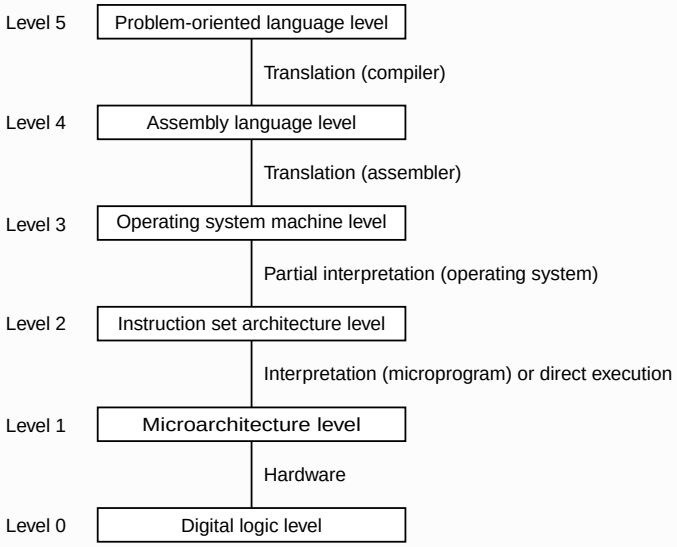


Analogico — Digitale

Nei calcolatori: segnale elettrico rappresenta l'informazione.
Due metodi:

- telegrafo: segnale presente - assente (digitale)
- telefono: segnale modulato (analogico)

Calcolatore digitale segue il paradigma del telegrafo:
– tolleranza ai disturbi, affidabilità
– semplicità di progettazione
– rappresentazione astratta dell'informazione
– less is more.

Codifica dell'informazione

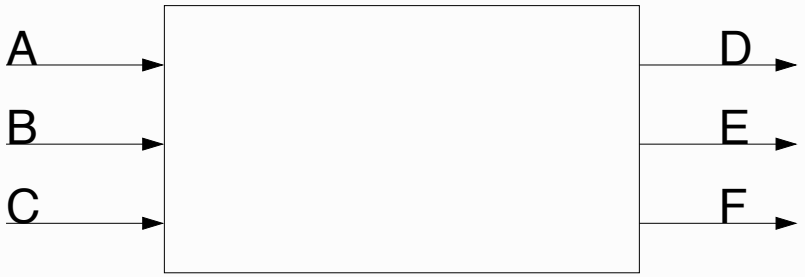
Tensione – segnale — cifra (binaria): bit .

- tensione presente: 2 - 3 volt — segnale presente — 1,
- tensione assente: $\sim$ 0 volt — segnale assente — 0.

In questo caso: logica positiva.
Logica negativa: all'assenza di tensione si associa 1.

Circuiti logici

Dispositivi con una serie di segnali di ingresso (input) e di segnali in uscita (output).



Combinatori – Sequenziali

Due classi di circuiti logici:

- COMBINATORI: l'uscita dipende solo dall'ingresso attuale, non c'è memoria.
- SEQUENZIALI: l'uscita dipende anche dagli ingressi passati (o dallo stato), hanno memoria della storia passata.

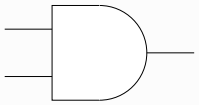
Porte logiche

Semplici circuiti combinatori.

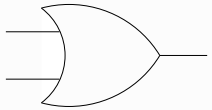
Elementi base della costruzione:

ogni circuito combinatorio può essere ottenuto collegando in modo opportuno un insieme di porte logiche.

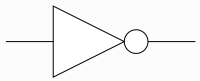
Porte logiche fondamentali



AND



OR



NOT

Comportamento delle porte logiche

- La porta logica AND ha uscita 1 se tutti gli ingressi sono a 1 (e zero altrimenti)
- La porta logica OR ha uscita 1 se almeno uno degli ingressi è a 1 (e zero altrimenti)
- La porta logica NOT ha uscita 1 se l'ingresso ha valore 0 (e zero altrimenti)

Le porte AND, OR possono avere più di 2 ingressi (con il comportamento descritto sopra).

