



UNIVERSITÀ DI PISA

CORSO DI LAUREA IN FISICA

LABORATORIO DI FISICA 3

D02 FUNZIONI LOGICHE

Prof. F. Forti (materiale by C. Roda, G.Punzi)

Sommario

2

- Implementazione di funzioni logiche arbitrarie
 - ▣ Somma di prodotti
 - ▣ Prodotti di somme
- Semplificazione con algebra booleana
- Mappe di Karnaugh
- Applicazioni
 - ▣ XOR, parità
 - ▣ Display a 7 segmenti
 - ▣ Sommatore
 - ▣ Comparatore

Forme standard di funzioni logiche

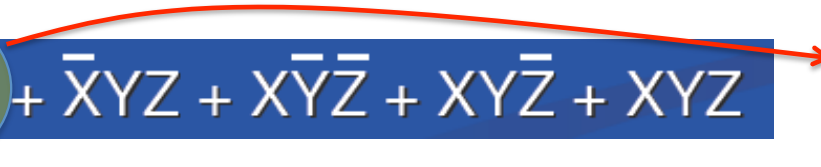
3

- Esistono due modi standard per esprimere funzioni logiche: **SOMMA DI PRODOTTI** e **PRODOTTI DI SOMME** degli ingressi eventualmente opportunamente negati.
- Tutte le funzioni logiche possono essere espresse in queste forme (lo vediamo tra un attimo)
- L'importante conseguenza è che tutte le funzioni logiche possono essere ottenute con porte logiche **AND, OR e NOT !**
- Abbiamo visto che AND, OR e NOT a più di due ingressi possono essere ottenuti banalmente da quelli a due ingressi
-> **tutte le funzioni logiche sono ottenibili da AND, OR e NOT a due ingressi**

Somma di minterm

4

- Consideriamo la funzione logica di tre ingressi ed 1 uscita descritta dalla tabella qui a fianco
- Il valore dell'uscita L puo' essere scritta come OR di tutte le possibili combinazioni degli ingressi che danno 1 in uscita:

$$L = \bar{X}Y\bar{Z} + \bar{X}YZ + X\bar{Y}\bar{Z} + XY\bar{Z} + XYZ$$


Ciascuno dei termini nella somma si indica con *minterm*
Ciascuna volta che una variabile o il suo complemento appaiono in una espressione si indica con *literal*
L ha 5 minterm e 15 literals

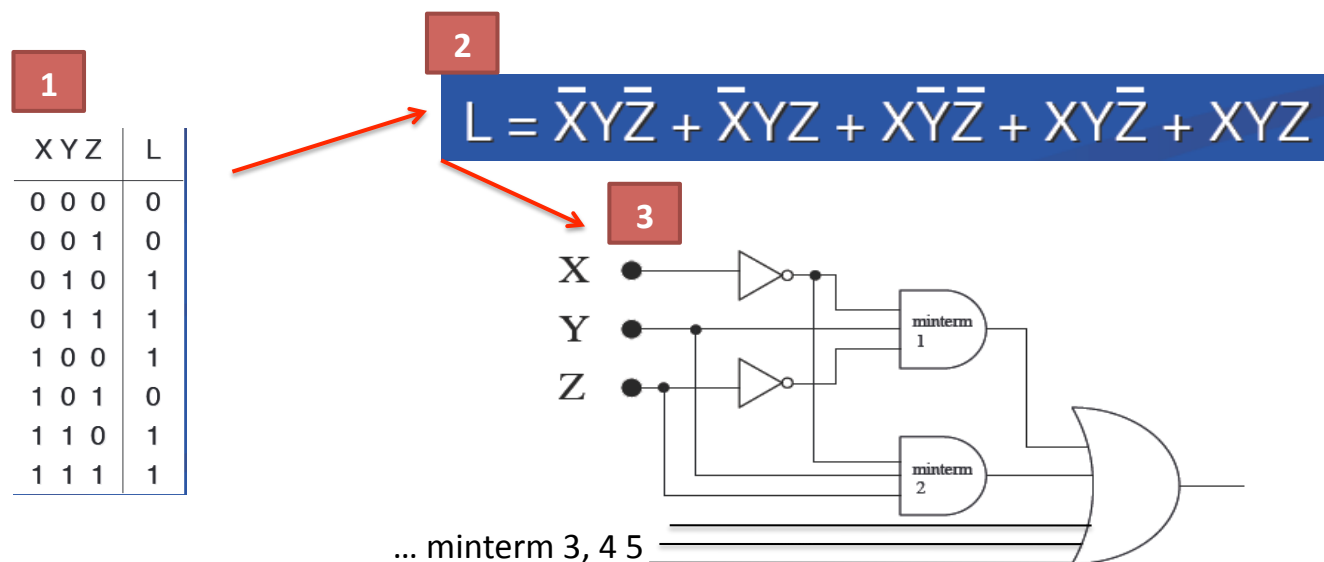
Si vede quindi che qualsiasi funzione logica puo' essere espressa tramite porte AND, OR e NOT.

Ingressi			Uscita
X	Y	Z	L
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Dalla funzione al circuito

5

- Possiamo implementare direttamente la funzione



- Naturalmente non è l'unica implementazione
- E non è necessariamente la migliore
 - ▣ Semplificazione e riduzione

Prodotto di maxterm

6

- Per ottenere l'espressione del prodotto di somme dalla tabella della verità devo fare l'operazione "complementare" a quella per ottenere la somma dei prodotti (OR di tutti miniterm che danno 1 in uscita, mnterm == AND delle variabili opportunamente negate se non 1):

1. Selezione tutte le righe che danno 0
2. costruisco l'OR delle variabili opportunamente negate se non danno 0 (**maxterm**)
3. costruisco l'AND di tutti maxterm

$$L = (X + Y + Z) \cdot (X + Y + \bar{Z}) \cdot (\bar{X} + Y + \bar{Z})$$

X	Y	Z	L
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Usando De Morgan....

