Calcolatori elettronici

definizioni, formule ed esempi

Pietro Barbiero

Quest'opera contiene informazioni tratte da wikipedia (http://www.wikipedia.en) e dalle dispense relative al corso di Calcolatori Elettronici tenuto dal professor Matteo Sonza Reorda del Dipartimento di Automatica e Informatica del Politecnico di Torino (IT).

Indice

Parte I

Progetto di circuiti logici

Capitolo 1

Introduzione

1.1 Sistemi digitali

1.1.1 Sistema

Un sistema è un insieme di elementi interconnessi tra di loro e/o con l'esterno che si comporta come un tutt'uno

1.1.2 Funzione di un sistema

La funzione di un sistema è determinata da: le funzioni svolte da ciascun componente; il modo con cui i componenti sono connessi

1.1.3 Sistema digitale

Un sistema digitale è un sistema che manipola in modo numerico segnali numerici di ingresso (che rappresentano dati o informazioni) per produrre segnali di uscita numerici (dati o informazioni)

1.1.4 Ciclo di vita di un prodotto

Il ciclo di vita di un prodotto si compone delle seguenti fasi:

- determinazione e descrizione del comportamento del sistema, dei sui vincoli e delle sue specifiche: velocità, consumo, affidabilità durata, costo (progetto, produzione, manutenzione, ...)
- attività di progetto il cui scopo è la costruzione/individuazione di un modello utilizzabile nella fase di produzione (in questa fase è possibile eseguire test e simulazioni per verificare il corretto funzionamento del modello)
- fabbricazione del prodotto
- operatività e manutenzione

1.1.5 Caratteristiche di un progetto

Le caratteristiche di un progetto sono:

- ogni componente deve essere il più indipendente possibile (in modo da poter essere sviluppato/modificato in modo indipendente)
- di ogni modulo/componente si conoscono le funzioni/caratteristiche e non i dettagli implementativi

1.1.6 Livelli di progetto di sistemi elettronici

I livelli di progetto di sistemi elettronici sono (top-down):

- sistema (CPU, processori I/O, memorie); unità di dato: blocchi di parole
- register (registri, reti combinatorie e sequenziali semplici); unità di dato: parole
- porte (porte logiche, flip-flop); unità di dato: bit

1.1.7 Tipologie di sistemi digitali

1.1.7.1 Sistema combinatorio

Un sistema combinatorio è un sistema digitale i cui valori di uscita sono funzione solo dei valori di ingresso in quell'istante

1.1.7.2 Sistema sequenziale

Un sistema sequenziale è un sistema digitale i cui valori di uscita sono funzione sia dei valori di ingresso in quell'istante sia dei valori di ingresso degli istanti precedenti

1.2 Algebra booleana

1.2.1 Variabile booleana

Una variabile booleana è una variabile che può assumere solo due valori (vero-falso, 1-0)

$$x \in B = \{0, 1\} \tag{1.1}$$

1.2.2 Tavola di verità

Una tavola di verità è una tabella matematica che per ogni combinazione di valori booleani di ingresso specifica i corrispondenti valori booleani di uscita

a	b	c	x_0	x_1
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

(1.2)

1.2.3 Funzione booleana

Una funzione booleana è una funzione che associa a n variabili booleane un valore booleano

$$f(x_1, \ldots, x_n) : B^n \to B = \{0, 1\} \tag{1.3}$$

1.2.4 Assiomi booleani (o postulati di Huntington) e leggi

Se $a, b \in B = \{0, 1\}$ allora:

- chiusura: $\implies ab \in B \quad \wedge \quad a + b \in B$
- identità: $a + 0 = a \quad \wedge \quad a \cdot 1 = a$
- commutatività: $a + b = b + a \quad \wedge \quad ab = ba$
- distributività: $a(b + c) = ab + ac \quad \wedge \quad a + (bc) = (a + b)(b + c)$
- inverso: $a + \bar{a} = 1 \quad \wedge \quad a\bar{a} = 0$
- associatività: $a + (b + c) = (a + b) + c \quad \wedge \quad a(bc) = (ab)c$
- idempotenza: $a + a = a \quad \wedge \quad aa = a$
- leggi di De Morgan: $\overline{a + b} = \bar{a}\bar{b} \quad \wedge \quad \overline{ab} = \bar{a} + \bar{b}$
- involuzione: $\bar{\bar{a}} = a$

1.2.5 Terminologia

1.2.5.1 Letterale

Un letterale è una variabile booleana (vera o falsa)

$$\dot{x}_i \in B \tag{1.4}$$

1.2.5.2 Minterm

Un minterm è un prodotto di tutti i letterali affermati o negati di una funzione booleana la cui combinazione fornisce un valore di uscita 1

$$\prod \dot{x}_i = 1 \tag{1.5}$$

1.2.5.3 Maxterm

Un maxterm è una somma di letterali affermati o negati di una funzione booleana la cui combinazione fornisce un valore di uscita 0

$$\sum \dot{x}_i = 0 \tag{1.6}$$

1.2.5.4 Cubo

Un cubo è un prodotto di letterali

$$\dot{x}_i \dots \dot{x}_k \tag{1.7}$$

1.2.5.5 Don't care

Un don't care è una combinazione di letterali che non prevede alcun valore in uscita

$$- \tag{1.8}$$

1.2.6 Forme canoniche di funzioni booleane

Le forme canoniche di espressioni booleane sono modelli di rappresentazione di un espressioni booleane ricavabili direttamente dalla tavola di verità

1.2.6.1 Somma di prodotti (o forma disgiuntiva)

Una somma di prodotti è una forma canonica di una funzione booleana costituita da somme di minterm

x_1	x_2	A	B
0	0	$\bar{A}\bar{B}$	
0	1	$\bar{A}B$	
1	0	$A\bar{B}$	
1	1	AB	

(1.9)

$$\begin{aligned} f(x_1, x_2, \dots, x_n) &= \sum_{i=1}^{n} \left(\prod_{j=1}^{n} \dot{x}_{ij} \right) = \\ &= (\bar{A}\bar{B}) + (\bar{A}B) + (A\bar{B}) + (AB) \end{aligned} \tag{1.10}$$

1.2.6.2 Prodotto di somme (o forma congiuntiva)

Un prodotto di somme è una forma canonica di una funzione booleana costituita da prodotti di maxterm

x_1	x_2	A	B
0	0	$A+B$	
0	1	$A+\bar{B}$	
1	0	$\bar{A}+B$	
1	1	$\bar{A}+\bar{B}$	

(1.11)

$$f(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \left(\sum_{j=1}^{n} \dot{x}_{ij} \right) = \\ = (A+B)(A+\bar{B})(\bar{A}+B)(\bar{A}+\bar{B}) \tag{1.12}$$

1.2.7 Funzione booleana minimale

Una funzione booleana minimale è una funzione minimale espressa in forma canonica in cui: il numero di prodotti (o somme) è minimo e nessun letterale può essere cancellato da un prodotto (o somma) senza modificare la funzione

1.2.8 Mappa di Karnaugh

Una mappa di Karnaugh è una rappresentazione grafica di una funzione booleana espressa in forma canonica

1.2.8.1 Procedura di sintesi di una funzione booleana

Per sintetizzare una funzione booleana minimale a partire da una funzione booleana espressa in forma canonica è necessario:

- costruire una mappa di Karnaugh corrispondente alla tavola di verità della funzione
- identificare il minimo insieme di cubi che coprono tutti gli 1 della mappa scegliendo quelli di dimensione massima (i cubi possono sovrapporsi)
- trasformare i cubi nella corrispondente espressione in forma di somma di prodotti

x_1	x_2	x_3	x_4	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	x
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	x
1	0	1	1	x
1	1	0	0	x
1	1	0	1	x
1	1	1	0	x
1	1	1	1	x

$\Longrightarrow$

$\frac{x_1x_2}{x_3x_4}$	00	01	11	10
00	0	0	x	0
01	0	0	x	1
11	1	0	x	x
10	0	x	x	x

$$\Longrightarrow f = x_1x_4 + \bar{x}_4x_3x_4 + x_2x_3\bar{x}_4 \qquad (1.13)$$

1.3 Porte logiche

1.3.1 Porta logica

Una porta logica è un circuito digitale in grado di implementare una particolare funzione logica di una o più variabili booleane

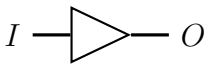
1.3.1.1 BUFFER

BUFFER è una porta logica che ha come valore di uscita 1 se e solo se la sua variabile booleana di ingresso x ha valore 1

$$f_{BUFFER} = x \tag{1.14}$$

x	f_{BUFFER}
0	0
1	1

(1.15)