Come si realizzano le porte logiche?

Paragone: elettronica — idraulica

Fornire un idea intuitiva del significato delle grandezze elettriche.

I due sistemi sono governate da leggi fisiche simili.

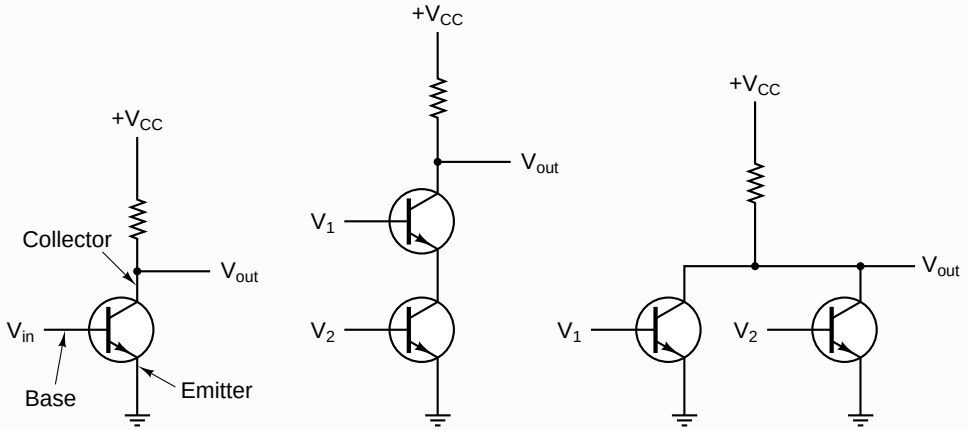
- corrente elettrica, flusso di elettroni (Ampere) — portata, flusso d'acqua;
- fili elettrici — tubi di un sistema idraulico;
- tensione elettrica (Volt) — pressione, spinta;
- batteria - alimentatore — pompa idraulica;
- resistenza (Ohm) — strozzatura, resistenza al flusso;

Paragone: elettronica — idraulica

- interruttore — rubinetto;
- condensatore — serbatoio;
- transistor — rubinetto regolato da una pressione idraulica.

Realizzazione fisica porte logiche

Transistor: elemento fondamentale per la realizzazione dei circuiti, amplifica il segnale.



Descrizione comportamento circuiti

Informale:

- La porta logica OR ha uscita 1 se almeno uno degli ingressi è a 1 (e zero altrimenti)
- D ha valore 1 se almeno uno degli ingressi ha valore 1 (D è vero se almeno uno degli ingressi è vero)
- E ha valore 1 se esattamente due ingressi hanno valore 1
- F ha valore 1 se tutte tre gli ingressi hanno valore 1

Formale: tabelle di verità

- Descrizione esaustiva del comportamento: per ogni combinazione dei valori di ingresso, specifica i valori di uscita;
- una tabella con tante righe quante le combinazioni di ingresso.
- Insieme finito di combinazioni di ingresso;
- se ci sono n ingressi, le possibili combinazioni sono: 2^n.

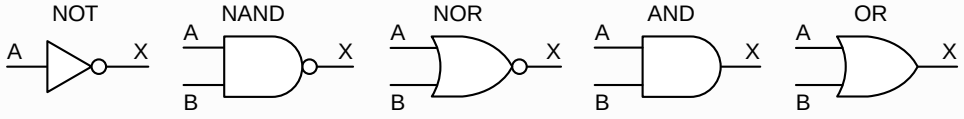
Tabella di verità

A	B	C	D	E	F
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Tabella di verità

A	B	C	D	E	F
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	0	1

Tabelle di verità per le porte logiche



A	X
0	1
1	0

(a)

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

(b)

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

(c)