7

- Es. La funzione L considerata prima
- Scriviamo NOT(L) e neghiamo

$$\bar{L} = \bar{X}\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z + X\bar{Y}\bar{Z} \quad \text{Righe con zeri}$$

$$L = \bar{\bar{L}} = \overline{\bar{X}\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z + X\bar{Y}\bar{Z}}$$

usando De Morgan:

$$L = (\overline{\bar{X}\bar{Y}\bar{Z}}) \cdot (\overline{\bar{X}\bar{Y}Z}) \cdot (\overline{X\bar{Y}\bar{Z}})$$

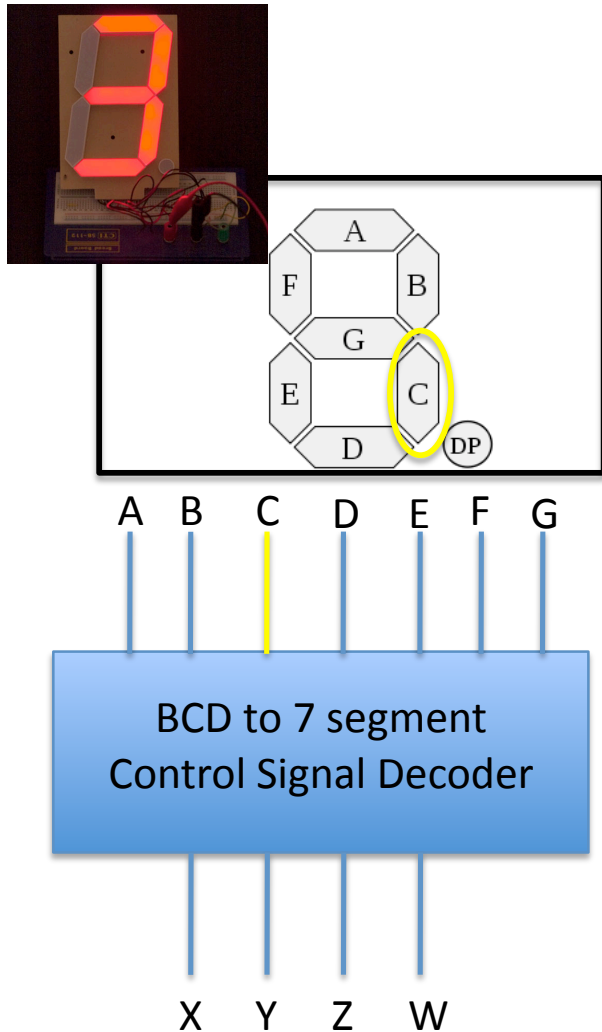
ancora De Morgan:

$$L = (X+Y+Z) \cdot (X+Y+\bar{Z}) \cdot (\bar{X}+Y+\bar{Z})$$

(ognuno dei fattori si chiama *maxterm*)

X	Y	Z	L	\bar{L}
0	0	0	0	1
0	0	1	0	1
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	0

Esempio1 : circuito per gestire un display a sette segmenti



X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1

Proviamo a scrivere l'espressione di C come somma di minterm...

Esempio 1: circuito per gestire un display a sette segmenti

$$C = \bar{X} \cdot \bar{Y} \cdot \bar{Z} \cdot \bar{W} + \bar{X} \cdot \bar{Y} \cdot \bar{Z} \cdot W + \bar{X} \cdot \bar{Y} \cdot Z \cdot W + \bar{X} \cdot Y \cdot \bar{Z} \cdot \bar{W} + \\ \bar{X} \cdot Y \cdot \bar{Z} \cdot W + \bar{X} \cdot Y \cdot Z \cdot \bar{W} + \bar{X} \cdot Y \cdot Z \cdot W + X \cdot \bar{Y} \cdot \bar{Z} \cdot \bar{W} + X \cdot \bar{Y} \cdot \bar{Z} \cdot W$$

Questa espressione ha 36 literals, 9 4-input AND ed 1 9-input OR ... vediamo come puo' essere semplificata

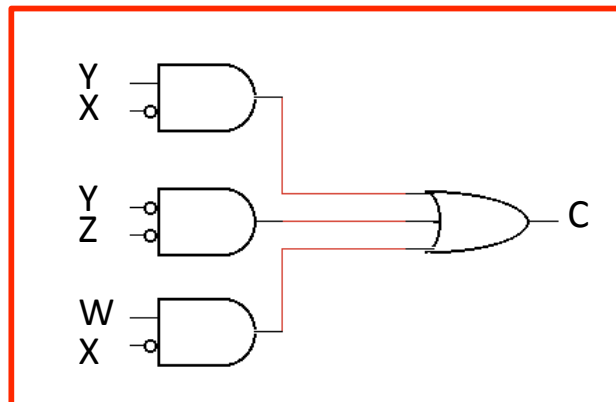
X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1

Esempio 1: circuito per gestire un display a sette segmenti

$$\begin{aligned}
 C &= \bar{X} \cdot \bar{Y} \cdot \bar{Z} \cdot \bar{W} + \bar{X} \cdot \bar{Y} \cdot \bar{Z} \cdot W + \bar{X} \cdot \bar{Y} \cdot Z \cdot W + \bar{X} \cdot Y \cdot \bar{Z} \cdot \bar{W} + \\
 &\bar{X} \cdot Y \cdot \bar{Z} \cdot W + \bar{X} \cdot Y \cdot Z \cdot \bar{W} + \bar{X} \cdot Y \cdot Z \cdot W + X \cdot \bar{Y} \cdot \bar{Z} \cdot \bar{W} + X \cdot \bar{Y} \cdot \bar{Z} \cdot W = \\
 &= \bar{X} \cdot Y \cdot (\bar{Z} \cdot \bar{W} + \bar{Z} \cdot W + Z \cdot \bar{W} + Z \cdot W) + \\
 &\bar{Y} \cdot \bar{Z} \cdot (\bar{X} \cdot \bar{W} + \bar{X} \cdot W + X \cdot \bar{W} + X \cdot W) + \\
 &+ \bar{X} \cdot W \cdot (\bar{Y} \cdot \bar{Z} + \bar{Y} \cdot Z + Y \cdot \bar{Z} + Y \cdot Z) = \\
 &\bar{X} \cdot Y + \bar{Y} \cdot \bar{Z} + \bar{X} \cdot W
 \end{aligned}$$

Raggruppando,
riordinando e
aggiungendo
termini
ridondanti

X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1



Esempio 1: circuito per gestire un display a sette segmenti

Oppure: notando che C e' sempre acceso meno che per il 2.

