Lezione 4

I circuiti digitali: dalle funzioni logiche ai circuiti

Proff. A. Borghese, F. Pedersini

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Sommario

- Funzioni logiche
- Semplificazione algebrica
- La prima forma canonica (SoP)
  - Implementazione circuitale mediante PLA o ROM.
- La seconda forma canonica (PoS)

Proprietà degli operatori logici

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

Proprietà di assorbimento

I: A · (A+B) = A
A + AB = A

Dim:
A(A+B) = AA + AB = A + AB = A(1+B) = A·1 = A c.v.d.

II: A + ~A·B = A + B
A · (~A+B) = A·B

Dim:
Proprietà distributiva di OR / AND:
A + ~A·B = (A + ~A)·(A + B)
Sviluppando il prodotto:
(A + B)·(A + ~A) = AA + A(~A) + BA + B(~A) =
= A + AB + (~A)B =
Raccogliendo B: = A + (A + ~A)B = A + B c.v.d.

Funzione logica / circuito logico

- **Funzione logica:** $f : \mathbf{B}^n \rightarrow \mathbf{B}$
  - Funzione booleana di **n** variabili booleane
  - Può essere rappresentata come un'opportuna combinazione di operatori elementari (NOT, AND, OR)
  - Definita per tutte le **2^n** combinazioni delle variabili (ingressi)

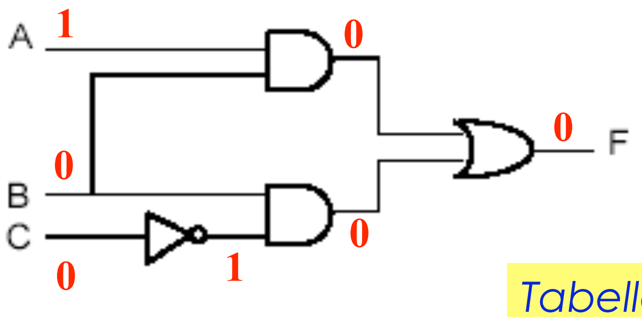
- **Espressione:** $Y = f(x_1, x_2, \ldots , x_n)$
  - Funzione booleana di **n** variabili booleane
  - Può essere rappresentata come un'opportuna combinazione di operatori elementari (NOT, AND, OR)
  - Definita per tutte le **2^n** combinazioni delle variabili (ingressi)

- **Tabella di verità** (Truth Table, TT)
  - Definizione della funzione per **elenco** di tutti i valori possibili delle variabili.

- **Circuito logico**
  - Uscita (booleana) = funzione logica di **n** ingressi (variabili) booleane
  - Può essere realizzata come un'opportuna combinazione di porte logiche elementari (not, and, or, nand, nor)

Funzione / circuito / tab. verità

F = (A and B) or (B and not(C))



3 ingressi: *F* = *f* (A,B,C)

→ 2^3 = 8 combinazioni

Tabella di verità

A	B	C	AandB	B and not(C)	F
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

Esempio di semplificazione algebrica

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

- raccolgo: $B\cdot \overline{C}$

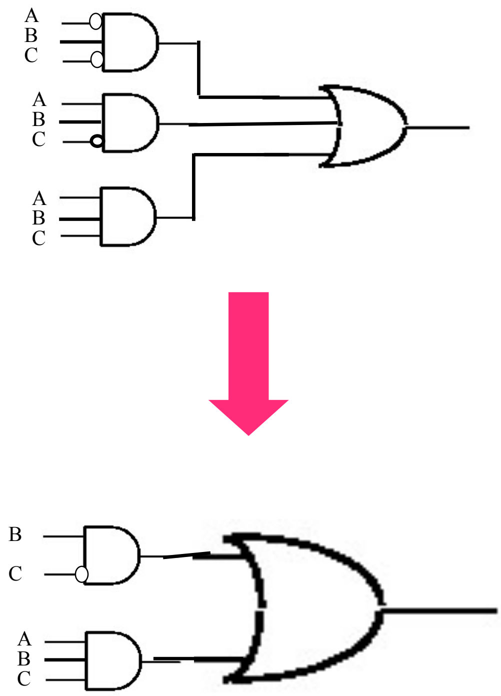
$$=\left(\overline{A} + A\right)\cdot B\cdot \overline{C} + A\cdot B\cdot C =$$