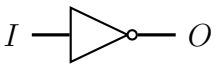
1.3.1.2 NOT

NOT è una porta logica che ha come valore di uscita 1 se e solo se la sua variabile booleana di ingresso x ha valore 0

$$f_{NOT} = \bar{x} \tag{1.16}$$

x	f_{NOT}
0	1
1	0

(1.17)



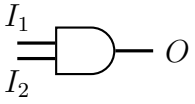
1.3.1.3 AND (porta di necessità)

AND è una porta logica che ha come valore di uscita 1 se e solo se tutte le variabili booleane di ingresso x_i hanno valore 1

$$f_{AND} = \prod x_i \tag{1.18}$$

x_1	x_2	f_{AND}
0	0	0
0	1	0
1	0	0
1	1	1

(1.19)



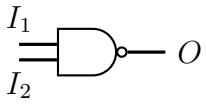
1.3.1.4 NAND

NAND è una porta logica che ha come valore di uscita 1 se e solo se le variabili booleane di ingresso x_i non hanno tutte valore 1

$$f_{NAND} = \prod \bar{x}_i \tag{1.20}$$

x_1	x_2	f_{NAND}
0	0	1
0	1	1
1	0	1
1	1	0

(1.21)



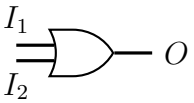
1.3.1.5 OR (porta di sufficienza)

OR è una porta logica che ha come valore di uscita 1 se e solo se almeno una variabile booleana di ingresso x_i ha valore 1

$$f_{OR} = \sum x_i \tag{1.22}$$

x_1	x_2	f_{OR}
0	0	0
0	1	1
1	0	1
1	1	1

(1.23)



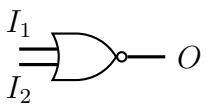
1.3.1.6 NOR

NOR è una porta logica che ha come valore di uscita 1 se e solo se tutte le variabili booleane di ingresso x_i hanno tutte valore 0

$$f_{NOR} = \sum \bar{x}_i \tag{1.24}$$

x_1	x_2	f_{NOR}
0	0	1
0	1	0
1	0	0
1	1	0

(1.25)



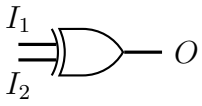
1.3.1.7 XOR

XOR è una porta logica che ha come valore di uscita 1 se e solo se il numero di variabili booleane di ingresso x_i che hanno valore 1 è dispari

$$f_{XOR} = \bigoplus x_i \tag{1.26}$$

x_1	x_2	f_{XOR}
0	0	0
0	1	1
1	0	1
1	1	0

(1.27)



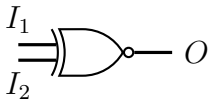
1.3.1.8 XNOR

XNOR è una porta logica che ha come valore di uscita 1 se e solo se il numero di variabili booleane di ingresso x_i che hanno valore 1 è pari oppure è zero

$$f_{XNOR} = \bigoplus \bar{x}_i \tag{1.28}$$

x_1	x_2	f_{XNOR}
0	0	1
0	1	0
1	0	0
1	1	1

(1.29)



1.3.2 Insieme di porte completo

Un insieme di porte completo è un insieme di porte logiche utilizzando le quali si può implementare qualunque funzione combinatoria
Insiemi completi sono:

- $\{NAND\}$
- $\{NOR\}$
- $\{AND, NOT\}$
- $\{OR, NOT\}$
- $\{AND, OR, NOT\}$

1.3.3 I/O di porte logiche

1.3.3.1 Fanin

Il fanin di una porta logica è il numero di segnali di ingresso della porta

1.3.3.2 Fanout

Il fanout di una porta logica è il numero di segnali di uscita della porta

1.3.4 Ritardo di una porta logica

Il ritardo associato ad una porta logica è il tempo richiesto alla porta per produrre il valore di uscita corretto rispetto ai suoi segnali di ingresso

1.4 Circuiti combinatori

1.4.1 Sistema combinatorio

Un sistema combinatorio è un sistema digitale i cui valori di uscita sono funzione solo dei valori di ingresso in quell'istante

1.4.2 Descrizione di sistemi combinatori

I sistemi combinatori possono essere descritti attraverso: una tavola di verità o una funzione booleana

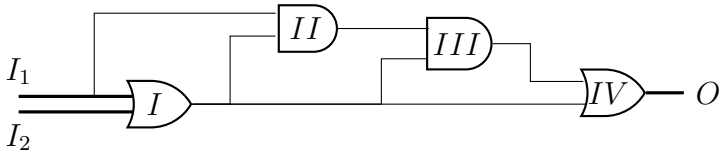
1.4.3 Circuiti combinatori ben formati (ccbf)

Un circuito combinatorio ben formato è un circuito combinatorio che soddisfa le seguenti proprietà:

- una singola linea o porta logica è un ccbf
- una giustapposizione di due ccbf è un ccbf
- se C_1 e C_2 sono due ccbf il circuito ottenuto connettendo un insieme di linee di uscita di C_1 ad un insieme di linee di ingresso di C_2 è un ccbf
- se x_i e x_j sono due linee di ingresso ad un ccbf il circuito ottenuto connettendo insieme x_i e x_j è un ccbf
- non contiene cicli

1.4.4 Profondità di un circuito combinatorio

La profondità di un circuito combinatorio è il numero massimo di porte logiche che si incontrano lungo i cammini che collegano gli ingressi alle uscite del circuito



1.4.5 Cammino critico di un circuito combinatorio

Il cammino critico di un circuito combinatorio è il cammino lungo il quale è massimo il ritardo con il quale una variazione dei valori di ingresso del circuito si propaga sui valori di uscita

1.4.6 Progetto di circuiti combinatori minimi

Il progetto di un circuito combinatorio segue i seguenti step:

- costruzione della mappa di Karnaugh a partire dalla tavola di verità o dalla funzione booleana
- sintesi della funzione booleana in forma di somma di prodotti
- trasformare i prodotti in porte AND e le somme in porte OR

Il progetto di un circuito combinatorio minimo deve soddisfare i seguenti vincoli:

- la profondità del circuito è minima
- il numero di porte logiche utilizzate è minimo
- il fanin massimo è minore di un valore assegnato
- il fanout massimo è minore di un certo valore

1.5 Circuiti sequenziali

1.5.1 Sistema sequenziale (o Finite State Machine)

Un sistema sequenziale è un sistema digitale i cui valori di uscita sono funzione sia dei valori di ingresso in quell'istante sia dei valori di ingresso degli istanti precedenti

1.5.1.1 FSM di Moore

Una FSM di Moore è un sistema sequenziale le cui uscite dipendono esclusivamente dallo stato del sistema

1.5.1.2 FSM di Mealy

Una FSM di Mealy è un sistema sequenziale le cui uscite dipendono sia dallo stato del sistema sia dai valori correnti di ingresso

1.5.2 Descrizione di sistemi sequenziali

I sistemi sequenziali possono essere descritti attraverso: una tavola degli stati/uscite, un diagramma degli stati/uscite e da una funzione di transizione degli stati e una funzione delle uscite

1.5.2.1 Tavola degli stati/uscite

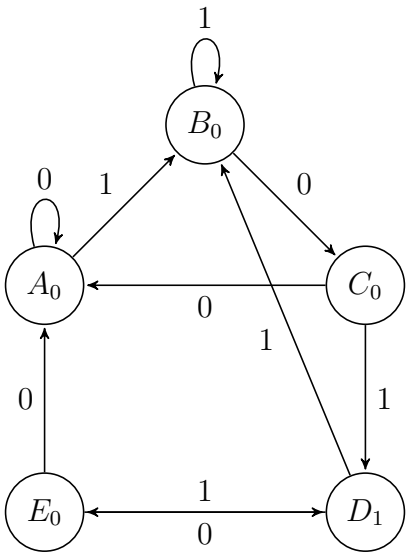
Una tavola degli stati/uscite è una tabella matematica in cui vengono rappresentati i valori degli stati futuri e delle uscite di un sistema in funzione dei valori degli stati correnti e degli ingressi

$S.Corr$	I	$S.Fut$	O
A	0	A	0
A	1	B	0
B	0	C	0
B	1	B	0
C	0	A	0
C	1	D	0
D	0	E	1
D	1	B	1
E	0	A	0
E	1	D	0

(1.30)

1.5.2.2 Diagramma degli stati/uscite

Un diagramma degli stati/uscite è un diagramma in cui viene rappresentato un grafo i cui nodi descrivono gli stati del sistema e i cui archi descrivono gli ingressi (i pedici dei nodi rappresentano le uscite)



