Corso di Laurea in
Informatica

Introduzione alle Reti Logiche

Architettura dei Calcolatori

Prof. Andrea Marongiu
andrea.marongiu@unimore.it

Anno accademico 2018/19

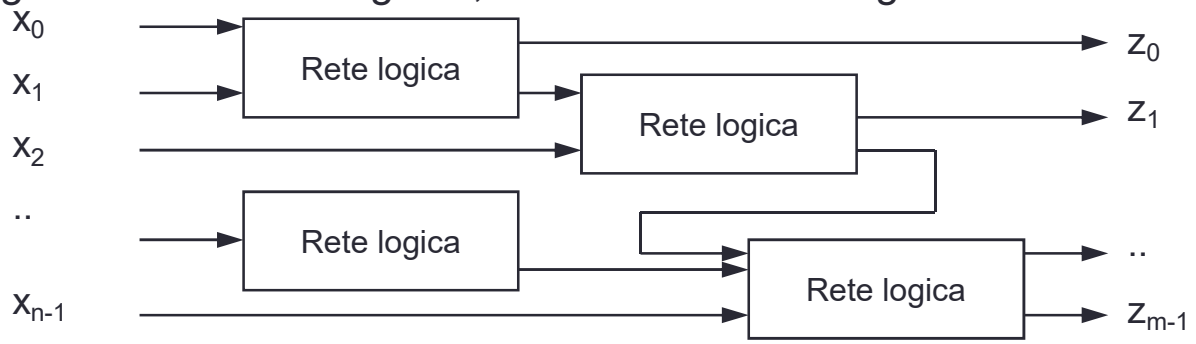
Reti logiche

- Livello di astrazione che studia i sistemi digitali a livello di componenti LOGICI elementari indipendentemente dalla tecnologia con cui il sistema viene realizzato.
- Rete logica: sistema digitale avente *n* segnali binari di ingresso ed *m* segnali binari di uscita.
- I segnali sono rigorosamente binari (0/1).

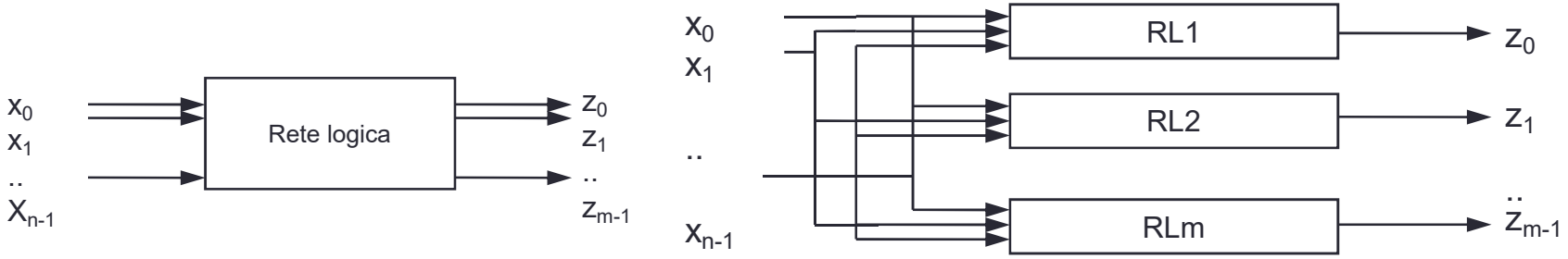
- I segnali sono grandezze funzioni del tempo
- X ={xn-1(t),…….x0(t)} Z={zm-1(t),……..z0(t)}
- zi(t)= fi(xn-1(t),…….x0(t))
- I segnali di ingresso ed uscita delle reti logiche possono essere singoli segnali binari (es. RESET) o segnali digitali composti in parole codificate come un insieme di segnali binari

Proprietà delle reti logiche

- **Proprietà di interconnessione:** l'interconnessione di più reti logiche, aventi per ingresso segnali esterni o uscite di altre reti logiche e per uscite segnali di uscita esterne o ingressi di altre reti logiche, è ancora una rete logica



- **Proprietà di decomposizione:** una rete logica complessa può essere decomposta in reti logiche più semplici (fino all'impiego di soli blocchi o gate elementari)
- **Proprietà di decomposizione in parallelo**: una rete logica a m uscite può essere decomposta in m reti logiche ad 1 uscita, aventi ingressi condivisi



Reti combinatorie e sequenziali

- Reti COMBINATORIE $z_i(t)=f(x_0(t),...,x_{n-1}(t))$
- Reti SEQUENZIALI $z_i(t)=f((x_0(t),...,x_{n-1}(t), t)$

- Rete **combinatoria**: ogni segnale di uscita dipende solo dai valori degli ingressi in quell'istante
- Rete **sequenziale**: ogni segnale di uscita dipende dai valori degli ingressi in quell'istante e dai valori che gli ingressi hanno assunto negli istanti precedenti

- Rete combinatoria: rete senza memoria (l'uscita cambia "istantaneamente" dopo che l'ingresso è cambiato)
- Rete sequenziale: rete con memoria; è una rete in cui l'uscita cambia in funzione del cambiamento dell'ingresso e della specifica configurazione interna in quell'istante (STATO). Lo stato «riassume» la sequenza degli ingressi precedenti