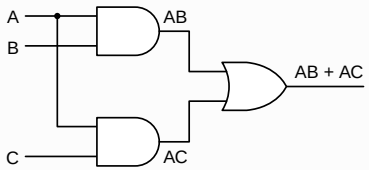
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

(d)

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

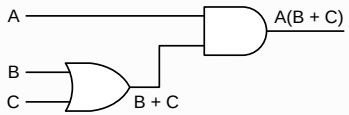
(e)

Dal circuito alla tabella di verità



A	B	C	AB	AC	AB + AC
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

(a)



A	B	C	A	B + C	A(B + C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

(b)

Sintesi di circuiti

Data una tabella di verità, definire un circuito.

Ricetta (algoritmo):

- considero tutte le righe della tabella di verità con uscita 1
- per ciascuna di esse costruisco un circuito che vale 1 solo quando gli ingressi (input) coincidono con gli input associati alla riga, uso porte NOT e AND
- riunisco le uscite delle porte AND con una porta OR.

Sintesi di circuiti

Altri metodi:

- esisto un metodo duale (principio di dualità: scambio $0 \leftrightarrow 1$, AND $\leftrightarrow$ OR)
- mappe di Karnaugh: permette di ottenere circuiti compatti, con poche porte.

Mappe di Karnaugh

Metodo per ottenere circuiti minimali.

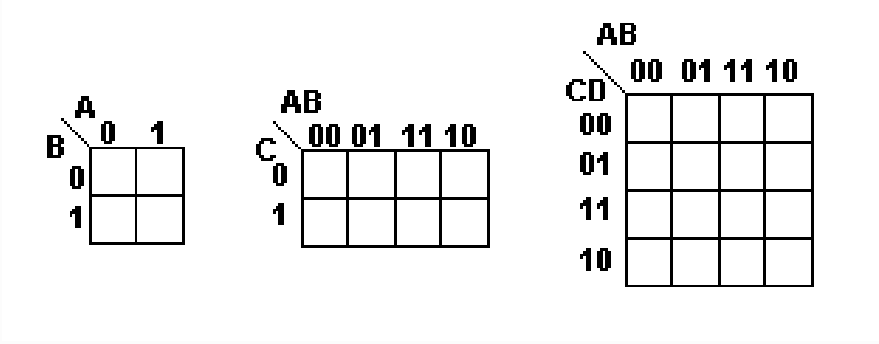
Il metodo precedente per la sintesi di circuiti crea circuiti con più porte logiche del necessario.

Può essere migliorato:

- spesso un gruppo di porte AND può essere sostituito da un'unica porta AND.
- ossia: un gruppo di 1 viene rappresentato da un unica porta AND.

Mappe di Karnaugh:

Tabelle di verità scritti nella forma di matrici:



notare l'ordine di scittura delle combinazioni
$00, 01, 11, 10$
ordine da seguire esattamente.

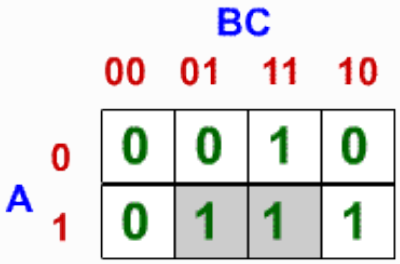
Mappe di Karnaugh:

Sono evidenziati i gruppi di 1 realizzabili con un unica porta AND:

- appaiono in un rettangolo (quadrato) di 2, 4, 8 ... (2^n) elementi;
- in ogni rettangolo:
  - alcune variabili assumono tutte le possibili combinazioni,
  - altre assumo sempre lo stesso valore,

nell'AND associato appaiono solo quest'ultime.

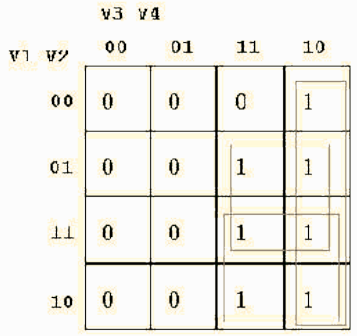
Esempio:



$F = AC + BC + AB$

La variabile che assume entrambi i valori, nella coppia di 1 adiacenti, può essere omessa.

