Logica Booleana e porte logiche

Introduzione

La logica booleana è una forma di logica che utilizza solo due valori, **vero** e **falso**. Importante ricordare che, in informatica:

1 = VERO
0 = FALSO

La logica booleana viene utilizzata in molte applicazioni informatiche, come ad esempio nella progettazione di circuiti elettronici, nella programmazione e nei diagrammi di flusso.

Uno strumento fondamentale per le porte logiche sono le **Truth Tables o Tabelle di verità**.
Le tabelle di verità sono strumenti utilizzati in logica booleana per rappresentare e analizzare il comportamento di funzioni booleane. Sono costituite da una tabella che mostra tutte le possibili combinazioni di input per una funzione booleana, insieme al corrispondente output.

AND, OR e NOT

Le operazioni booleane più comuni sono l'AND, l'OR e il NOT. Vediamo di seguito come funzionano e come vengono rappresentati tramite porte logiche.

1) **AND**
   L'AND è un'operazione booleana che restituisce il valore vero solo se entrambi i valori di ingresso sono veri. Altrimenti restituisce falso. L'AND viene rappresentato dalla porta logica AND, che ha due ingressi e un'uscita.

   Esempio: A = 1, B = 0 allora l'operazione A AND B = 0

   Di seguito la tabella di verità della porta AND

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

2) **OR**
   L'OR è un'operazione booleana che restituisce il valore vero se almeno uno dei valori di ingresso è vero. Restituisce falso solo se entrambi i valori di ingresso sono falsi. L'OR viene rappresentato dalla porta logica OR, che ha due ingressi e un'uscita.

   Esempio: A = 1, B = 0 allora l'operazione A OR B = 1

   Di seguito la tabella di verità della porta OR

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

3) NOT

Il NOT è un'operazione booleana che inverte il valore di ingresso. Restituisce vero se il valore di ingresso è falso, e viceversa. Il NOT viene rappresentato dalla porta logica NOT, che ha un ingresso e un'uscita.

Esempio: A = 1 allora l'operazione NOT A = 0

Di seguito la tabella di verità della porta NOT

A	***B***	***NOT A***	***NOT B***
0	*0*	***1***	***1***
0	*1*	***1***	***0***
1	*0*	***0***	***1***
1	*1*	***0***	***0***

Espressioni Logiche / Booleane

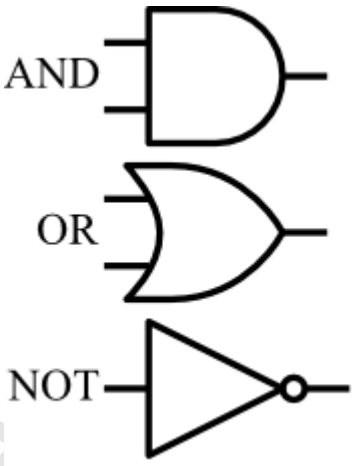
Le espressioni booleane/logiche sono un tipo di espressione matematica che coinvolgono solo due valori possibili, ovvero vero e falso.
Quelle che abbiamo visto noi sono definite dall'insieme delle porte logiche AND, OR e NOT.
Alcuni esempi di espressioni booleane:

(A AND B) OR C
(C OR B) AND (A OR B)
B OR (A AND (C OR D))
Etc..

Le possibilità sono pressochè infinite, dalle espressioni logiche è possibile ricavare i circuiti, utilizzando le porte logiche.

Circuiti Logici

Di seguito sono disegnate le rappresentazioni delle porte logiche che sono state definite prima:

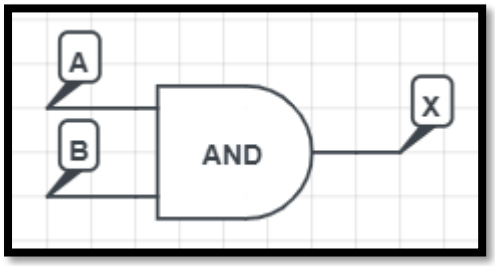


Come potete vedere:

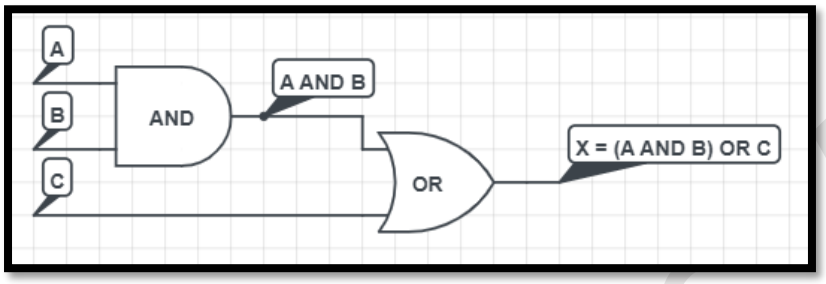
- La **AND** ha due ingressi e un'uscita. Questo perché, come visto prima, la AND richiede due input e prevede un solo output
- La **OR** ha due ingressi e un'uscita. Questo perché, come visto prima la OR richiede due input e prevede un solo output
- La **NOT** ha un ingresso e un'uscita. Questo perché, come visto prima, la NOT richiede un solo input e un solo output.

Come Facciamo ad unire queste porte logiche per fare circuiti piu complessi? Semplice, unendole e rappresentando quello che ci viene chiesto.

Prendiamo ad esempio il circuito semplicissimo X = *A AND B*. Come si rappresenta? Con una banale porta AND:

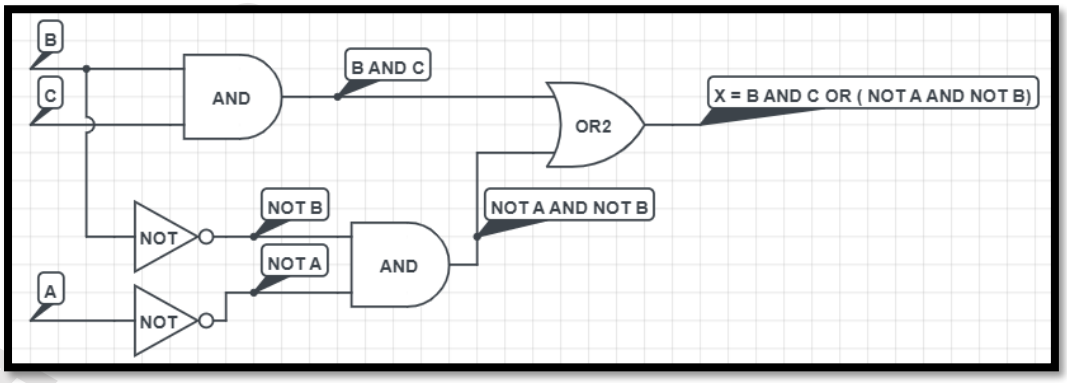


Adesso, complichiamo un po' le cose e proviamo ad aggiungere una OR e quindi creare il circuito rappresentato da *X=(A AND B) OR C*



Importante: Notiamo come le parentesi siano importanti, se avessi scritto solo A AND B OR C, come sarebbe dovuto essere il circuito? Non lo sappiamo, perché è ambiguo, dobbiamo mettere A in AND con B e poi fare OR con C oppure fare A in AND con l'OR tra B e C? Per questa ragione sono fondamentali le parentesi, ci permettono di capire l'ordine delle operazioni. Per facilitarvi la comprensione, pensate come fate in matematica: Le parentesi servono a spiegarvi quale operazione fare prima. Scrivere "5*2+4" è diverso da scrivere "5*(2+4)", la prima espressione fa come risultato 14, la seconda espressione fa come risultato 30!

Aggiungiamo ancora un passo di difficoltà, proviamo ora a fare questo circuito: *B AND C OR (NOT A AND NOT B)*