1.5.2.3 Funzione di transizione stati/uscite

Una funzione di transizione degli stati e di uscite è una funzione booleana che associa ai valori di ingresso X e di stato corrente Y i valori di stato futuro Y e di uscita Z

$$f : XY \to YZ \tag{1.31}$$

$$\begin{aligned} &f(0, A) \to A, 0 \\ &f(1, A) \to B, 0 \\ &f(0, B) \to C, 0 \\ &f(1, B) \to B, 0 \\ &f(0, C) \to A, 0 \\ &f(1, C) \to D, 0 \\ &f(0, D) \to E, 1 \\ &f(1, D) \to B, 1 \\ &f(0, E) \to A, 0 \\ &f(1, E) \to D, 0 \end{aligned} \tag{1.32}$$

1.5.3 Stati equivalenti

Due stati si dicono equivalenti se per ogni sequenza di ingresso il loro comportamento in termini di valori prodotti sulle uscite è identico

1.5.4 Clock

Il clock è un segnale periodico (generalmente un'onda quadra) che permette ai segnali di ingresso di giungere ad una porta logica nello stesso istante

1.5.4.1 Circuito sincrono

Un circuito sincrono è un circuito dotato di clock in cui le porte commutano tutte nello stesso istante

1.5.4.2 Circuito asincrono

Un circuito asincrono è un circuito privo di clock in cui le porte commutano a seconda del ritardo di ciascuna porta

1.5.4.3 Periodo di clock

Il periodo di clock T è il tempo che intercorre tra due fronti di clock e deve essere maggiore della somma tra il massimo ritardo dei circuiti combinatori del sistema Δ e il ritardo dei flip-flop δ

$$T > \Delta + \delta \tag{1.33}$$

1.5.4.4 Frequenza di clock

La frequenza di clock f è il reciproco del periodo di clock e indica il numero di fronti di clock che si possono avere nel periodo di teempo T

$$f = \frac{1}{T} \tag{1.34}$$

1.5.5 Flip-Flop

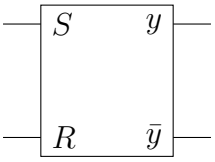
Un flip-flop è un circuito sequenziale in grado di mantenere il valore delle uscite costante nel tempo (può essere utilizzato per memorizzare gli stati di un circuito sequenziale)

1.5.5.1 Flip-Flop SR

Un flip-flop SR asincrono è un flip-flop asincrono in cui: se $S = R = 0$ le uscite y e $\bar{y}$ restano costanti; se $S = 0$ e $R = 1$ allora $y = 0$ e $\bar{y} = 1$ (reset); se $S = 1$ e $R = 0$ allora $y = 1$ e $\bar{y} = 0$ (set); la combinazione $S = R = 1$ non è permessa

S	R	y	$\bar{y}$
0	0	=	=
0	1	0	1
1	0	1	0
1	1	/	/

(1.35)

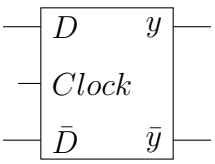


1.5.5.2 Data Flip-Flop

Un data flip-flop è un flip-flop sincrono in cui: se $D = 0$ allora $y = 0$; se $D = 1$ allora $y = 1$ (ad ogni colpo di clock il valore di D viene memorizzato

D	$\bar{D}$	y	$\bar{y}$
0	1	0	1
1	0	1	0

(1.36)



1.5.6 Progetto di circuiti sequenziali sincroni

Il progetto di un circuito sequenziale sincrono segue i seguenti step:

- costruzione della tavola degli stati/uscite
- minimizzazione (eliminazione degli stati equivalenti)
- assegnazione degli stati
- costruzione della tavola di verità o della funzione booleana del circuito combinatorio
- sintesi del circuito combinatorio

Capitolo 2

Componenti a livello di registri

2.1 Componenti combinatori

2.1.1 Multiplexer

Un multiplexer è un sistema combinatorio che seleziona uno dei suoi k valori di ingresso X_i in base ad un segnale di selezione S

$$\begin{cases} S = 0\ldots000 \implies Y = X_0 \\ S = 0\ldots001 \implies Y = X_1 \\ S = 0\ldots010 \implies Y = X_2 \\ S = 0\ldots011 \implies Y = X_3 \\ \ldots \end{cases} \implies \begin{array}{cccccc|c} X_0 & X_1 & X_2 & \ldots & S & & Y \\ \hline 0 & 0 & 0 & 0 & 0\ldots000 & & 0 \\ 0 & 1 & 0 & 0 & 0\ldots000 & & 0 \\ 1 & 0 & 0 & 0 & 0\ldots000 & & 1 \\ 1 & 1 & 0 & 0 & 0\ldots000 & & 1 \\ 0 & 0 & 0 & 0 & 0\ldots001 & & 0 \\ 0 & 1 & 0 & 0 & 0\ldots001 & & 1 \\ 1 & 0 & 0 & 0 & 0\ldots001 & & 0 \\ 1 & 1 & 0 & 0 & 0\ldots001 & & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & & \ldots \end{array} \tag{2.1}$$

2.1.2 Decodificatori

Un decodificatore è un sistema combinatorio che seleziona solo l'uscita Y_X corrispondente al valore X applicato in ingresso (può essere presente un segnale di enable E)

$$\begin{cases} E = 0 \implies Y = / \\ E = 1 \implies Y_X = Y_{\ldots X_3X_2X_1X_0} \end{cases} \implies \begin{array}{cc|c} X = \ldots X_3X_2X_1X_0 & E & Y_X \\ \hline \forall X & 0 & / \\ 0\ldots0000 & 1 & Y_0 \\ 0\ldots0001 & 1 & Y_1 \\ 0\ldots0010 & 1 & Y_2 \\ 0\ldots0011 & 1 & Y_3 \\ 0\ldots0100 & 1 & Y_4 \\ 0\ldots0101 & 1 & Y_5 \\ 0\ldots0110 & 1 & Y_6 \\ 0\ldots0111 & 1 & Y_7 \\ \ldots & \ldots & \ldots \end{array} \tag{2.2}$$

2.1.3 Codificatori

Un codificatore è un sistema combinatorio che riporta sull'uscita Y il codice binario dell'indice dell'entrata corrispondente

$$Y = X_{binario} \implies \begin{array}{c|c} X = \ldots X_3X_2X_1X_0 & Y \\ \hline 0\ldots0000 & 0\ldots0000 \\ 0\ldots0001 & 0\ldots0001 \\ 0\ldots001x & 0\ldots0010 \\ 0\ldots01xx & 0\ldots0011 \\ \ldots & \ldots \end{array} \tag{2.3}$$

2.1.4 Moduli aritmetici

2.1.4.1 Sommatore

Un sommatore è un sistema combinatorio che somma due numeri

2.1.4.1.1 Sommatore seriale Un sommatore seriale è un sommatore composto da una porta AND che somma un bit alla volta i due numeri e da un flip-flop D che memorizza a ogni colpo di clock l'eventuale riporto della somma

2.1.4.1.2 Sommatore combinatorio Un sommatore combinatorio è un sommatore composto da un circuito a due livelli progettato ad hoc per sommare 2 numeri su n bit

2.1.4.1.3 Full-adder Un full-adder è un sommatore che riceve in ingresso 2 bit x_i e y_i e un bit di carry c_i e produce un risultato z_i e un bit di carry c_{i+1}

2.1.4.1.4 Ripple carry adder Un ripple carry adder è un sommatore composto da m full-adder in cascata

2.1.4.1.5 Carry-lookahaed generator Un carry-lookahaed generator è un circuito combinatorio in grado di generare in parallelo n bit di carry

2.1.4.1.6 Sommatore con carry-lookahaed Un sommatore con carry-lookahaed è un ripple carry adder in cui i bit di carry vengono generati in parallelo da un carry-lookahaed generator

2.1.4.2 Arithmetic and Logic Unit (ALU)

Un'ALU è un sistema combinatorio che integra le principali funzioni aritmetiche e logiche (somma, sottrazione, NOT, AND, ...)

