PRESTAZIONI

Le prestazioni si valutano in maniera diversa a seconda dell'applicazione, le grandezze da considerare sono :

- Tempo di risposta o di esecuzione
- Throughput

Per una macchina qualsiasi vale la relazione :

$$\text{prestazioni }(X) = \frac{1}{\text{tempo di esecuzione di } X}$$

Se per due macchine X e Y, le prestazioni di X sono migliori di Y si ha che :

prestazioni di X > prestazioni di Y

tempo di esecuzione di Y > tempo di esecuzione di X

Relazione tra le prestazioni di due macchine X e Y:

$$\frac{\text{prestazioni } X}{\text{prestazioni di } Y} = \frac{\text{tempo di esecuzione di } Y}{\text{tempo di esecuzione di } X} = n$$

Tempo di CPU –> tempo speso dalla CPU nell'eseguire un determinato programma.

Tempo di CPU = Cicli di clock della CPU x Periodo (durata) di ciclo del clock

$$\text{Tempo di CPU} = \frac{\text{Cicli di clock della CPU}}{\text{frequenza di clock}}$$

CPI –> clock per istruzione, indica il numero medio di cicli di clock per istruzione

Cicli di clock della CPU = n. istruzioni del programma x CPI

quindi..

$$\text{Tempo di CPU} = \frac{\text{Cicli di clock della CPU}}{\text{frequenza di clock}} = \frac{\text{n. istruzioni del programma} \times \text{CPI}}{\text{frequenza di clock}}$$

Tempo di CPU = Cicli di clock della CPU x Periodo (durata) di ciclo del clock

= n. istruzioni del programma x CPI x Periodo (durata) di ciclo del clock

ESEMPIO TEMPO DI CPU

Computer A: 2GHz clock, 10s CPU time

Designing Computer B

- Aim for 6s CPU time
- Can do faster clock, but causes 1.2 × clock cycles

How fast must Computer B clock be?

$$\overset{\text{Frequenza}}{\text{Clock Rate}_B} = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6s}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10s \times 2GHz = 20 \times 10^9$$

$$\overset{\text{Frequenza}}{\text{Clock Rate}_B} = \frac{1.2 \times 20 \times 10^9}{6s} = \frac{24 \times 10^9}{6s} = 4GHz$$

ESEMPIO CPI

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$

$$= I \times 2.0 \times 250ps = I \times 500ps$$

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$

$$= I \times 1.2 \times 500ps = I \times 600ps$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600ps}{I \times 500ps} = 1.2$$

Cicli di clock della CPU analizzando i diversi tipi di istruzioni :

$$\text{Cicli di clock della CPU} = \sum_{i=1}^{n} (\text{CPI}_i \, \text{C}_i)$$

LEGGENDA:

C(i)–> numero medio dei cicli per le istruzioni della classe i

CPI(i)–> numero medio dei cicli per le istruzioni della classe i

n–> numero delle classi

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5

Legge di Amdhal

$$\text{Tempo migliorato} = \frac{\text{Tempo affected}}{\text{Fattore di miglioramento}} + \text{Tempo unaffected}$$

$$\text{Speedup} = \frac{\text{Tempo vecchio}}{\text{Tempo migliorato}}$$

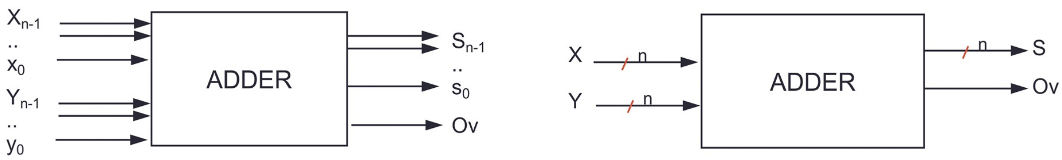
IPC → Istruzioni per ciclo, inverso di CPI

$$ICP = \frac{\text{n. di istruzioni programma}}{\text{n. cicli di clock della CPU}}$$

RETI LOGICHE

Definizione -> Una rete logia è un sistema digitale avente:
- n -> segnali binari in ingresso
- m -> segnali binari in uscita

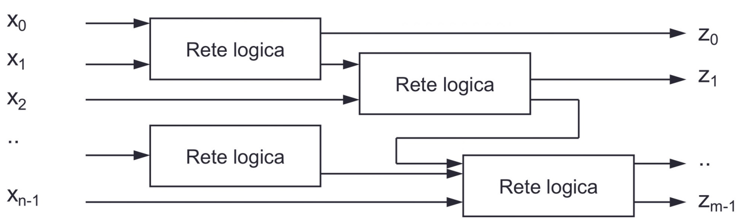
I segnali di ingresso e di uscita possono essere singoli(un solo bit) oppure composti (n bit)



Proprietà delle reti logiche

Proprietà di interconnessione:

- L'interconnessione di più reti logiche, aventi per ingresso segnali esterni o uscite di altre reti logiche e per uscite segnali di uscita esterne o ingressi di altre reti logiche, è ancora una rete logica

