I circuiti digitali: dalle funzioni logiche ai circuiti (le SOP)

Prof. Alberto Borghese
Dipartimento di Informatica
alberto.borghese@unimi.it

Università degli Studi di Milano

Riferimento al testo: Patterson Hennessy, Sezione B.3 on-line;
Approfondimento sulle forme canoniche: Fummi et al., Progettazione Digitale, McGrawHill, capitolo 3.

Sommario

I circuiti combinatori.

Semplificazione algebrica.

Dalla tabella della verità al circuito: la prima forma canonica: SOP.

Criteri di ottimalità.

Circuiti combinatori

- Circuiti logici digitali in cui le operazioni (logiche) dipendono solo da una combinazione degli input. Come nelle funzioni a valori reali, il risultato dell'elaborazione del circuito viene aggiornato immediatamente dopo il cambiamento dell'input (si suppone il tempo di commutazione trascurabile; tempo di attesa prima di guardare l'output sufficientemente ampio per permettere a tutti i circuiti la commutazione).

- Circuiti senza memoria. Ogni volta che si inseriscono in ingresso gli stessi valori, si ottengono le stesse uscite. Il risultato non dipende dallo stato del circuito.

- I circuiti combinatori descrivono delle funzioni Booleane. Queste funzioni si ottengono combinando tra loro (in parallelo o in cascata) gli operatori logici: **NOT, AND, OR**.

- Il loro funzionamento può essere descritto come **tabella della verità**.

- Dato un circuito è univoca la funzione algebrica che ne rappresenta il funzionamento e viceversa.

Un po' di tassonomia

- **Espressione logica.** Combinazione di operatori logici. Ad ogni espressione logica è associato un ben preciso circuito.

$$AB + \bar{B}C$$

- **Funzione logica.** Corrispondenza tra un insieme di ingresso (valori possibili di A, B, C) e un insieme di uscita (valori possibili di Y).

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

Una funzione logica viene rappresentata come espressione logica.

Regole manipolazione algebrica

	AND	OR
Doppia Inversione	$\overline{\overline{x}} = x$	
Identità	$1\,x = x$	$0 + x = x$
Elemento nullo	$0\,x = 0$	$1 + x = 1$
Idempotenza	$x\,x = x$	$x + x = x$
Inverso	$x\,\overline{x} = 0$	$x + \overline{x} = 1$
Commutativa	$x\,y = y\,x$	$x + y = y + x$
Associativa	$(x\,y)\,z = x\,(y\,z)$	$(x + y) + z = x + (y + z)$
	AND rispetto ad OR	**OR rispetto ad AND**
Distributiva	$x\,(y + z) = x\,y + x\,z$	$x + y\,z = (x + y)\,(x + z)$
Assorbimento	$x\,(x + y) = x$	$x + x\,y = x$
De Morgan	$\overline{xy} = \overline{x} + \overline{y}$	$\overline{x + y} = \overline{x}\,\overline{y}$

Si possono dimostrare sostituendo 0/1 alle variabili.

Regole algebriche su più variabili

Commutativa $x\,y\,z=y\,x\,z=z\,x\,y$ $\quad x+y+z=y+x+z=z+x+y$

	AND rispetto ad OR	OR rispetto ad AND
Distributiva	$x\,(yh+z) = xyh+xz$	$xh + yz = (xh+y)\,(xh+z)$
De Morgan	$\overline{xyz} = \overline{x} + \overline{y} + \overline{z}$	$\overline{x+y+z} = \overline{x}\,\overline{y}\,\overline{z}$

Si possono dimostrare sostituendo 0/1 alle variabili.