Reti combinatorie e sequenziali (2)

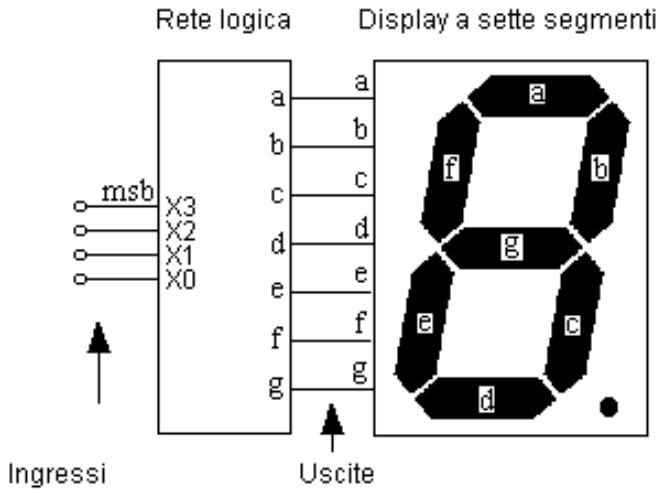
- Una rete combinatoria, quindi non ha STATO. Non ricorda gli ingressi precedenti. Transitori a parte, basta conoscere gli ingressi in un istante per sapere esattamente quali saranno tutte le uscite nel medesimo istante.
- Le reti sequenziali, invece, hanno memoria. Per sapere l'uscita in un certo istante ho due possibilità:
  - Mi ricordo TUTTI gli ingressi che si sono presentati alla rete dalla sua accensione
  - Memorizzo uno STATO del sistema, che riassume in qualche modo tutti gli ingressi precedenti al fine di valutare il valore delle uscite.

Esempio di rete combinatoria

Conversione di valori BCD su display a sette segmenti

- Descrizione comportamentale (a parole):
  progettare una rete logica che permette la visualizzazione su un display a sette segmenti di un valore in codice BCD.
- **Codifica BCD:** impiego di 4 cifre binarie per la rappresentazione di un numero decimale da 0 a 9.
- Es:

15	decimale
1111	binario
0001 0101	BCD

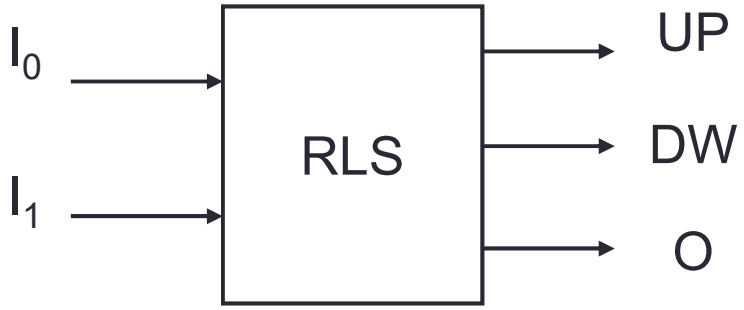


- L'uscita Z={a,b,…g} dipende in ogni istante dalla configurazione degli ingressi $\{x_3,x_2,x_1,x_0\}$

Esempio di rete sequenziale

Progettare la rete logica di gestione di un ascensore.

- La rete ha tre uscite UP, DW e O. UP, DW indicano le direzioni su e giù mentre O vale 1 se la porta deve essere aperta e 0 altrimenti. La rete ha come ingresso due segnali che indicano il piano {0,1,2,3} corrispondente al tasto premuto. Per calcolare l'uscita è necessario conoscere il piano corrente che indica lo stato interno.



Descrizione delle reti combinatorie

1) **Descrizione comportamentale a parole:** descrizione a parole del comportamento della rete logica (poco formale e precisa)
2) **Tabelle di verità:** descrizione esaustiva di tutte le configurazioni di uscita per ogni possibile configurazione di ingresso
3) **Mappe:** altra rappresentazione delle tabelle della verità
4) **Espressioni dell'algebra Booleana**
5) **Schema logico:** descrizione strutturale
6) **Forme d'onda:** descrizione comportamentale in funzione del tempo
7) **Linguaggi di descrizione dell'hardware**

- **Tabella di verità**: tabella che associa tutte le possibili combinazioni degli ingressi alle corrispondenti configurazioni delle uscite e indica esaustivamente il comportamento della rete logica
- Se la rete combinatoria ha n ingressi e m uscite, allora la tabella di verità ha (n+m) colonne e 2^n righe
- Oppure per la proprietà di decomposizione si possono definire tante tabelle quante sono le uscite

Tabelle di verità (1/2)

- Si dicono **COMPLETAMENTE SPECIFICATE** se ogni valore della tabella assume il valore logico di vero o falso (1, 0)

- Si dicono **NON COMPLETAMENTE SPECIFICATE** se contengono condizioni di indifferenza. Si verifica in due casi:

C.1) *se alcune configurazioni di ingressi sono vietate*

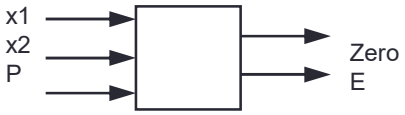
Es: conversione BCD 7 segmenti

x3	*x2*	*x1*	*x0*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	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	-	-	-	-	-	-	-

Tabelle di verità (2/2)

C.2) *se le uscite sono indifferenti per alcune configurazioni di ingresso*

Esempio: progettare una rete che indichi se due ingressi binari sono entrambi uguali a zero, se il segnale di parità pari e' corretto, altrimenti indichi errore

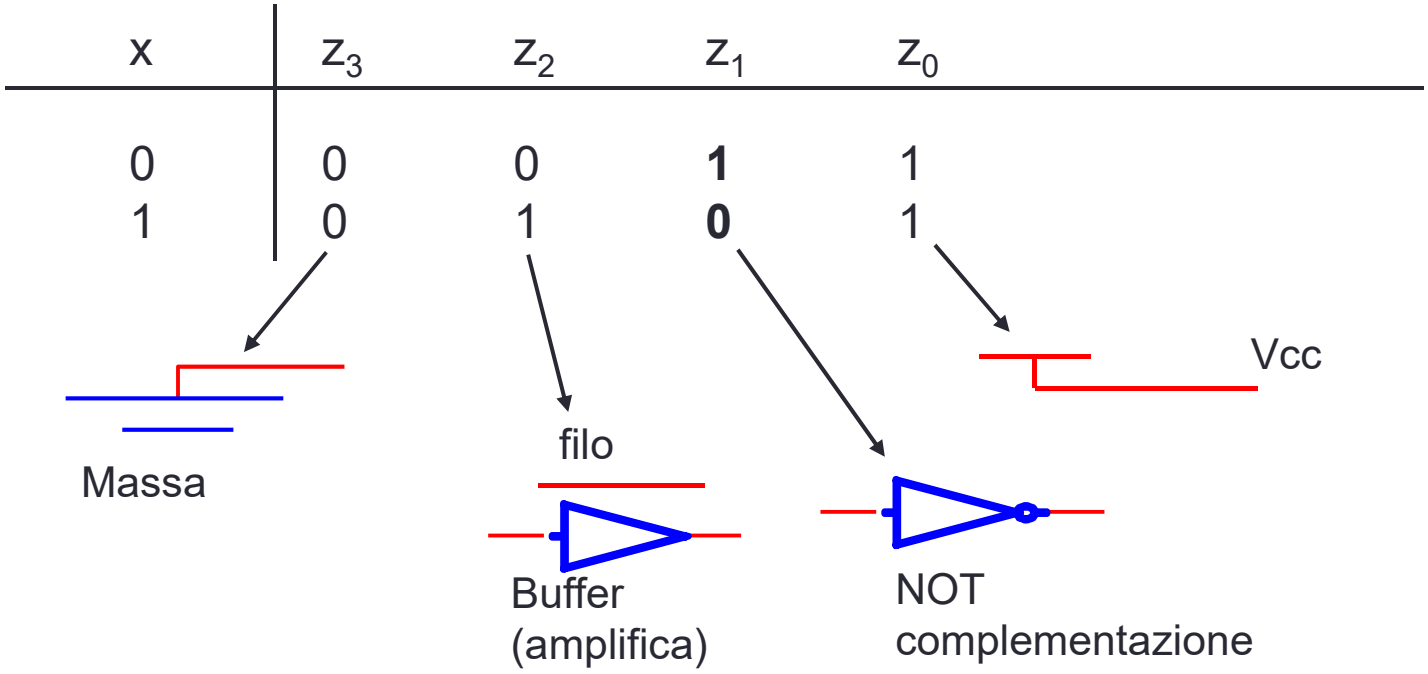


x1	x2	P	Zero	E
0	0	0	1	0
0	0	1	-	1
0	1	0	-	1
0	1	1	0	0
1	0	0	-	1
1	0	1	0	0
1	1	0	0	0
1	1	1	-	1

Funzioni combinatorie e gate elementari

- Le reti logiche combinatorie sintetizzano funzioni combinatorie.
- Per ogni n, è finito il numero di funzioni combinatorie di n variabili di ingresso. Alcune funzioni combinatorie elementari hanno una rappresentazione logica e grafica elementare (gate)

Funzioni di 1 sola variabile indipendente



Funzioni di 2 variabili indipendenti

x_1 x_0	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7
0 0	0	**0**	0	0	0	0	**0**	**0**
0 1	0	**0**	0	0	1	1	**1**	**1**
1 0	0	**0**	1	1	0	0	**1**	**1**
1 1	0	**1**	0	1	0	1	**0**	**1**

z_1 → AND

vale 1 se e solo se tutti gli ingressi valgono 1 (equivale al prodotto logico in logica positiva)

z_7 → OR

vale 1 se e solo se almeno uno degli ingressi vale 1 (equivale alla somma logica in logica positiva)

z_6 → EXOR

vale 1 se e solo se x_1 o x_2 valgono 1 ma non entrambi (diseguaglianza)

