Capitolo 4

Reti logiche

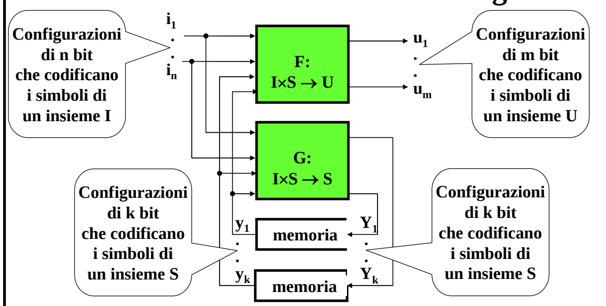
4.1 - Funzioni, espressioni e schemi logici
4.2 - Algebra di commutazione
4.3 - Famiglie logiche

4.1 Funzioni, espressioni e schemi logici

Logica e Reti logiche

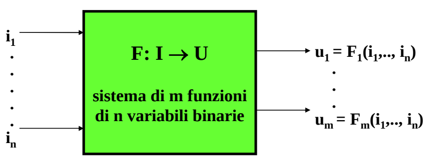
__Rete logica__ -Modello matematico che assume come *primitive* alcune semplici modalità di elaborazione di segnali binari e deduce da queste in modo rigoroso
- quale struttura soddisfa un dato comportamento,
- quale comportamento ha una data struttura.

Il modello strutturale delle reti logiche



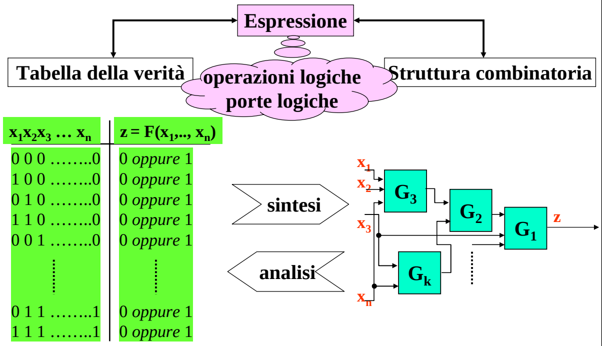
Rete logica sequenziale sincrona	Rete logica sequenziale asincrona	Rete logica combinatoria
retroazioni con FF D	*retroazioni dirette*	*nessuna retroazione*

Rete logica combinatoria



__Rete logica combinatoria__ - I valori dei segnali d'uscita dipendono solo dai valori contemporanei dei segnali d'ingresso.

Struttura & Comportamento di una rete logica combinatoria

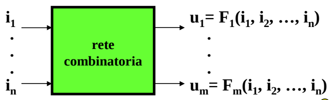


Descrizione matematica del comportamento delle reti combinatorie

- **Variabili binarie**: indipendenti e dipendenti
- **Funzioni booleane**: complete e incomplete
- **Operazioni logiche**: simboli e regole
- **Espressioni logiche**: funzioni e schemi

Funzioni booleane

Funzioni di variabili binarie



Funzione completa di n variabili binarie $z = F(x_1, x_2, \ldots, x_n)$
Insieme di 2^n coppie ordinate $\{x, z \mid x \in B^n, z \in B\}$ formate da una configurazione di valori delle variabili indipendenti x_i e dal corrispondente valore della variabile dipendente z.

Il numero di distinte funzioni di n variabili binarie è finito.

$\Phi(n) = 2^{2^n}$

4 funzioni di 1 variabile,
16 funzioni di 2 variabili,
256 funzioni di 3 variabili,
65.536 funzioni di 4 variabili, ecc.