Funzione come espressione logica o come tabella delle verità

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

A B C	A and B	B and not(C)	Y
0 0 0	0	0	0
0 0 1	0	0	0
0 1 0	0	1	1
0 1 1	0	0	0
1 0 0	0	0	0
1 0 1	0	0	0
1 1 0	1	1	1
1 1 1	1	0	1

Una seconda rappresentazione - I

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

Applico la doppia negazione ai singoli addendi

Applico De Morgan ai prodotti logici: $\overline{xy} = \overline{x}+\overline{y}$

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

NB !B e !B non si sommano mediante somma logica perché prima vanno calcolate le negazioni!!

Potrei effettuare la somma se fosse:

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

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

A B C	Y
0 0 0	0
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	1

Una seconda rappresentazione - II

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

Voglio sostituire la somma con un prodotto logico.
Applico De Morgan:

$\overline{x} + \overline{y} = \overline{xy}$

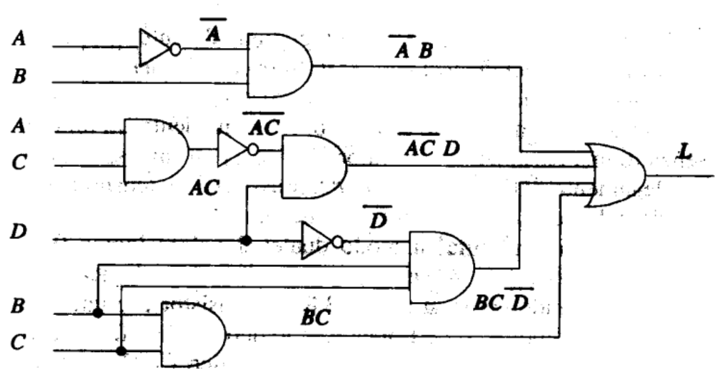
E' la funzione AND costruita con un'espressione logica diversa!

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

A B C	Y
0 0 0	0
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	1

Esempio – rappresentazione 1

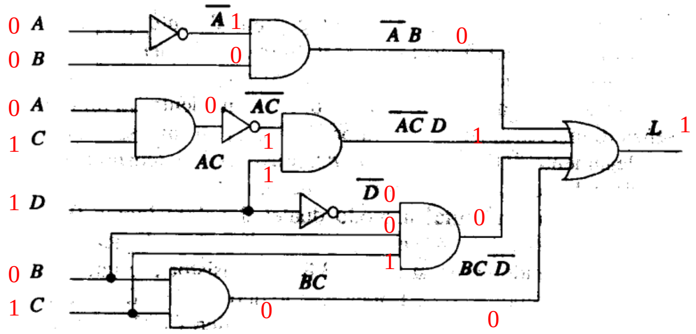
$L = \overline{A}\,B + \overline{AC}\,D + BC + BC\,\overline{D}$



Esempio – tabella verità

A	B	C	D	L
0	0	0	0	0
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	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

$L = \bar{A}B + \overline{AC}D + BC + BC\bar{D}$



Manipolazione algebrica

Applichiamo De Morgan.

$$L = \bar{A}B + \overline{AC}D + BC\bar{D} + BC =$$

$$= \overline{A+\bar{B}} + \overline{AC+\bar{D}} + \overline{\overline{BC}+D} + \overline{\bar{B}+\bar{C}} =$$

$$\bar{x}\,\bar{y} = \overline{x+y}$$

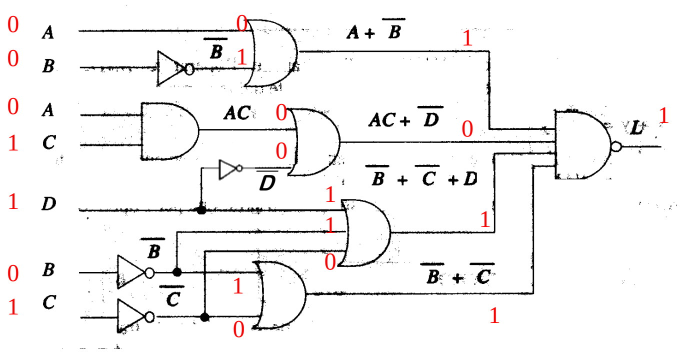
$$= \overline{\underbrace{(A+\bar{B})}_{x}\underbrace{(AC+\bar{D})}_{y}\underbrace{(\bar{B}+\bar{C})}_{z}\underbrace{(\overline{BC}+D)}_{h}}$$