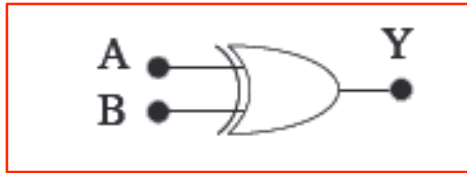
1. Selezione tutte le righe che danno 0
2. costruisco l'OR delle variabili opportunamente negate se non danno 0 (**maxterm**)
3. costruisco l'AND di tutti maxterm

$$C = X + Y + \bar{Z} + W$$

X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1

Commento: la semplificazione e' un'arte, anche se ci sono delle tecniche

Esempio 2: implementazione XOR

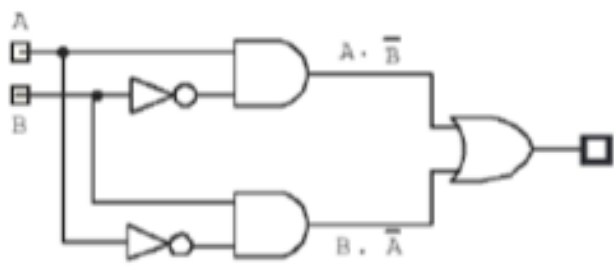


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

1

XOR come somma di minterm
dalla tabela della veria'

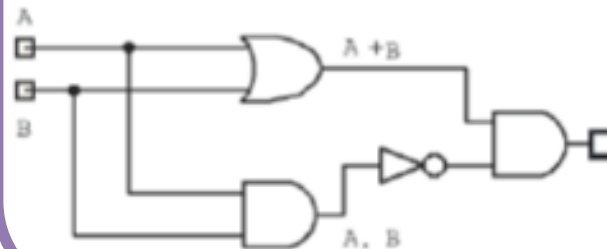
$$A \oplus B = \bar{A} \cdot B + A \cdot \bar{B}$$



2

Oppure: XOR e' vero se e' vero
l'OR e l'AND e' falso

$$A \oplus B = (A + B) \cdot \overline{A \cdot B}$$

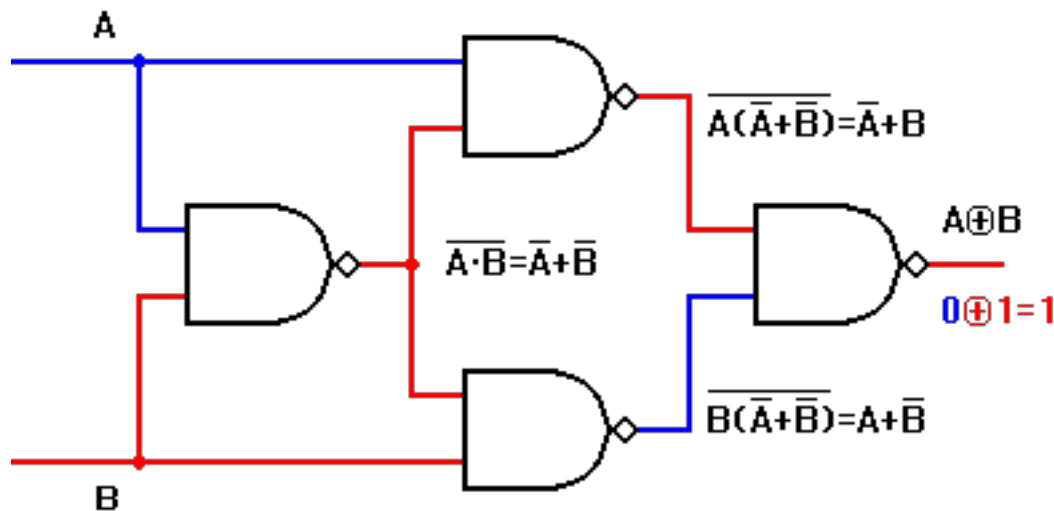


XOR con solo 4 porte NAND

13

□ Trasformazione in NAND

$$A \oplus B = (A+B) \cdot (\overline{A \cdot B}) = \overline{\overline{(A \cdot (\overline{A \cdot B})) + (B \cdot (\overline{A \cdot B}))}} = \overline{(A \cdot (\overline{A \cdot B})) \cdot (B \cdot (\overline{A \cdot B}))}$$



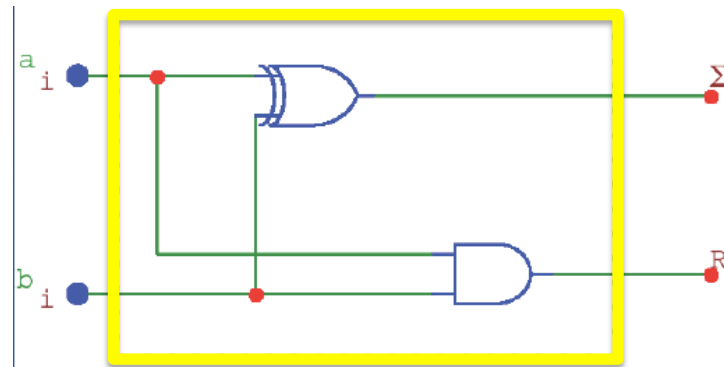
Esempio 3: Somma di nri binari

Consideriamo inizialmente la somma di due nri da 1 bit. Indico con Sum il bit somma e con R il bit riporto

a_i	b_i	Σ	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\Sigma = a_i \oplus b_i$$

