Architettura degli Elaboratori
4 - Reti Combinatorie e Algebra di Boole

Zeynep KIZILTAN

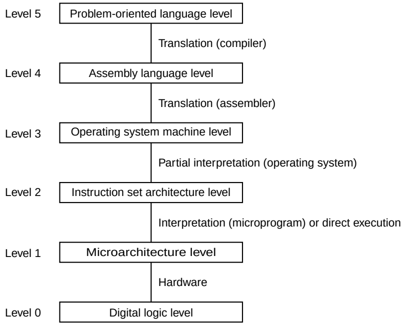
Dipartimento di Scienze dell'Informazione
Università degli Studi di Bologna

Anno Accademico 2007/2008

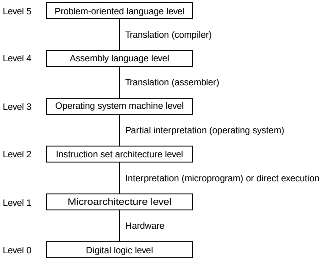
Sommario

Il Livello Logico-Digitale

- Il ciclo di lezioni che andiamo ad iniziare riguarda il **livello logico-digitale**.
- Come abbiamo introdotto all'inizio del corso, il livello logico-digitale sta **alla base** delle gerarchia di macchine virtuali.

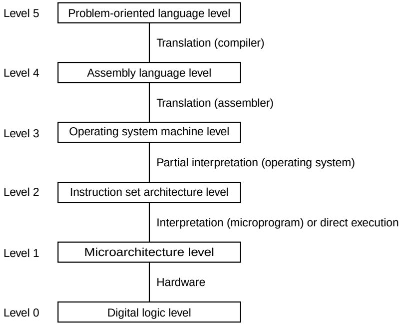


Il Livello Logico-Digitale



- In questo livello, i programmi sono costitutiti da **circuiti digitali** (o **reti logiche**), i cui componenti fondamentali sono detti **porte logiche**.
- I circuiti sono entità concrete e tangibili, per cui non vanno tradotti o interpretati.
- Il funzionamento interno delle porte logiche si può comprendere solo ad un livello inferiore, detto **livello dei dispositivi**.
  - Questo livello non è mostrato nella figura.
  - Ricade nel campo del'ingegneria elettronica.

Il Livello Logico-Digitale

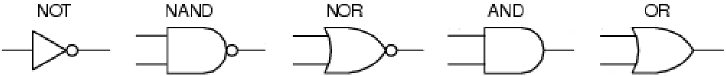


Nel livello logico-digitale, esistono due classi di reti logiche:

- Le **reti combinatorie**, che non hanno uno stato interno e che verranno studiate in questa parte del corso. Esempio: **ALU**.
- Le **reti sequenziali**, che hanno uno stato interno e che verranno studiate nella prossima parte del corso. Esempio: **registri**.

Circuiti Digitali

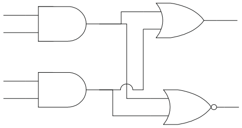
- Un **circuito digitale** è un'interconnessione di porte logiche.
- Nei fili che interconnettono le porte logiche in un circuito sono presenti solo due valori logici:
  - Il valore **0**, rappresentato da un segnale compreso tra 0 e 1 volt.
  - Il valore **1**, rappresentato da un segnale compreso tra 3 e 5 volt.
  - Le tensioni al fuori di questi intervalli non sono ammesse.
- Esistono almeno cinque tipi diversi di porte logiche: **NOT**, **NAND**, **NOR**, **AND**, **OR**.



- La figura mostra i simboli usati per indicare queste porte logiche.

Circuiti Digitali

- Un esempio di circuito è composto da 4 porte logiche dove:
  - le uscite delle porte AND sono connesse alle porte OR e NOR.



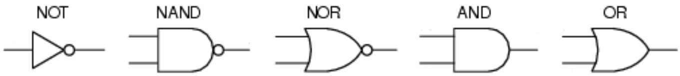
- Si noti che:
  - Ogni porta logica ha uno o più **ingressi** e un'**uscita**.

  - Ogni circuito ha uno o più **ingressi** e una o più **uscite**.

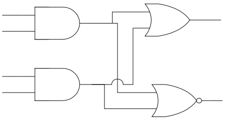
Circuiti Digitali

- Gli ingressi di un circuito possono essere connessi:
  - Ad uno degli ingressi di una porta logica.
  - Ad una delle uscite del circuito.
- In un circuito, l'uscita di una porta logica può essere connessa:
  - Ad uno degli ingressi di un'altra porta logica.
  - Ad una delle uscite del circuito.