- inverso: $\overline{A} + A = 1$

$$= 1\cdot B\cdot \overline{C} + A\cdot B\cdot C =$$

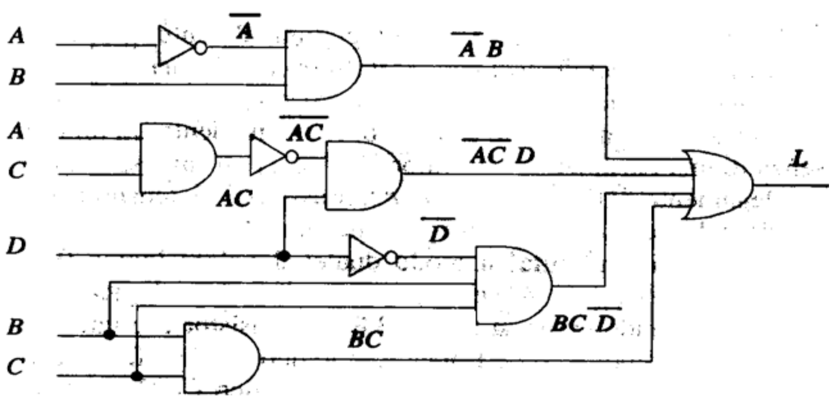
- identità: $(1\cdot B = B)$

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



Esempio – rappresentazione 1

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



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

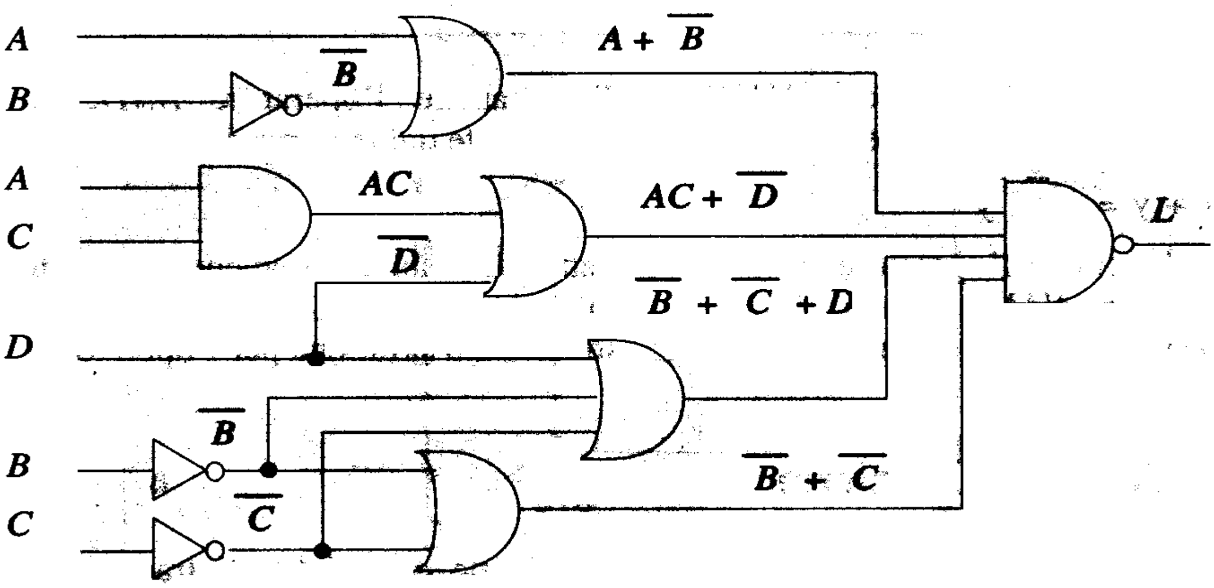
Manipolazione algebrica

Applichiamo DeMorgan :
$$\begin{cases} \overline{AB} = \overline{A} + \overline{B} \\ \overline{A+B} = \overline{A}\cdot\overline{B} \end{cases}$$