$$\bar{x}+\bar{y}+\bar{z}+\bar{h} = \overline{xyzh}$$

Esempio – rappresentazione 2

A B C D	L
0 0 0 0	0
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	0
1 0 0 1	1
1 0 1 0	0
1 0 1 1	0
1 1 0 0	0
1 1 0 1	1
1 1 1 0	1
1 1 1 1	1

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



Sommario

I circuiti combinatori.

Semplificazione algebrica.

Dalla tabella della verità al circuito: la prima forma canonica: SOP.

Criteri di ottimalità.

Semplificazioni notevoli

Dimostrare che: $A + \bar{A}B = A + B$

Proprietà distributiva di OR rispetto ad AND:

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

Sviluppando il prodotto:

$(A + B)(A + \bar{A}) = AA + A\bar{A} + BA + B\bar{A} = A + AB + \bar{A}B$

Raccogliendo B:

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

NB: posso anche identificare i 3 «1» della funzione OR:

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

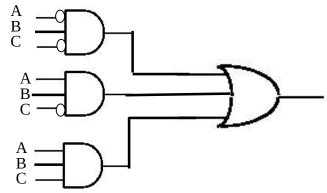
Semplificazioni notevoli

Dimostrare che: $(A + \bar{B})(B + C) = AB + AC + \bar{B}C$

Dimostrare che: $\bar{A} + AB = \bar{A} + B$

Esempio di semplificazione algebrica (esercizio)

$Y = \bar{A}\bar{B}C + AB\bar{C} + ABC =$



Raccogliendo $B\bar{C}$:

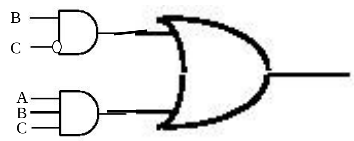
$(\bar{A} + A)B\bar{C} + ABC =$

Proprietà dell'inverso: " $\bar{A} + A = 1$ *"*

$= 1B\bar{C} + ABC =$

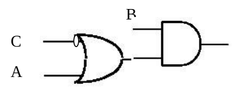
Proprietà dell'identità: "1B = B"

$= B\bar{C} + ABC =$



Dalla slide precedente:

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



Esempi di manipolazione algebrica

Y = !xyv + yz + !y!zv + !xy!v + x!yv =

Y = A !B !C + A B C + A B !C + A !B C = A

Somma di prodotti di 3 variabili: A, B, C (inverso dell'esercizio precedente):

Esercizi

• Calcolare le TT per le seguenti funzioni
Y = DA + AC + !B
Y = A + B + C + D
Y = !D!ABC + !DABC + !D!AB!C + !DAB!C

• Trasformare in funzioni equivalenti le seguenti funzioni, semplificandole:
Y = !(ABCD)
Y = !(DA) + !(B + !C)

Sommario

I circuiti combinatori.

Dall'espressione logica al circuito. Semplificazione algebrica.

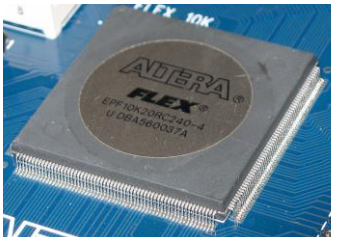
Dalla tabella della verità al circuito: la prima forma canonica: SOP.

Criteri di ottimalità.

FPGA

Field Programmable Gate Array (Matrice di porte logiche programmabili sul campo)

2020 mercato di 8.5 miliardi di dollari.