Porte Logiche

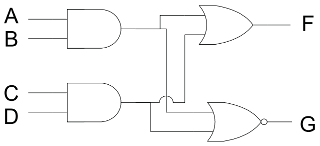


Circuito di Esempio



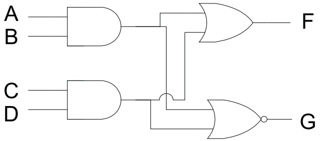
Reti Combinatorie

- Le reti combinatorie sono circuiti digitali che **non contengono cicli**.
  - In altre parole, non possono esistere cammini ciclici all'interno del circuito.
- Un **input** per il circuito consiste in una sequenza di valori binari, uno per ogni ingresso del circuito.
  - ABCD nel nostro esempio.
- Un **output** per un circuito consiste in una sequenza di valori binari, uno per ogni uscita del circuito.
  - FG nel nostro esempio.



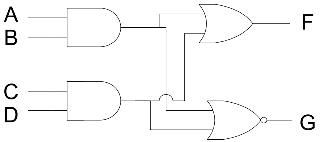
Reti Combinatorie

- L'output di una rete combinatoria in corrispondenza di un certo input si può calcolare come segue:
  - Per ogni porta logica, se sono noti i valori binari in corrispondenza dei suoi ingressi, allora sarà noto anche il valore binario in corrispondenza della sua uscita.
  - Nel nostro esempio:
    - Dati i valori degli ingressi A e B, la porta AND di sopra produce un'uscita F1.
    - F1 diventa il primo ingresso della porta OR.
    - Analogamente, i valori degli ingressi C e D portano all'uscita F2 alla porta AND di sotto.
    - F2 diventa il secondo ingresso della porta OR.
    - L'uscita F si può ottenere usando i valori F1 e F2 all'ingresso della porta OR.



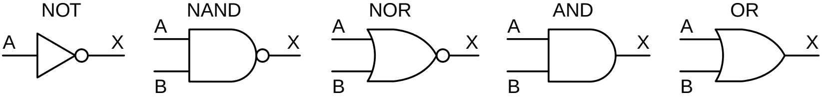
Reti Combinatorie

- Le regole che permettono di determinare, per ogni porta logica, quale sia il valore della sua uscita in corrispondenza di ciascun valore delle entrate dipendono solo dal tipo di porta.
- Vediamo adesso i comportamenti funzionali delle porte logiche di base...



Porte Logiche - Comportamenti Funzionali

- Nella figura:
  - A e B rappresentano gli ingressi delle porte;
  - X rappresenta l'uscita delle porte;
  - ogni riga specifica il valore in uscita, data una particolare combinazione dei valori in ingresso.



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)

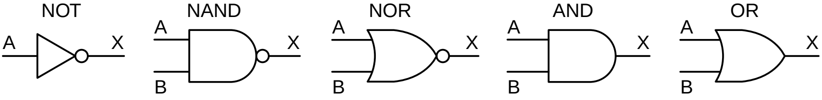
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

(d)

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

(e)

Porte Logiche - Comportamenti Funzionali



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

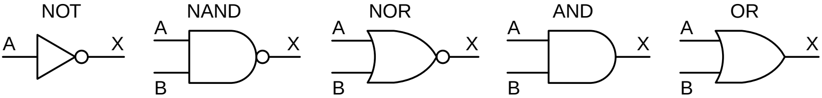
(d)

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

(e)

- La porta NOT è un invertitore che:
  - converte un valore logico 0 in un valore logico 1;
  - converte un valore logico 1 in un valore logico 0.
- La porta NAND produce un segnale di uscita 0 logico se e solo se entrambi i segnali di ingresso valgano 1.
- La porta NOR risponde con un segnale di uscita 1 logico se e solo se entrambi i segnali di ingresso valgano 0.

