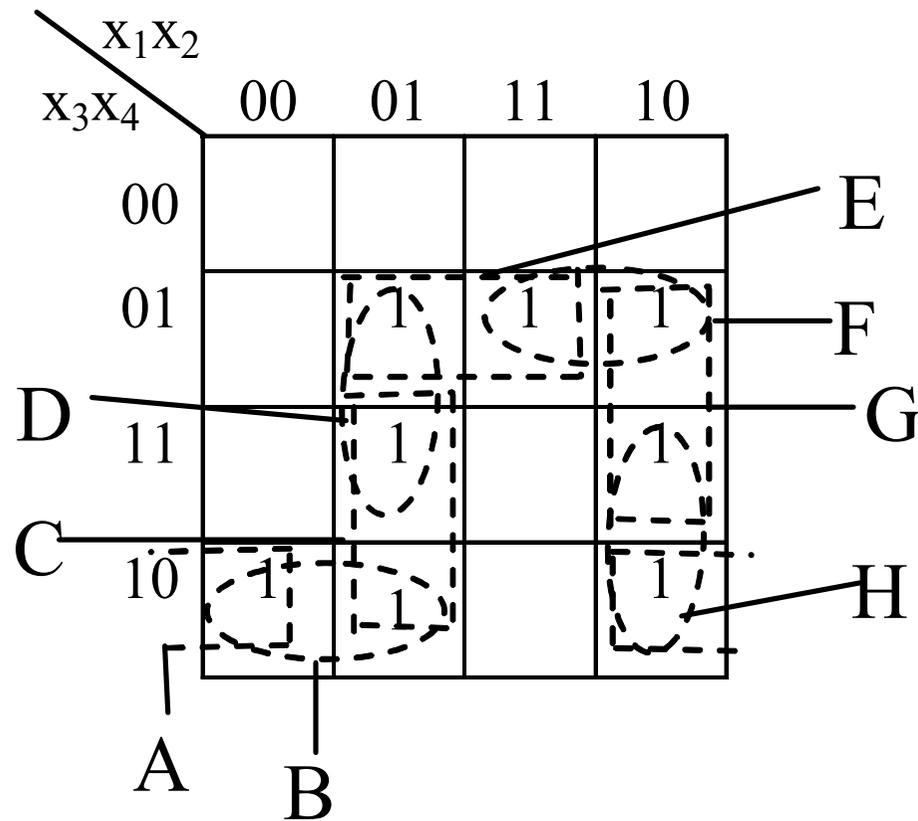
Righe A e B dominate dalla **C**

# Esempio

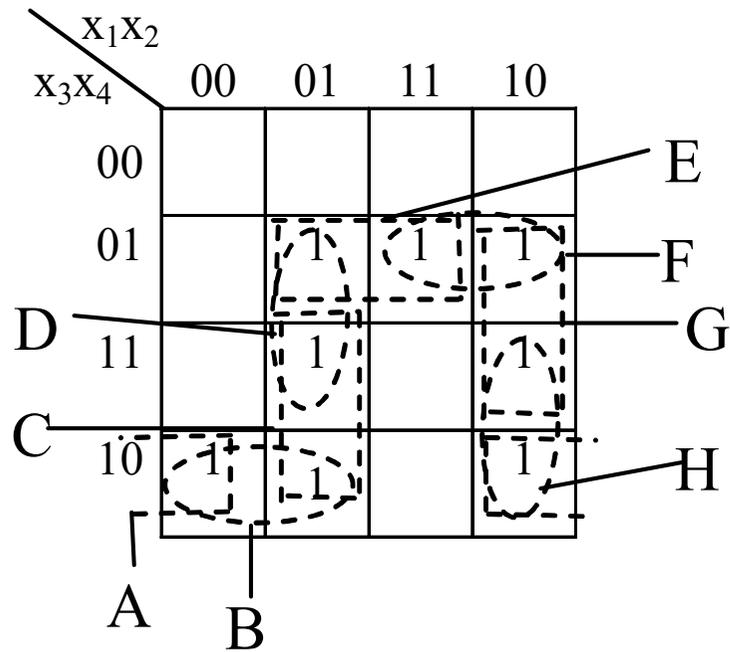
$$\begin{aligned}
 f(x_1, x_2, x_3, x_4, x_5) &= \sum (0,1,2,8,9,15,17,21,24,25,27,28,31) = \\
 &= \overline{x_1}x_2x_3x_4x_5 + x_1\overline{x_2}x_3x_4x_5 + \overline{x_1}x_2\overline{x_3}x_4x_5 + \dots = \\
 &= E + G + H + J + F + C =
 \end{aligned}$$

$$\begin{aligned}
 &= x_2x_3x_4x_5 + x_1\overline{x_2}x_4x_5 + \overline{x_1}x_2\overline{x_3}x_4x_5 + \overline{x_1}x_2x_3x_5 \\
 &+ \overline{x_1}x_2x_3x_5 + \overline{x_1}x_3x_4
 \end{aligned}$$

# Altro esempio



# Altro esempio



	$P_3$	$P_6$	$P_7$	$P_8$	$P_9$	$P_{10}$	$P_{13}$	$P_{14}$
A				1		1		
B				1	1			
C					1		1	
D	1						1	
E	1		1					
F		1	1					
G		1						1
H						1		1

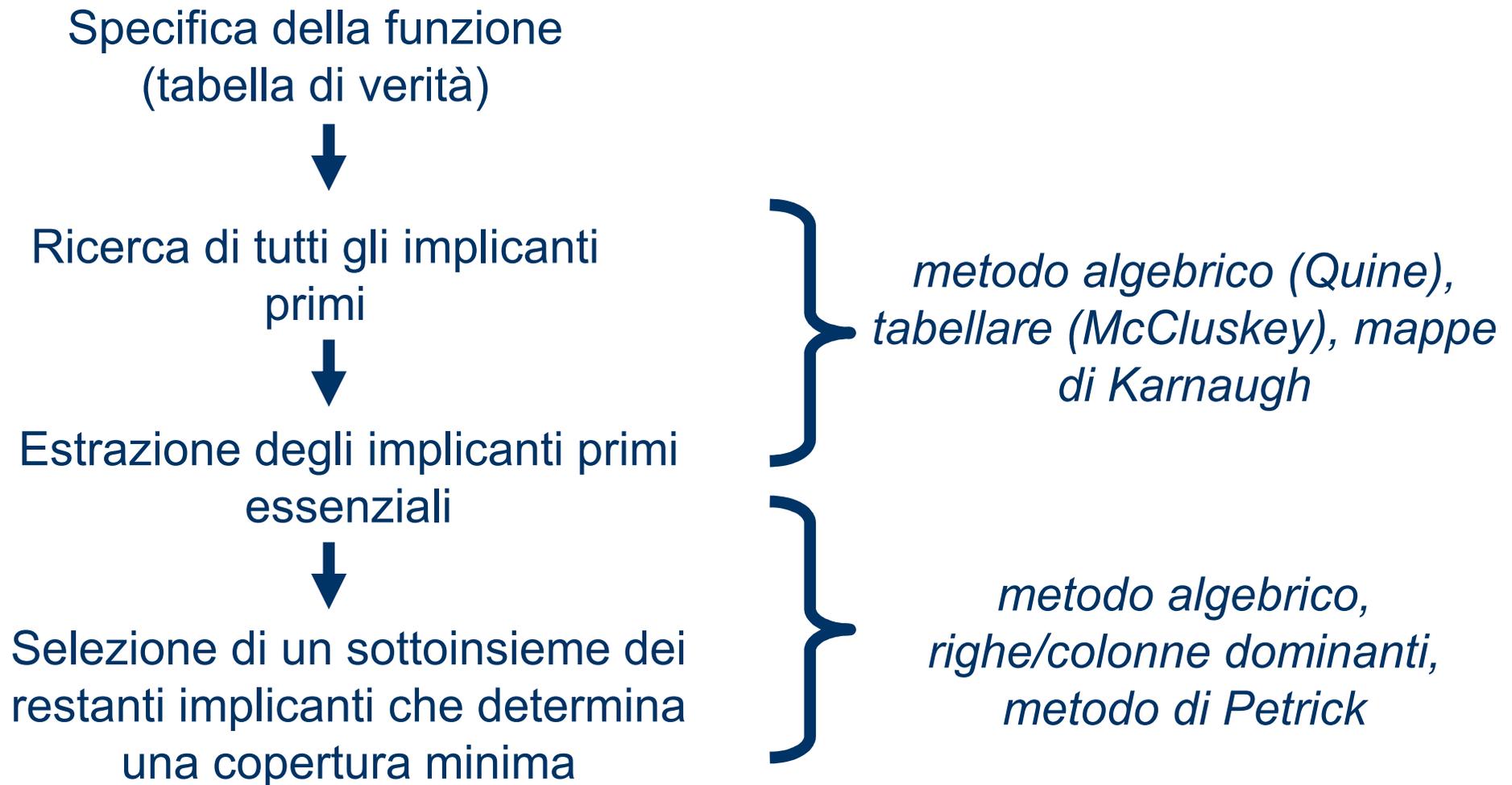
- ◆ Metodo di Petrick:

$$\begin{aligned} & (A+B)(B+C)(C+D)(D+E)(E+F)(F+G)(G+H)(H+A)= \\ & = ABCDEFGH + BCDEFGH + \dots + BDFH + \dots + ACEG + \dots = \\ & = BDFH + ACEG \end{aligned}$$

e quindi si individuano due coperture minime:  
 $\{B, D, F, H\}$  e  $\{A, C, E, G\}$ .

# Ricapitolando

## Procedimento completo per la minimizzazione di una funzione logica



# Punti di indeterminazione (don't care)

- ◆ rappresentano combinazioni di ingresso per le quali non ci interessa il comportamento della funzione
- ◆ indicati con –
- ◆ ha senso usarli per la **specificità** di una funzione, ovvero in fase di **sintesi**

<i>a</i>	<i>b</i>	<i>c</i>	<i>y</i>
0	0	0	-
0	0	1	-
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



		bc			
		00	01	11	10
a	0	-	-	1	1
	1			1	

# Punti di indeterminazione (don't care)

- ◆ nell'esempio, non ci interessa il comportamento della funzione quando gli ingressi sono '000' oppure '001'
- ◆ in effetti, la tabella specifica non una singola funzione ma **4** differenti funzioni tra loro **compatibili**
- ◆ se ne può scegliere una qualsiasi in fase di sintesi

<i>a</i>	<i>b</i>	<i>c</i>	<i>y</i>
0	0	0	-
0	0	1	-
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

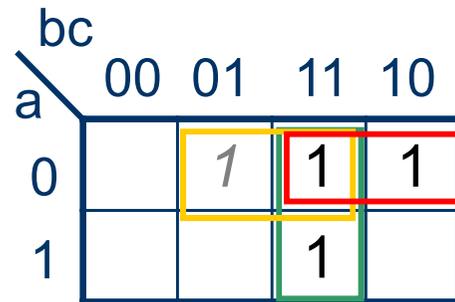


		bc			
		00	01	11	10
a	0	-	-	1	1
	1			1	

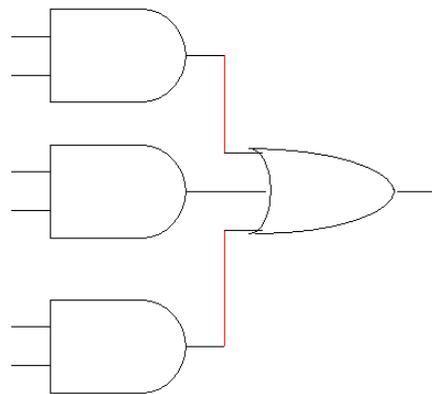
# Punti di indeterminazione (don't care)

due differenti funzioni **compatibili** in virtù dei don't care:

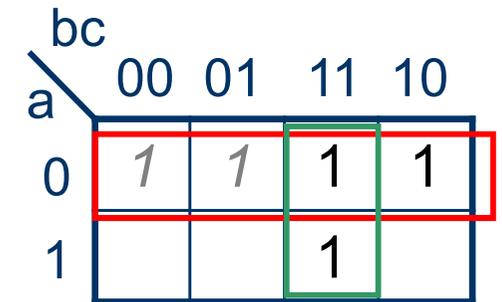
a	b	c	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



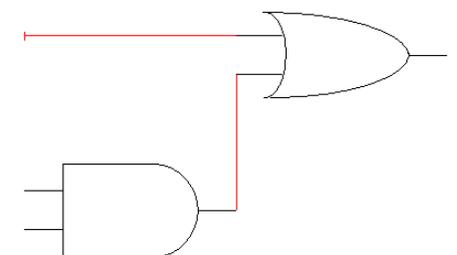
$$f = \bar{a}c + bc + \bar{a}b$$



a	b	c	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



$$f = \bar{a} + bc$$



# Minimizzazione con don't care

---

- ◆ I don't care possono essere sfruttati per minimizzare ulteriormente la struttura di una funzione logica
  - *si può cercare tra tutte le funzioni compatibili quella che ha costo minimo*

# Presenza di don't care

- ◆ Notate che più '1' nella tabella di verità consentono di ottenere implicanti più "larghi"
- ◆ D'altro canto, un maggior numero di '0' nella tabella di verità riduce il numero di mintermini da coprire

bc	00	01	11	10
a				
0	-	-	1	1
1	-		1	

bc	00	01	11	10
a				
0	-	1	1	1
1	-		1	

bc	00	01	11	10
a				
0	-	1	1	1
1	1		1	

bc	00	01	11	10
a				
0	1	1	1	1
1	-		1	

# Presenza di don't care

---

- ◆ Notate che più '1' nella tabella di verità consentono di ottenere implicanti più "larghi"
- ◆ D'altro canto, un maggior numero di '0' nella tabella di verità riduce il numero di mintermini da coprire
  - conviene considerare i don't care come '1' quando si cercano gli implicanti primi, e come '0' quando si ricerca la copertura

# Presenza di don't care

- ◆ conviene considerare i don't care come '1' quando si cercano gli implicantti primi, e come '0' quando si ricerca la copertura

ricerca degli implicantti primi

a \ bc	00	01	11	10
0	1	1	1	1
1	1		1	

$$\{ \bar{a}, \bar{bc}, bc \}$$

copertura

a \ bc	00	01	11	10
0	-	-	1	1
1	-		1	

$$f = \bar{a} + bc$$

# Esempio

	bc			
a \	00	01	11	10
0	-	-		1
1	-	1	1	1

	P <sub>2</sub>	P <sub>5</sub>	P <sub>6</sub>	P <sub>7</sub>
$a$	A = 1--	X	X	X
$\bar{b}$	B = -0-	X		
$\bar{c}$	C = --0	X	X	

implicanti primi ottenuti  
considerando i don't care  
pari a 1

matrice di copertura  
considerando i don't care  
pari a 0 (presenti solo 4  
mintermini)