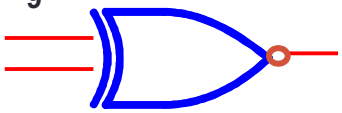
Funzioni di 2 variabili indipendenti

x_1 x_0	z_8	z_9	z_{10}	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}
0 0	**1**	**1**	1	1	1	1	**1**	1
0 1	**0**	**0**	0	0	1	1	**1**	1
1 0	**0**	**0**	1	1	0	0	**1**	1
1 1	**0**	**1**	0	1	0	1	**0**	1

z_8 → NOR

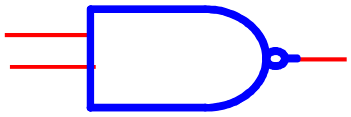
vale 1 se e solo se nè x_1 nè x_2 valgono 1 (l'uscita è il complemento di z_7)

z_9 → EXNOR



EQUIVALENCE: vale 1 se e solo se x_1 e x_2 sono uguali (l'uscita è il complemento di z_6)

z_{14} → NAND



vale 0 se e solo se nè x_1 nè x_2 valgono 0 (l'uscita è il complemento di z_1)

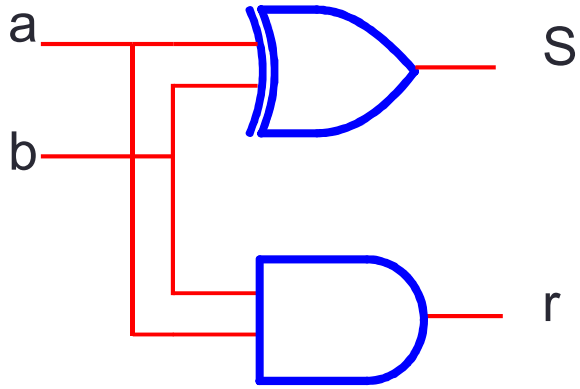
Funzioni combinatorie

- Quante sono le possibili funzioni binarie di n variabili ?
- Tutte le combinazioni delle uscite per ogni configurazione di ingresso, ossia 2 elevato al numero delle possibili configurazioni di ingresso

$$N.\ conf = 2^{(2^n)}$$

- **Esempio di rete logica con gate elementari**: Progettare un HALF ADDER, ossia un sommatore senza riporto in ingresso

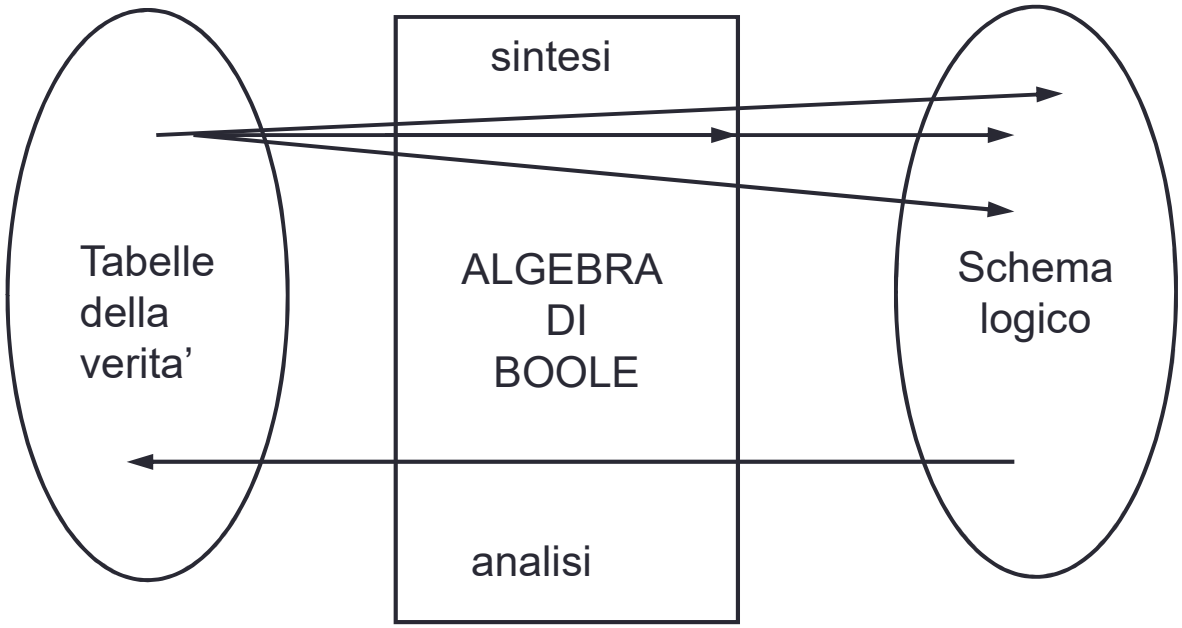
b a	S r
0 0	0 **0**
0 1	1 **0**
1 0	1 **0**
1 1	0 **1**



Algebra di Boole (1/2)

- Uno strumento potente di rappresentazione delle reti logiche combinatorie è data dalle espressioni dell'ALGEBRA DI BOOLE o ALGEBRA DI COMMUTAZIONE