La struttura di un FPGA è in generale costituita da una matrice di blocchi logici configurabili, detti CLB (*Configurable Logic Blocks*), connessi fra loro attraverso interconnessioni programmabili. Ai margini di tale matrice vi sono i blocchi di ingresso/uscita, detti IOB (*Input Output Block*).

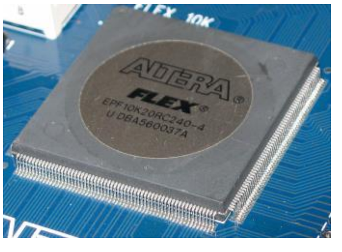
Da circuiti logici relativamente semplici fino a microprocessori interi:
https://www.arm.com/resources/designstart/designstart-fpga
https://venturebeat.com/2018/10/01/xilinx-will-use-arm-cores-in-fpga-chips/

Dall'espressione logica / tabella della verità al circuito

Definizione della funzione logica
Semplificazione e fitting sulla FPGA

Programmazione in linguaggi specifici (Verilog e VHDL)

Programmazione mediante POD che si collegano ai piedini di programmazione.



Insiemi di porte logiche le cui connessioni vengono definite (bruciate) attraverso un pod di programmazione collegato con USB a un PC. Questo consente di personalizzare il circuito logico e di implementare il circuito specificato via SW dal PC.

Da circuiti logici relativamente semplici fino a microprocessori interi:
https://www.arm.com/resources/designstart/designstart-fpga
https://venturebeat.com/2018/10/01/xilinx-will-use-arm-cores-in-fpga-chips/

Funzione come espressione logica o come tabella delle verità

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

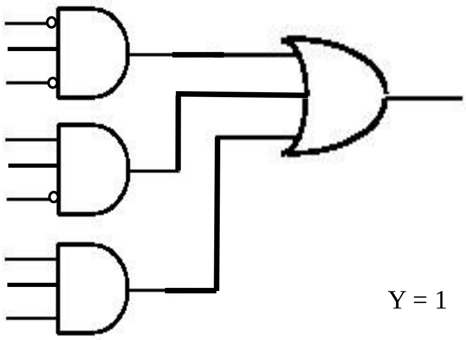
A B C	Y
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	0
1 0 0	0
1 0 1	0
1 1 0	1
1 1 1	1

Y = 1

iff

A = 0 B = 1 C = 0
OR
A = 1 B = 1 C = 0
OR
A = 1 B = 1 C = 1

Circuito associato



A B C	Y
0 0 0	0
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	1

Y = 1

iff

A = 0 B = 1 C = 0
OR
A = 1 B = 1 C = 0
OR
A = 1 B = 1 C = 1

La prima forma canonica

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

Implicante: prodotto delle variabili (in forma asserita o negata) per le quali la funzione vale 1

	A B C	Y
0	0 0 0	0
1	0 0 1	0
2	0 1 0	1
3	0 1 1	0
4	1 0 0	0
5	1 0 1	0
6	1 1 0	1
7	1 1 1	1

Mintermine, m_j,: un implicante che contiene tutte le N variabili della funzione (e.g. ABC) associato agli 1 della funzione.

j indica il numero progressivo in base 10.

Prima forma canonica: $F = \sum_{i=1}^{Q} m_i$

$0 \leq Q < 2^N$

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

Mintermini e Maxtermini

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

A B C	Y
0 0 0	0
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	1

Mintermine, m_j,: un implicante che contiene tutte le N variabili della funzione (e.g. ABC) associato agli 1 della funzione.

j indica il numero progressivo in base 10.

$F = \bar{A}\,B\,\bar{C} + A\,B\,\bar{C} + A\,B\,C$

Maxtermine, M_k: contiene tutte le N variabili della funzione ed è tale che il loro prodotto logico è uno 0 della funzione.