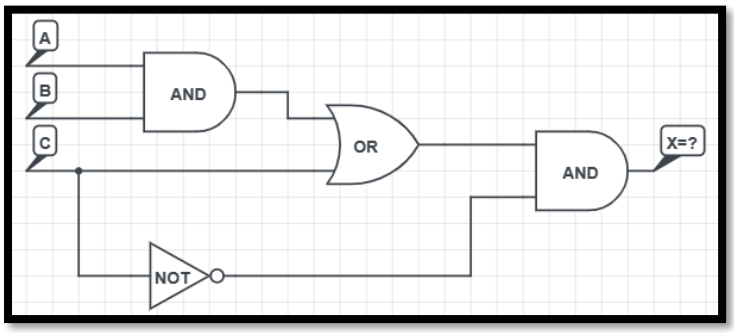


Importante: *Come potete vedere, nell'espressione B AND C OR (NOT A AND NOT B), la B è presente due volte. Non significa però che ci siano due input di B diversi. Posso ripetere la lettera di un input quante volte voglio, ma si riferirà sempre e solo ad uno stesso input. Osservate il disegno e vedete che ho messo solo una volta la lettera B e poi, per fare la NOT B, ho semplicemente fatto una linea che parte dalla B e va dentro la NOT. In sintesi quindi, non scrivete piu di una volta una lettera come INPUT*

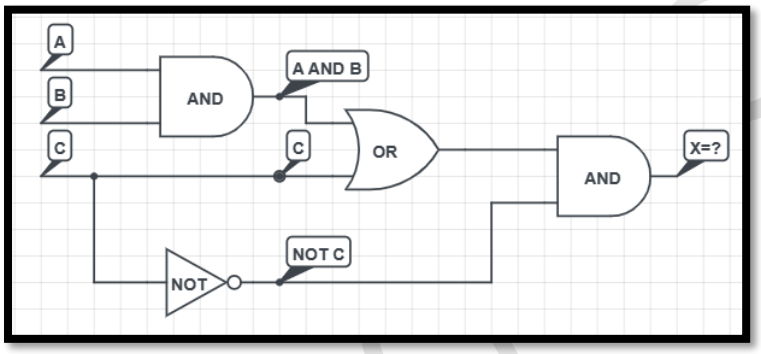
Dal circuito logico all'espressione booleana

Una volta capito come fare un circuito data un'espressione, è importante anche capire come ottenere un'espressione dato un circuito.
Per quale ragione dico che è importante? Perché significa che avete capito bene l'argomento e che quindi sapete manipolare porte logiche e logica booleana senza problemi, elemento fondamentale per un informatico / elettronico.

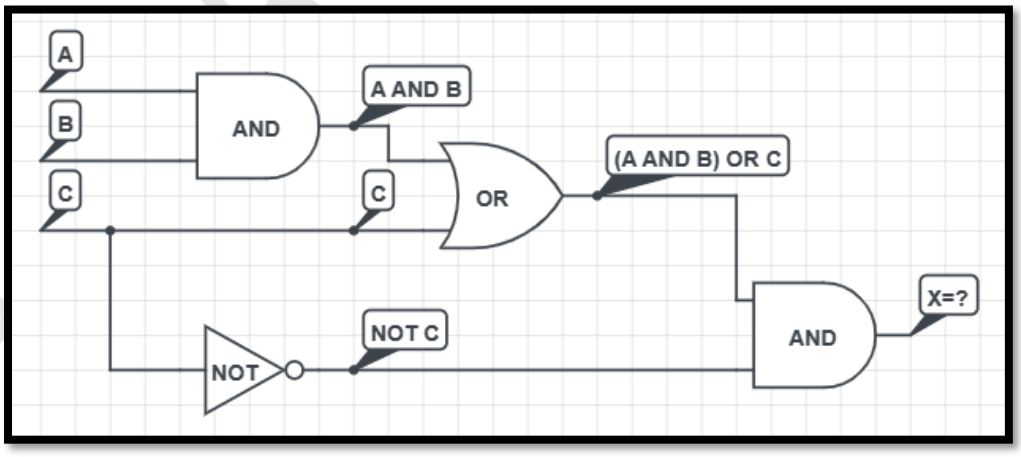
Consideriamo il seguente circuito



Da dove si parte? Il mio consiglio è quello di partire da sinistra e capire cosa succede agli input iniziali. Come vediamo A e B finiscono dentro una AND mentre l'input C finisce dentro una OR e una NOT. Dopo aver valutato queste cose, scriviamole sul grafico:



Adesso, continuiamo sempre da sinistra verso destra per arrivare all'ingresso della AND finale e scriviamo nel circuito:



Infine, pensiamo, cosa entra nell'ultima AND? **NOT C** e **(A AND B) OR C** quindi il risultato finale di questo circuito sarà **X = (NOT C) AND ((A AND B) OR C)**