2.2 Componenti sequenziali

2.2.1 Registri a scorrimento (shift register)

Un registro a scorrimento è un sistema sequenziale composto da n flip-flop D in cascata i quali consentono lo scorrimento di ciscun bit da un flip-flop a quello adiacente ad ogni colpo di clock (utile per implementare code FIFO)

2.2.2 Contatori

Un contatore è un sistema sequenziale che assume gradualmente n stati in risposta a n segnali di ingresso

2.2.2.1 Contatore asincrono (ripple acounter)

Un contatore asincrono è un contatore le cui uscite non commutano contemporaneamente

2.2.2.2 Contatore sincrono

Un contatore sincrono è un contatore le cui uscite commutano contemporaneamente

2.2.3 Field Programmable Gate Array (FPGA)

Un FPGA è un circuito sequenziale integrato composto da una matrice di Configurable Logic Blocks (CLB) programmabili (si può definire la funzione di ogni CLB) connessi da una rete di interconnessione programmabile (si può definire ogni connessione) ai cui margini sono presenti degli Input and Output Blocks (IOB) programmabili

2.2.3.1 Metodi di programmazione di FPGA

I metodi di programmazione di FPGA sono:

- "bruciare" alcuni elementi del dispositivo al primo utilizzo (si può eseguire una sola volta)
- caricare in una memoria Flash interna un bitstream (una serie di bit) che determinano il comportamento del dispositivo (si può eseguire ogni volta che si vuole cambiare il circuito modificando il bitstream)
- caricare in una memoria RAM interna un bitstream (viene eseguita ogni volta che il dispositivo viene alimentato)

2.2.4 Bus

Un Bus è un canale di comunicazione che permette ai componenti di un sistema elettronico di: leggere i valori presenti nel bus e scrivere valori sul bus (non più di un componente alla volta può pilotarne il valore in scrittura)

2.2.5 Buffer tri-state

Un buffer tri-state è un buffer che funge da interfaccia verso il bus e possiede n ingressi ed n uscite di dato e un ingresso di enable

2.2.6 Transceiver

Un transceiver è un sistema elettronico che permette di collegare al bus un dispositivo che necessita di trasmettere dati sia in scrittura sia in lettura

2.2.7 Memorie

Una memoria è un sistema elettronico composto da un insieme di celle di memorizzazione in grado di memorizzare 1 bit; le memorie sono organizzate in parole identificate da un indirizzo i cui bit vengono letti e scritti tutti insieme in un'operazione di I/O da o verso la memoria; la lettura o scrittura di una parola avviene applicando agli ingressi della memoria l'indirizzo corrispondente

2.2.7.1 Random Access Memory (RAM)

Una RAM è una memoria che consente l'accesso a qualunque indirizzo di memoria con lo stesso tempo di accesso

2.2.7.2 Read Only Memory

Una ROM è una memoria il cui contenuto non è modificabile durante il funzionamento del dispositivo

Parte II

Processori

Capitolo 3

Introduzione

3.1 Architettura dei processori

3.1.1 Processore

Un processore è un sistema elettronico dedicato all'esecuzione di istruzioni

3.1.2 Control Process Unit (CPU)

Una CPU è un sistema di processori composto da un'unità di controllo e da un'unità di elaborazione (di cui fanno parte l'ALU e i registri)

3.1.2.1 Control Unit (CU)

Una CU è un processore che coordina tutte le operazioni necessarie per l'esecuzione di un'istruzione

3.1.2.2 Arithmetic and Logic Unit (ALU)

Un'ALU è un processore che esegue operazioni aritmetiche e logiche

3.1.2.3 Registri

Un registro è una cella di memoria particolarmente veloce che permette alla CPU di avere a disposizione nel minor tempo possibile le informazioni necessarie per svolgere un'istruzione

3.1.2.3.1 Tipologie di registri I registri possono essere suddivisi in categorie quali:

- registri dato: registri destinati alla memorizzazione di dati
- registri indice: registri destinati alla memorizzazione di indirizzi di memoria
- registri contatore: registri destinati all'esecuzione di operazioni di conteggio
- registri di stato: registro che memorizza due tipi di bit (o flag)
  - flag di stato: bit forzati dal verificarsi di particolari condizioni durante o al seguito dell'esecuzione di un'istruzione (carry, overflow, ...) o testati da istruzioni di salto condizionato
  - flag di controllo: bit di interrupt controlla la sensibilità agli interrupt; bit di trap attiva o disattiva la modalità debug del processore

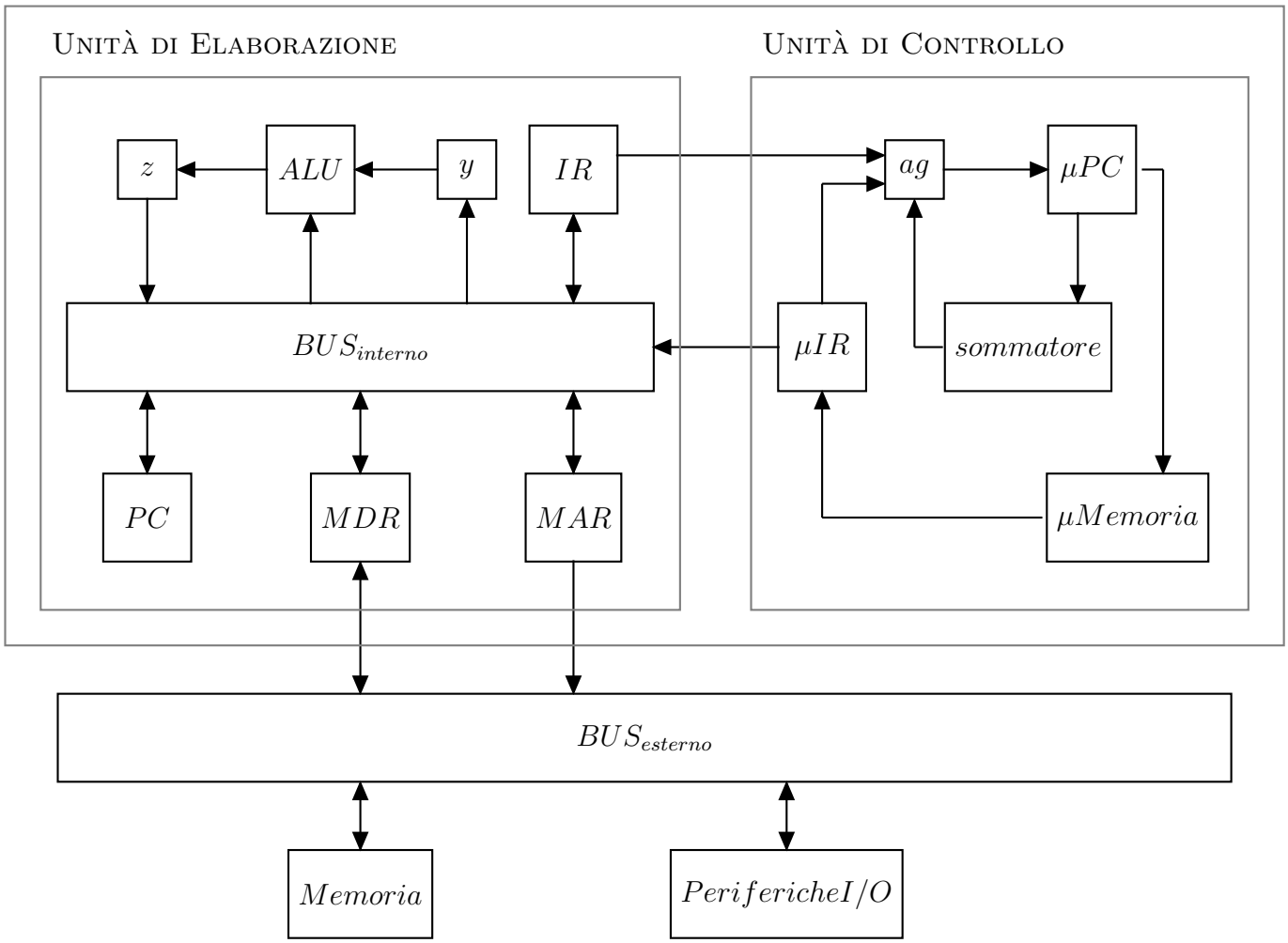
3.1.2.3.2 Program Counter (PC)(o Instruction Pointer) Un PC è un registro indice dove viene conservato temporaneamente l'indirizzo di memoria dell'istruzione successiva

3.1.2.3.3 Instruction Register (IR) Un IR è un registro indice dove viene memorizzata temporaneamente un'istruzione prelevata dalla memoria durante la decodifica dell'istruzione stessa

3.1.2.3.4 Memory Address Register (MAR) Un MAR è un registro indice che interfaccia la CPU con la memoria centrale (RAM) e contiene gli indirizzi delle locazioni di memoria sulle quali la CPU deve operare