Funzione come espressione logica nella seconda forma canonica

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

A B C	Y
0 0 0	0
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	1

Y = 1

iff

NOT(A = 0 B = 0 C = 0)
AND
NOT(A = 0 B = 0 C = 1)
AND
NOT(A = 0 B = 1 C = 1)
AND
NOT(A = 1 B = 0 C = 0)
AND
NOT(A = 1 B = 0 C = 1)

Dall'espressione algebrica alla SOP (Sum Of Product)

- Passare attraverso la tabella della verità:

A B C	Y
0 0 0	0
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	1

- Manipolazione algebrica:

$Y = \bar{A}\,B\,\bar{C} + AB =$

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

$\bar{A}B\bar{C} + AB\bar{C} + ABC = m_2 + m_6 + m_7$

La SOP è la prima forma canonica

- La forma canonica di una funzione è la somma dei suoi mintermini.
- Qualunque funzione è esprimibile in forma canonica.

Esempio: $Y = f(A,B,C,D) = AC + BC + \overline{A}\,\overline{B}\,\overline{C}$

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

$= \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}C\overline{D} + A\overline{B}CD + ABC\overline{D} + ABCD$

La stessa espressione si ricaverebbe dalla tabella della verità:

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

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

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

Perchè SOP è una forma canonica

•Forma universale mediante la quale è possibile rappresentare qualunque funzione booleana.

• In generale una forma canonica non è una forma ottima, ma un punto di partenza per l'ottimizzazione.

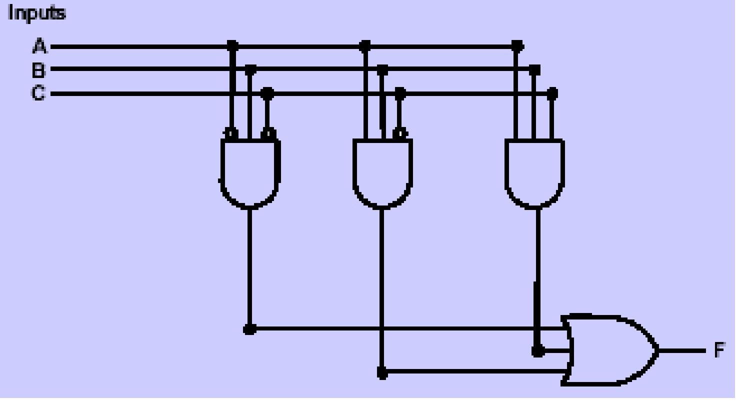
• Si basa su componenti caratterizzanti la struttura della funzione (mintermine), che traducono le condizioni logiche espresse dalla funzione.

Minternine, m_i:

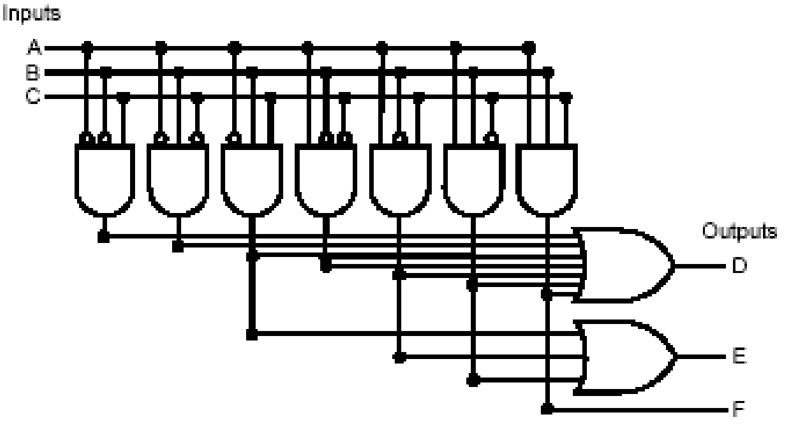
- E' una funzione booleana a n ingressi che vale 1 in corrispondenza della sola i-esima configurazione di ingresso.
- Al massimo, 2^n mintermini per ogni n variabili.
- ogni mintermine è rappresentabile con un AND con n ingressi.