- E' il sistema matematico usato per la sintesi e per l'analisi, per passare dalle tabelle della verità allo schema logico e viceversa



Algebra di Boole (2/2)

- L'algebra di Boole è un sistema matematico che descrive funzioni di variabili binarie: è composto da
  - un insieme di simboli **B={0,1}**
  - un insieme di operazioni **O={+,•,'}**
    - + somma logica
    - • prodotto logico
    - ' complementazione
  - un insieme di postulati (assiomi) **P:**

P1) 0+0=0	P5) 0 •0= 0	P9) 0' =1
P2) 0+1=1	P6) 0 •1= 0	P10) 1' =0
P3) 1+0=1	P7) 1 •0= 0	
P4) 1+1=1	P8) 1 •1= 1	

Proprietà di chiusura:

per ogni a, b $\in$B

a+b $\in$ B

a•b $\in$ B

- **COSTANTI** dell'algebra: le costanti 0 ed 1
- **VARIABILE:** un qualsiasi simbolo che può essere sostituito da una delle due costanti

Funzioni Booleane

- Una **funzione completamente specificata** *di n variabili* $f(x_{n-1},..., x_1,x_0)$ è l'insieme di tutte le possibili coppie formate da un elemento di B^n (dominio) e da un elemento di B (codominio).

- La tabella della verità è un tipico modo per descrivere una funzione dell'algebra di Boole.

- Esiste corrispondenza 1:1 tra una tabella della verità e funzione Booleana.

f(x2,x1,x0):BxBxB →B

x2	x1	x0	f(x2,x1,x0)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- **Complementazione**: il valore complementato di A si indica come A' oppure $\overline{A}$.
- Il simbolo • del prodotto logico viene spesso omesso.

Espressioni Booleane

Un'**espressione** secondo l'algebra di Boole è una stringa di elementi di B che soddisfa una delle seguenti regole:

- una costante è un'espressione;
- una variabile è un'espressione;
- se X è un'espressione allora il complemento di X è un'espressione;
- se X,Y sono espressioni allora la somma logica di X e Y è un'espressione;
- se X,Y sono espressioni allora il prodotto logico di X e Y è un'espressione.

TEOR: *ogni espressione di n variabili descrive una funzione completamente specificata che può essere **valutata** attribuendo ad ogni variabile un valore assegnato.*

es: dalla tabella della verità precedente:

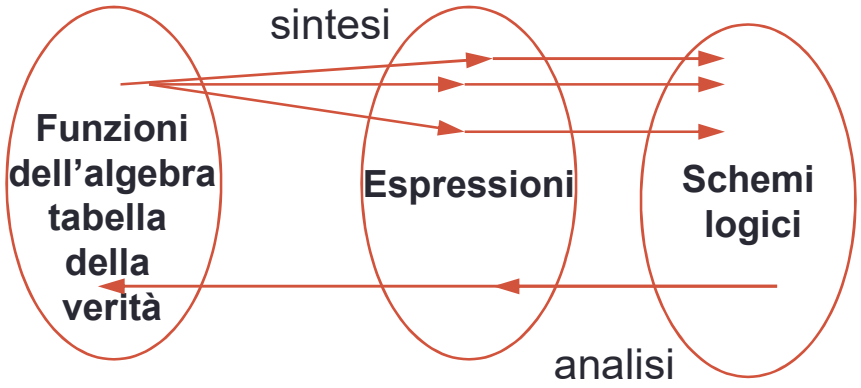
f(x3,x2,x1)= x1'x2'x3+x1x2'x3'+x1x2x3

- Se ogni espressione definisce univocamente una funzione non e' vero il contrario: per ogni funzione esistono più espressioni che la descrivono e si dicono logicamente **equivalenti.**

TEOR: *una espressione di n variabili descrive in maniera univoca uno schema logico di AND, OR e NOT*

Analisi di uno schema logico

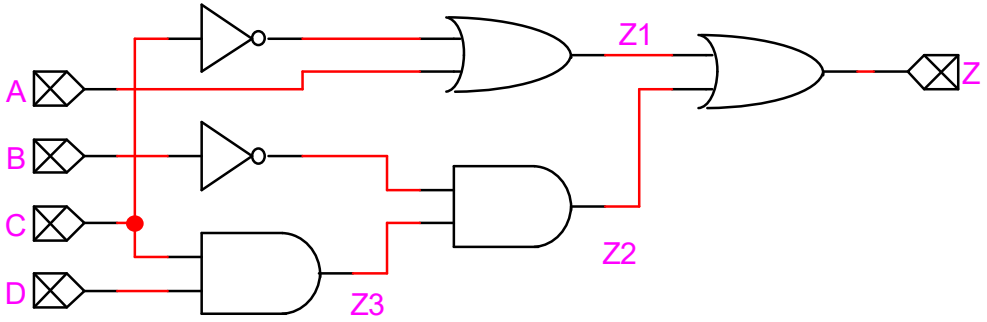
- Dallo schema logico tramite le espressioni è possibile ricavare il comportamento di una rete logica



Analisi:

1) nominando tutte le uscite dei gate logici
2) per sostituzione a partire dalle uscite si ottiene una funzione Booleana delle sole variabili di ingresso

Esercizio: Eseguire l'analisi del seguente schema



Teoremi dell'algebra di Boole (1/4)

Principio di Dualità:

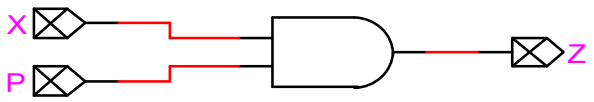
- *ogni espressione algebrica presenta una forma duale ottenuta scambiando l'operatore OR con AND, la costante 0 con la costante 1 e mantenendo i letterali invariati.*
- *ogni proprietà vera per un'espressione è vera anche per la sua duale.*
- il principio di dualità è indispensabile per trattare segnali attivi alti e segnali attivi bassi.

