Informatica Generale
03 - Algebra di Boole e Circuiti

- **Cosa vedremo:**
  - Algebra di Boole
  - Proprietà degli operatori AND, OR e NOT
  - Semplificazione di espressioni booleane
  - Forma normale e circuiti AND-OR
  - *Riduzione di circuiti e di espressioni booleane*
    - *usando le leggi dell'algebra di Boole*
    - *usando Mappe di Karnaugh*
  - *Altri operatori universali: NAND e NOR*

Algebra di Boole

- Insieme di regole algebriche della logica binaria che stanno alla base del funzionamento dei calcolatori
- È costituita da:
  - un insieme di **variabili booleane** A,B,C,... che possono assumere solo i valori **1** (**vero**) o **0** (**falso**).
  - un insieme di **funzioni** (**operazioni**) che operano su valori booleani di input e danno valori booleani di output
  - un insieme di **leggi** (**assiomi**) che definiscono le proprietà delle funzioni.

Algebra di Boole

Le tre funzioni principali sono:

- AND (*): congiunzione, “e”:

  A*B (AB) è vera se sia A sia B sono vere

- OR (+): disgiunzione, “oppure”:

  A+B è vera se almeno uno tra A e B è vero

- NOT (¯ oppure ¬): negazione:

  Ā (o **¬A**) è vera se A è falsa

Tavole di verità

- Le tavole di verità servono a visualizzare i valori assunti dalle funzioni a partire da tutti i possibili valori delle variabili.

A	B	A*B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

A	$\bar{A}$
0	1
1	0

Tavole di verità

- Ogni funzione booleana è equivalente a un'espressione booleana costruita con variabili, **AND**, **OR** e **NOT**: possiamo "leggere" la tavola come un'espressione booleana che descrive l'output in base all'input.
- Esempio:

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

A XOR B =

(¬A * ¬B * 0) +

(¬A * B * 1) +

(A * ¬B * 1) +

(A * B * 0)

Proprietà delle funzioni logiche (leggi dell'algebra booleana)

- proprietà **commutativa**
  - $A * B = B * A$ $A + B = B + A$
- proprietà **associativa**
  - $(A * B) * C = A * (B * C)$ $(A + B) + C = A + (B + C)$
- proprietà **distributiva**
  - $A * (B + C) = (A * B) + (A * C)$ $\mathbf{A + (B * C) = (A + B) * (A + C)}$
- leggi di **idempotenza** e **complementazione**
  - $A + 0 = A$ $\mathbf{A * 0 = 0}$ $A + 1 = 1$ $\mathbf{A * 1 = A}$ $A + A = A$
  - $A * A = A$ $\neg(\neg A) = A$ $A + \neg A = 1$ $A * \neg A = 0$
- leggi di **De Morgan**
  - $\neg(A + B) = \neg A * \neg B =$ $\neg(A * B) = \neg A + \neg B$

Uso delle leggi per semplificare espressioni booleane

A XOR B =

$(\neg A * \neg B * 0) + (\neg A * B * 1) + (A * \neg B * 1) + (A * B * 0) =$

$[X * 0 = 0]$ $[X * 0 = 0]$

$0 + (\neg A * B * 1) + (A * \neg B * 1) + 0 =$

$[X + 0 = X]$ $[X * 1 = X,\ 2\ volte]$ $[X + 0 = X]$

$(\neg A * B) + (A * \neg B) =$

$(\neg A * B) + (A * \neg B)$

- Quindi, partendo dalla tavola, basta considerare solo le combinazioni degli input per cui la funzione vale 1.

Uso delle leggi per semplificare espressioni booleane

(X AND (NOT Y)) OR ((NOT X) XOR (NOT Y)) =

[espr di XOR]

(X * ¬Y) + ((¬¬X * ¬Y) + (¬X * ¬¬Y)) =

[¬¬A = A, 2 volte] [assoc di +]

(X * ¬Y) + (X * ¬Y) + (¬X * Y) =

[A + A = A]

(X * ¬Y) + (¬X * Y) [= X XOR Y!!]

- Ogni espressione booleana può essere ridotta in una forma canonica (**forma normale disgiuntiva, AND-OR**), che corrisponde a un circuito con tre livelli di porte

Esercizi

- Per ciascuna delle seguenti funzioni scrivere tavole e espressioni booleane equivalenti con AND-OR-NOT, e disegnare il circuito che la calcola:

  1) Bit di parità: dati A, B e C, vale 1 se la somma di 1 in input è dispari, vale 0 altrimenti.

  2) Contatore: dati A, B e C, calcola la somma binaria degli 1 in input.

  3) Sommatore: dati due numeri binari da due cifre ciascuno, ne esegue la somma.

     - Quante funzioni servono per 2) e 3) ?