# Esempio

**C** è un implicante primo *essenziale* (è l'unico a coprire **P<sub>2</sub>**):

sarà quindi incluso  
nella forma minima della  
funzione

	P <sub>2</sub>	P <sub>5</sub>	P <sub>6</sub>	P <sub>7</sub>
A = 1--		X	X	X
B = -0-		X		
C = --0	X		X	

Poiché **C** è stato incluso nella forma minima della funzione, abbiamo la sicurezza che **P<sub>2</sub>** e **P<sub>6</sub>** siano coperti e possiamo non preoccuparcene più, ovvero eliminarli dalla matrice

# Esempio

Nella restante matrice, la riga **A** “domina” **B**, ovvero copre un sovrainsieme dei mintermini di **B**, che può quindi essere **eliminata**.

A questo punto, tutti i mintermini risultano coperti.

**f** può quindi essere scritta come somma di **C** ed **A**:

$$f = \underline{a} + \bar{c}$$

	P <sub>5</sub>	P <sub>7</sub>
A = 1--	X	X
B = -0-	X	

bc	00	01	11	10
a				
0	-	-		1
1	-	1	1	1

# Esempio

Sulla mappa di Karnaugh (utilizzabile per funzioni con pochi ingressi) il procedimento è ancora una volta molto intuitivo:

cercare gli implicant primari più grandi, coprendo anche i *don't care*, e scegliere per la forma minima solo quelli che effettivamente coprono degli '1'

$$f = \underline{a} + \underline{\bar{c}}$$

bc	00	01	11	10
a				
0	-	-		1
1	-	1	1	1



# Minimizzazione in forma S

- ◆ i consensi sono generati come segue:

$$f(a, b, c, d) = (a + \bar{b}) (\bar{a} + b + \bar{c} + \bar{d}) \dots (\bar{a} + b + c + \bar{d}) (a + \bar{d}) =$$

$$(a + \bar{b}) (c\bar{c} + \bar{a} + b + \bar{d}) \dots (a + \bar{d}) = (a + \bar{b}) (\bar{a} + b + \bar{d}) \dots (a + \bar{d})$$

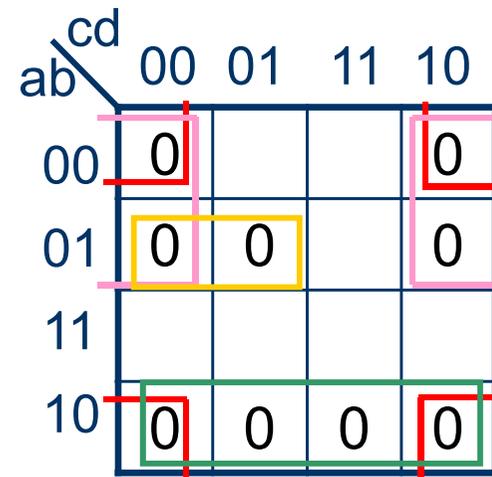
<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>y</b>
...	...	...	....	...
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
...	...	...	....	...
...	...	...	....	...



notare che maxtermini e somme sono definiti in maniera duale rispetto a mintermini e clausole, e corrispondono agli zero della funzione

# Minimizzazione in forma S

<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>y</b>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

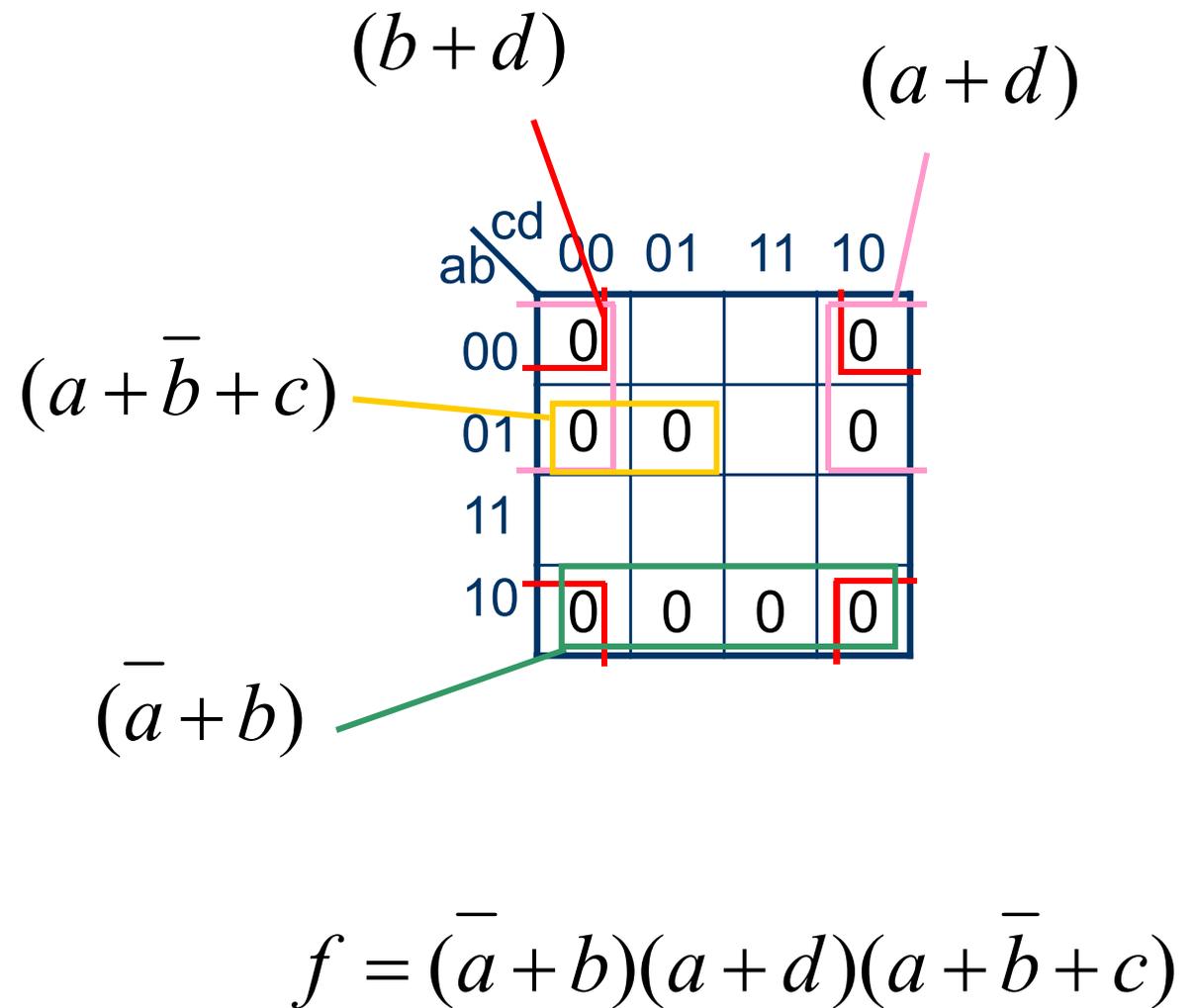


è possibile individuare sulla mappa l'equivalente degli Implicanti Primi per la forma P:

i cubi di area massima che coprono soltanto zero della funzione

# Minimizzazione in forma S

<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>y</b>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