Porte Logiche - Comportamenti Funzionali



(a)

A	X
0	1
1	0

(b)

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

(c)

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

(d)

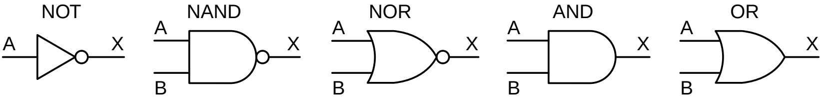
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

(e)

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

- Facendo passare il valore in uscita della porta NAND in un circuito invertitore, si ottiene la porta AND il cui comportamento è l'opposto di quello della porta NAND.
  - L'uscita vale 1 se e soltanto se entrambi gli ingressi valgono 1.
- Analogamente, è possibile collegare la porta NOR a un invertitore, in modo da ottenere la porta OR il cui comportamento è l'opposto di quello della porta NOR.
  - L'uscita vale 1 se almeno uno dei ingressi vale 1.

Porte Logiche - Comportamenti Funzionali



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

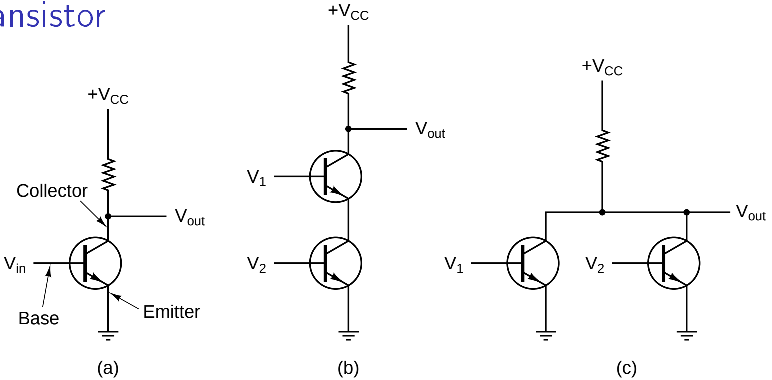
(e)

- Quindi, i cerchi alla fine delle porte NAND e NOR rappresentano un'inversione.
- Si noti che le porte NAND, NOR, AND, OR possono avere più di due ingressi.
- Si può supporre, a questo punto, che il **tempo** necessario alla valorizzazione dell'uscita di una porta logica sia trascurabile.

Come Sono Realizzate le Porte Logiche?

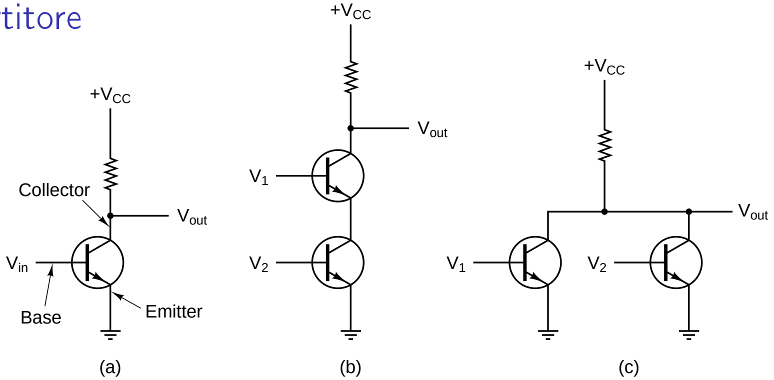
- Il funzionamento interno delle porte logiche è un problema di **livello dei dispositivi**.
- Questo livello si trova sotto il nostro livello 0 (livello logico digitale): quindi non ce ne occuperemo in dettaglio.
  - Però è importante avere un'idea di base.
- La moderna logica digitale si basa sui **transistor**.
- Il transistor è un dispositivo elettronico, che funziona come un interruttore.
- Uno o più transistor collegati tra loro danno origine a porte logiche (NAND, OR, NOT, ...).
- Vediamo in breve come è fatto un transistor.

Il Transistor



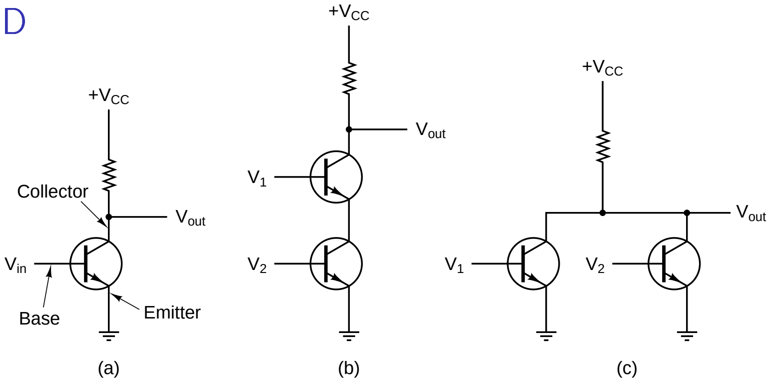
- Il transistor ha tre connessioni verso il mondo esterno: il **collettore**, la **base**, e l' **emettitore**.
  - La base funziona come uno strumento di controllo, che viene attivato con una tensione di ingresso: V_{in} in figura (a).
  - Se V_{in} è una tensione "alta" (e.g., da 2 a 5V) la corrente è libera di passare dal collettore all'emettitore.
  - Se V_{out} è una tensione "bassa" (e.g., da 0 a 1V) il collettore viene separato dall'emettitore, e la corrente non può passare.