3.1.2.3.5 Memory Data Register (MDR) Un MDR è un registro dati che interfaccia la CPU con la memoria centrale (RAM): tutti i dati che devono essere elaborati da una CPU e tutti i risultati di un'elaborazione di una CPU passano in un MDR

Control Process Unit



3.2 Istruzioni

3.2.1 Ciclo di istruzione

Un ciclo di istruzione è la sequenza di operazioni che un processore svolge per eseguire un'istruzione

- fetch (prelievo): il processore preleva un'istruzione dalla memoria
- decode (decodifica): il processore decodifica l'istruzione
- execute (esecuzione): il processore esegue l'istruzione

3.2.2 Microistruzione

Una microistruzione consiste in un insieme di segnali di controllo della CPU; un insieme di microistruzioni costituisce un'istruzione

3.2.3 Operazioni della CPU

La CPU può effettuare le seguenti operazioni:

- lettura di una parola dalla memoria
- scrittura di una parola in memoria
- trasferimento di una parola tra due registri
- esecuzione di un'operazione logico-aritmetica

3.3 Ciclo di vita di un programma

Il ciclo di vita di un programma è l'insieme di operazioni svolte da un sistema elettronico per effettuare l'esecuzione di un programma

3.3.1 Assemblaggio

L'assemblaggio di un programma consiste nella traduzione di un file sorgente (avviato dall'utente) in un file oggetto corrispondente attraverso i seguenti passaggi:

- rimozione dei commenti del file sorgente
- associazione di ciascuna variabile simbolica ad una locazione di memoria
- traduzione in codice macchina di ciascuna istruzione

3.3.2 Link

Il link di un programma consiste nella creazione di un file eseguibile a partire da più file oggetto

3.3.3 Load

Il load di un programma consiste nei seguenti passaggi:

- reperimento del file eseguibile all'interno di un disco di memoria
- caricamento del file in memoria principale
- inizio dell'esecuzione del programma

3.3.4 Debug

Il debug di un programma consiste in un'interazione tra l'utente e i processi di assemblaggio, link e load che consente all'utente di individuare e correggere eventuali errori contenuti nel programma

Capitolo 4

Unità di Controllo (CU)

4.1 Modello di CU

4.1.1 Funzionamento di una CU

Una CU funziona come un circuito sequenziale che ad ogni colpo di clock riceve segnali dall'esterno e dall'unità di elaborazione e produce segnali di controllo per l'esterno e per l'unità di elaborazione

4.2 Segnali di controllo della CU

4.2.1 Trasferimento di una parola tra due registri

I segnali di controllo inviati dalla CU per effettuare il trasferimento di una parola tra due registri ($R1$ e $R2$) sono:

- $R1_{out}$: segnale che attiva il caricamento del valore di $R1$ sul bus dati interno
- $R2_{in}$: segnale che attiva il caricamento del valore del bus dati interno sul registro $R2$

4.2.2 Lettura di una parola dalla memoria

I segnali di controllo inviati dalla CU per effettuare la lettura di una parola (il cui indirizzo è contenuto nel registro $R1$) sono:

- $R1_{out}$: segnale che attiva il caricamento del valore di $R1$ sul bus indirizzi
- $MAR_{internal-in}$: segnale che attiva il caricamento del valore del bus indirizzi interno sul MAR
- $MAR_{external-out}$: segnale che attiva il caricamento del valore del MAR sul bus indirizzi esterno
- $READ$: segnale che attiva la lettura della cella indirizzata: la memoria carica sul bus dati il contenuto della cella indirizzata
- attendi MFC (Memory Function Completed): la CU attende che la memoria abbia caricato i dati nel bus dati
- $MDR_{external-in}$ segnale che attiva il caricamento del valore del bus dati esterno sul MDR
- $MDR_{internal-out}$: segnale che attiva il caricamento del valore del MDR sul bus dati interno della della CPU
- $R2_{in}$: segnale che attiva il caricamento del valore del bus dati interno sul registro $R2$

4.2.3 Scrittura di una parola in memoria

I segnali di controllo inviati dalla CU per effettuare la scrittura di una parola (il cui indirizzo si trova nel registro $R1$ e il cui valore si trova nel registro $R2$) sono:

- $R1_{out}$: segnale che attiva il caricamento del valore di $R1$ sul bus indirizzi interno
- $MAR_{internal-in}$: segnale che attiva il caricamento del valore del bus indirizzi interno sul MAR
- $MAR_{external-out}$: segnale che attiva il caricamento del valore del MAR sul bus indirizzi esterno
- $R2_{out}$: segnale che attiva il caricamento del valore di $R1$ sul bus dati
- $MDR_{internal-in}$: segnale che attiva il caricamento del valore del bus dati interno sul MDR
- $MDR_{external-out}$: segnale che attiva il caricamento del valore del MDR sul bus dati esterno
- $WRITE$: segnale che attiva la scrittura della cella indirizzata: la memoria carica nella cella indirizzata il dato contenuto nel bus dati
- attendi MFC (Memory Function Completed): la CU attende che la memoria abbia salvato il dato

4.2.4 Operazione logico-aritmetica

I segnali di controllo inviati dalla CU per effettuare un'operazione logico-aritmetica (i cui operandi si trovano nei registri $R1$ ed $R2$ e il cui risultato andrà in $R3$) sono:

- $R1_{out}$: segnale che attiva il caricamento del valore di $R1$ sul bus dati interno
- y_{in}: segnale che attiva il caricamento del valore del bus dati interno sul registro y dedicato all'ALU
- $R2_{out}$: segnale che attiva il caricamento del valore di $R2$ sul bus dati interno
- $OPERATION$ (ADD, MUL, AND, ...): segnale che attiva l'ALU la quale compie l'operazione tra i due operandi
- z_{in}: segnale che attiva il caricamento del risultato dell'operazione dell'ALU sul registro z dedicato all'ALU
- z_{out}: segnale che attiva il caricamento del valore di z sul bus dati interno
- $R3_{in}$: segnale che attiva il caricamento del valore del bus dati interno sul registro $R3$

4.3 Progetto di CU

4.3.1 CU cablate (o hardwired)

Una CU cablata è un sistema sequenziale al quale si possono applicare i metodi di progetto dei circuiti sequenziali
Caratteristiche:

- le CU cablate sono molto difficilmente modificabili (la struttura dell'hardware viene fissata durante la fase di produzione)
- la dimensione della tabella degli stati/uscite può essere intrattabile
- vengono perse alcune informazioni (presenza di cicli)

4.3.2 CU microprogrammate

Una CU microprogrammata è un sistema elettronico in cui ogni operazione viene descritta da una microistruzione e i valori dei segnali di controllo da assegnare alle uscite sono immagazzinati in una micromemoria
Caratteristiche:

- flessibilità: si può modificare o aggiungere un'istruzione con la semplice modifica della memoria di microprogramma
- lentezza: l'esecuzione di un'istruzione può richiedere diversi accessi alla memoria di microprogramma
- costo

Funzionamento (operazioni eseguite ad ogni colpo di clock):

- lettura di una parola dalla memoria di microcodice utilizzando come indirizzo il contenuto del μPC
- caricamento della parola sul μIR (il quale pilota l'unità di elaborazione e la logica di generazione dell'indirizzo della successiva microistruzione)
- generazione di un nuovo indirizzo da caricare nel μPC

4.3.2.1 Tipologie di microprogrammazione

Esistono due tipi di microprogrammazione:

- microprogrammazione orizzontale: ogni microistruzione contiene un bit per ogni segnale di controllo (molto veloce ma ingombrante)
- microprogrammazione verticale: ogni microistruzione contiene in maniera codificata le informazioni da inviare ai segnali di controllo (meno veloce ma meno ingombrante)

4.3.2.2 Segnali compatibili

Due segnali si dicono compatibili se non sono mai attivati dalla stessa microistruzione

4.3.2.3 Classe di compatibilità

Una classe di compatibilità è un insieme di segnali di controllo a due a due compatibili

4.3.2.4 Minimizzazione del microcodice

Per minimizzare il parallelismo della memoria di microcodice è necessario che sia minima la somma dei logaritmi in base 2 del numero di segnali di controllo di ciascuna classe n_i (dove K è il numero di classi)

$$\sum_{i=0}^{K} \log_2 n_i \tag{4.1}$$

Parte III

Memorie

Capitolo 5

Introduzione

5.1 Caratteristiche generali

5.1.1 Memoria

Una memoria digitale è un sistema elettronico in grado di memorizzare dati e istruzioni

5.1.2 Caratteristiche delle memorie

5.1.2.1 Costo

Il costo di una memoria dipende dalla grandezza della memoria, dal software di interfaccia e dalla qualità del circuito utilizzato