# Minimizzazione in forma S

---

- ◆ mappe di Karnaugh
- ◆ metodo di Quine/McCluskey
- ◆ metodo di Petrick
- ◆ matrice di copertura  
etc...

possono essere applicati in maniera identica  
al caso di minimizzazione in forma S

# Minimizzazione con *don't care*

<b>a</b>	<b>b</b>	<b>c</b>	<b>y</b>
0	0	0	-
0	0	1	-
0	1	0	1
0	1	1	1
1	0	0	-
1	0	1	0
1	1	0	0
1	1	1	1

minimizzazione  
in forma P

		bc			
		00	01	11	10
a	0	-	-	1	1
	1	-		1	

$$f = \bar{a} + bc$$

minimizzazione  
in forma S

		bc			
		00	01	11	10
a	0	-	-		
	1	-	0		0

$$f = b(\bar{a} + c)$$

$$= \bar{a}b + bc$$

sono due funzioni diverse...

... ma **compatibili**



# Tempificazione nelle reti combinatorie

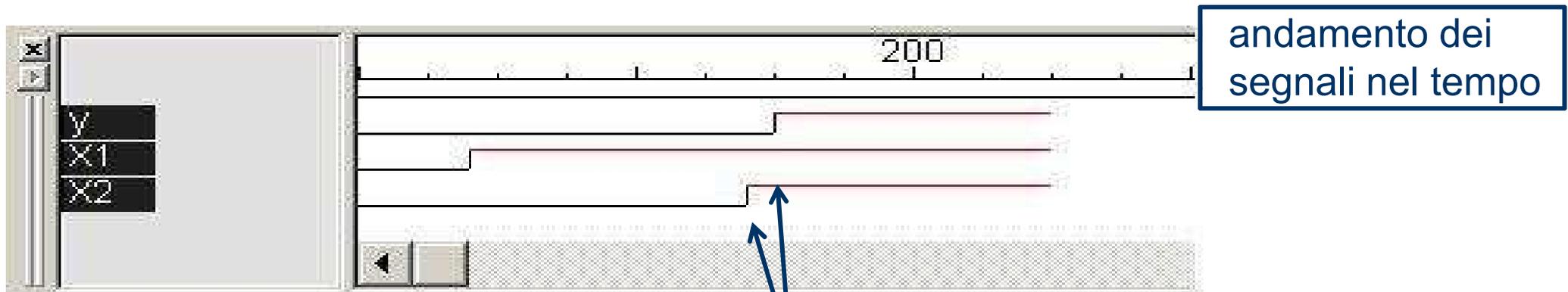
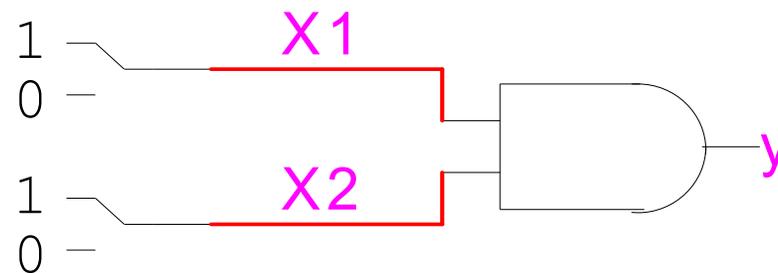
Materiale facoltativo

- ◆ Ritardi nelle reti combinatorie
- ◆ Alee statiche
- ◆ Alee dinamiche
- ◆ Alee multiple
- ◆ Impulsi concomitanti
- ◆ Codifica degli ingressi

# Il Tempo di risposta

Materiale facoltativo

Ritardo  $d = t_f - t_i$  con il quale una variazione sull'ingresso è seguita da una variazione sull'uscita

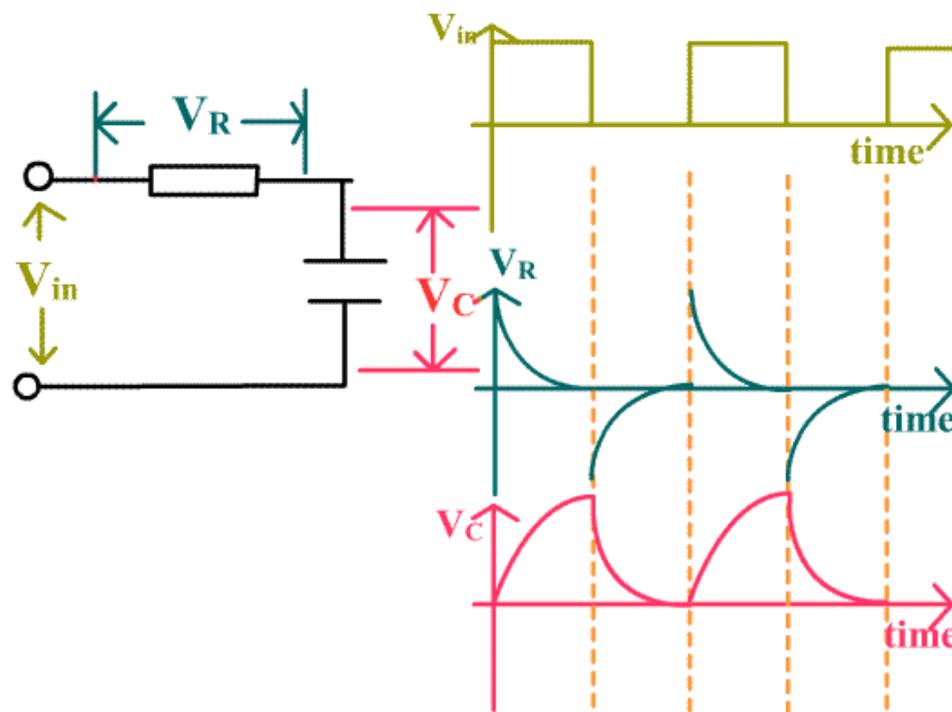


l'uscita  $y$  cambia con un certo ritardo

# Il Ritardo Inerziale

Materiale facoltativo

- ◆ ritardo dovuto a fenomeni fisici dinamici causati dalla realizzazione delle porte che “smussano” le transizioni



segnale “logico”: esprime la sequenza 1→0→1→0→1 ...

segnale fisico: a causa dell’andamento smussato, potrebbe essere interpretato come una sequenza errata di ‘1’ e ‘0’

# Le Alee (o “*hazards*”)

Materiale facoltativo

- ◆ La presenza di ritardi nei dispositivi utilizzati può avere l'effetto di **modificare** il comportamento delle uscite in alcuni casi
- ◆ Si chiamano **Alee** (*hazards*) quelle situazioni a causa delle quali, per brevi intervalli di tempo (*alee transitorie*) o in maniera permanente (*alee di regime*), le uscite assumono dei valori imprevisti.