Teor. di Identità

- (T1) $X + 0 = X$ (T1') $X \cdot 1 = X$

Teor. di Elementi nulli

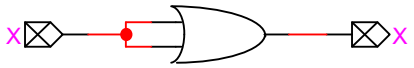
- (T2) $X + 1 = 1$ (T2') $X \cdot 0 = 0$
- sono molto utili nella sintesi di reti logiche: gli elementi nulli permettono di "lasciar passare" un segnale di ingresso in determinate condizioni
- ad es: progettare una rete logica che fornisca in uscita il valore di X se un pulsante P viene premuto altrimenti l'uscita valga sempre 0



Teoremi dell'algebra di Boole (2/4)

Idempotenza

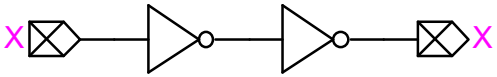
- (T3) $X + X = X$
- (T3') $X \cdot X = X$



si usa per l'amplificazione dei segnali ed eliminazione disturbi

Involuzione

- (T4) $(X ')' = X$

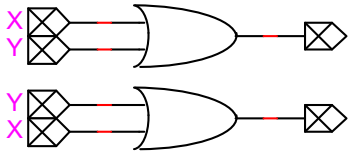


Complementarietà

- (T5) $X + X' = 1$
- (T5') $X \cdot X' = 0$

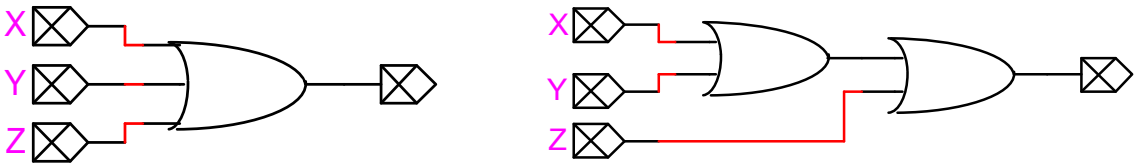
Proprietà commutativa

- (T6) $X + Y = Y + X$
- (T6') $X \cdot Y = Y \cdot X$



Proprietà associativa

- (T7) $(X + Y) + Z = X + (Y + Z) = X + Y + Z$
- (T7') $(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) = X \cdot Y \cdot Z$



Teoremi dell'algebra di Boole (3/4)

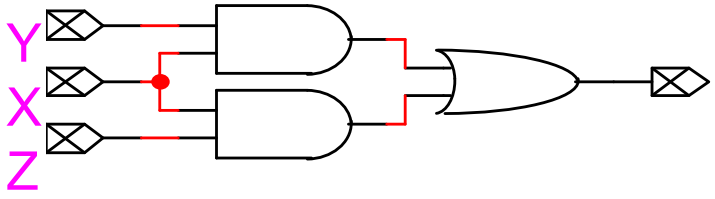
Proprietà di assorbimento

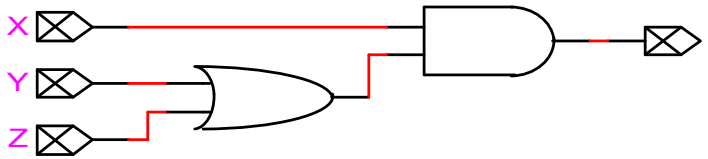
- (T8) $X + X \cdot Y = X$
- (T8') $X \cdot (X + Y) = X$

permette di minimizzare il n. di gate

Proprietà distributiva

- (T9) $X \cdot Y + X \cdot Z = X \cdot (Y + Z)$
- (T9') $(X + Y) \cdot (X + Z) = X + Y \cdot Z$





Teoremi dell'algebra di Boole (4/4)

Proprietà della combinazione

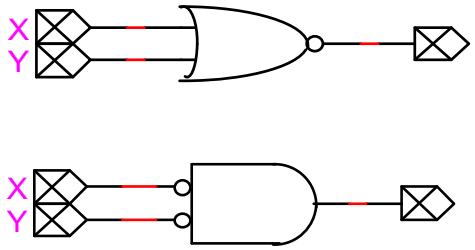
- (T10) $(X + Y) \cdot (X' + Y) = Y$
- (T10') $X \cdot Y + X' \cdot Y = Y$

Proprietà del consenso

- (T11) $(X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z)$
- (T11') $X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$