Invertitore



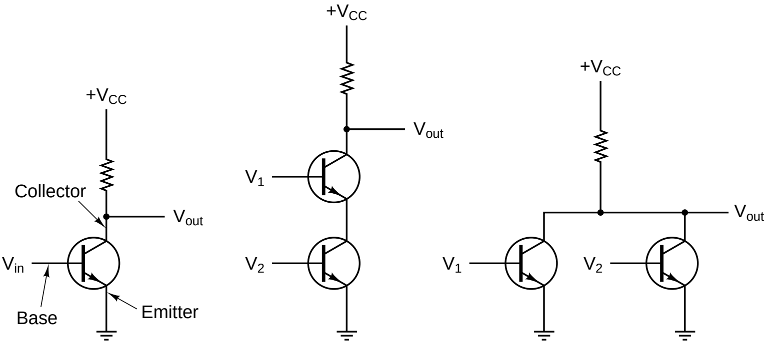
- Se il transistor ha il collettore collegato a una tensione $+V_{CC}$ tramite una resistenza, e l'emettitore collegato a terra (a):
  - quando V_{in} è alta, V_{out} è bassa (al livello della terra)
  - quando V_{in} è bassa, V_{out} è alta (vicina alla tensione $+V_{CC}$)
- Questo comporta una **inversione** del segnale di ingresso V_{in}.
- Per questo motivo, il circuito (a) si chiama **invertitore**.
  - Se diciamo che "alto" corrisponde a "1" e "basso" a "0", abbiamo realizzato un NOT logico!

NAND



- La figura (b) mostra un altro circuito, in cui abbiamo due transistor in serie, con basi V_1 e V_2.
- La corrente è libera di circolare attraverso i due transistor *solo se entrambe* le basi sono a un livello *alto*.
- Quindi V_{out} è **basso** *se e solo se* V_1 è **alto E** V_2 è **alto**.
- Di conseguenza, (b) può rappresentare un NAND logico.
- Per ottenere un AND bisogna inserire un invertitore dopo V_{out}
  - L'AND è più costoso del NAND, perché richiede 3 transistor!

NOR



(a) (b) (c)

- La figura (c) mostra un terzo circuito, in cui abbiamo due transistor collegati in parallelo.
- La corrente è libera di circolare *se almeno una* base è a un livello *alto*.
- Quindi V_{out} è **basso** se V_1 è **alto OPPURE** V_2 è **alto**.
- Di conseguenza, (c) può rappresentare un NOR logico.
- Per ottenere un OR bisogna inserire un invertitore dopo V_{out}
  - L'OR è più costoso del NOR, perché richiede 3 transistor!

Aspetti Tecnologici dei Transistor

- Una tensione tipica per V_{CC} è 5V.
- Il tempo che un transistor impiega per passare da uno stato all'altro (e.g. da $V_{out} = 5V$ a $V_{out} = 0V$) è di pochi **nanosecondi**.
- Esistono due principali tecnologie di costruzione dei gate:
  - la tecnologia **bipolare**, che può essere di vari tipi:
    - **TTL** (Transistor-Transistor Logic);
    - **ECL** (Emitter-Coupled Logic), più veloce della prima;
  - la tecnologia **MOS** (Metal Oxide Semiconductor).
    - I gate MOS sono più lenti dei TTL ed ECL, però sono più piccoli e consumano meno potenza.
    - Quindi scaldano meno!
    - Le CPU e memorie moderne usano la tecnologia CMOS, che funziona con V_{CC} a 3,3V.