5.1.2.2 Velocità

La velocità di una memoria è misurata attraverso tre parametri:

- tempo di accesso: tempo che intercorre tra l'istante in cui giunge alla memoria la richiesta di effettuare un'operazione e l'istante in cui l'operazione viene effettuata
- tempo di ciclo: tempo che intercorre tra l'inizio di un ciclo di accesso e l'inizio del ciclo successivo
- tasso di trasferimento: quantità di dati che possono essere trasferiti in una certo periodo di tempo da o verso la memoria
  - memorie ad accesso casuale: $\frac{1}{T_{ciclo}}\left[\frac{bit}{s}\right]$
  - altre memorie (T_n: tempo medio per leggere o scrivere n bit): $\frac{n}{(T_n - T_{accesso})}\left[\frac{bit}{s}\right]$

5.1.2.3 Modi di accesso

I modi di acceso ad una memoria sono quattro:

- sequenziale: le informazioni possono essere scritte solo in un ordine predeterminato (nastri)
- diretto: ogni blocco ha un indirizzo e la ricerca del blocco è sequenziale (dischi magnetici)
- casuale: ogni unità di dato ha un indirizzo, l'accesso alle informazioni può avvenire in qualsiasi ordine e il tempo di accesso è uguale per tutte le locazioni di memoria (memorie a semiconduttore)
- associativo: l'accesso avviene tramite un confronto tra il contenuto di una cella e il contenuto di una maschera (cache)

5.1.2.4 Alterabilità

Le memorie possono avere un contenuto inalterabile (Read Only Memory) oppure modificabile off-line (Programmable ROM)

5.1.2.5 Durevolezza del contenuto

La durevolezza del contenuto di una memoria può essere:

- destructive readout (DRO): la lettura causa la cancellazione del dato memorizzato (dopo ogni lettura è necessario riscrivere il dato)
- refreshing: dopo un certo periodo di tempo i bit di memoria commutano a 0; perciò è necessario riscrivere i bit periodicamente
- volatili: perdono il contenuto quando non sono alimentate

5.1.2.6 Guasti

I guasti di una memoria possono essere:

- transitori: uno o più cambiano valore ma la memoria continua a funzionare correttamente
- permanenti: la memoria smette di funzionare

5.1.2.7 Affidabilità

L'affidabilità di una memoria è misurata attraverso due parametri:

- Mean Time To Failure (MTTF): tempo medio prima di avere un guasto permanente
- failure rate: frequenza media dei guasti transitori

5.1.2.8 Altre caratteristiche

Altre caratteristiche sono: consumo, portabilità, robustezza e dimensione

5.2 Gerarchie di memorie

5.2.1 Gerarchia di memorie attuale

Le memorie sono classificate in base al costo, al tempo di accesso e alla dimensione in categorie

5.2.1.1 Memoria interna alla CPU

La memoria interna alla CPU (registri, cache) ha una velocità dell'ordine dei ns e ha una dimensione dell'ordine dei $Kbyte$; di solito è realizzata tramite circuiti di RAM statica

5.2.1.2 Memoria principale

La memoria principale ha una velocità dell'ordine di $10ns$ e ha una dimensione che va da qualche $Mbyte$ fino a qualche $Gbyte$; di solito è realizzata tramite circuiti di RAM dinamica

5.2.1.3 Memoria secondaria

La memoria secondaria ha una velocità dell'ordine di $10ms$ e ha una dimensione che va fino a qualche $Tbyte$; di solito è realizzata tramite dischi magnetici o memorie flash

5.2.1.4 Memoria off-line

La memoria off-line ha una velocità dell'ordine di $10s$ e ha una dimensione che va fino a qualche *Pbyte*; di solito è realizzata tramite dischi ottici o nastri

Capitolo 6

Memorie ad accesso seriale

6.1 Memorie a disco magnetico

6.1.1 Memoria a disco magnetico

Una memoria a disco magnetico è una memoria ad accesso seriale costituita da uno o più dischi ricoperti di materiale magnetico composto da una serie di tracce concentriche

6.1.2 Organizzazione del disco magnetico

Il disco magnetico è organizzato in due superfici dette facce dotate di una testina in grado di muoversi radialmente sulle tracce di una faccia; tutte le tracce contengono lo stesso numero di bit ma hanno lunghezza e densità di bit diversa (la densità aumenta verso il centro); ogni traccia è organizzata in settori; ogni settore corrisponde ad un'unità di trasferimento; ogni settore è composto da: un campo identificatore e un campo dati

6.1.3 Accesso alla memoria

L'accesso ad una memoria a disco magnetico avviene tramite le seguenti operazioni: posizionamento della testina sulla traccia; posizionamento del settore sotto la testina; accesso al settore

6.1.4 Tempo di accesso

Il tempo di accesso ad una memoria a disco magnetico è dato dalla somma tra: il tempo necessario per posizionare la testina (seek time $(5 - 15ns)$), il tempo per posizionare il settore (latency time) e il tempo necessario per la lettura seriale dei dati (data-transfer time)

$$t_a = t_s + t_l + t_d \tag{6.1}$$

6.1.5 Tasso di trasferimento dati

Il tasso di trasferimento dei dati $(40 - 130MB/s)$ è dato dal prodotto tra la densità di bit del settore $\left[\frac{bit}{cm}\right]$ e la velocità della testina $\left[\frac{cm}{s}\right]$ $(5400 - 15000gpm)$

6.2 Memorie a nastro magnetico

6.2.1 Memoria a nastro magnetico

Una memoria a nastro magnetico è una memoria ad accesso seriale costituita da un nastro magnetico di plastica flessibile sul quale sono memorizzate 9 tracce parallele

6.2.2 Organizzazione del nastro magnetico

Il nastro magnetico è organizzato in 9 tracce parallele ognuna delle quali è dotata di una testina (è quindi possibili accedere contemporaneamente a più tracce); i dati sono organizzati in record nelle prime 8 tracce; la nona traccia contiene un codice di parità

6.3 Memorie ottiche

6.3.1 Memoria ottica

Una memoria ottica è una memoria ad accesso seriale costituita da un disco ottico

6.3.2 Organizzazione del disco ottico

6.3.2.1 Costant Angular Velocity (CAV)

Un disco ottico con una soluzione CAV è un disco ottico organizzato in tracce concentriche di densità crescente verso il centro costituite da settori accessibili direttamente specificando traccia e numero del settore (soluzione analoga ai dischi magnetici)

6.3.2.2 Costant Linear Velocity (CLV)

Un disco ottico con una soluzione CLV è un disco ottico sul quale vengono incise delle fosse (pits) e delle piazzole (land) che memorizzano un'informazione binaria scandita da un raggio laser (a seconda di come il raggio viene riflesso viene decodificata l'informazione) il quale gira con velocità lineare costante seguendo una traiettoria a spirale

6.3.3 Tipologie di dischi ottici

6.3.3.1 Compact Disc (CD)

Un CD è un disco ottico non cancellabile che memorizza dati audio

6.3.3.2 Compact Disc-Read Only Memory (CD-ROM)

Un CD-ROM è un disco ottico non cancellabile che memorizza dati in formato elettronico

6.3.3.3 Compact Disc-Rewritable (CD-RW)

Un CD-RW è un disco ottico riscrivibile (tramite riscaldamento); la velocità di accesso di un CD-RW viene specificata attraverso tre parametri: velocità di scrittura, velocità di riscrittura e velocità di lettura

6.3.3.4 Digital Video Disk (DVD

Un DVD è un disco ottico in grado di memorizzare informazioni di vario genere:

- DVD-ROM: per grandi quantità di dati
- DVD-Video: per video
- DVD-Audio: per audio
- DVD-R: scrivibile una sola volta
- DVD-RAM
- DVD-RW
- DVD+RW

6.3.3.5 Blu-ray Disk (BD)

Un BD è un disco ottico utilizzato per memorizzare dati ad alta definizione

Capitolo 7

Memorie ad accesso casuale

7.1 Caratteristiche generali

7.1.1 Memoria ad accesso casuale

Una memoria ad accesso casuale è una memoria in cui ogni unità di dato ha un indirizzo, l'accesso alle informazioni può avvenire in qualsiasi ordine e il tempo di accesso è uguale per tutte le locazioni di memoria

7.1.2 Funzionamento

7.1.2.1 Ciclo di lettura

Il ciclo di lettura di una memoria ad accesso casuale avviene in due fasi:

- la CU carica sul bus indirizzi l'indirizzo della cella di memoria da leggere e sul bus di controllo attiva i segnali di lettura
- la memoria carica sul bus dati il contenuto della cella indirizzata