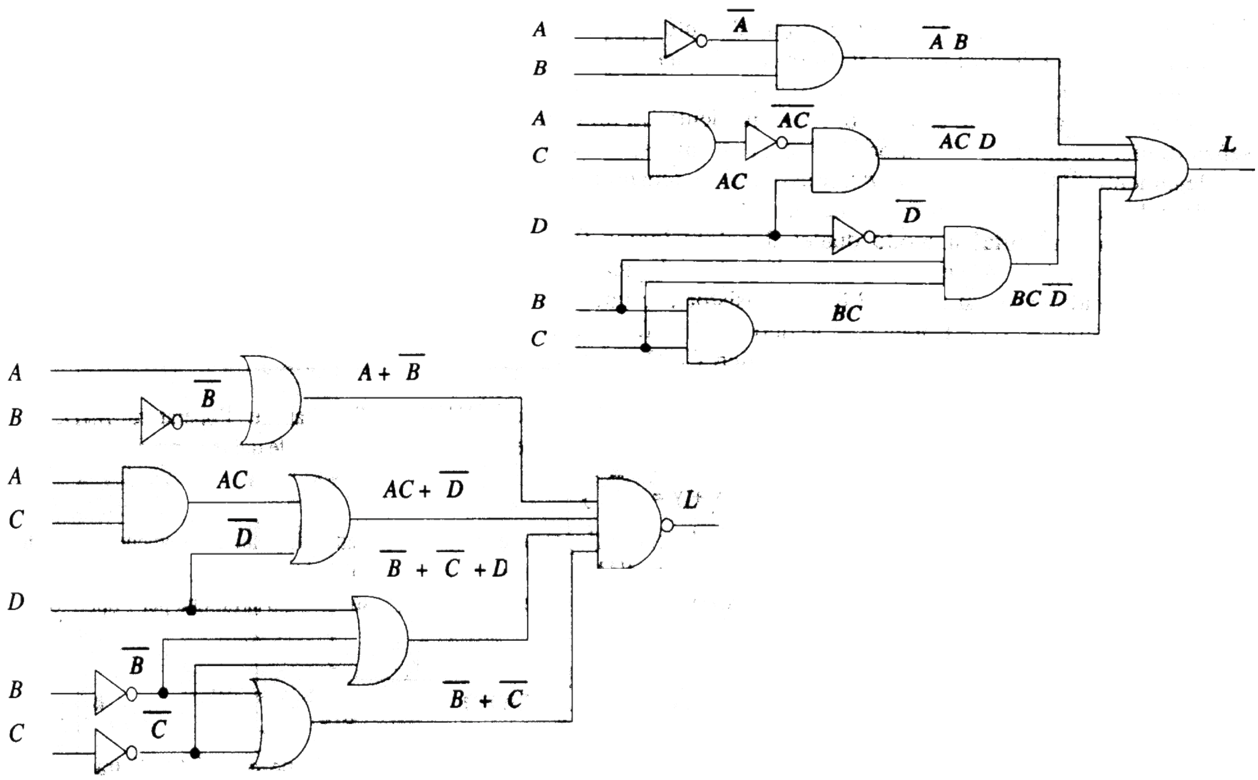
$$\begin{aligned} L &= \overline{A}B + \overline{AC}D + BC + BC\overline{D} = \\ &= \overline{A+\overline{B}} + \overline{AC+\overline{D}} + \overline{\overline{B}+\overline{C}} + \overline{\overline{BC}+D} = \\ &= \overline{(A+\overline{B})(AC+\overline{D})(\overline{B}+\overline{C})(\overline{BC}+D)} = \\ &= \overline{(A+\overline{B})(AC+\overline{D})(\overline{B}+\overline{C})(\overline{B}+\overline{C}+D)} \end{aligned}$$

Esempio – rappresentazione 2

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



Esempio: le 2 rappresentazioni equivalenti



Semplificazione di funzioni

- Espressioni equivalenti
  - 2 espressioni si dicono equivalenti se hanno la stessa tabella di verità
- Quale è la “migliore”?
  - La più semplice
  - La più “veloce”
- Metodi di semplificazione
  - Sfruttando le proprietà dell’algebra Booleana
    - Sulle espressioni logiche
  - Mappe di Karnaugh
    - Sulle tabelle di verità

Funzione: espressione / tabella di verità

espressione logica ↔ tabella delle verità

F = (**A** AND **B**) OR (**B** AND NOT(**C**))