Funzione incompleta o non completamente specificata
Il dominio è un sottoinsieme di B^n

Esempio: BCD →7 segmenti

Tabelle della verità

Tabella della verità - Descrizione tabellare di una funzione di variabili binarie.

n+1 colonne

2^n righe

$x_1\ x_2 \ldots x_n$	$F(x_1, x_2, \ldots, x_n)$
0 0 0 ……..0	0 *oppure* 1 *oppure* -
1 0 0 ……..0	0 *oppure* 1 *oppure* -
0 1 0 ……..0	0 *oppure* 1 *oppure* -
1 1 0 ……..0	0 *oppure* 1 *oppure* -
0 0 1 ……..0	0 *oppure* 1 *oppure* -
⋮	⋮
0 1 1 ……..1	0 *oppure* 1 *oppure* -
1 1 1 ……..1	0 *oppure* 1 *oppure* -

Funzioni incomplete

Funzioni di una e di due variabili

x	f_0	f_3	f_1	f_2
0	0	1	0	1
1	0	1	1	0

4 funzioni di una variabile

f_0, f_3 : costanti 0 e 1
f_1: identità o buffer
f_2: not

$x_0\ x_1$	f_0	f_{15}	f_3	f_5	f_{12}	f_{10}	f_1	f_{14}	f_7	f_8	f_9	f_6	f_{13}	f_2	f_{11}	f_4
0 0	0	1	0	0	1	1	0	1	0	1	1	0	1	0	1	0
0 1	0	1	0	1	1	0	0	1	1	0	0	1	1	0	0	1
1 0	0	1	1	0	0	1	0	1	1	0	0	1	0	1	1	0
1 1	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1	0

16 funzioni di due variabili

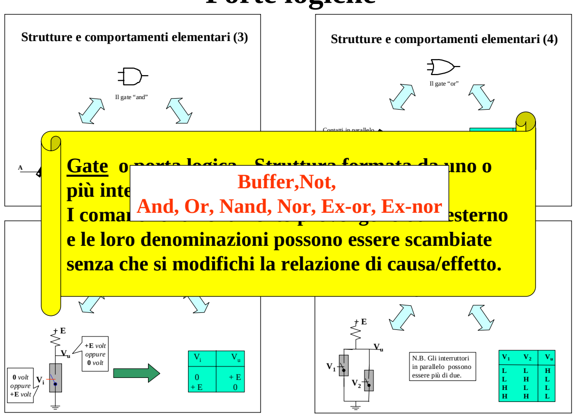
f_0, f_{15} : costanti 0 e 1
f_3 , f_5 : identità o buffer
f_{12} , f_{10} : not

f_1 : and
f_{14} : nand
f_7 : or
f_8 : nor
f_9: equivalence
f_6: ex-or

f_{13}: x_0=1 implica x_1=1
f_{11}: x_1=1 implica x_0=1
f_2: complemento di f_{13}
f_4 : complemento di f_{11}

funzioni complementari

Porte logiche

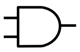


Gate o porta logica - Struttura formata da uno o più inte… I coman… esterno e le loro denominazioni possono essere scambiate senza che si modifichi la relazione di causa/effetto.

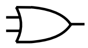
Buffer,Not,
And, Or, Nand, Nor, Ex-or, Ex-nor

Dualità tra “and” e “or”(1)

Logica positiva



I1	I2	AB
0	0	0
1	0	0
0	1	0
1	1	1

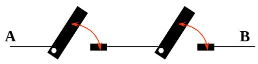


I1	I2	AB
0	0	0
1	0	1
0	1	1
1	1	1

Due differenti astrazioni!

{aperto = 0, chiuso = 1} {aperto = 1, chiuso = 0}

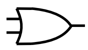
Contatti in serie



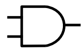
I1	I2	AB
aperto	aperto	aperto
aperto	chiuso	aperto
chiuso	aperto	aperto
chiuso	chiuso	chiuso

Dualità tra “and” e “or”(2)

Logica positiva



I1	I2	AB
0	0	0
1	0	1
0	1	1
1	1	1

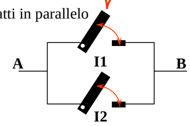


I1	I2	AB
0	0	0
1	0	0
0	1	0
1	1	1

Due differenti astrazioni!