7.1.2.2 Ciclo di scrittura

Il ciclo di scrittura di una memoria ad accesso casuale avviene in due fasi:

- la CU carica sul bus indirizzi l'indirizzo della cella di memoria da scrivere, sul bus di controllo attiva i segnali di scrittura e sul bus dati il valore da memorizzare
- la memoria carica sulla cella indirizzata il contenuto del bus dati

7.1.3 Organizzazione di una memoria ad accesso casuale

Una memoria ad accesso casuale è organizzata in n parole (unità di dato a cui si accede in blocco) composte da m bit ciascuna a cui si accede attraverso $\log_2 n$ segnali di indirizzo ed m segnali per ciascun dato

7.1.3.1 Organizzazione a vettore

Una memoria ad accesso casuale organizzata a vettore è composta da: un decoder $\log_2 N \to N$ e N driver

7.1.3.2 Organizzazione a matrice

Una memoria ad accesso casuale organizzata a matrice è composta da: due decoder (uno per l'indirizzo della riga e uno per l'indirizzo della colonna) $\log_2 \sqrt{N} \rightarrow \sqrt{N}$ e $2\sqrt{N}$ driver

7.1.3.2.1 Row Address Strobe (RAS) e Column Address Strobe (CAS) Le memorie ad accesso casuale organizzate come matrici possono prevedere un protocollo di accesso ai dati attraverso un numero ridotto di segnali di indirizzo (la metà) in cui l'indirizzo viene fornito in due fasi distinte (utilizzando gli stessi segnali):

- alla memoria viene fornito il segnale di indirizzo della riga e un segnale di RAS
- alla memoria viene fornito il segnale di indirizzo della colonna e un segnale di CAS

7.1.3.2.2 Page Mode Il Page Mode è un segnale che permette di accedere a celle di una memoria organizzata a matrice poste consecutivamente in modo più rapido; il Page Mode attiva quattro operazioni:

- attivazione dei segnali RAS e CAS
- accesso al primo dato
- attivazione o del segnale di RAS (celle sulla stessa colonna) o del segnale di CAS (celle sulla stessa riga)
- accesso al secondo dato

7.2 Classificazione delle memorie ad accesso casuale

7.2.1 Read Only Memory (ROM)

Una ROM è una memoria il cui contenuto viene stabilito durante la fase di progettazione e in seguito non è più modificabile

7.2.2 Programmable Read Only Memory (PROM)

Una PROM è una memoria il cui contenuto viene stabilito attraverso attrezzature speciali chiamate programmatori che bruciano (una volta per sempre) alcune parti del circuito

7.2.3 Electrically Programmable Read Only Memory (EPROM)

Una EPROM è una memoria il cui contenuto può essere cancellato e riscritto un numero indefinito di volte attraverso un'esposizione prolungata a luce ultravioletta

7.2.4 Electrically Erasable Programmable Read Only Memory (EEPROM)

Una EEPROM è una memoria il cui contenuto può essere riprogrammato byte per byte tramite i bus e opportuni segnali di controllo

7.2.5 Flash

Una Flash è una memoria che in lettura si comporta come le RAM mentre in scrittura richiedono una precedente operazione di cancellazione

7.2.6 Random Access Memory (RAM)

7.2.6.1 Static RAM (SRAM)

Una SRAM è una memoria ad accesso casuale in cui ogni cella corrisponde ad un flip-flop

7.2.6.1.1 Accesso ai dati L'accesso ai dati di una SRAM avviene attraverso l'attivazione di una linea di parola che permette: la lettura dei valori di uscita (opposti) del flip-flop o la scrittura di un nuovo valore nel flip-flop

7.2.6.2 Dinamic RAM (DRAM)

Una DRAM è una memoria ad accesso casuale in cui ogni cella corrisponde ad un condensatore e ad un transistor e l'informazione è memorizzata sotto forma di carica del condensatore

7.2.6.2.1 Accesso ai dati L'accesso ai dati di una DRAM avviene attraverso l'attivazione di una linea di parola che:

- collega il condensatore alla linea di dato che assume il valore del condensatore (lettura)
- collega la linea di dato al condensatore che viene caricato o scaricato a seconda del valore della linea (scrittura)

7.2.6.2.2 Rinfresco Il rinfresco di una DRAM consiste nell'amplificazione della carica dei condensatori che contengono un valore alto (1 che tende a 0 per la scarica del condensatore) tramite operazioni di lettura fittizie

7.2.6.2.3 Affidabilità L'affidabilità di una DRAM è minata da particelle cariche che possono colpire la carica immagazzinata nel condensatore (facendo cambiare il valore memorizzato)

7.2.6.2.4 Codice di protezione Un codice di protezione di una DRAM è un codice che aumenta l'affidabilità della memoria

Codice di parità Un codice di partità è un codice di protezione che funziona nel seguente modo:

- in scrittura: la memoria calcola il bit di parità della parola insieme al nuovo valore della parola
- in lettura: la memoria calcola il codice di parità associato alla parola letta e lo confronta con il bit di parità memorizzato

in caso di diversità la memoria invia una segnalazione di errore

Codici di Hamming (o SECDED) I codici di Hamming o Single Error Correction Double Error Detection (SECDED) sono codici che richiedono $\log_2 n$ bit di codice e permettono: di rilevare e correggere tutti gli errori singoli; di rilevare tutti gli errori doppi

7.2.6.2.5 Synchronous DRAM (SDRAM) Una SDRAM è una DRAM caratterizzata dal possedere uno o più ingressi di clock

7.2.6.3 Confronto SRAM-DRAM

Le SRAM rispetto alle DRAM sono: più veloci, più costose, più semplici da utilizzare, più affidabili

7.2.6.4 Memoria interlacciata

Una memoria interlacciata è un sistema di memorie in cui le parole sono disposte in maniera alternata tra i vari moduli di memoria cui si può accedere in parallelo (riducendo il tempo di accesso complessivo nel caso di accessi a blocchi di parole)

Capitolo 8

Memorie cache

8.1 Introduzione

8.1.1 Memoria cache

Una memoria cache è una memoria interposta tra il processore e la memoria principale avente: piccole dimensioni e un'elevata velocità

8.1.2 Principi di località dei riferimenti

8.1.2.1 Principio di località temporale

Il principio di località temporale afferma che se un processore accede ad una cella di memoria nell'istante t è molto probabile che vi acceda nuovamente nell'istante $t + D$

8.1.2.2 Principio di località spaziale

Il principio di località spaziale afferma che se un processore accese ad una cella di memoria di indirizzo X nell'istante t è molto probabile che acceda entro l'istante $t + D$ anche alla cella di indirizzo $X + e$

8.1.3 Funzionamento

Una memoria cache si attiva quando il processore esegue un accesso alla memoria e funziona nel seguente modo:

- intercetta l'indirizzo della cella alla quale il processore vuole accedere
- verifica se la parola cercata è presente in uno dei suoi blocchi di memoria
  - se è presente (hit ≈ 0.9): la cache estrae la parola e la fornisce al processore (da 2 a 6 volte più velocemente della memoria principale)
  - se non è presente (miss ≈ 0.1): a seconda del proprio hardware la cache:
    - accede alla memoria principale, carica il blocco mancante per i principi di località dei riferimenti e poi fornisce la parola richiesta (tempo di accesso superiore a quello senza memoria cache)(meno costoso)
    - accede alla memoria principale, fornisce subito la parola richiesta e poi carica il blocco mancante per i principi di località dei riferimenti (load-through o early restart)(più costoso)

8.1.4 Prestazioni

Le prestazioni di una memoria cache sono una funzione: del tasso di hit della cache $h \approx 0.9$; del tempo di accesso alla parola il caso di hit C; del tempo di accesso alla parola in caso di miss M

$$t_{medio} = hC + (1 - h)M \tag{8.1}$$

8.2 Architettura

8.2.1 Architettura generale

Una memoria cache è composta da linee; ogni linea è composta da un campo indirizzi chiamato tag e da un blocco di memoria; la cache è posizionata (di solito) tra il processore e il bus esterno permettendo: di alleggerire il carico del bus e di progettare dispositivi multiprocessore

8.2.2 Architettura di von Neumann

L'architettura di von Neumann è un'architettura hardware in cui vi è un'unica memoria contenente dati e istruzioni

8.2.3 Archiettura di Harvard

L'architettura di Harvard è un'architettura hardware in cui sono presenti due memorie separate alle quali il processore può accedere in parallelo: una contenente i dati e una contenente le istruzioni