Mappe di Karnaugh:



$F = V_3\overline{V_4} + V_2V_3 + V_1V_3$

Mappe di Karnaugh: lati opposti.

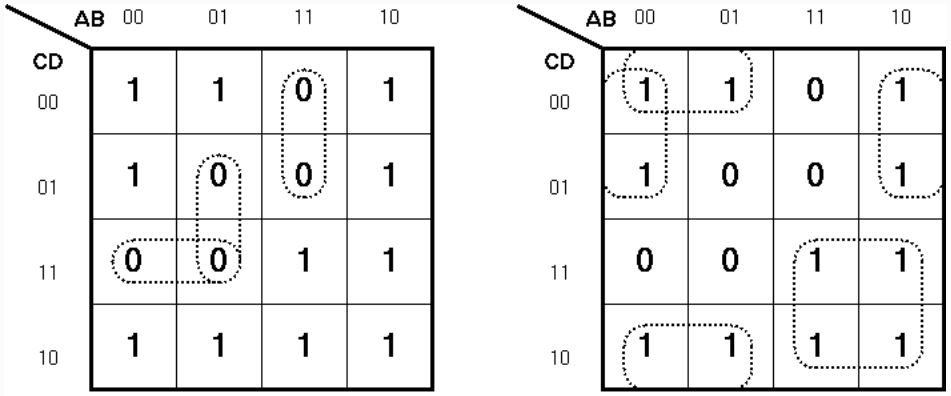
I lati opposti vanno considerati contigui.

WX \ YZ	00	01	11	10
00	1	0	0	1
01	1	1	0	0
11	1	0	0	0
10	1	0	0	1

$$F = \overline{YZ} + \overline{XZ} + \overline{W}X\overline{Y}$$

Principio di dualità

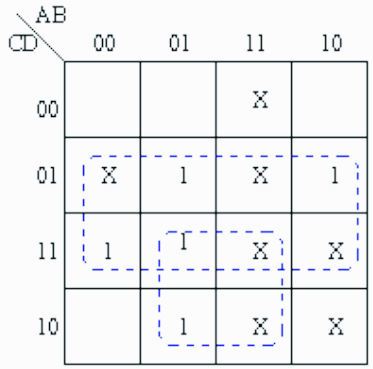
Versione duale delle mappe di Karnaugh:



$S_n = \overline{A}CD + AB\overline{C} + \overline{A}BD$ $S_p = \overline{A}\overline{C} + BC + AD$

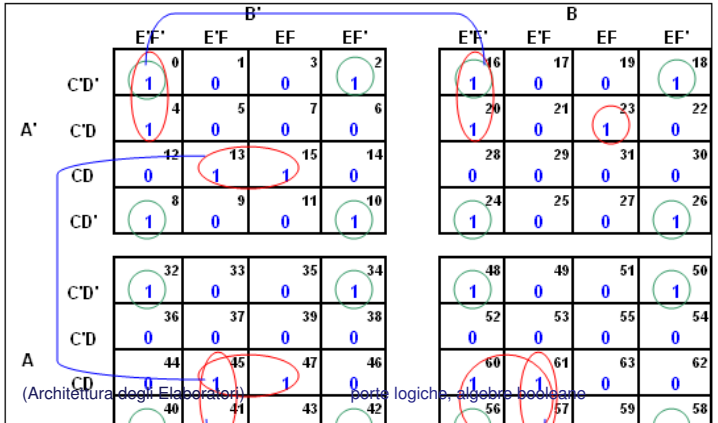
Valori indefiniti

La specifica può non definire il comportamento su tutto le combinazioni di input. Alcune combinazioni lasciate indeterminate. L'implementazione sceglie i valori che portano al circuito più semplice.