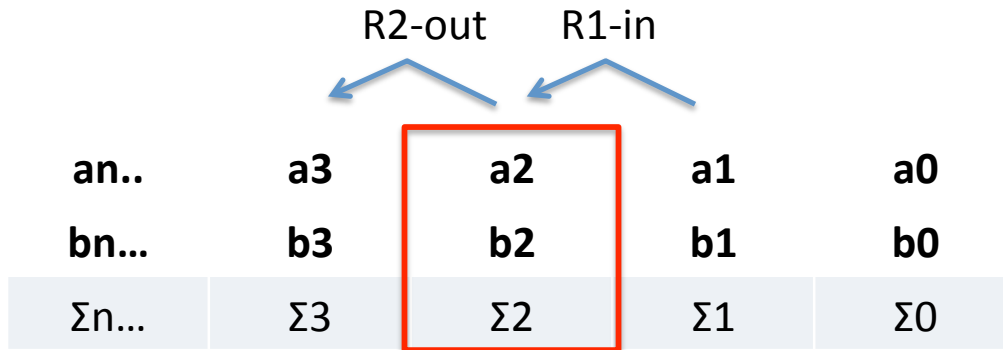
$$R = a_i \cdot b_i$$



Questo circuito si chiama **HALF ADDER**

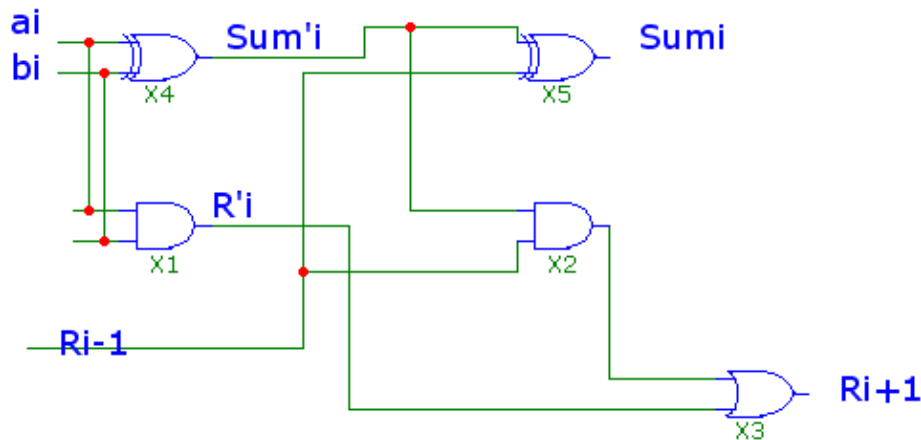
Per poter generalizzare questo circuito alla somma di un numero ad N bit devo considerare il riporto: quando somma i due bit nella posizione “i” devo considerare il riporto dai bit “i-1”

Somma di numeri binari



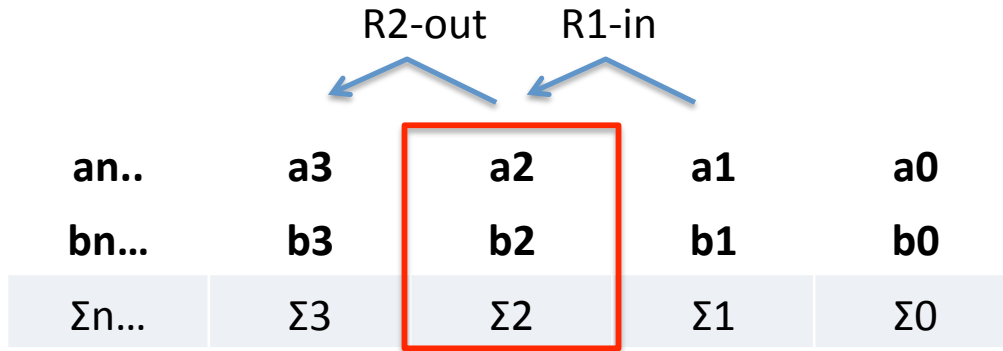
Per ottenere la somma dei due bit in posizione "i" di un nro ad N bit devo sommare alla somma dei due bit implementata con l'HALF ADDER il riporto della somma dei bit nella posizione "i-1"

a_i	b_i	R_{i-1}	Σ	R
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



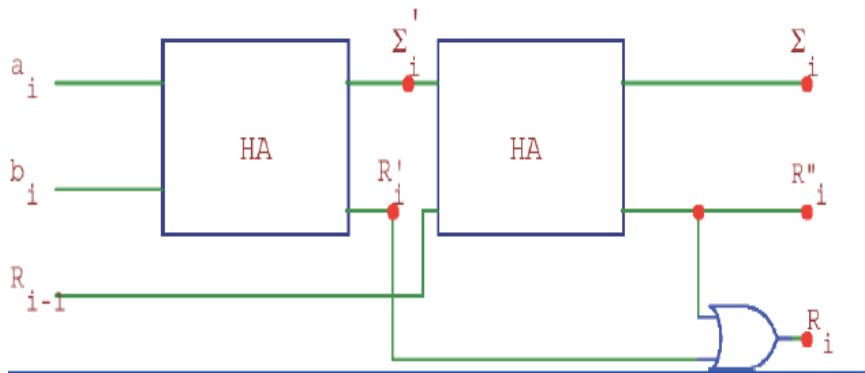
Potete verificare con la tabella della verita' che il circuito implementa la funzione voluta

Somma di numeri binari



Per ottenere la somma dei due bit in posizione “i” di un nro ad N bit devo sommare alla somma dei due bit implementata con l’HALF ADDER il riporto della somma dei bit nella posizione “i-1”

a_i	b_i	R_{i-1}	Σ	R
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



Potete verificare con la tabella della verita' che il circuito implementa la funzione voluta

Semplificazione dei circuiti

17

Semplificazione e ottimizzazione rispetto a cosa ?

- Numero di input (literals)
 - ▣ Riduce il numero di connessioni e di transistors
- Numero di porte logiche
 - ▣ Impatto diretto sull'area del circuito
- Numero di livelli
 - ▣ Riduzione del ritardo di propagazione
- Possibilità di usare logica programmabile
 - ▣ Uso di strutture standardizzate programmabili che semplificano l'implementazione del circuito

Metodi di semplificazione di funzioni

18

- Partendo dai minterm o maxterm
 - ▣ Scegliere il più conveniente. Sfruttare i “don’t care”
- Semplificazione algebrica
 - ▣ Non algoritmica; difficile dire quando ho trovato la soluzione migliore
- Computer-aided
 - ▣ Tempi di calcolo molto grandi per funzioni complesse
 - ▣ Necessaria una guida euristica
- L'intervento manuale è rilevante
 - ▣ Comprensione dei sistemi automatici
 - ▣ Controllo dei risultati su esempi ridotti

La madre delle semplificazioni

19

- Teorema di unificazione $A \cdot (B + \overline{B}) = A$
- Trovare dei subset dei minterm dove solo una variabile cambia valore \rightarrow si può eliminare quella variabile.