A	B	C	F
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

F = 1 ***se e solo se:***

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

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

La prima forma canonica

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

A	B	C	F
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

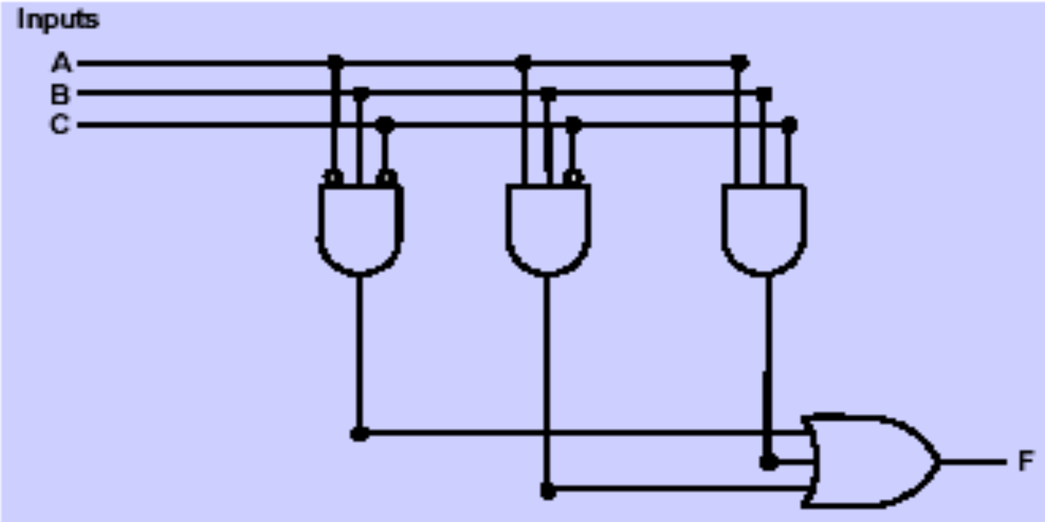
Implicante:
Prodotto delle variabili (in forma naturale o negata) per le quali la funzione vale 1

Mintermine m_j:
implicante che contiene tutte le ***n*** variabili della funzione (e.g. ABC).

$$\boxed{\text{Prima forma canonica (SoP)}: \quad F = \sum_{j=1}^{Q} m_j \quad , \quad Q \le 2^n}$$

Circuito della I forma canonica: SOP

F = ~AB~C + AB~C + ABC



Prima forma canonica: Sum-of-Products (SoP)

- Forma universale mediante la quale è possibile rappresentare qualunque funzione booleana.
  - Non è una forma ottima, ma un punto di partenza per l'ottimizzazione
- Si basa su componenti caratterizzanti la struttura della funzione (**mintermini**), che esprimono le condizioni logiche di verità (**1**) della funzione
- **Mintermine**, m_i:
  - Funzione booleana a **n** ingressi che **vale 1** in corrispondenza della **sola i-esima configurazione di ingresso**
  - Per ***n* variabili,** al più ***2^n^* mintermini**
  - Implementabile mediante un **AND ad *n* ingressi**

La prima forma canonica: SoP

Prima forma canonica di una funzione (SoP):
la somma dei suoi mintermini

❖ Qualunque funzione è esprimibile in forma canonica:

Z(A,B,C,D) = AC + BC + ~A~B~C =

= A(B+ ~B)C(D+ ~D) + (A+~A)BC(D+~D) + ~A~B~C(D+~D) =

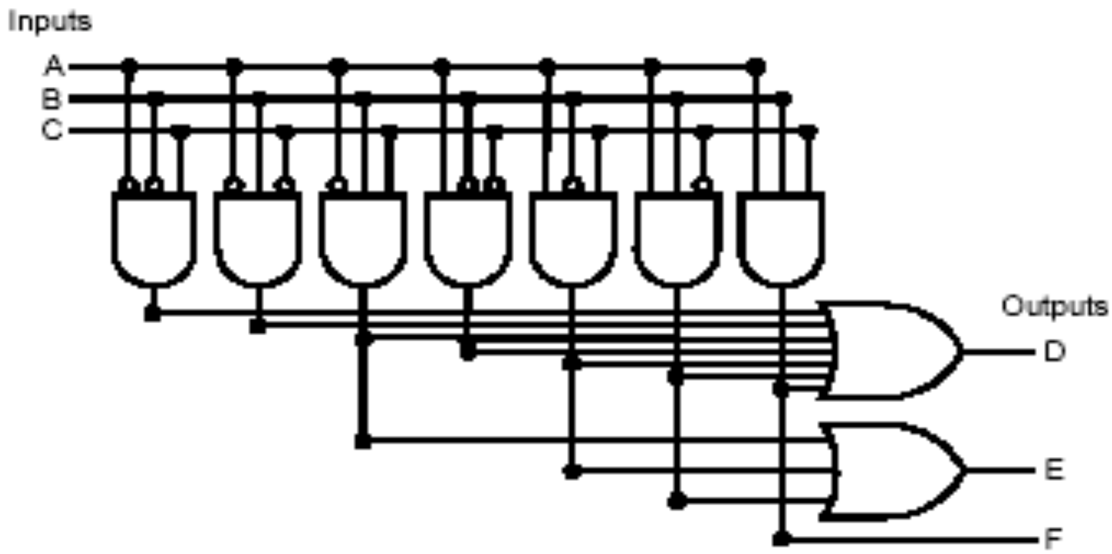
= ABCD + A~BCD + ABC~D + A~BC~D +

+ <u>ABCD</u> + ~ABCD + <u>ABC~D</u> + ~ABC~D +

+ ~A~B~C~D + ~A~B~CD

➢ La stessa espressione si può ricavare a partire dalla **tabella di verità**

SoP a più uscite



Esercizio: Ricavare la funzione in forma di tabella della verità

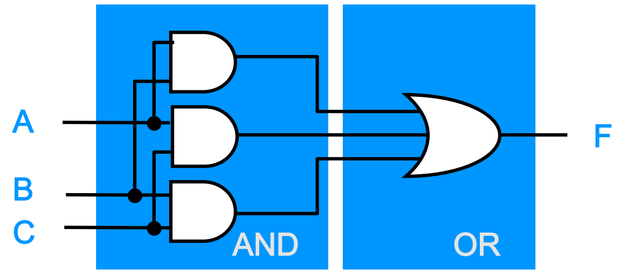
Esercizio: *funzione maggioranza*

❖ 3 ingressi, 1 uscita

1. Costruzione tabella di verità o espressione logica
2. Trasformazione a forma SOP
3. Eventuale semplificazione

$$F(A,B,C) = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC =$$
$$= \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC + ABC + ABC =$$
$$= AB(C+\overline{C}) + AC(B+\overline{B}) + BC(A+\overline{A}) =$$
$$= AB + AC + BC$$

A	B	C	F
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



Uscite indifferenti di una funzione logica

Situazione tipica in <u>sintesi (progetto)</u> di funzioni/circuiti logici:

❖ *Per alcune combinazioni degli ingressi, il valore assunto dall'uscita è <u>INDIFFERENTE</u>*

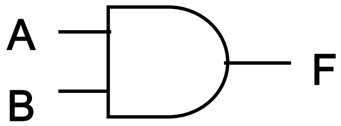
➢ **Simbolo: X**

❖ Come si risolve?

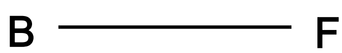
➢ **Si sceglie il caso che rende il circuito <u>più semplice</u>**

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

X=0 → F=AB



X=1 → F=B



Dalla SoP al circuito

- Dalla forma canonica (somma di mintermini) è facile passare al circuito:
  - Ogni mintermine si realizza con una porta **AND**
  - La somma dei mintermini si realizza con una porta **OR**
- Implementazione regolare
  - Solo due livelli di porte
  - Tempo di commutazione: $2 * t_{GATE}$
- Blocchi generali personalizzabili (PLA, ROM)
  - purché ci sia un numero sufficiente di componenti elementari

Tipi di circuiti che implementano SOP

PLA: ***P**rogrammable **L**ogic **A**rray*

- Matrici regolari AND e OR in successione, personalizzabili dall'utente.