Il circuito della prima forma canonica: SOP

$$F = \bar{A}B\bar{C} + AB\bar{C} + ABC$$



SOP a più uscite



Riutilizzo alcune "parti", in questo caso alcuni mintermini:
quelli che sono contenuti in D, E, F.

Ricavare la funzione in forma di tabella della verita'

Dalla SOP al circuito: osservazioni

- Dalla forma canonica (somma di mintermini) è facile passare al circuito:
  Ogni mintermine è un AND
  Tutti gli AND entrano in un OR
- Implementazione regolare
- Solo due livelli di porte
- Blocchi generali personalizzabili purché ci sia un numero sufficiente di componenti elementari.

Sommario

I circuiti combinatori.

Dall'espressione logica al circuito. Semplificazione algebrica.

Dalla tabella della verità al circuito: la prima forma canonica: SOP.

Criteri di ottimalità.

Dall’espressione logica al circuito

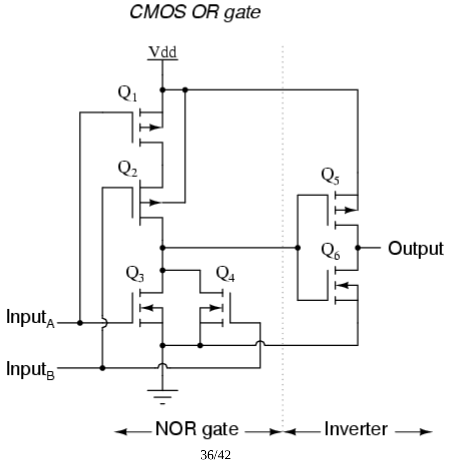
Ad ogni espressione logica corrisponde un circuito, ad ogni circuito corrisponde una tabella delle verità, ad ogni tabella della verità, in generale, **non corrisponde** un unico circuito possibile.

- Esistono più espressioni tra loro equivalenti: 2 espressioni sono equivalenti se hanno la stessa tabella di verità.
- Quale è la “migliore”?
- È possibile trovare un metodo di semplificazione sfruttando le proprietà dell’algebra booleana.
- Esistono tecniche automatiche o semi-automatiche di semplificazione.

Valutazione della semplicità di un circuito

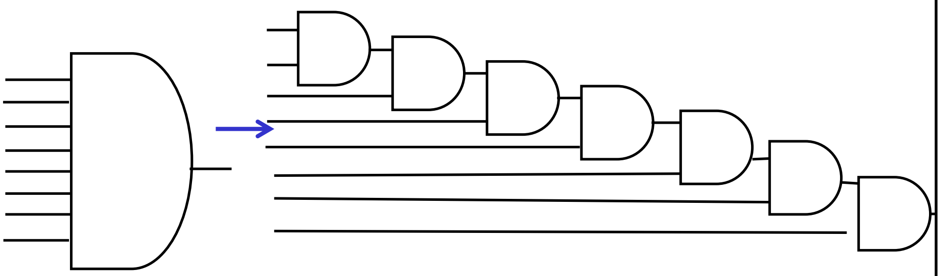
Area (numero di porte) = “ampiezza”
Tempo di commutazione (numero di transistor attraversati = “profondità”)

Soddisfazione di vincoli, potenza dissipata, facilità di debug...