Mappe di Karnaugh con 5 o 6 variabili

Si usa la terza dimensione. Disegnare 2 o 4 mappe di Karnaugh a 4 variabile e considerarle sovrapposte.



Algebra booleana

Descrive un circuito con un espressione algebrica, utile per:

- rappresentare sinteticamente un circuito,
- verificare l'equivalenza tra due circuiti,
- semplificare un circuito: trasformarlo in un circuito equivalente più semplice.

[George Bool (1815-1864), Claude Shannon (1916-2001)]

Algebra booleana

Algebra dei valori di verità:

- ideata nell'ambito della logica [Bool].
- In logica: è necessario formalizzazione linguaggio;
- la valutazione della verità di una espressione viene ridotta ad un calcolo;
- si considerano solo due possibili valori di verità: vero (1), falso (0).

Formalizzazione del linguaggio

Frasi composte da proposizioni base (atomiche) come:
splende il sole — piove — nevica;
combinate con l'uso di connettivi (e, o, non):

- piove e splende il sole
- o piove o splende il sole
- non piove
- se piove allora – non splende il sole e – non nevica.

Si calcola la verità di una frase a partire dalla verità delle asserzioni atomiche.

Algebra booleana

Operazioni (connettivi base)

AND	$\cdot$	$\wedge$	prodotto logico	(binaria)
OR	$+$	$\vee$	somma logica	(binaria)
NOT	$\overline{}$	$\neg$	negazione	(unaria)

Valori (valori di verità)

TRUE	1	*t*
FALSE	0	*f*

Connetivi e valori hanno diverse scritture a seconda dell'ambito: logica, progettazione di circuiti.

Comportamento delle operazioni

Coincide con quello delle porte logiche:

- l'espressione (*A* AND *B*) è vera se *A* è vera **e** anche *B* è vera
  (es: piove e splende il sole)
- (*A* OR *B*) è vera se almeno una tra *A* e *B* sono vere
  (es: o piove o splende il sole)
- (NOT *A*) è vera se *A* è falsa.
  (es: non piove)

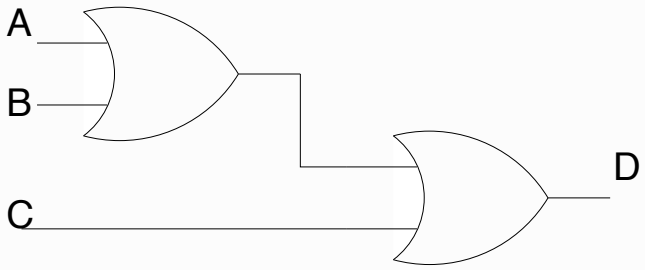
Applicazioni all'informatica

Rappresentare un circuito mediante un espressione algebrica [Shannon]

circuiti logici	$\Rightarrow$	funzioni, espressioni
porte logiche	$\Rightarrow$	operazioni base
input	$\Rightarrow$	argomenti (variabili)
output	$\Rightarrow$	valore dell'espressione
segnali intermedi	$\Rightarrow$	sotto-espressioni

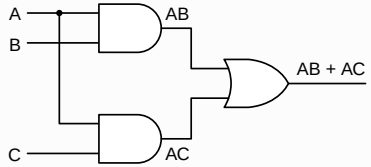
Corrispondenza circuiti - espressioni

Ai circuiti formati da porte logiche posso associare espressioni (e viceversa).