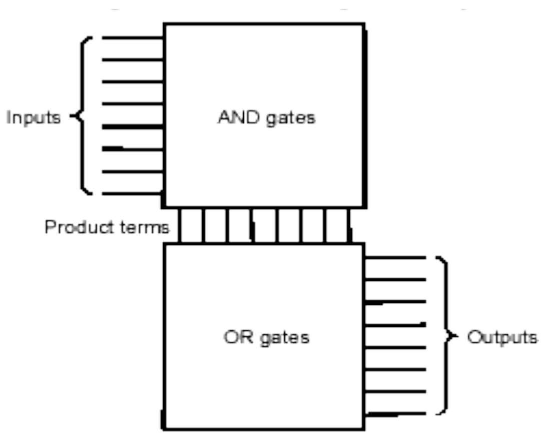
ROM: ***R**ead **O**nly **M**emory*

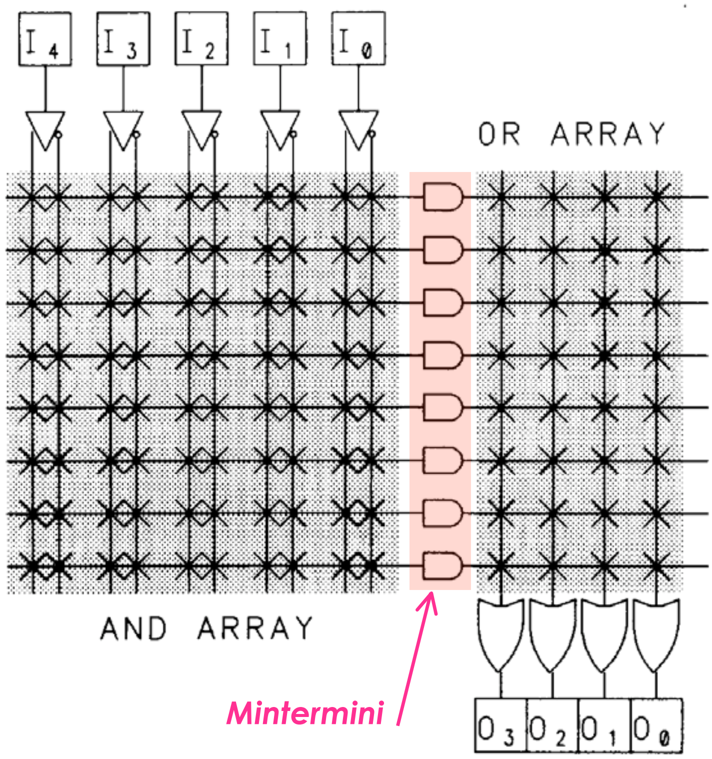
- Circuiti *ad-hoc* che implementano una particolare funzione in modo irreversibile.

PLA (Programmable Logic Array)

- La matrice degli AND ha ***n*** linee di ingresso
  - Ciascuna porta ha a disposizione ***2n*** segnali: ***n*** ingressi + ***n*** ingressi negati
- L'utente fornisce la matrice che dice quale linea entra in quale porta AND
  - Crea la matrice dei mintermini, bruciando in ingresso alle porte AND le linee che non servono.
- Le uscite della matrice AND entrano nella matrice OR con linee definite dall'utente
  - Matrice di programmazione OR
  - Si utilizza una porta OR per ogni funzione.

Struttura di una PLA

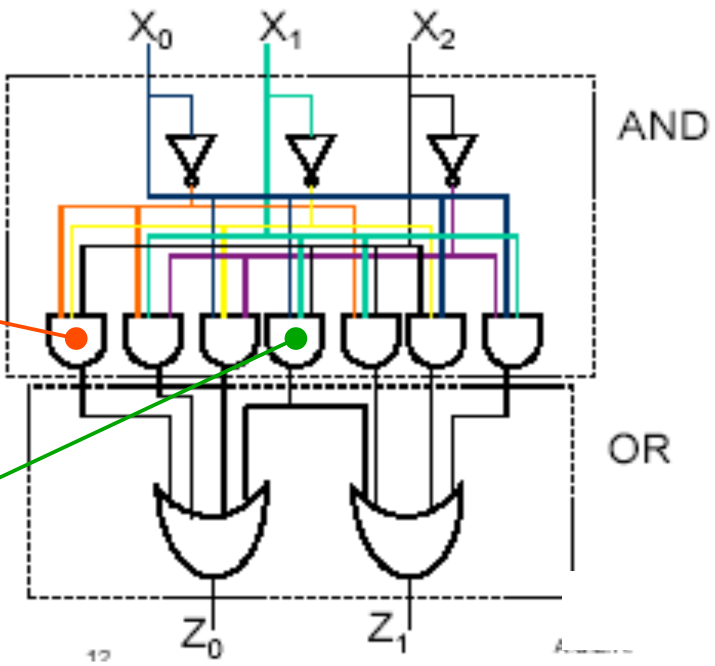




Esempio di PLA

- Realizzare con un PLA la funzione descritta dalla seguente TT

X_0	X_1	X_2	Z_0	Z_1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Esercizi sulla PLA