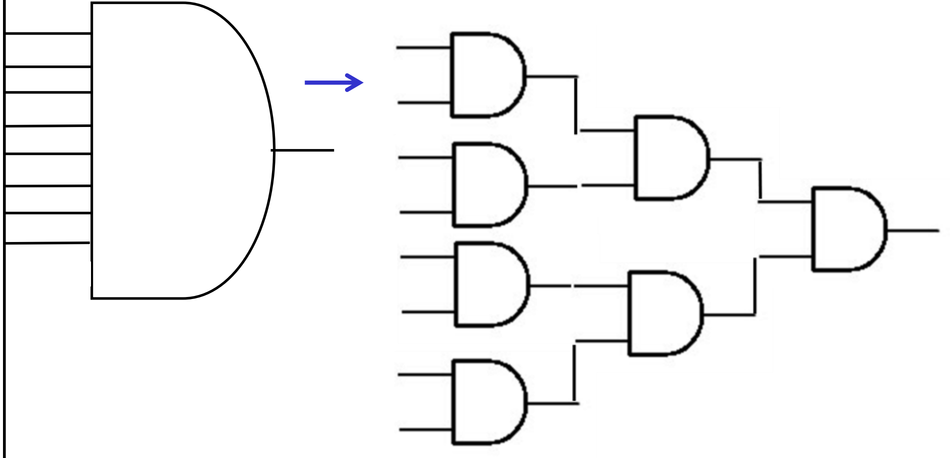
Esempio di trasformazione in un'implementazione con porte a 2 ingressi di un AND a 5 ingressi

- Gli elementi costruttivi di base tipici sono porte a 2 ingressi
  - Porta a N ingressi → N-1 porte a 2 ingressi



Numero di porte: N-1= 7
Cammino Critico: N-1 = 7 ⇨ Non è efficiente

Parallelizzazione

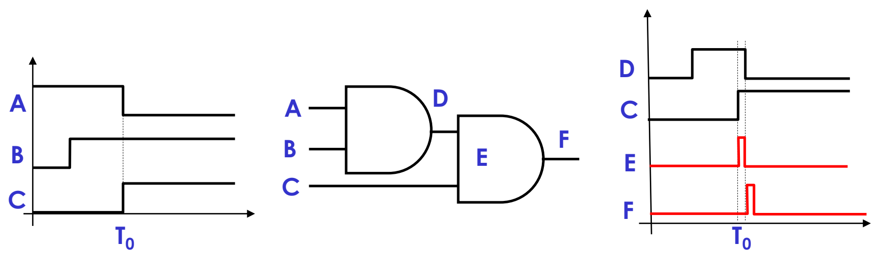


Numero di porte: 7 (stesso numero di porte) Parallelizzazione! «Divide et impera»
Cammino Critico: 3

Transitori

- Ogni circuito logico è caratterizzato da un **tempo di commutazione**
  - Più porte devo attraversare, più è lungo il tempo della **transizione del circuito nel suo complesso.**

- **CAMMINO CRITICO**
  - max numero di porte da attraversare da ingresso a uscita
  - Non si contano gli inverters (inclusi nelle porte)



A e C commutano contemporaneamente in To, E raggiunge il valore corretto dopo un tempo 2 ΔT (la commutazione di D segue la commutazione di B con un ritardo ΔT.

SOP dell'OR

Sintetizziamo la funzione OR come SOP

$Y = \bar{A}B + A\bar{B} + AB$ Complessità = 5 – Cammino critico = 3

Semplifico:

$Y = \bar{A}B + A\bar{B} + AB = B(\bar{A} + A) + A\bar{B} = B + A\bar{B} = B + A = A + B = OR(A,B)$

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

Ottimalità all'opera

SOP:

$Y = \overline{A}B\overline{C} + AB\overline{C} + ABC = m_2 + m_6 + m_7$

A	B	C	Y
0	0	0	0
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	1

POS:

$Y = \overline{(\overline{A}\overline{B}\overline{C})}\ \overline{(\overline{A}\overline{B}C)}\ \overline{(\overline{A}BC)}\ \overline{(A\overline{B}\overline{C})}\ \overline{(A\overline{B}C)}$

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

Quale implementazione è migliore? Perché?

Sommario

I circuiti combinatori.

Dall'espressione logica al circuito. Semplificazione algebrica.

Dalla tabella della verità al circuito: la prima forma canonica: SOP.

Criteri di ottimalità.