L'Algebra di Boole

- L'Algebra di Boole è un utile strumento per l'analisi e la sintesi delle reti combinatorie e sequenziali.
- Le **Algebre di Boole** sono strutture algebriche che soddisfano alcune proprietà.
  - Lo studio delle Algebre di Boole in generale richiederebbe troppo tempo.
- A noi interessa una particolare Algebra di Boole, che consiste nell'insieme $B = \{0, 1\}$ e in tre operazioni su questo insieme:
  - L'operazione binaria di **addizione**, indicata con $+$ oppure con OR. Valgono le seguenti identità: $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$ e $1 + 1 = 1$.
  - L'operazione binaria di **moltiplicazione**, indicata con $\cdot$, con AND oppure tramite la semplice giustapposizione degli operandi. Valgono le seguenti identità: $0 \cdot 0 = 0$, $0 \cdot 1 = 0$, $1 \cdot 0 = 0$ e $1 \cdot 1 = 1$.
  - L'operazione unaria di **negazione**, indicata con il simbolo NOT oppure con una linea sopra il relativo operando. Valgono le seguenti identità: $\overline{0} = 1$ e $\overline{1} = 0$.

Proprietà dell'Algebra di Boole

- Si notino le identità di base dell'algebra booleana.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- Le prime nuove identità coinvolgono una singola variabile.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- L'identità 9 stabilisce che la doppia negazione restituisce la variabile originale.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\ \overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- L'identità 10 e 11 sono le leggi commutative.
- Stabiliscono che l'ordine in cui le variabili sono presenti non influenza il risultato quando si usano gli operatori AND e OR.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- L'identità 12 e 13 sono le leggi associative.
- Stabiliscono che il risultato di un'operazione con 3 variabili è independente dell'ordine con cui queste sono considerate.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- L'identità 14 e 15 sono le leggi distributive.
  - L'identità 14 è ben nota dell'algebra ordinaria.
  - L'identità 15 non si trova nell'algebra ordinaria.

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Proprietà dell'Algebra di Boole

- L'identità 16 e 17 sono il Teorema di De Morgan.
  - Si tratta di un teorema molto importante utilizzato per ottenere il complemento di un'espressione.
  - Il teorema può essere esteso a tre o più variabili:

$$\overline{X_1 + X_2 + \ldots + X_n} = \overline{X_1}\,\overline{X_2} \ldots \overline{X_n}$$
$$\overline{X_1\,X_2 \ldots X_n} = \overline{X_1} + \overline{X_2} + \ldots + \overline{X_n}$$

Elemento neutro	1. $0 + A = A$	2. $1A = A$
Assorbimento	3. $1 + A = 1$	4. $0A = 0$
Idempotenza	5. $A + A = A$	6. $AA = A$
Complementazione	7. $A + \overline{A} = 1$	8. $A\overline{A} = 0$
Doppia Negazione	9. $\overline{\overline{A}} = A$	
Commutativa	10. $A + B = B + A$	11. $AB = BA$
Associativa	12. $(A + B) + C = A + (B + C)$	13. $(AB)C = A(BC)$
Distributiva	14. $A(B + C) = AB + AC$	15. $A + BC = (A + B)(A + C)$
De Morgan	16. $\overline{A + B} = \overline{A}\,\overline{B}$	17. $\overline{AB} = \overline{A} + \overline{B}$

Espressioni Booleane

- Le **variabili booleane** sono espresse usando le lettere $X, Y, Z, A, B, C, \ldots$. Si suppone che le variabili booleane prendano un valore in $\{0, 1\}$.
  - Abbiamo appena utilizzato le variabili booleane nel dare le proprietà dell'Algebra di Boole, che valgono qualunque sia il valore sostituito a A, B o C.
- A partire dalle costanti 0 e 1 e dalle variabili booleane, possiamo costruire **espressioni booleane** utilizzando le operazioni di addizione, moltiplicazione e negazione.
  - Esempi di espressioni booleane sono $A + BC$, $\overline{A + B + \overline{C}} + C$, etc.
- Una volta assegnato un valore di verità in $\{0, 1\}$ ad ognuna delle variabili in un espressione booleana, è possibile valutare l'espressione in corrispondenza di tale assegnamento.
  - E.g., $A + BC$
  - Se $A = 0$ $B = 1$ $C = 1$, allora $A + BC = 1$.