- Esempio:

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

(richiede 2 AND, 2 NOT, 1 OR)

$$\text{ma } Y = (A + \overline{A}) \cdot \overline{B} = \overline{B}$$

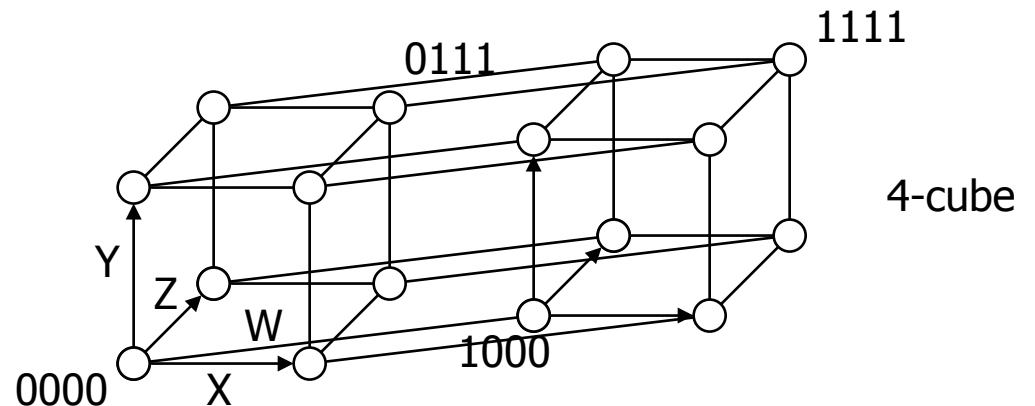
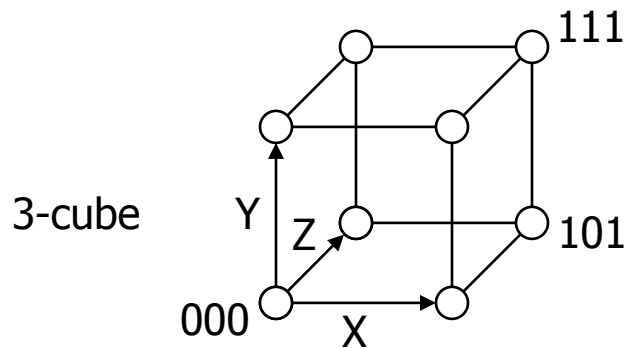
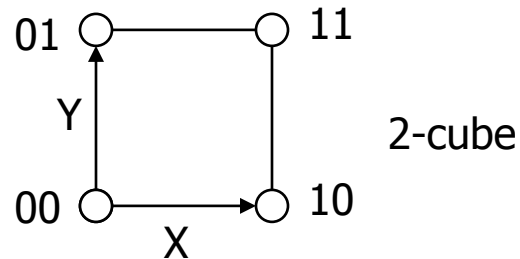
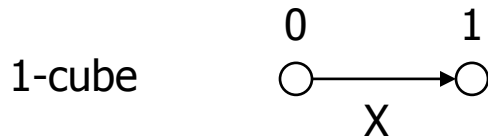
(e' sufficiente un NOT !)

A	B	Y
0	0	1
0	1	0
1	0	1
1	1	0

Metodo dei cubi booleani

20

- N input = cubo N-dimensionale
- Identificare gruppi contigui di nodi “1” o neri

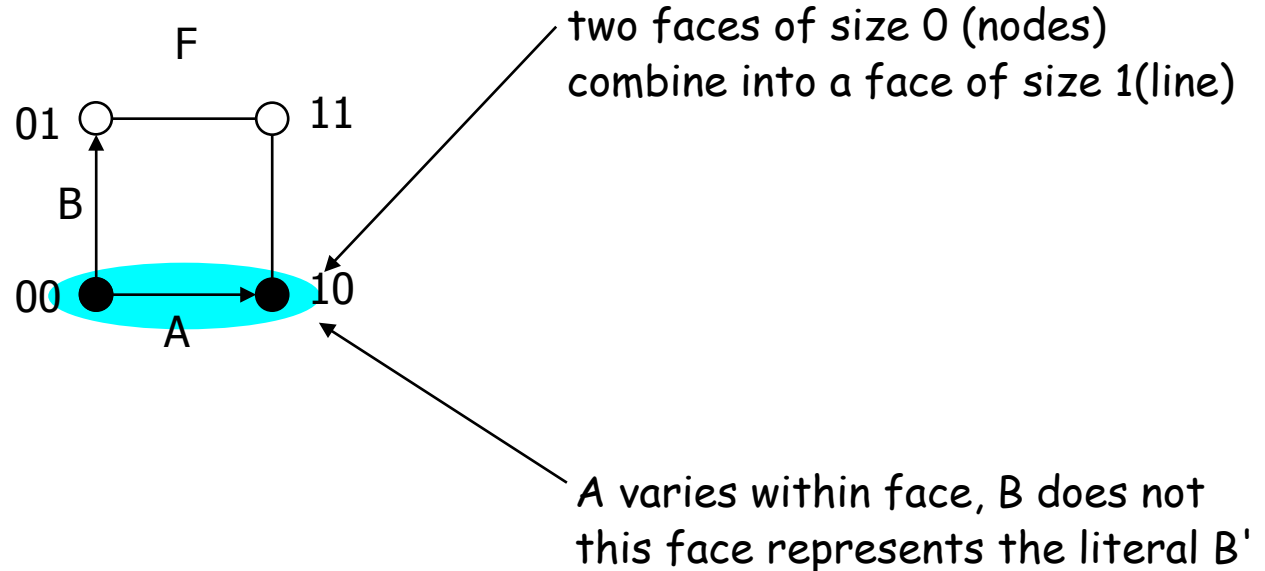


Mappatura delle tabelle di verità

21

- Si possono unire più “facce” dell’ipercubo in elementi più semplici.

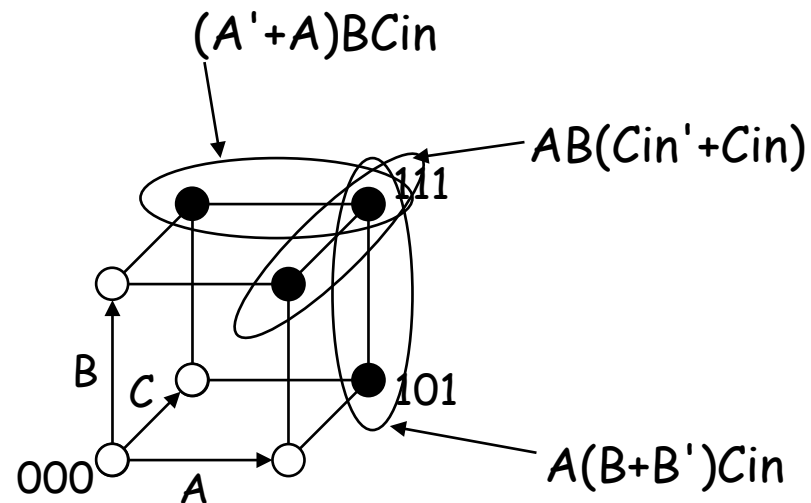
A	B	F
0	0	1
0	1	0
1	0	1
1	1	0



Half-adder

22

A	B	Cin	Cout
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



the on-set is completely covered by the combination (OR) of the subcubes of lower dimensionality - note that “111” is covered three times

$$Cout = BCin + AB + ACin$$

Mappe di Karnaugh

23

- Proiezioni bi-dimensionali dei cubi booleani
 - ▣ Aiutano a trovare le adiacenze
 - ▣ In ogni caso utilizzabili solo per poche variabili
- Sui due assi si mettono i possibili valori di A e B
- Nelle caselle si mette il valore di Y
- Si identificano gli “1” adiacenti e si accorpano
- I termini isolati devono essere espressi come funzioni di entrambe le variabili