Teorema di De Morgan

- (T12) $(X + Y)' = (X' \bullet Y')$
- (T12') $(X \bullet Y)' = (X' + Y')$
- generalizzabile per *n* variabili



Dai teoremi dell'assorbimento o dalla proprietà distributiva:

$XY' + Y = XY' + XY + Y = X + Y$

$XY' + Y = (X + Y)(Y' + Y) = X + Y$

Parità (1/2)

- I codici **rilevatori d'errori** sono codici in cui è possibile rilevare se sono stati commessi errori nella trasmissione
- *Codici ridondanti:* in cui l'insieme dei simboli dell'alfabeto è minore dell'insieme di configurazioni rappresentabili col codice
- Codici con **bit di parità:** alla codifica binaria si aggiunge un bit di parità (codice ridondante in quanto usa 1 bit in più del necessario)
- **parità pari** rende pari il numero di 1 presenti nella parola (vale 1 se ci sono un n. dispari di 1)
- **parità dispari:** il contrario
- I codici di parità rilevano la presenza di un numero dispari di errori (e quindi di errori singoli)
- es. valore definito con 8 bit 11001011
  con 9 bit con parità (pari) 110010111

Parità (2)

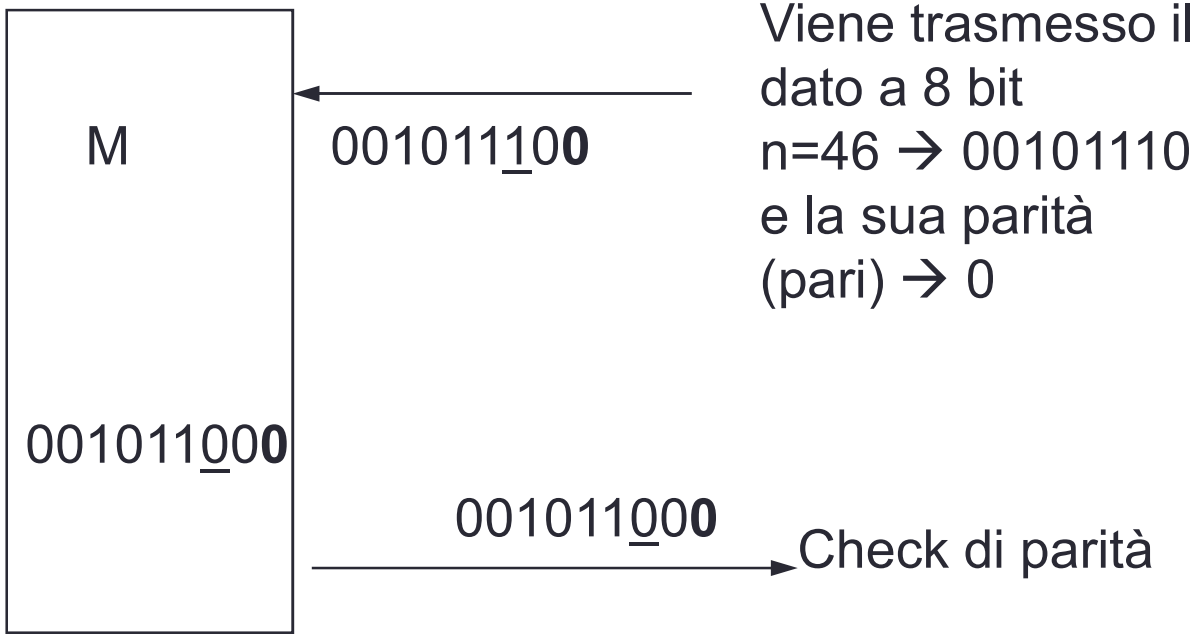
Simboli alfabeto	cod. Binaria	cod. Binaria con parità pari
0	000	000 0
1	001	001 1
2	010	010 1
3	011	011 0
4	100	100 1
5	101	101 0
6	110	110 0
7	111	111 1

Ad ogni simbolo dell'alfabeto corrisponde una configurazione a parità pari.

Le configurazioni a parità dispari non codificano alcun simbolo dell'alfabeto.

Se viene rilevata una configurazione a parità dispari significa che si è verificato un errore che ha alterato un numero dispari di bit (1, 3, 5, 7, ...).

Esempio



- Supponiamo un errore di trasmissione durante la scrittura in memoria così che il numero memorizzato sia 001011000 .
- Quando il dato viene riletto ed utilizzato viene fatto il check di parità e si verifica che quel numero non è ammissibile per la codifica binaria con parità pari perché la somma dei bit a 1 è dispari.
- Quindi viene rilevato un errore.