Tabelle di Verità

- Il valore di un'espressione in corrispondenza di tutti i possibili assegnamenti è riassunto nella sua **tabella di verità**.
- Una tabella di verità è costituita da due parti:
  - nella parte sinistra, vengono riportate tutte le combinazioni che possono essere assegnate alle variabili binarie;
  - nella parte destra, vengono riportati i valori assunti dall'espressione.
- E.g.,

$$AB + AC$$

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

Tabelle di Verità

- Due espressioni booleane (contenenti le stesse variabili booelane) si dicono **equivalenti** quando il valore delle rispettive tabelle di verità coincide in ogni riga.
- E.g., $AB + AC = A(B + C)$

$AB + AC$

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

$A(B + C)$

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

Manipolazioni Algebriche

- Un modo molto semplice per trasformare un'espressione booleana in un'altra espressione ad essa equivalente consiste nell'impiego delle proprietà algebriche dell'algebra booleana.
  - Se si dimostra che esiste una catena di uguaglianze (indotte dalle proprietà dell'algebra booleana) che permettono di riscrivere un'espressione in un'altra, allora le due espressioni sono equivalenti.
- In questo modo è possibile, in particolare, semplificare un'espressione booleana data in un'espressione più semplice (ma equivalente).
- Esempi:

$$AB + AC = A(B + C)$$

$$\begin{aligned} \overline{A}BC + \overline{A}B\overline{C} + AC &= \overline{A}B(C + \overline{C}) + AC \\ &= \overline{A}B \cdot 1 + AC \\ &= \overline{A}B + AC \end{aligned}$$

Funzioni Booleane

- Date n variabili booleane $A_1, \ldots, A_n$, una **funzione booleana** associa un assegnamento di verità per un'altra variabile F ad ogni assegnamento di verità alle variabili $A_1, \ldots, A_n$.
- Un modo esplicito per definire una funzione booleana consiste nel dare la sua tavola di verità, che elencherà il valore di F per ogni possibile valore di verità per $A_1, \ldots, A_n$.
- E.g., una funzione F definita sulle 3 variabili A, B e 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 assume il valore 1 quando:

$$A = 0 \quad B = 1 \quad C = 0$$
$$A = 1 \quad B = 1 \quad C = 0$$
$$A = 1 \quad B = 1 \quad C = 1$$

Funzioni Booleane

- Data una funzione booleana, il suo **complemento** è definito come la funzione booleana (sulle stesse variabili) che vale 0 quando la funzione di partenza vale 1 e, viceversa, vale 1 quando la funzione di partenza vale 0.
- E.g., la stessa funzione introdotta in precedenza:

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

  - $\overline{F}$ assume il valore 1 quando:

$$
\begin{array}{lll}
A = 0 & B = 0 & C = 0 \\
A = 0 & B = 0 & C = 1 \\
A = 0 & B = 1 & C = 1 \\
A = 1 & B = 0 & C = 0 \\
A = 1 & B = 0 & C = 1
\end{array}
$$

Dalle Espressioni alle Funzioni

- Ad ogni espressione booleana E può essere **associata** una funzione booleana F, che si indica spesso con:

$$F = E$$

- Il valore della funzione F su un assegnamento di verità alle variabili booleane in E sarà semplicemente il valore di E in corrispondenza di tale assegnamento.
- Esempi:

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

- In generale, espressioni booleane **diverse** possono corrispondere alla **stessa** funzione booleana.
  - Ad esempio, le tabella di verità di $F = AB + AC$ e di $F = A(B + C)$ sono uguali.
  - Più in generale, tutte le espressioni equivalenti corrispondono alla stessa funzione.

Dalle Funzioni alle Espressioni

- Un **letterale** è una variabile booleana oppure la negazione di una variabile booleana.
  - X, $\overline{X}$, Y, ...
- Data una funzione booleana, si può **sempre** costruire una espressione booleana corrispondente (tra le tante possibili).
- In particolare, si può procedere considerando ogni riga della tabella di verità relativa alla funzione di partenza.
- Si associa **ad ogni riga** il prodotto di letterali ad essa corrispondente.
- L'espressione che cerchiamo sarà una **somma di prodotti** di letterali. Se la funzione vale 1 in una certa riga, allora il corrispondente prodotto di letterali farà parte della somma. Viceversa, se la funzione vale 0 in una certa riga, allora il corrispondente prodotto di letterali non farà parte della somma.
- In questo modo si possono ottenere espressioni booleane molto complesse...
  - ... che possono però essere semplificate tramite l'uso delle proprietà.