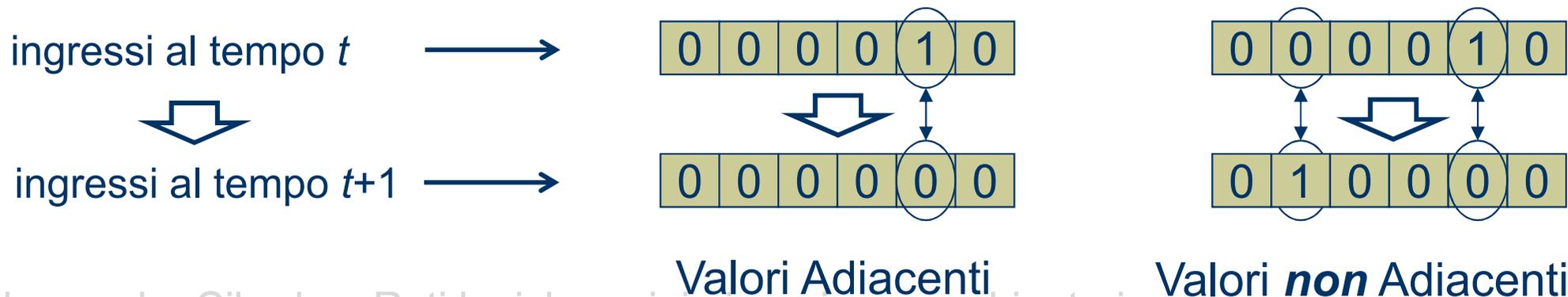
# Tipologie di transizioni in ingresso

Materiale facoltativo

◆ Per lo studio delle alee è importante valutare il modo nel quale si passa da una configurazione dei valori di ingresso a quella successiva nel tempo.

◆ **Valori adiacenti** di una codifica:

Data una codifica, due configurazioni di ingressi si dicono adiacenti se differiscono per una sola variabile



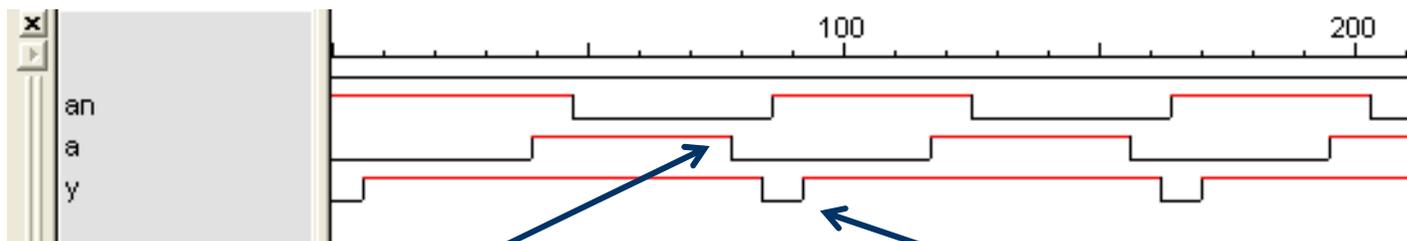
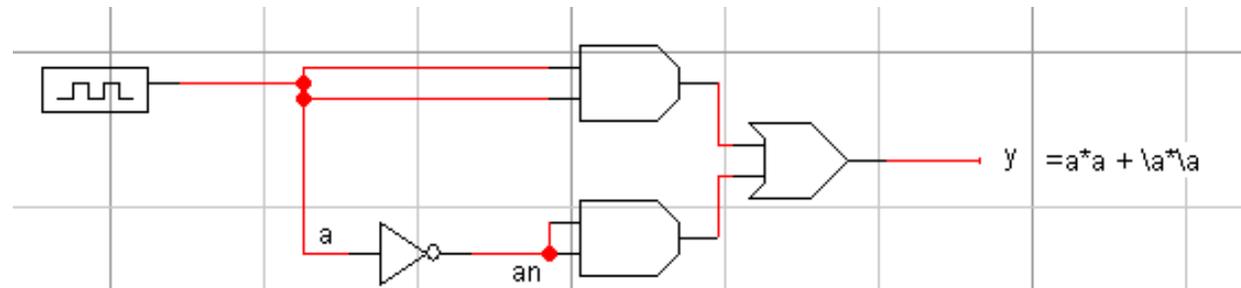
# Alee Statiche

Materiale facoltativo

Si ha un'**Alea Statica** se avendo due ingressi  $i_1$  e  $i_2$  in sequenza con uscite uguali  $f = f(i_1) = f(i_2)$ , l'uscita assume per un breve istante di tempo nel passaggio  $i_1 \rightarrow i_2$ , il valore  $NOT(f)$  (sbagliato)

Es.

$$Y = a a + \bar{a} \bar{a}$$



$a$  passa da 1 a 0, quindi  $\bar{a}$  passa da 1 a 0, ma con un certo ritardo...

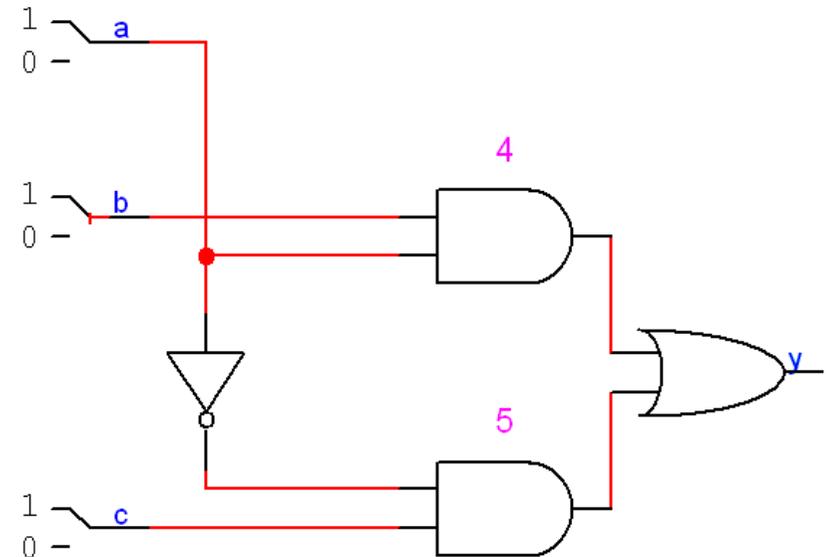
...questo fa sì che  $y$  assuma un valore sbagliato per un certo (breve) tempo

Si manifesta anche per transizioni tra ingressi **adiacenti!**

# Alea Statica - Soluzione

Materiale facoltativo

- ◆ Il problema è legato ad una doppia variazione dei valori interni della rete a partire dalla variazione di un singolo ingresso
- ◆ Aggiungendo gli implicanti ridondanti si “coprono” le variazioni che determinano l’alea
- ◆ Non è una buona idea, invece, cercare di bilanciare i ritardi circuitali
  - da un punto di vista elettronico è molto difficile controllare i ritardi delle diverse parti di un circuito



a \ bc	00	01	11	10
0			1	1
1		1	1	

Implicante  
Ridondante

# Alea Statica - Soluzione

Materiale facoltativo

- ◆ Una rete AND-OR è esente da alee statiche se nel primo livello della rete sono presenti tutti gli implicant che coprono transizioni  $1 \rightarrow 1$
- ◆ La rete è esente da alee per transizioni  $0 \rightarrow 0$
- ◆ Dualmente per le reti OR-AND
- ◆ Le alee statiche si possono eliminare aggiungendo *ridondanza* alla rete, come visto prima