A	B	Y
0	0	1
0	1	0
1	0	1
1	1	0

B \ A	0	1
0	1	1
1	0	0

$$\overline{A} \cdot \overline{B} + A \cdot \overline{B} = (\overline{A} + A) \cdot \overline{B} = \overline{B}$$

Altro esempio due ingressi

24

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

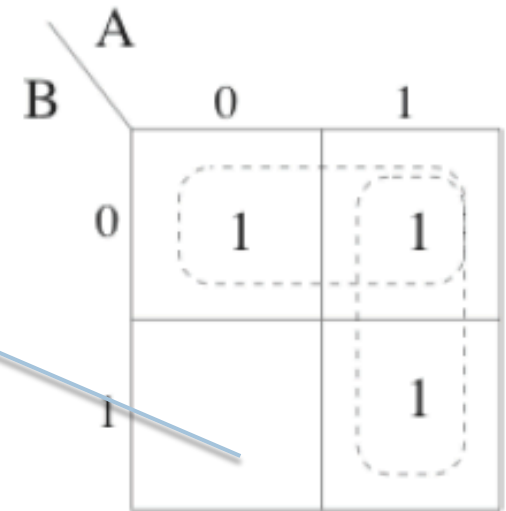
$$Y = (\overline{A} \cdot \overline{B} + A \cdot \overline{B}) + (A \cdot \overline{B} + A \cdot B)$$

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

□ Per i maxterm →
adiacenze di “0”

$$A \cdot B + A \cdot \overline{B} = A \cdot (B + \overline{B}) = A$$

$$(A + B) \cdot (A + \overline{B}) = A + B \cdot \overline{B} = A$$



Tre ingressi

25

- Raggruppare due ingressi su un asse ed ordinarli per codice Gray, in modo che solo un bit cambia
- Si gira intorno: wrap around

$$Y = \bar{A} \cdot \bar{B} \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C + \bar{A} \cdot B \cdot \bar{C} + A \cdot B \cdot \bar{C}$$

Su un asse 1 variabile e
sull'altra le altre 2 scritte in
modo che solo 1 bit cambi di
valore tra due celle adiacenti

		A=B=1 ; A=1, B=0			
C \ B	A	00	01	11	10
	0	1	1	1	
1	1	1	1		

- Si deve immaginare la mappa che si estenda a dx e sx
- 4 celle adiacenti – linea o quadrato – si combinando in modo da fornire una funzione di 1 sola variabile
- 2 celle adiacenti si combinano in modo da fornire una funzione di 2 variabili
- 1 cella isolata: termine a 3 variabili

$$Y = \bar{A} + B \cdot \bar{C}$$

Quattro ingressi

26

La mappa va immaginata
“periodica” su tutto il piano

$$Y = \bar{A} \cdot \bar{B} \cdot \bar{C} + B \cdot \bar{D} + A \cdot B$$

C \ B	00	01	11	10
D 00	1	1	1	
01	1		1	
11			1	
10		1	1	

- 8 celle (rettangolo) : termine a 1 variabile
- 4 celle : termine a 2 variabili
- 2 celle : termine a 3 variabili

Esempio 1: comparatore di nri binari

- Confronto di due nri binari a 2 bit: A e B

	Y
A>B	1
A<=B	0

A1	A0	B1	B0	Y
0	1	0	0	1
1	0	0	0	1
1	1	0	0	1
1	0	0	1	1
1	1	0	1	1
1	1	1	0	1

Per tutti gli altri casi: Y = 0

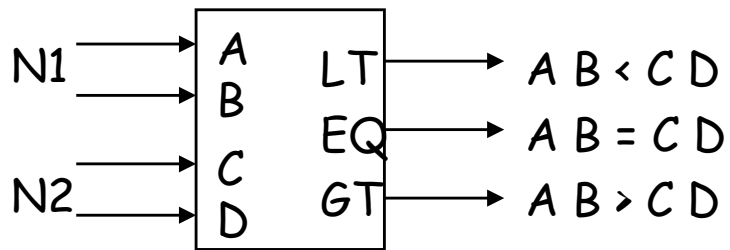
Mappa di Karnaugh

		A1 A0 B1 B0	00	01	11	10
00				1	1	1
01					1	1
11						
10					1	

$$Y = \overline{B1} \cdot A1 + A0 \cdot \overline{B0} \cdot \overline{B1} + A0 \cdot A1 \cdot \overline{B0}$$

Anche EQ e LT

28

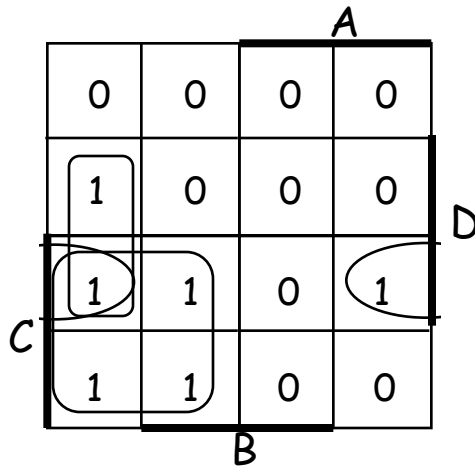


block diagram
and
truth table

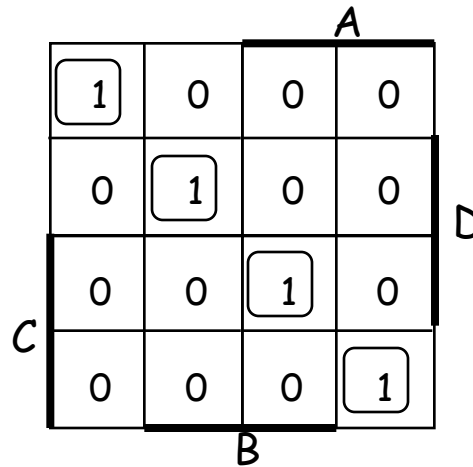
A	B	C	D	LT	EQ	GT
0	0	0	0	0	1	0
		0	1	1	0	0
		1	0	1	0	0
		1	1	1	0	0
0	1	0	0	0	0	1
		0	1	0	1	0
		1	0	1	0	0
		1	1	1	0	0
1	0	0	0	0	0	1
		0	1	0	0	1
		1	0	0	1	0
		1	1	1	0	0
1	1	0	0	0	0	1
		0	1	0	0	1
		1	0	0	0	1
		1	1	0	1	0