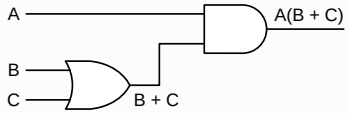
$D = ((A + B) + C)$

Corrispondenza circuiti - espressioni



A	B	C	AB	AC	AB + AC
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

(a)

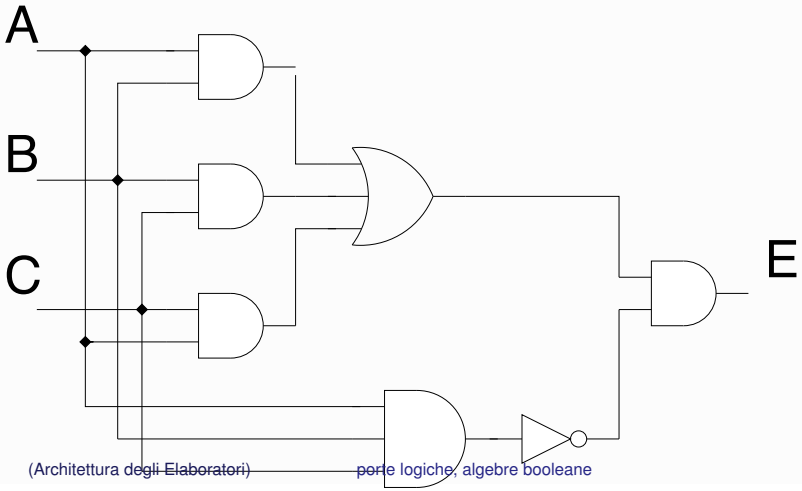


A	B	C	A	B + C	A(B + C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

(b)

Corrispondenza espressioni - circuiti

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

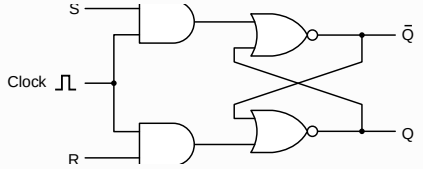


Corrispondenza non perfetta

Ambiguità: un circuito può essere rappresentato con più formule.

Alcuni circuiti non hanno una corrispondente espressione algebrica.

- circuiti in cui un'uscita comanda più ingressi;
- circuiti con retroazione:



C'è maggiore libertà nel collegare fili che nello scrivere espressioni.

Convenzioni nella scrittura

Le usali convenzioni dell'algebra:

- il segno di prodotto può essere omesso, e volte indicato con $\times$ o $*$ (AB, $A \times B$, $A * B$);
- il prodotto ha precedenza sulla somma;
- per l'associatività delle operazioni di AND e OR, si possono omettere le parentesi
  (A + B) + C = A + B + C = A + (B + C).

Inoltre:

- la negazione può essere anche rappresentata da $'$ ($A' = \overline{A}$);

Equivalenze algebriche

Ogni algebra caratterizza da un insieme di equivalenze.

Equivalenze booleane utili per:

- stabilire equivalenze tra circuiti;
- semplificare circuiti.

Equivalenze booleane

Identità	$0 + A = A$
Elem. nullo	$1 + A = 1$
Idempoten.	$A + A = A$
Inverso	$A + \overline{A} = 1$
Commutat.	$A + B = B + A$
Associat.	$(A + B) + C = A + (B + C)$
Distribut.	$A(B + C) = AB + AC$
Assorbim.	$A + (A \cdot B) = A$
De Morgan	$\overline{A + B} = \overline{A} \cdot \overline{B}$
Negazione	$\overline{\overline{A}} = A$

Equivalenze duali

Identità	$1 \cdot A = A$
Elem. nullo	$0 \cdot A = 0$
Idempoten.	$A \cdot A = A$
Inverso	$A \cdot \overline{A} = 0$
Commutat.	$A \cdot B = B \cdot A$
Associat.	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
Distribut.	$A + BC = (A + B)(A + C)$
Assorbim.	$A \cdot (A + B) = A$
De Morgan	$\overline{A \cdot B} = \overline{A} + \overline{B}$
Negazione	$\overline{\overline{A}} = A$

Considerazioni

- Alcune equivalenze: Identità, Commutativa, Associativa, Distributiva (somma sul prodotto), Elemento nullo $\cdot$, sono valide anche per l'algebra degli interi (le altre no).
- Tutte equivalenze della algebra degli interi restano valide tenuto conto che negazione non ha nessuna attinenza con le operazioni di opposto o inverso.
- Equivalenze dimostrabili mediante tabelle di verità,
  oppure mediante argomenti di logica.