{aperto = 0, chiuso = 1} {aperto = 1, chiuso = 0}

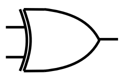
Contatti in parallelo



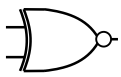
I1	I2	AB
aperto	aperto	aperto
aperto	chiuso	chiuso
chiuso	aperto	chiuso
chiuso	chiuso	chiuso

Dualità tra “ex-or” e “ex-nor”(3)

Logica positiva

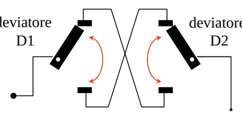


I1	I2	AB
0	0	0
1	0	1
0	1	1
1	1	0



I1	I2	AB
0	0	1
1	0	0
0	1	0
1	1	1

{alto = 0, basso = 1} {aperto = 0, chiuso = 1}
{alto = 1, basso = 0} {aperto = 1, chiuso = 0}



D1	D2	AB
alto	alto	aperto
basso	alto	chiuso
alto	basso	chiuso
basso	basso	aperto

Funzioni e operazioni

F = “F è descritta da ..” *** operatore**

Operazioni a un operando

F(x) = *(x)
F(x) = (x)*

Esempi:
radice
logaritmo
potenza
derivata
modulo

Operazioni a due operandi

F(x,y) = *(x,y)
F(x,y) = x* y

Esempi:
addizione
sottrazione
moltiplicazione
divisione
Min, Max

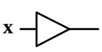
Identità : z = x

Regole:
0 = 0
1 = 1

Funzione:

x	z
0	0
1	1

Realizzazione:



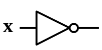
Complementazione : x’ , $\bar{x}$, ⌉x

Regole:
0’ = 1
1’ = 0

Funzione:

x	z
0	1
1	0

Realizzazione:



= : il complemento di 0 **vale** 1

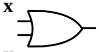
Somma logica: x + y , x ∨ y

Regole:
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 1

Funzione:

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

Realizzazione:



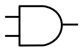
Prodotto logico: x . y , xy , x ∧ y

Regole:
0 . 0 = 0
0 . 1 = 0
1 . 0 = 0
1 . 1 = 1

Funzione:

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

Realizzazione:



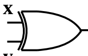
Somma modulo due: x ⊕ y

Regole:
0 ⊕ 0 = 0
0 ⊕ 1 = 1
1 ⊕ 0 = 1
1 ⊕ 1 = 0

Funzione:

x	y	z
0	0	0
0	1	1
1	0	1
1	1	0

Realizzazione:



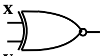
Equivalenza: x ≡ y

Regole:
0 ≡ 0 = 1
0 ≡ 1 = 0
1 ≡ 0 = 0
1 ≡ 1 = 1

Funzione:

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1

Realizzazione:



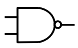
Nand (operazione di Shaffer): z = x ↑ y

Regole:
0 ↑ 0 = 1
0 ↑ 1 = 1
1 ↑ 0 = 1
1 ↑ 1 = 0

Funzione:

x	y	z
0	0	1
0	1	1
1	0	1
1	1	0

Realizzazione:



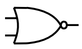
Nor (operazione di Pierce): z = x ↓ y

Regole:
0 ↓ 0 = 1
0 ↓ 1 = 0
1 ↓ 0 = 0
1 ↓ 1 = 0

Funzione:

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0

Realizzazione:



Operazioni e Espressioni

$f_1(x) = x$ $f_7(x,y) = x + y$ $f_1(x,y) = x . y$ $f_6(x,y) = x \oplus y$

$f_2(x) = x'$ $f_8(x,y) = x \downarrow y$ $f_{14}(x,y) = x \uparrow y$ $f_9(x,y) = x \equiv y$

Espressione logica - Stringa formata da costanti, bit, operatori logici e parentesi.

Esempi: (x ⊕ y) ⊕ (z ⊕ w) a + (b.c)

(x ↓ y) ↓ 0

Valutazione di una espressione

Valutazione di una espressione di n variabili per una n-pla di valori
1 - Si sostituisce ad ogni variabile il valore che le compete.
2 - Partendo dalle parentesi più interne si sostituisce ogni operazione con il suo risultato fino ad ottenere o la costante 0 o la costante 1.

Esempio: E(a,b,c) = a+(b.c) per a=0, b=1, c=0
= 0+(1.0)
= 0+0
= 0

N° di valutazioni - Una espressione di n variabili può essere valutata in 2^n modi diversi.