Anche LT, EQ, GT

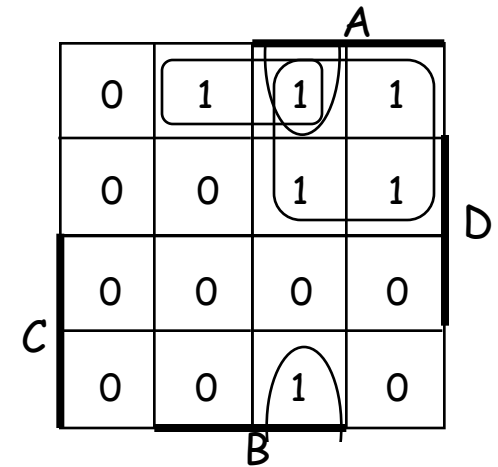
29



K-map for LT



K-map for EQ



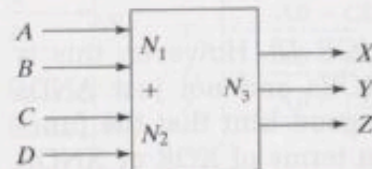
K-map for GT

SCRIVERE LE FUNZIONI PER I TRE OUTPUT

Sommatore a due bit

30

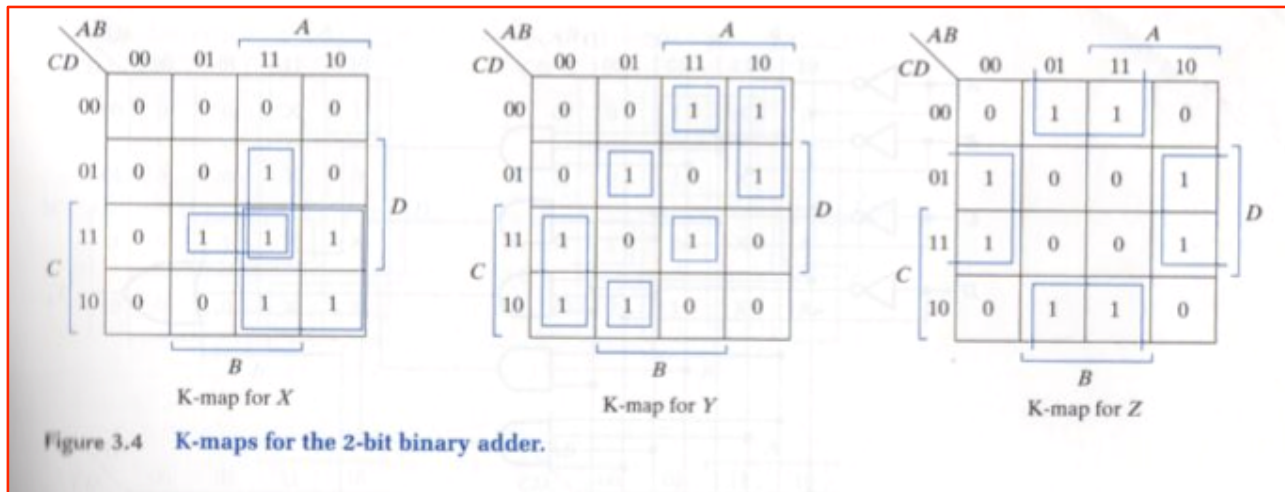
- Ingressi: N1(A,B) N2(C,D) a due bit
- Uscita: N3(X,Y,Z) 3 bit



A	B	C	D	X	Y	Z
0	0	0	0	0	0	0
		0	1	0	0	1
		1	0	0	1	0
		1	1	0	1	1
0	1	0	0	0	0	1
		0	1	0	1	0
		1	0	0	1	1
		1	1	1	0	0
1	0	0	0	0	1	0
		0	1	0	1	1
		1	0	1	0	0
		1	1	1	0	1
1	1	0	0	0	1	1
		0	1	1	0	0
		1	0	1	0	1
		1	1	1	1	0

Figure 3.3 Block diagram and truth table of 2-bit binary adder.

Esempio 2: sommatore di nri a 2 bit



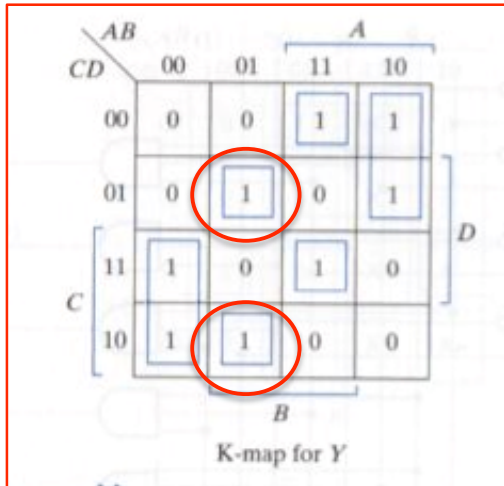
$$X = AC + BCD + ABD$$

$$Z = \overline{B}\overline{D} + \overline{B}D = B \oplus D$$

Per Y, dalla mappa di Karnaugh posso esprimere come:

- 2 termini a 3 variabili + 4 termini a 4 variabili
- oppure
- 4 termini di 3 variabili + 2 termini di 4 variabili

Esempio 2: sommatore di nri a 2 bit



Per Y, dalla mappa di Karnaugh posso esprimere come:

- 2 termini a 3 variabili + 4 termini a 4 variabili

oppure

- 4 termini di 3 variabili + 2 termini di 4 variabili

Considero due dei termini a 4 variabili:

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

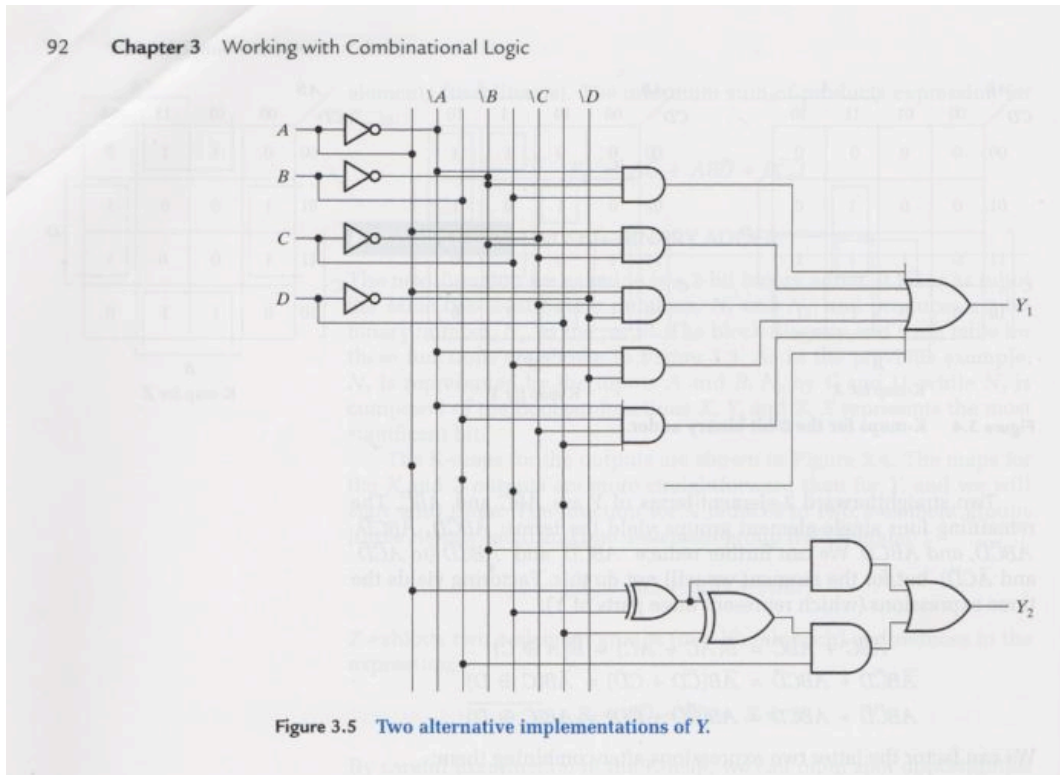
Facendo analoghe elaborazioni per gli altri termini:

$$Y_1 = \overline{B}(A \oplus C) + B(A \oplus C \oplus D)$$

Nel secondo caso :

$$Y_2 = \overline{A}\overline{B}C + \overline{A}B\overline{C} + \overline{A}C\overline{D} + \overline{A}C\overline{D} + \overline{A}B\overline{C}D + \overline{A}B\overline{C}D$$

Esempio 2: sommatore di nri a 2 bit



Y2 richiede 2 AND a 4 ingressi e 4 AND a 3 ingressi ed 1 OR a 6 ingressi per un totale di 7 gate e 20 literals

Y2 richiede solo 5 gate a 2 ingressi tuttavia due gate sono XOR.

$$Y_1 = \overline{B}(A \oplus C) + B(A \oplus C \oplus D)$$

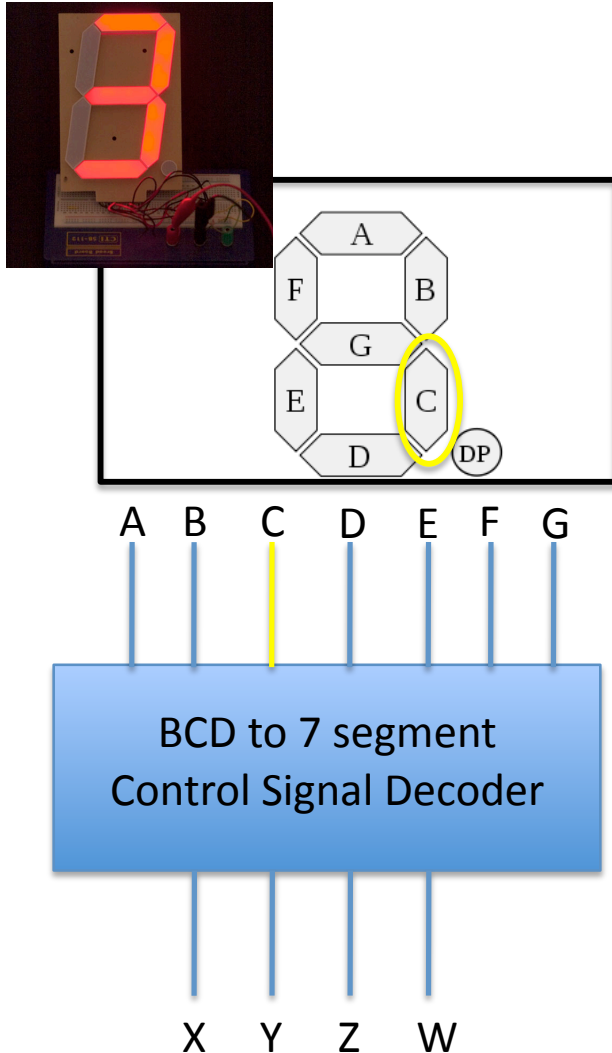
$$Y_2 = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{C}\overline{D} + \overline{A}C\overline{D} + \overline{A}B\overline{C}D + ABCD$$

Don't cares

34

- In molti casi le funzioni non sono specificate per tutti gli ingressi possibili
 - ▣ Si indica con X il don't care = riga per cui non importa quale valore assume la funzione
- Di solito sono casi che non si presentano mai
- Si può scegliere 0 oppure 1 a seconda della convenienza → semplificazione ulteriore

Mappa di Karnaugh e “don't cares”



X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X

Visto la volta scorsa

Parte della funzione non definita

Esempio1 : circuito per gestire un display a sette segmenti

X	Y	Z	W	C
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	x
1	0	1	1	x

La parte della funzione non definita puo` essere sfruttata per semplificare il circuito.

Mappa di Karnaugh

$x \backslash z$ w	00	01	11	10
00	1	1	x	1
01	1	1	x	1
11	1	1	x	x
10	0	1	x	x

Parte della
funzione
non
definita

$$C = X + Y + \bar{Z} + W =$$

$$= Y + \bar{Z} + W$$