Semplificazione di espressioni

- Le proprietà Identità, Elemento nullo permettono di eliminare le costanti 0 e 1, dalle espressioni.
- Le leggi di De Morgan e Negazione permettono di portare la negazione a livello di variabili.
- Le proprietà Commutativa, Associativa, Inverso, Idempotenza e Assorbimento permettono di riordinare i termini ed eliminare doppioni.
- La proprietà distributiva permette di riscrivere ogni espressione come somma di prodotti (forma normale) (polinomio).

Esempi di applicazione

Attraverso le regole indicate si possono semplificare le seguenti espressioni.

- De Morgan: $\overline{AB + \overline{C}}$, $\overline{ABC}$, $\overline{A + B + C}$.
- Distributiva: $((A + B)C + A)B$
- Con alcune regole: $A(1 + B) + 1A + 0$
  $\overline{A}BC + A\overline{B}C + ABC + 0$.

Assorbimento: mostrare come la proprietà di Assorbimento si possa derivare dalle altre (Identità, Distributiva, Elemento nullo).

Principio di dualità

Definizione: data un espressione booleana, E, l'espressione duale $\widetilde{E}$ si ottiene scambiando tra loro le operazioni $+$ e $\times$ e le costanti 0 e 1.

Esempio:

- $E = A + \overline{A} \cdot (B + 0)$
- $\widetilde{E} = A \cdot (\overline{A} + B \cdot 1)$

Proprietà: se $E_1 = E_2$ allora $\widetilde{E_1} = \widetilde{E_2}$.

In ogni equivalenza booleana, se scambiamo tra loro le operazioni $+$ e $\times$ e le costanti 0 e 1, otteniamo una nuova equivalenza (duale della precedente) valida.

Altre porte logiche

NAND NOR XOR XNOR

A	B	A NAND B	A NOR B	A XOR B	A XNOR B
0	0	1	1	0	1
0	1	1	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1

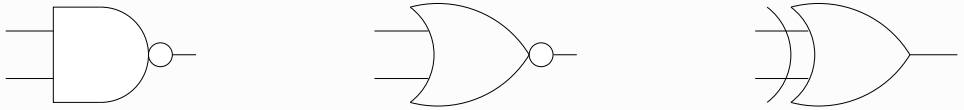
Definizione mediante espressioni

A NAND $B = \overline{A \cdot B}$
A NOR $B = \overline{A + B}$
A XOR $B = (A \cdot \overline{B}) + \overline{A} \cdot B = (A + B) \cdot (\overline{A} + \overline{B})$
A XNOR $B = A \cdot B + \overline{A} \cdot \overline{B}$

Le porte NAND e NOR possono essere generalizzate ad n ingressi.
Una NAND con n ingressi restituisce il valore negato di una porte AND con n ingressi.

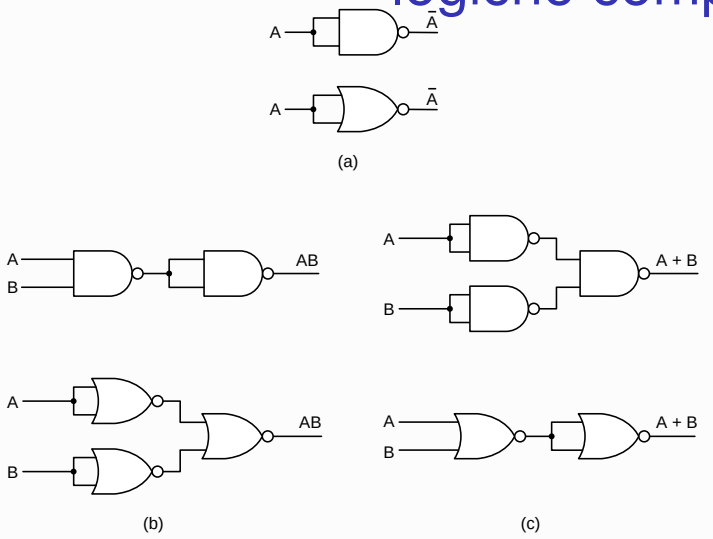
Esercizio Ha senso definire una porta XOR con n ingressi? Se "sì", descriverne il comportamento.