Esercizi (1/4)

Esercizio 1

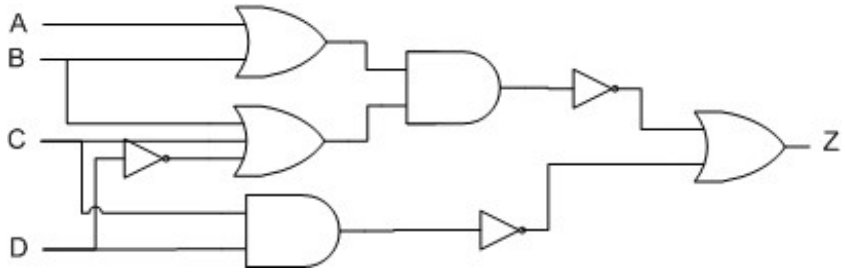
Date le seguenti funzioni logiche ricavare le corrispondenti reti logiche realizzate utilizzando solo gate elementari AND, OR e NOT

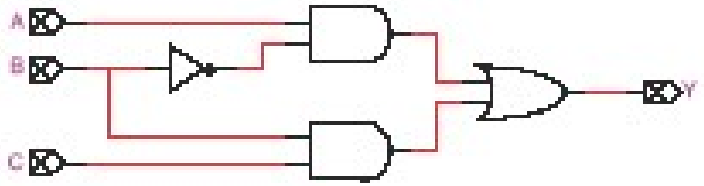
$$F = X(Y + Z)$$

$$F = \overline{X} + Y + X\overline{Z}$$

Esercizio 2

Date le seguenti reti logiche determinare le tabella di verità e le funzioni logiche corrispondenti

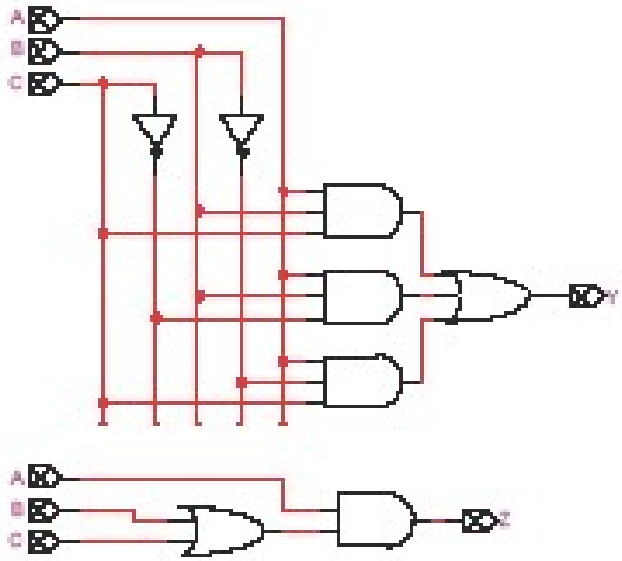




Esercizi (2/4)

Esercizio 3

Date le reti di figura ricavare le tabelle di verità, le funzioni logiche in forma algebrica e dimostrare, facendo uso dei teoremi dell'algebra di Boole, che risultano logicamente equivalenti.



Esercizio 4

Ricavare le tabelle di verità delle seguenti espressioni

Z = W'X+Y'Z'+X'Z+Y

Z = W+X'(Y'+Z)

Z = WX+Y(Z'+X)+Z(X'+Y')

Z = ABC+(A'+B'+C)C'

Esercizi (3/4)

Esercizio 5

Ricavare le tabelle di verità e semplificare le seguenti funzioni. Indicare anche il teorema utilizzato per ciascun passaggio della semplificazione:

Y = (A+B)(A+BC) + A'B' +A'C'
Y = ABC+ABC'+A'BD+ABD+A'D
F = (X+Y+W')(X+Y+W)(X+Y'+W)(X'+Y'+W)
Y = A'C(A'BD)'+A'BC'D'+AB'C
Y = (A'+B)(A+B+D)D'
Y = A'B'C'D+A'B'CD+A'BC'D+AB'C'D
W = X'Y+X'Y'Z

Esercizio 6:

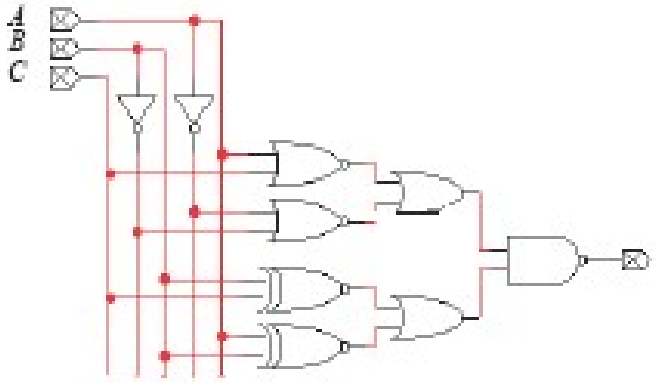
Una assicurazione è disposta a fornire una assicurazione nei seguenti casi: il contraente è maschio e ha meno di 30 anni oppure ha più di 30 anni ed ha figli; il contraente ha più di 30 anni, non ha figli e, o è maschio o è sposato; il contraente ha più di 30 anni, non ha figli e non è sposato.

Valutazione: una donna con figlio non sposata e con meno di 30 anni può essere assicurata?

Esercizi (4/4)

Esercizio 7

Ricavare la funzione logica in forma algebrica e semplificare applicando i teoremi dell'algebra booleana. Disegnare il diagramma della rete semplificata.



Esercizio 8

Ricavare la funzione logica in forma algebrica e semplificare applicando i teoremi dell'algebra booleana. Disegnare il diagramma della rete semplificata.