Calcolatori Elettronici

Nicola Ferru

22 marzo 2022

Capitolo 1

Introduzione

1.1 Programma

1. Reti logiche;
2. Unità di memoria;
3. CPU - Set di Istruzioni;
4. CPU - ALU;
5. Unità di Input/Output.

In questo corso il linguaggio di programmazione che verrà utilizzato, sarà l'assembler per l'architettura Mips,

1.2 Rete logica

Rete Logica 1. *In elettronica una rete logica è un insieme di dispositivi interconnessi che realizza un'elaborazione ovvero una certa funzione logica. Possono essere di natura elettronica, nell'accezione più comune, ma potenzialmente anche di altra natura, se in grado di trasmettere e elaborare un segnale, come nella fotonica, eccetera. Esistono essenzialmente due tipi principali di reti logiche: le reti unilaterali e le reti bilaterali.*

1.3 Algebra di Bool

Primarie

AND rende il valore unitario solo nel caso in cui entrambe le variabili sono ad 1, in caso contrario rende 0, questo è molto utile in programmazione per controllare se i valori contenuti in due variabili sono uguali o due condizioni sono uguali. La formula matematica è $A \wedge B$. Questa operazione logica viene definita anche prodotto logico.
OR rende il valore unitario se almeno una delle due variabili è a 1, infatti, in questo caso vale la regola del uno, l'altro o tutti e due. La formula matematica è $A \vee B$. Questa operazione viene definita anche somma logica.
NOT nega il contenuto di una variabile quindi se il valore è pari a 1 il risultato sarà 0 e viceversa.

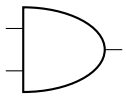

Figura 1.1: AND porta logica

a	b	risultato
0	0	0
0	1	0
1	0	0
1	1	1

Tabella 1.1: AND

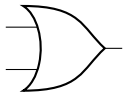

Figura 1.2: OR porta logica

a	b	risultato
0	0	0
0	1	1
1	0	1
1	1	1

Tabella 1.2: OR

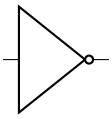

Figura 1.3: NOT porta logica

a	risultato
1	0
0	1

Tabella 1.3: NOT

Composte

NAND è la congiunzione negata di AND, $\overline{A \wedge B} = A \overline{\wedge} B$, come è normale che sia negando la porta AND l'unico caso in cui dara 0 sarà quando tutti e due i valori passati in ingresso saranno a 1.

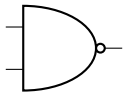

Figura 1.4: NAND porta logica

A	B	$A \wedge B$
0	0	1
0	1	1
1	0	1
1	1	0

Tabella 1.4: NAND

La negazione della disgiunzione $\neg(A \vee B) \equiv A \overline{\vee} B$, e l'unione delle congiunzioni $\neg A \wedge \neg B$ risultano così di

seguito:

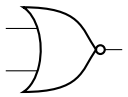

Figura 1.5: NOR porta logica

A	B	$A \vee B$	$A \overline{\vee} B$	$\neg A$	$\neg B$	$\neg A \wedge \neg B$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Tabella 1.5: NOR

La XOR è un operatore che è simile alla OR ma va ad escludere i valori uguali, quindi se A = B il risultato reso sarà 0.

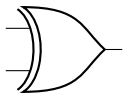

Figura 1.6: XOR porta logica

a	b	risultato
0	0	0
0	1	1
1	0	1
1	1	0

Tabella 1.6: XOR

Ovviamente come tutti le perte esiste la sua versione negata quindi esiste la XNOR.

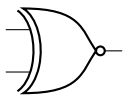

Figura 1.7: XNOR porta logica

a	b	risultato
0	0	1
0	1	0
1	0	0
1	1	1

Tabella 1.7: XOR

Sostanzialmente gli operatori che vengono più utilizzati nella programmazione sono AND, OR, NOT. Gli altri sono molto secondari e vengono utilizzati raramente. In alcuni manuali lo 0 diventa Falso e 1 diventa Vero. Diamo per assodato che tutte le funzioni logiche possono essere eseguite con gli operatori AND, OR e NOT, nulla di più.

1.3.1 Leggi legata alle porte logiche

L'algebra di bool è un algebra complessa, quindi è giusto che per semplificare potenziali calcoli siano state inventate delle regole, che consentono di individuare dei casi e di risparmiare parecchio tempo e fatiche. Ovviamente essendo AND e OR i principali operatori logici, le formule sono tarati su quelli.

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\bar{A} = 0$	$A + \bar{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$A(B + C) = AB + AC$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absirption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$

Tabella 1.8: Leggi legate alle porte Logiche AND e OR

1.4 Rappresentazione grafica dei latch

Latch 1. *In elettronica digitale, il latch (letteralmente "serratura", "chiavistello") è un circuito elettronico bistabile, caratterizzato quindi da almeno due stati stabili, in grado di memorizzare un bit di informazione nei sistemi a logica sequenziale asincrona. Il latch modifica lo stato logico dell'uscita al variare del segnale di ingresso, mentre il flip-flop, basato sulla struttura del latch, cambia lo stato logico dell'uscita solamente quando il segnale di clock è nel semiperiodo attivo.*

l latch costituisce l'elemento base di tutti i circuiti sequenziali ma trova anche delle applicazioni come elemento singolo, ad esempio per eliminare i rimbalzi dei componenti elettromeccanici come pulsanti, interruttori e commutatori.

Spesso i latch sono usati in gruppi, alcuni dei quali hanno nomi speciali come il quad latch (gruppo di quattro) e l'octal latch (gruppo di otto). Molti tipi di display a 7 segmenti o alfanumerici contenenti il circuito di decodifica, dispongono di un pin collegato a questo circuito, il quale permette, tramite il cambio del livello logico, di "stoppare" il valore in quel momento visualizzato.