Rappresentazione grafica



NAND NOR XOR

NAND e NOR: porte logiche complete



NAND e NOR: porte logiche complete

Ogni “polinomio” può essere implementato con porte NAND:

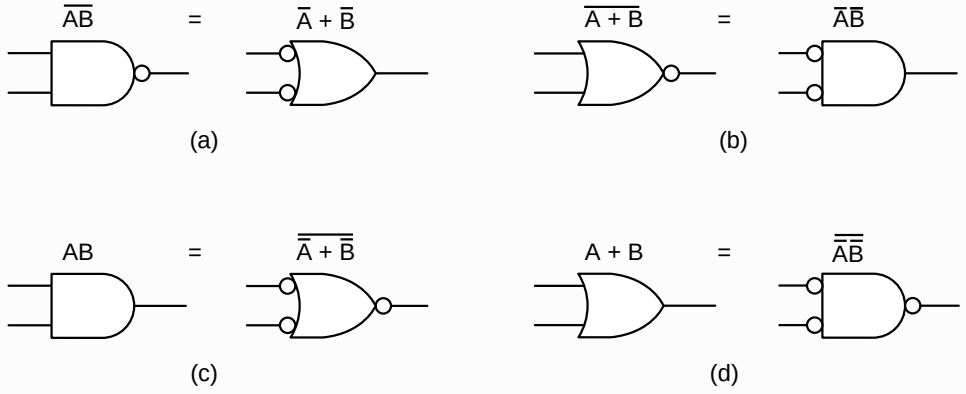
$$A_1 \cdot A_2 \cdot A_3 + B_1 \cdot B_2 + C_1 \cdot C_2 \cdot C_3 \cdot C_4 =$$
$$\overline{\overline{A_1 \cdot A_2 \cdot A_3 + B_1 \cdot B_2 + C_1 \cdot C_2 \cdot C_3 \cdot C_4}} =$$
$$\overline{\overline{A_1 \cdot A_2 \cdot A_3} \cdot \overline{B_1 \cdot B_2} \cdot \overline{C_1 \cdot C_2 \cdot C_3 \cdot C_4}}$$

Nei circuiti integrati vengono impiegate esclusivamente porte NAND o NOR: più economiche da implementare.

Esercizio: come rappresentare, in modo efficiente, ogni circuito con porte NOR?

Bolla di inversione

può apparire anche all’ingresso delle porte logiche:



Algebre di Bool in matematica

Un insieme A con tre operazioni: $+, \times, -$ e due costanti: $0, 1$,
che soddisfano le proprietà sopra elencate: Identità, Elemento nullo, Inverso, ...
Esempio: (l'insieme delle parti) l'insieme formato dai sottoinsiemi di numeri naturali
con le operazioni: unione $+$, intersezione $\times$, e complemento $-$,
e con le costanti: insieme vuoto, insieme completo.
Nella progettazione di circuiti ci si limita all'algebra di commutazione (0, 1)

Esercizi

- Esemplificare le espressioni:
  - $A + \overline{A}B$
  - $\overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + AB\overline{C}$
  - $\overline{A} + \overline{\overline{B} + \overline{C}} + ABC$
  - $\overline{ABC} + A\overline{BC} + \overline{AB}C + AB\overline{C}$

  sia algebricamente che attraverso le mappe di Karnaugh.