# Alee Dinamiche

Materiale facoltativo

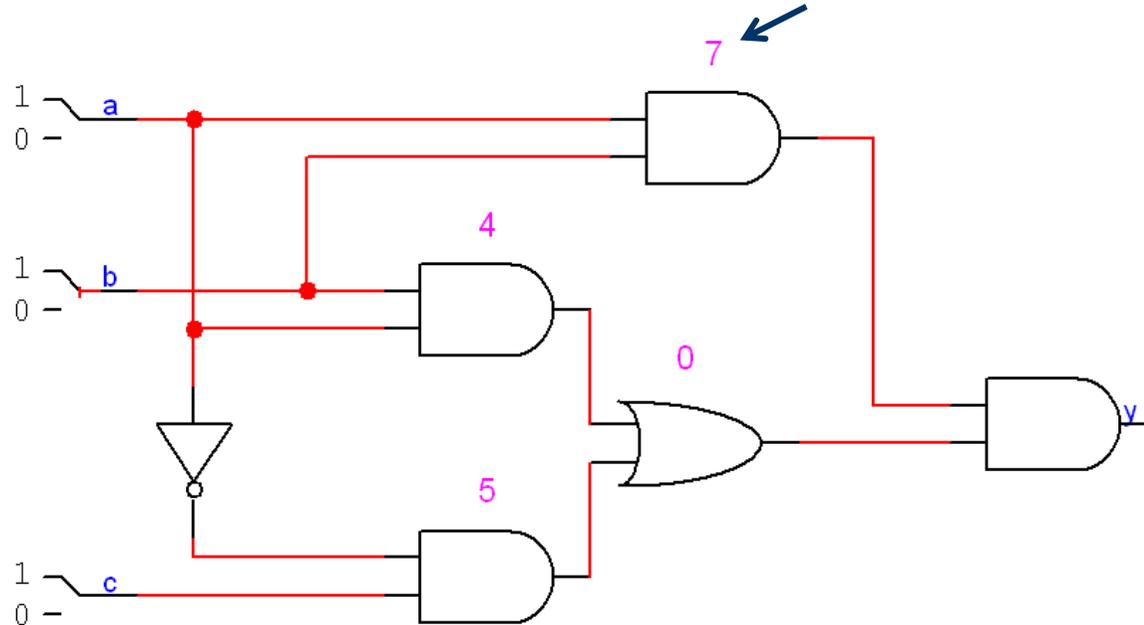
Le alee dinamiche riguardano situazioni in cui, in corrispondenza di un cambiamento degli ingressi, si ha un cambiamento dell'uscita, ma...  
prima di assumere il valore corretto, l'uscita cambia più volte  
valore

ritardi delle porte

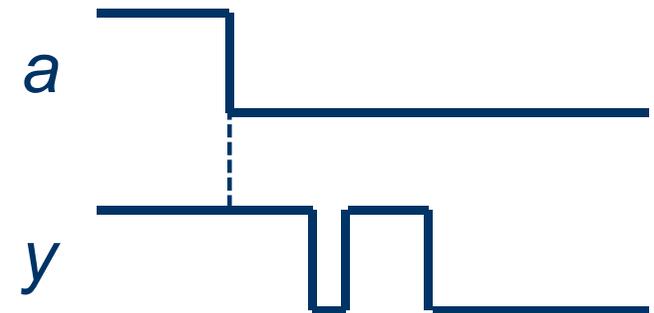
$a=1 \rightarrow 0$

$b=1$

$c=1$



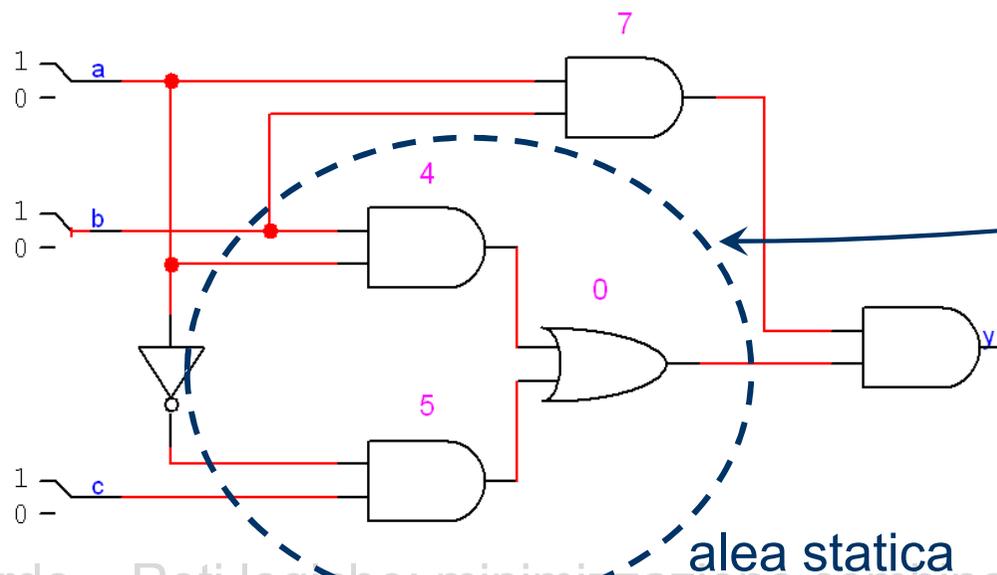
Ciò accade per esempio qui quando i tre ingressi **abc** passano da **111** a **011**. A causa dei ritardi delle porte, l'uscita **y** avrà il seguente andamento:



# Alee Dinamiche

Materiale facoltativo

- ◆ Si verificano solo in reti a più di due livelli.
- ◆ Sono dovute ad alee statiche nei livelli precedenti
- ◆ Si eliminano eliminando le alee statiche nelle sottoreti componenti



# Alee Multiple

Materiale facoltativo

Si ha in generale un'*Alea Multipla* in corrispondenza di una variazione di ingressi in cui almeno due bit cambino contemporaneamente (gli ingressi non sono adiacenti)

**Es.**

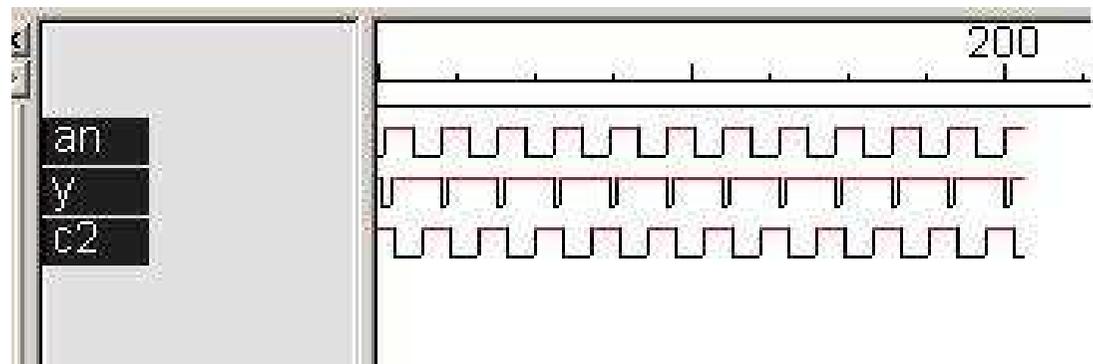
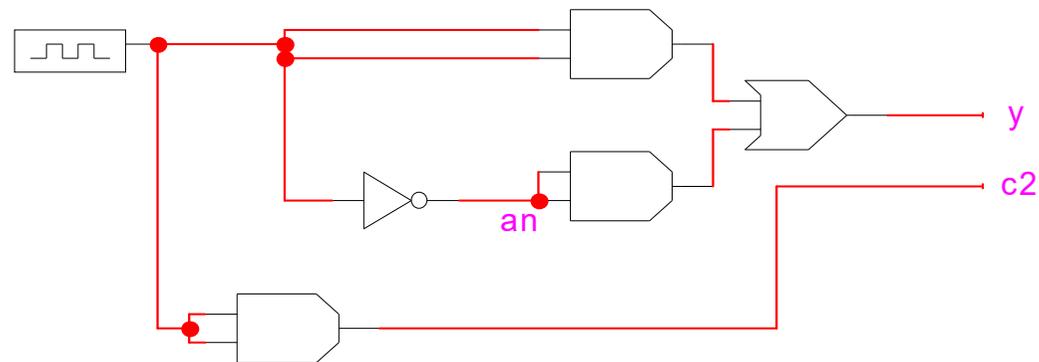
$$Y = X_1X_2 + \overline{X_1}\overline{X_2}$$