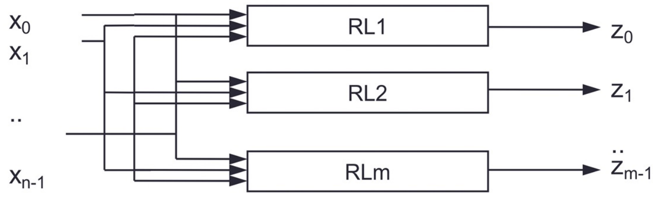


Proprietà di decomposizione:

- Una rete logica complessa può essere decomposta in reti logiche più semplici (fino all'impiego di soli blocchi o gate elementari)

Proprietà di decomposizione in parallelo:

- Una rete logica a m uscite può essere decomposta in m reti logiche ad 1 uscita, aventi ingressi condivisi



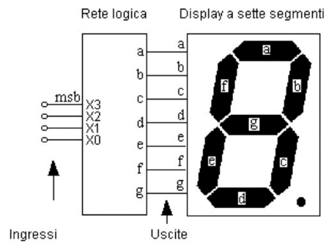
Rete Combinatoria vs Rete Sequenziale

- Rete COMBINATORIA -> ogni segnale di uscita dipende solo dai valori degli ingressi in quell'istante. E' una rete senza memoria non ha quindi stato e non ricorda gli ingressi precedenti.

Esempio di rete combinatoria

Conversione di valori BCD su display a sette segmenti

- Descrizione comportamentale (a parole): progettare una rete logica che permette la visualizzazione su un display a sette segmenti di un valore in codice BCD.
- **Codifica BCD:** impiego di 4 cifre binarie per la rappresentazione di un numero decimale da 0 a 9.
- ***Es***: *15 decimale*
  1111 binario
  0001 0101 BCD



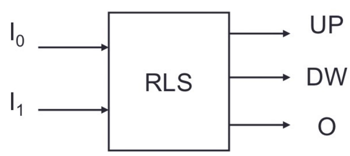
- L'uscita Z={a,b,…g} dipende in ogni istante dalla configurazione degli ingressi $\{x_3,x_2,x_1,x_0\}$

- Rete SEQUENZIALE -> ogni segnale di uscita dipende dai valori degli ingressi in quell'istante e dai valori che gli ingressi hanno assunto negli stati precedenti. E' una rete con memoria, ha uno STATO che riassume la sequenza degli ingressi precedenti.

Esempio di rete sequenziale

Progettare la rete logica di gestione di un ascensore.

- La rete ha tre uscite UP, DW e O. UP, DW indicano le direzioni su e giù mentre O vale 1 se la porta deve essere aperta e 0 altrimenti. La rete ha come ingresso due segnali che indicano il piano {0,1,2,3} corrispondente al tasto premuto. Per calcolare l'uscita è necessario conoscere il piano corrente che indica lo stato interno.



Descrizione delle reti combinatorie

Tabella di verità:

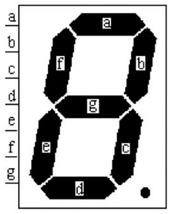
- tabella che associa tutte le possibili combinazioni degli ingressi alle corrispondenti configurazioni delle uscite e indica esaustivamente il comportamento della rete logica
  - Se la rete combinatoria ha n ingressi e m uscite, allora la tabella di verità ha (n+m) colonne e 2n righe

- Si dicono **COMPLETAMENTE SPECIFICATE** se ogni valore della tabella assume il valore logico di vero o falso (1, 0)

- Si dicono **NON COMPLETAMENTE SPECIFICATE** se contengono condizioni di indifferenza. Si verifica in due casi:

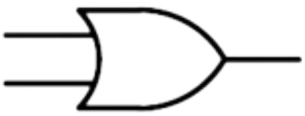
  C.1) *se alcune configurazioni di ingressi sono vietate*

***Es**: conversione BCD 7 segmenti*



x3	*x2*	*x1*	*x0*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	-	-	-	-	-	-	-
1	0	1	1	-	-	-	-	-	-	-
1	1	0	0	-	-	-	-	-	-	-
1	1	0	1	-	-	-	-	-	-	-
1	1	1	0	-	-	-	-	-	-	-
1	1	1	1	-	-	-	-	-	-	-

Rappresentazione grafica e logica di funzioni combinatorie



OR

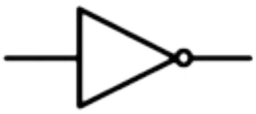
X_0X_1	Z
00	0
01	1
10	1
11	1

Vale 1 se e solo se tutti gli ingressi valgono 1

AND

X_0X_1	Z
00	0
01	0
10	0
11	1

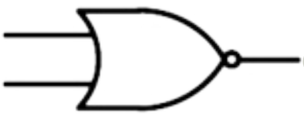
Vale 1 se e solo se tutti gli ingressi valgono 1



