Architettura degli Elaboratori - Introduzione

Andrea Malvezzi

19 Settembre, 2024

A cosa serve questo corso?

L'architettura degli elaboratori cerca di spiegare il funzionamento dei calcolatori dal punto di vista dell'hardware *(HW)*.

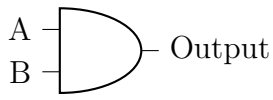
1 La composizione dei calcolatori

1.1 Le porte logiche

Un calcolatore è composto da migliaia di componenti, tra cui le **PORTE LOGICHE**.
Esistono diversi tipi di porte logiche, ma due di queste - la NAND e la NOR - nutrono di una peculiarità: queste sono dette "**UNIVERSALI**" in quanto combinandole tra loro è possibile produrre tutte le altre funzioni logiche (o *booleane*).

1.2 In dettaglio

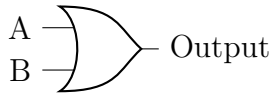
1.2.1 Porta logica AND



A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

La funzione logica AND restituisce VERO (*1*) soltanto quando entrambi i valori in ingresso sono VERO.

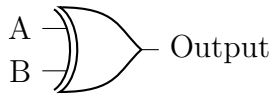
1.2.2 Porta logica OR



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

La funzione logica OR restituisce VERO quando almeno uno dei due valori in ingresso è VERO.

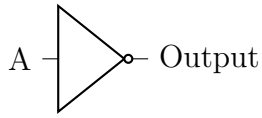
1.2.3 Porta loica XOR



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

La funzione logica XOR restituisce VERO quando soltanto uno dei due valori in ingresso è VERO.

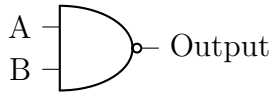
1.2.4 Porta logica NOT



A	Output
0	1
1	0

La funzione logica NOT "nega" effettivamente il valore entrante, invertendolo. Quindi da un valore vero se ne otterrà uno falso e viceversa.
Quando collocata in seguito ad altre porte logiche si indica con un piccolo cerchio.

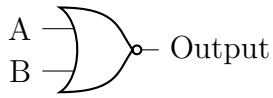
1.2.5 Porta logica NAND



A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

La funzione logica NAND è la porta AND seguita da una NOT, quindi basta invertire i risultati della sua tabella di verità.

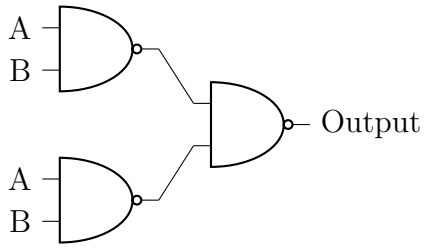
1.2.6 Porta logica NOR



A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

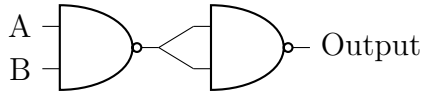
La funzione logica NOR, come la NAND, altro non è che una porta OR seguita da una NOT. Quindi basterà invertire i valori di verità della funzione logica OR per ottenere quelli della sua controparte negata.

1.2.7 La funzione logica OR con la porta NAND



Per verificare la correttezza del circuito, inserire i valori di verità desiderati e paragonare l'output con quello della tabella di verità della porta OR.

1.2.8 La funzione logica AND con la porta NAND



Per verificare la correttezza del circuito, inserire i valori di verità desiderati e paragonare l'output con quello della tabella di verità della porta AND.

1.3 Il principio di Astrazione/Implementazione

Per realizzare sistemi complessi partendo da queste porte si usa il **principio di "Astrazione/Implementazione"**:

- Astrazione: si frammenta il problema di partenza in problemi minori, più semplici da risolvere.
- Implementazione: si realizza la soluzione al problema di partenza combinando tra loro le soluzioni minori precedentemente escogitate.

2 Elaboratori multilivello

Un elaboratore può essere composto da più livelli e generalmente è corretto dire che i livelli superiori sono destinati alla rsoluzione di problemi mediante linguaggi adatti, mentre quelli inferiori sono improntati alla gestione delle risorse e alla comunicazione tra HW e SW.

Figure 1: Un esempio della struttura di un elaboratore a più livelli.

2.1 Livello 0: livello logico digitale

Qui si hanno le porte logiche e conseguentemente è dove avvengono le relative operazioni.

2.2 Livello 1: livello di microarchitettura

La microarchitettura è ciò che in un elaboratore gestisce il flusso di dati tra i vari componenti.
Può trattarsi sia di HW che di SW.

2.3 Livello 2: livello ISA(Instruction Set Architecture)

Funge da interfaccia da HW e SW.
Contiene le istruzioni che vengono eseguite nella microarchitettura.

2.4 Livello 3-4: O.S. e Assembly

Qui si programmano i livelli sottostanti, si gestiscono le risorse del sistema e si eseguono i programmi.

2.5 Livello 5: livello del linguaggio

Qui è dove si usano linguaggi più vicini agli esseri umani per risolvere problemi e fornire servizi. Il codice di questi linguaggi dovrà essere tradotto in linguaggio macchina per essere eseguito.
Vedi "Livello applicativo" nell'immagine 1.