- Realizzare mediante PLA con 3 ingressi:
  - la funzione maggioranza.
  - la funzione che vale 1 se e solo se 1 solo bit di ingresso vale 1
  - un decoder
  - la funzione che vale 0 se l’input è pari, 1 se dispari
  - la funzione che calcola i multipli di 3 (con 4 ingressi)

Rappresentazione circuitale mediante ROM

- **Read-Only Memory, memoria di sola lettura.**
  - Funge anche da modulo combinatorio a uscita multipla.

- ***n*** linee di ingresso, ***m*** linee di uscita (**ampiezza**)
  - a ciascuna delle 2^n (**altezza**) configurazioni di ingresso (parole di memoria) è associata permanentemente una combinazione delle ***m*** linee di uscita
  - **Ad ogni parola corrisponde un mintermine, definito dal suo indirizzo di memoria**

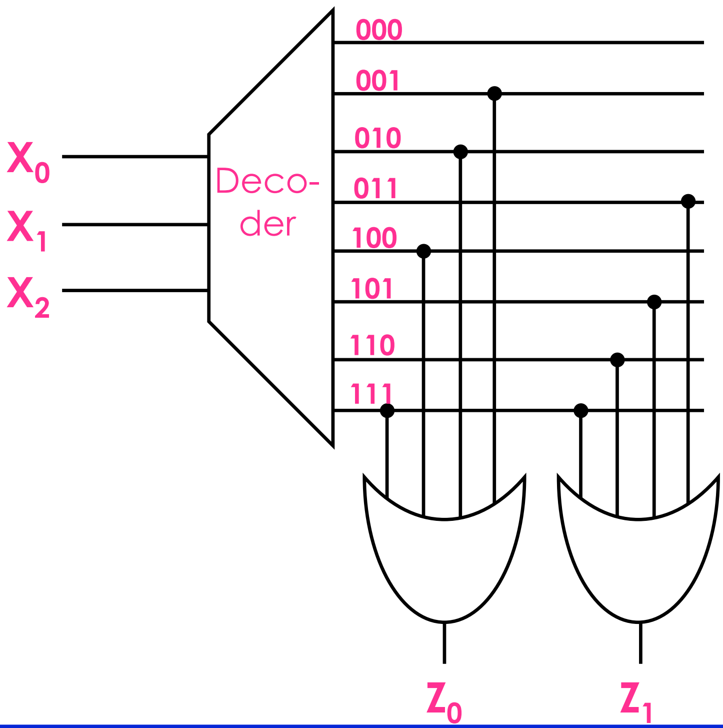
- L' ingresso seleziona la parola da leggere di ***m*** bit
  - Il **contenuto della parola** di memoria corrisponde all'**uscita** relativa a tale mintermine.

Decoder $n \rightarrow 2^n$ seguito da una matrice di ***m*** porte OR

Esempio di ROM

- *Realizzare con una ROM la funzione descritta dalla seguente tabella di verità:*

X_0	X_1	X_2	Z_0	Z_1
0	0	0	**0**	**0**
0	0	1	**1**	**0**
0	1	0	**1**	**0**
0	1	1	**0**	**1**
1	0	0	**1**	**0**
1	0	1	**0**	**1**
1	1	0	**0**	**1**
1	1	1	**1**	**1**



Forme canoniche

- Esiste un metodo per ricavare automaticamente un circuito che implementi una tabella di verità?

→ Forme canoniche

I. Somma di Prodotti (SOP)

II. DUALE: Prodotto di Somme (POS)

Prima forma canonica (SoP): Somma dei mintermini

$$F = \sum_{i=1}^{Q} m_i$$

$$F = A \cdot B + B \cdot \overline{C} =$$
$$= AB\left(C + \overline{C}\right) + \left(A + \overline{A}\right)B\overline{C} =$$
$$= ABC + AB\overline{C} + AB\overline{C} + \overline{A}B\overline{C} =$$
$$\boxed{= ABC + AB\overline{C} + \overline{A}B\overline{C}}$$