quando:

$$X_1X_2 = 00 \rightarrow 11$$

*Unica Soluzione:*

Imporre come vincolo l'applicazione di sequenze di ingressi sempre adiacenti



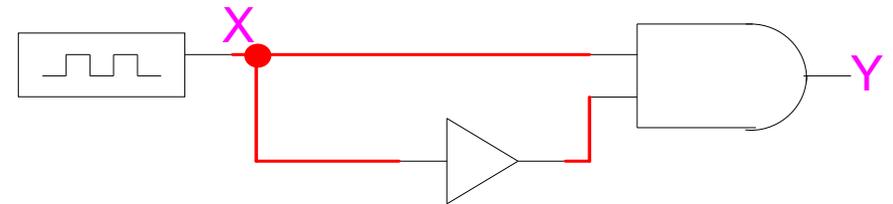
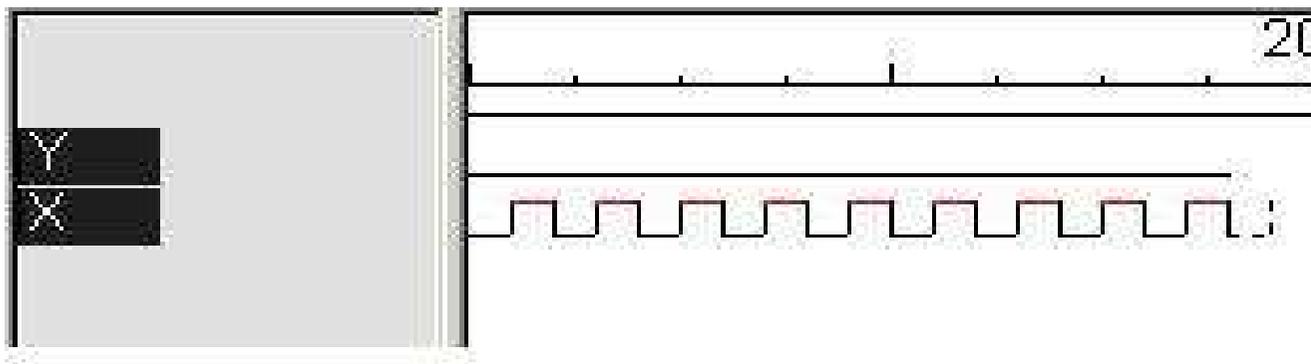
# Impulsi Concomitanti

Materiale facoltativo

Si ha un'*Alea da impulsi concomitanti* quando il funzionamento del sistema dipende dal fatto che due brevi impulsi brevi si presentino nello stesso momento su due distinti segnali del sistema

**Es.**

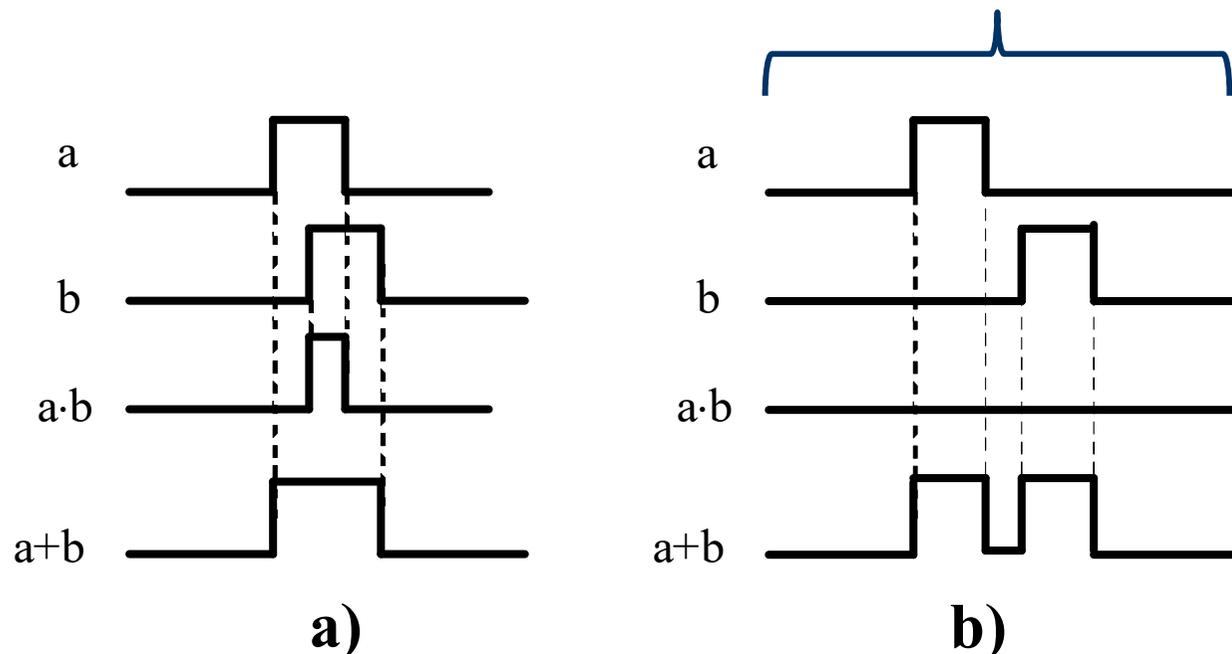
$$Y = X_1 X_2$$



# Impulsi Concomitanti

Materiale facoltativo

- ◆ bastano piccoli disallineamenti accidentali dei due impulsi per alterare la sequenza dei valori prodotti



disallineamento trascurabile

disallineamento più pronunciato

in questo caso, ad esempio, il disallineamento dei due impulsi su **a** e **b** è tale per cui la loro AND ( $ab$ ) e la loro OR ( $a+b$ ) sono totalmente stravolte rispetto al funzionamento corretto