Espressioni e Funzioni

Le 2^n valutazioni di una espressione $E(x_1, x_2, \ldots, x_n)$ creano 2^n coppie x, z $\{x, z \mid x \in B^n, z \in B\}$

Esempio: E(a,b,c) = a+(b.c)

	a b c	E
E(0,0,0) = 0+(0.0) = 0	0 0 0	0
E(0,0,1) = 0+(0.1) = 0	0 0 1	0
E(0,1,0) = 0+(1.0) = 0	0 1 0	0
E(0,1,1) = 0+(1.1) = 1	0 1 1	1
E(1,0,0) = 1+(0.0) = 1	1 0 0	1
E(1,0,1) = 1+(0.1) = 1	1 0 1	1
E(1,1,0) = 1+(1.0) = 1	1 1 0	1
E(1,1,1) = 1+(1.1) = 1	1 1 1	1

T1) Ogni espressione descrive una e una sola funzione completa.

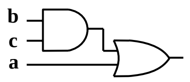
Espressioni e Schemi logici

T2) Ogni espressione descrive una struttura formata da gate connessi in serie e/o in parallelo.

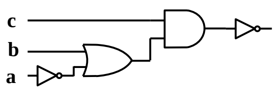
Per individuare lo schema descritto da una espressione:
1 - si parte dalle parentesi più interne e si traccia il simbolo del gate corrispondente all'operazione, collegandone gli ingressi ai segnali esterni;
2 - si procede in modo analogo con le altre coppie di parentesi, considerando via via come ingressi dei nuovi gate anche le uscite di quelli già tracciati.

Esempi

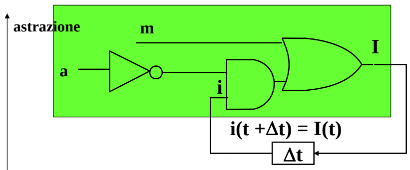
a+(b.c)



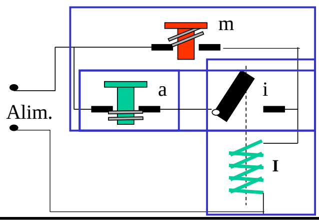
(((a)' + b) . c)'



N.B. - Lo schema logico di una espressione non può avere segnali in <u>retroazione</u> (l'uscita di ogni gate dipende da segnali d'ingresso e/o da uscite di gate disposti "a monte").

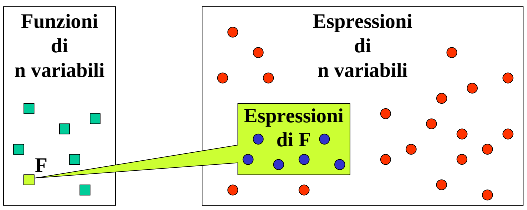


I = f(m,a,i) = m + a'.i



Equivalenza tra espressioni

<u>Espressioni equivalenti</u> - Due espressioni E_1, E_2 sono equivalenti, e si scrive $E_1 = E_2$, se e solo se descrivono la stessa funzione.



Metodi per dimostrare l'equivalenza: **induzione perfetta**
manipolazione algebrica

Proprietà

T3) proprietà commutativa (+, ., ↓, ↑, ⊕, ≡)

a * b = b * a

T4) proprietà associativa (+, ., ⊕)

(a * b) * c = a * (b * c) **= a * b * c**

T5) complementi:

(x + y)' = x ↓ y

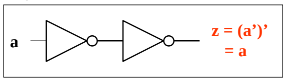
(x . y)' = x ↑ y

(x ≡ y)' = x ⊕ y

N.B. il pallino!

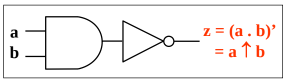
Insiemi di gate (1)

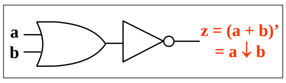
<u>Insieme AND, OR, NOT</u> - Disponendo opportunamente in serie/parallelo soltanto questi tre tipi di gate è possibile ottenere il comportamento di tutti gli altri.