Seconda forma canonica

- DUALE della I forma canonica:
  - considero i casi in cui: **F = 0**

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

A B C	F
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

F = 0 *se e solo se:*

A=0 and B=0 and C=0

OR

A=0 and B=0 and C=1

OR

A=0 and B=1 and C=1

OR

A=1 and B=0 and C=0

OR

A=1 and B=0 and C=1

Seconda forma canonica

- Nuova definizione di **F:**
  - Elenco dei termini per cui: **F = 0 → ~F = 1**

$$\overline{\mathrm{F}} = \sum_{i=1}^{W} M_i \quad , \quad W \le 2^N$$

Maxtermine, M_j:

Prodotto di tutte le variabili di ingresso al quale corrisponde un valore di funzione = 0

I forma can.: $F = \sum_{j=1}^{Q} m_j \quad , \quad Q \le 2^N \longrightarrow$ $\boxed{Q + W = 2^N}$

Seconda forma canonica

- Esprimiamo F come: **somma di MAXtermini**:

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

A	B	C	F
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

$$\overline{\mathrm{F}} = \sum_{i=1}^{W} M_i$$

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

Seconda Forma Canonica: POS

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

❖ Negando entrambi i membri
ed applicando il II teorema di De Morgan si ottiene:

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

In generale:

II Forma Canonica: PoS (Product of Sums)

$$\overline{F} = \sum_{i=1}^{W} M_i, \quad W \le 2^N$$

$$\overline{\overline{F}} = F = \overline{\left(\sum_{i=1}^{W} M_i\right)} = (2^\circ \text{ Th. De Morgan}) = \prod_{i=1}^{W} \overline{M_i}$$

$$M_i = a \cdot b \cdot c \longrightarrow \overline{M_i} = \overline{a} + \overline{b} + \overline{c}$$

Somma di Prodotti

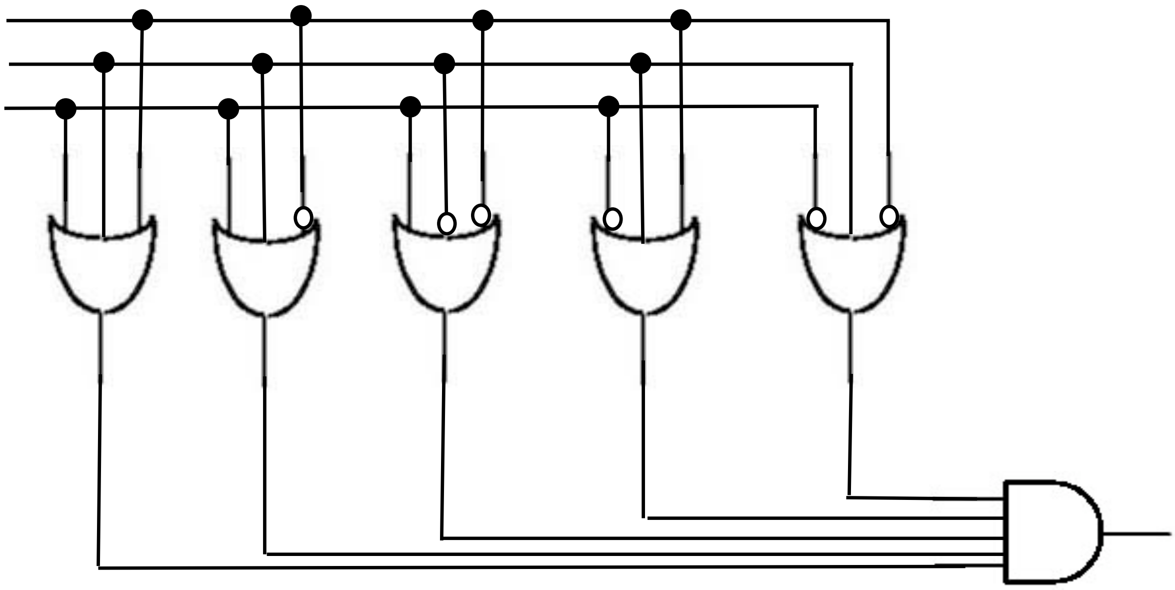
❖ I termini-somma sono i casi in cui: **F = 0**

$$\overline{M_i} = 0 \longrightarrow F = 0, \quad \forall i = 1..N$$

A	B	C	F
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

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

Circuito 2a forma canonica: POS



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