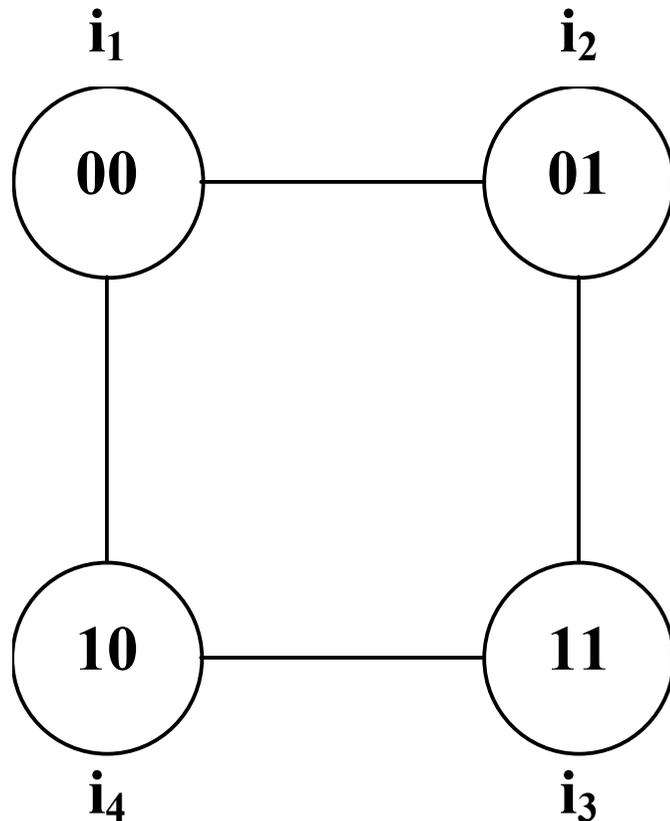
# Progetto orientato all'affidabilità

Materiale facoltativo

- ◆ La presenza di alee *influenza* il progetto delle reti logiche
  - Aggiunta di ridondanza (alee statiche e dinamiche)
  - Codifica degli ingressi (alee multiple)
    - forzare transizioni tra ingressi adiacenti
    - eventualmente aggiungendo variabili (e quindi codifiche dell'ingresso)

# Grafo di transizione degli ingressi

Materiale facoltativo



Il grafo esprime tutti i possibili modi in cui da un ingresso si può passare ad un altro

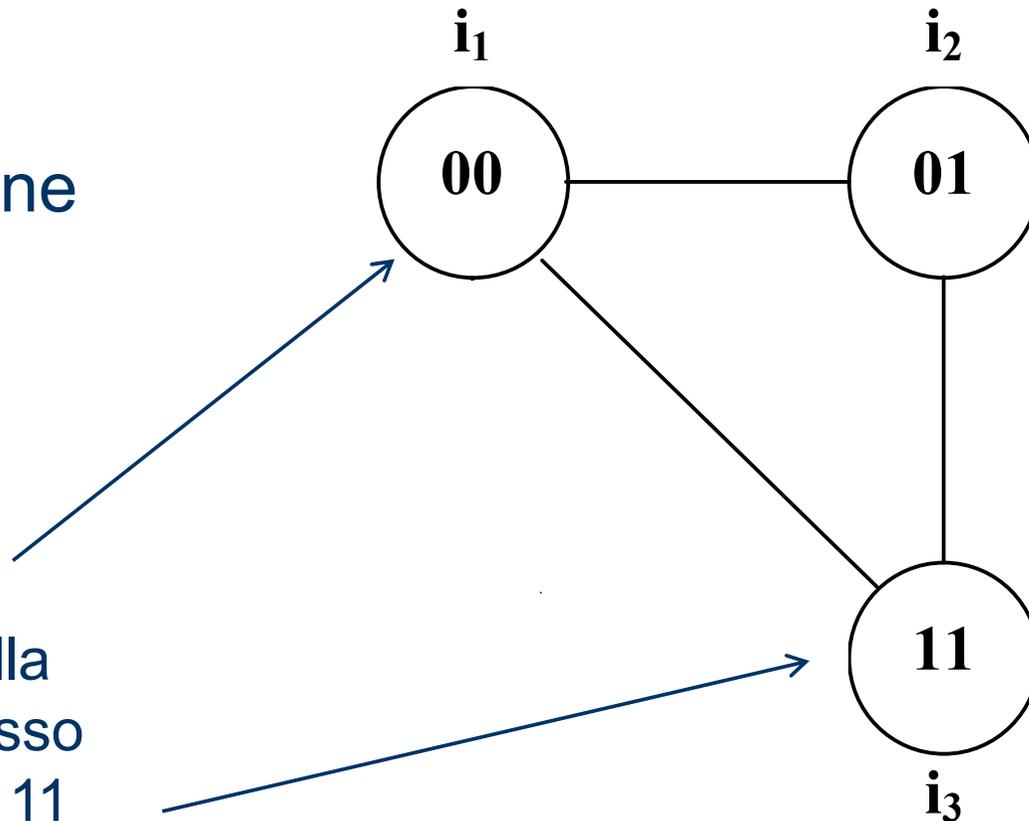
In questo esempio, le transizioni di ingresso ammesse sono tutte tra **ingressi adiacenti** (ovvero, ingressi che differiscono per un solo bit)

→ non ci sono *Alee Multiple*

# Grafo di transizione degli ingressi

Materiale facoltativo

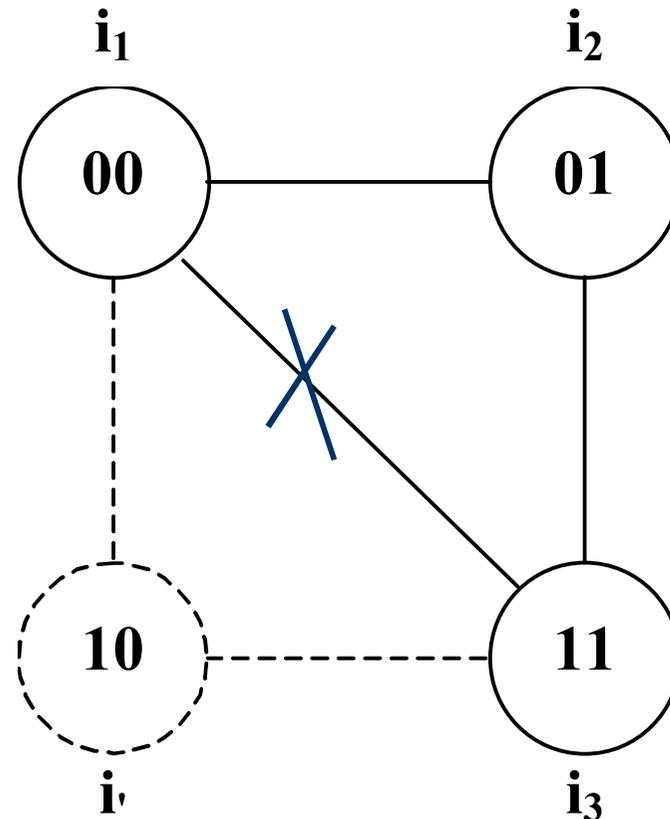
Supponiamo di avere questo grafo di transizione degli ingressi



Il passaggio diretto dalla configurazione d'ingresso 00 e la configurazione 11 introduce un'**alea multipla**

# Aggiunta di configurazioni di ingresso

Materiale facoltativo



per ottenere sempre transizioni tra **ingressi adiacenti**, aggiungiamo “artificialmente” un passaggio intermedio tra **00** e **11** attraverso la configurazione **10**

# Aggiunta di configurazioni di ingresso

Materiale facoltativo

Se il grafo iniziale prevede molte transizioni, può diventare necessario aggiungere molti passaggi in modo da avere solo transizioni tra ingressi adiacenti