Dalle Funzioni alle Espressioni

- Consideriamo la funzione booleana introdotta in precedenza.
- In corrispondenza di ogni riga della tabella di verità, scriviamo il prodotto di letterali:

A	B	C	F	
0	0	0	0	$\overline{A}\ \overline{B}\ \overline{C}$
0	0	1	0	$\overline{A}\ \overline{B}\ C$
0	1	0	1	$\overline{A}\ B\ \overline{C}$
0	1	1	0	$\overline{A}\ B\ C$
1	0	0	0	$A\ \overline{B}\ \overline{C}$
1	0	1	0	$A\ \overline{B}\ C$
1	1	0	1	$A\ B\ \overline{C}$
1	1	1	1	$A\ B\ C$

- Le righe in cui F vale 1 sono la terza, la penultima e l'ultima. Possiamo quindi concludere che l'espressione booleana che ci interessa è

$$\overline{A}B\overline{C} + AB\overline{C} + ABC$$

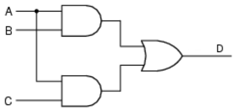
- Utilizzando le proprietà dell'algebra booleana otteniamo:

$$\begin{aligned} F &= \overline{A}B\overline{C} + AB\overline{C} + ABC = \overline{A}B\overline{C} + AB \\ &= B(\overline{A}\ \overline{C} + A) \end{aligned}$$

Dai Circuiti alle Funzioni

- Ad ogni uscita di ogni rete combinatoria corrisponde una funzione booleana.
  - Prima di tutto associamo n variabili booleane $A_1, \ldots, A_n$ agli n ingressi e una variabile F all'uscita che ci interessa.
  - Per ogni assegnamento di valori di verità alle variabili booleane $A_1, \ldots, A_n$, il corrispondente valore di F sarà quello ottenuto valutando il circuito.
  - In questo modo, si possono associare funzioni booleane ad ogni uscita del circuito. Diremo che il circuito in questione **implementa** tali funzioni booleane.
- Una volta costruita una funzione, si potrà poi ricavare un'espressione booleana

Dai Circuiti alle Funzioni: Esempio



$\Rightarrow$

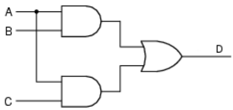
A	B	C	D
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$\Downarrow$

$$A\overline{B}C + AB\overline{C} + ABC$$

- Associamo 3 variabili booleane A, B, C agli ingressi.
- Associamo 1 variabile booleana D all'uscita.
- La tabella di verità è ottenuta valutando il circuito.
- La funzione definita dalla tabella da l'espressione booleana.

Dai Circuiti alle Funzioni: Esempio



$\Rightarrow$

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

$\Downarrow$

$$A\overline{B}C + AB\overline{C} + ABC$$

- Due circuiti si dicono **equivalenti** se implementano la stessa funzione.

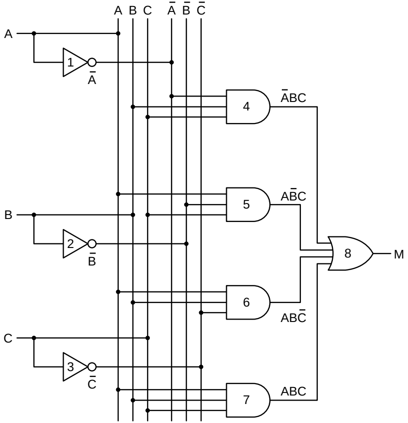
Dalle Espressioni ai Circuiti

- Ad ogni sequenza di espressioni booleane sulle stesse variabili corriponde un circuito.
  - Data un'espressione booleana E contenente le variabili booleane $A_1, \ldots, A_n$, un circuito con n entrate e un'uscita si può costruire seguendo la struttura di E.
  - La funzione implementata dal circuito sarà la funzione corrispondente ad E.
  - Partendo da una sequenza di espressioni, si potrà costruire un **singolo** circuito con più uscite.
- Per ottenere le espressioni booleane di partenza, si può procedere applicando ad altrettante funzioni booleane la procedura vista in precedenza.
  - In questo modo, si può passare da n funzioni booleane su m variabili ad un circuito con m ingressi ed n uscite.

Dalle Funzioni ai Circuiti: Esempio

A	B	C	M
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

(a)



(b)

$\Downarrow$ $\Uparrow$

$$\overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$