NOT

X	Z
0	1
0	1
1	0
1	0

Restituisce il valore inverso dell'ingresso



NOR

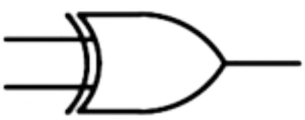
X_0X_1	Z
00	1
01	0
10	0
11	0

Vale 1 se e solo se ne x_0 ne x_1 valgono 1

NAND

X_0X_1	Z
00	1
01	1
10	1
11	0

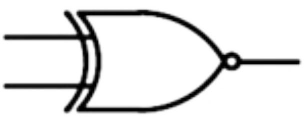
Vale 0 se e solo se ne x_0 ne x_1 valgono 0



XOR

X_0X_1	Z
00	0
01	1
10	1
11	0

Vale 1 se e solo se x_0 e x_1 valgono 1 ma non entrambi



XNOR

X_0X_1	Z
00	1
01	0
10	0
11	1

Vale 1 se e solo se x_0 e x_1 sono uguali

Algebra di Boole

L'algebra di Boole è un sistema matematico che descrive funzioni di variabili binarie

OR	AND	NOT
P1) $0 + 0 = 0$	P5) $0 \cdot 0 = 0$	P9) $0' = 1$
P2) $0 + 1 = 1$	P6) $0 \cdot 1 = 0$	P10) $1' = 0$
P3) $1 + 0 = 1$	P7) $1 \cdot 0 = 0$	
P4) $1 + 1 = 1$	P8) $1 \cdot 1 = 1$	

Funzioni Booleane

- Esiste corrispondenza 1:1 tra una tabella della verità e funzione Booleana.

$f(x2,x1,x0): B \times B \times B \rightarrow B$

x2	x1	x0	f(x2,x1,x0)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- **Complementazione**: A complementato si indica come A' oppure $\overline{A}$.
- Il simbolo • del prodotto logico viene spesso omesso.

Teorema 1 –> Ogni espressione di n variabili descrive una funzione completamente specificata che può essere valutata attribuendo ad ogni variabile un valore assegnato.

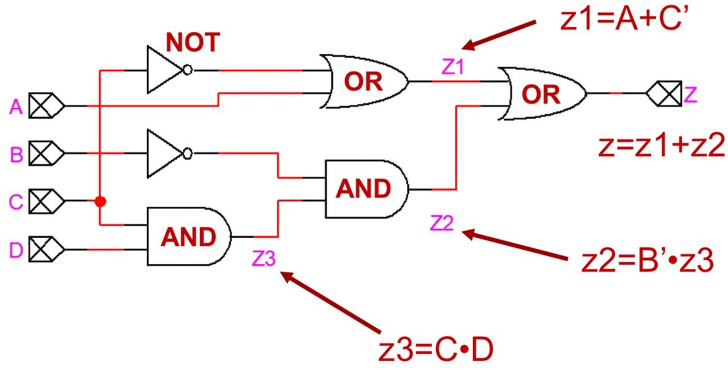
x2	x1	x0	f(x2,x1,x0)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

↓

$$f(x2,x1,x0) = x2'x1'x0 + x2x1'x0' + x2x1x0$$

Da uno schema logico ad una funzione booleana

Esercizio: Eseguire l'analisi del seguente schema



z=z1+z2
z1=A+C'
z2=B'•z3
z3= C•D

z=z1+z2
z1=A+C'
z2=B'•C•D

z=A+C'+B'•C•D

Teoremi dell' algebra di Boole

Teor. di Identità

- (T1) $X + 0 = X$
  (T1') $X \cdot 1 = X$

Teor. di Elementi nulli

- (T2) $X + 1 = 1$
  (T2') $X \cdot 0 = 0$

Idempotenza

- (T3) $X + X = X$
- (T3') $X \cdot X = X$

Involuzione

- (T4) $(X')' = X$

Complementarietà

- (T5) $X + X' = 1$
- (T5') $X \cdot X' = 0$

Proprietà commutativa

- (T6) $X + Y = Y + X$
- (T6') $X \cdot Y = Y \cdot X$

Proprietà associativa

- (T7) $(X + Y) + Z = X + (Y + Z) = X + Y + Z$
- (T7') $(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) = X \cdot Y \cdot Z$

Proprietà di assorbimento

- (T8) $X + X \cdot Y = X$
- (T8') $X \cdot (X + Y) = X$

Proprietà distributiva

- (T9) $X \cdot Y + X \cdot Z = X \cdot (Y + Z)$
- (T9') $(X + Y) \cdot (X + Z) = X + Y \cdot Z$

Proprietà della combinazione

- (T10) $(X + Y) \cdot (X' + Y) = Y$
- (T10') $X \cdot Y + X' \cdot Y = Y$

Proprietà del consenso

- (T11) $(X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z)$
- (T11') $X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$

Teorema di De Morgan

- (T12) $(X + Y)' = (X' \bullet Y')$
- (T12') $(X \bullet Y)' = (X' + Y')$