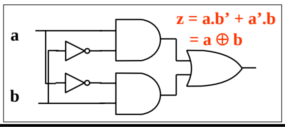


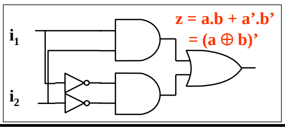
infatti:

a	a'	(a')'
0	1	0
1	0	1



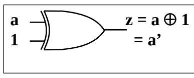






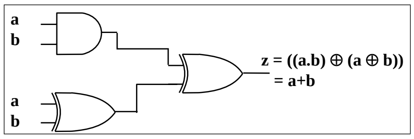
Insiemi di gate (2)

<u>Insieme EX-OR, AND</u> - Disponendo opportunamente in serie/parallelo soltanto questi due tipi di gate è possibile ottenere il comportamento di tutti gli altri.



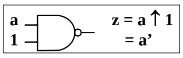
infatti

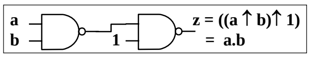
a	a ⊕ 1
0	1
1	0

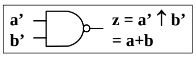


Insiemi di gate (3)

<u>NAND</u> - Disponendo opportunamente in serie/parallelo solo questo tipo di gate è possibile ottenere il comportamento di tutti gli altri.







Dimostrazione per induzione perfetta →

a b	a+b	a'.b'	a'↑b'
0 0	0	1	0
0 1	1	0	1
1 0	1	0	1
1 1	1	0	1

Calcolo delle proposizioni

Assegnata una qualsiasi funzione di variabili binarie, è possibile descriverla con una espressione contenente solo le operazioni eseguite dai gate?

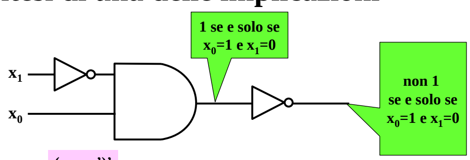
<u>Proposizione</u> - Frase o "vera" o "falsa", formata da affermazioni o "vere" o "false" unite dai connettivi o, e, non.

Sia b la proposizione "il bit b vale 1".

La frase "F(x,y) vale 1 se o x vale 1 o y vale 1"
- **descrive la funzione "or"**
- **è equivalente alla proposizione "o x o y"**
  (*vera* per 01,10,11 e *falsa* per 00)
- **è equivalente all'espressione x + y**

"vero"→ 1
"falso"→ 0
"e"→ .
"o"→ +
"non"→ '

Sintesi di una delle implicazioni



x_0 x_1	f_{13}
0 0	1
0 1	1
1 0	0
1 1	1

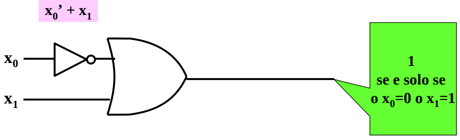
$(x_0 . x_1')'$

non (x_0 e non x_1)

non x_0 o x_1

$x_0' + x_1$

$f_{13} = x_0 \Rightarrow x_1$

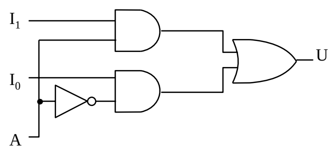


Sintesi di un SELETTORE a due vie

A I_0 I_1	U
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

o non A e I_0 oppure A e I_1

$A' . I_0 + A . I_1$



Algebre binarie

<u>Algebra binaria</u> - Sistema matematico formato da un insieme di operatori definiti assiomaticamente ed atti a descrivere con una espressione ogni funzione di variabili binarie

Calcolo delle proposizioni
{***vero***, ***falso***} {***e***, ***o***, ***non***}
tre operatori

Crisippo (250 a.c.)
G. Boole (1854)

Algebra di commutazione
{**0, 1**} {**+, . , '**}
tre operatori

C. Shannon (1938)

Algebra del nand
{**0, 1**} {$\uparrow$}
un operatore

Algebra del nor
{**0, 1**} {$\downarrow$}
un operatore

Algebra lineare
{**0, 1**} { $\oplus$ **, .**}
due operatori