8.2.3.1 Instruction Cache (I-Cache)

Una I-Cache è una cache dedicata alla memorizzazione di istruzioni

8.2.3.2 Data Cache (D-Cache)

Una D-Cache è una cache dedicata alla memorizzazione di dati

8.3 Caratteristiche

8.3.1 Dimensioni

Le dimensioni di una cache variano tra qualche $Kbyte$ a qualche $Mbyte$; al crescere della dimensione: aumenta il costo; diminuiscono le prestazioni

8.3.2 Funzione di traduzione (mapping)

Una funzione di mapping è un algoritmo che definisce in quale linea di cache memorizzare un blocco di memoria

8.3.2.1 Direct mapping

Una funzione direct mapping mette in corrispondenza fissa ogni blocco i di memoria con la linea k di cache (N è il numero di linee di cache)

$$k = i \mod N \tag{8.2}$$

8.3.2.1.1 Vantaggi e svantaggi Vantaggi: funzione facilmente implementabile in hardware
Svantaggi: se il processore fa accesso frequentemente a 2 blocchi corrispondenti alla stessa linea della cache, ad ogni accesso si verifica un miss

8.3.2.1.2 Struttura dell'indirizzo Un indirizzo di un'architettura con direct mapping è composto da:

- etichetta: identifica il blocco di memoria e viene confrontata con il tag della cache
- blocco: identifica la linea di cache attraverso la funzione di mapping
- parola: identifica la parola all'interno di un blocco

8.3.2.2 Associative mapping

Una funzione associative mapping è una funzione di mapping che permette ad ogni blocco di memoria di essere memorizzato in un qualunque blocco di cache

8.3.2.2.1 Vantaggi e svantaggi Vantaggi: massima flessibilità nella scelta del blocco
Svantaggi: complessità dell'hardware di ricerca

8.3.2.2.2 Struttura dell'indirizzo Un indirizzo di un'architettura con associative mapping è composto da:

- etichetta: identifica il blocco di memoria e viene confrontata con il tag della cache
- parola: identifica la parola all'interno di un blocco

8.3.2.3 Set associative mapping

Una funzione set associative mapping è una funzione di mapping in cui: le linee di cache sono suddivise in S sottoinsiemi; un blocco i è associato al k-esimo sottoinsieme se $k = i \mod S$; il blocco i può essere messo in una linea qualunque del k-esimo sottoinsieme

8.3.2.3.1 Struttura dell'indirizzo Un indirizzo di un'architettura con set associative mapping è composto da:

- etichetta: identifica il blocco di memoria i e viene confrontata con il tag della cache
- blocco: identifica il k-esimo sottoinsieme di linee di cache attraverso la funzione di mapping
- parola: identifica la parola all'interno di un blocco

8.3.3 Algoritmi di rimpiazzamento

Gli algoritmi di rimpiazzamento sono algoritmi che definiscono quale linea di cache tra quelle associate ad un blocco debba essere utilizzata per memorizzarlo:

- Least Recently Used (LRU): più usato
- First-In First-Out (FIFO): più economico
- Least Frequently Used (LFU): più efficiente
- random: semplice ed efficace

8.3.4 Metodo di aggiornamento della memoria principale

8.3.4.1 Metodo write-back

Il metodo write-back è un metodo di aggiornamento della memoria principale nel quale per ogni blocco della cache si tiene aggiornato un flag chiamato dirty bit ricorda se il blocco ha subito modifiche o meno da quando è stato caricato nella cache; quando il blocco viene eliminato dalla cache, se il dirty bit è settato il blocco viene copiato nella memoria principale

8.3.4.2 Metodo write-through

Il metodo write-through è un metodo di aggiornamento della memoria principale nel quale, ogni volta che il processore esegue un'operazione di scrittura, la esegue sia sul dato nella cache sia sul dato nella memoria principale

8.3.5 Coerenza

La coerenza della cache è un problema nei sistemi multiprocessore con memoria principale condivisa in cui ogni processore utilizza una propria cache; per mantenere la coerenza della memoria si possono utilizzare i seguenti metodi:

- bus watching con write-through: i controllori delle cache rilevano le operazioni di write-through sul bus e invalidano (attraverso un bit che viene disattivato quando il blocco viene modificato chiamato bit di validità) le linee corrispondenti nella propria cache
- non-cacheable memory: la memoria condivisa non può essere caricata nelle cache

8.3.6 Cache a livelli

Una cache a livelli è un sistema di cache in cui sono presenti:

- una cache di primo livello (C1) piccola e molto veloce
- una cache di secondo livello (C2) più grande e un po' più lenta
- una cache di terzo livello (C3) grande e lenta

Capitolo 9

Memorie virtuali

9.1 Introduzione

9.1.1 Memoria virtuale

La memoria virtuale è un meccanismo attraverso il quale il processore: può fare accesso ad uno spazio di indirizzamento molto maggiore rispetto alle dimensioni della memoria principale realmente disponibile; utilizza indirizzi logici che vengono poi trasformati in indirizzi fisici

9.2 Architettura

9.2.1 Suddivisione in pagine

La memoria virtuale è suddivisa in blocchi chiamati pagine per i principi di località dei riferimenti

9.2.2 Memory Address Table (MAT)

Una MAT è una tabella che permette la trasformazione degli indirizzi logici in indirizzi fisici; per ogni pagina virtuale la MAT memorizza: o l'indirizzo fisico in cui si trova la pagina nella memoria principale o la posizione della pagina nella memoria secondaria

9.2.3 Translation Lookaside Buffer (TLB)

Un TLB è una memoria cache per la MAT che memorizza le entry delle pagine accedute più di recente

9.2.4 Memory Management Unit (MMU)

Una MMU è un sistema hardware che utilizza una MAT per verificare che la parola indirizzata logicamente dal processore si trovi fisicamente nella memoria principale:

- se esiste: la MMU produce l'indirizzo fisico corrispondente della memoria principale
- se non esiste (page fault): la MMU richiede l'intervento del sistema operativo che provvede a spostare nella memoria principale il relativo blocco di memoria

9.3 Caratteristiche

9.3.1 Vantaggi

La memoria virtuale offre i seguenti vantaggi:

- rende i programmi indipendenti dalla configurazione reale della memoria
- offre ai programmi una memoria complessiva molto ampia, con tempi di accesso ridotti e a basso costo per bit
- permette un uso efficiente della memoria in ambienti multiutente

9.3.2 Differenze tra memoria virtuale e cache

Le differenze tra memoria virtuale e cache sono:

- dimensioni delle pagine: $4 - 32$ byte per la cache; $1 - 64$ Kbyte per la memoria virtuale
- gestione: tramite hardware apposito per la cache; tramite il supporto del sistema operativo per la memoria virtuale

Parte IV

Trasferimento dati

Capitolo 10

Sotto sistema di Input/Output (I/O)

10.1 Introduzione

10.1.1 Sotto sistema di I/O

Il sotto sistema di I/O di un calcolatore è un sistema composto da: dispositivi di I/O (periferiche); unità di controllo delle periferiche; software per la gestione delle periferiche

10.1.2 Porta

Una porta è un registro che permette la comunicazione tra il processore e una periferica

10.1.2.1 Modalità memory-mapped I/O

La connessione al bus con memory-mapped I/O è una connessione in cui le porte sono connesse come normali celle di memoria

10.1.2.2 Modalità isolated I/O

La connessione al bus con isolated I/O è una connessione in cui gli spazi di indirizzamento per la memoria e per le porte di I/O sono separati e sono attivati alternativamente da appositi segnali

10.2 Meccanismi di gestione dell'I/O

10.2.1 I/O programmato

L'I/O programmato è un meccanismo di gestione delle periferiche totalmente demandato alla CPU (lettura, scrittura, spostamento di dati): la CPU controlla ciclicamente le necessità del dispositivo (polling) e in caso di necessità interviene

10.2.1.1 Caratteristiche

L'I/O programmato è: poco costoso in termini di hardware; poco efficiente (tutto deve passare attraverso la CPU perché non c'è un collegamento diretto gtra periferica e memoria)

10.2.2 Interrupt

L'interrupt è un meccanismo di gestione delle periferiche in cui il dispositivo di I/O invia alla CPU un segnale asincrono quando ha bisogno di un servizio (viene evitato il polling della CPU): la CPU interrompe le operazioni che stava svolgendo e salta all'esecuzione di una ISR

10.2.2.1 Interrupt Service Routine (ISR)

Una ISR dell'interrupt è una procedura la cui attivazione interrompe l'esecuzione di un programma; al termine dei una ISR è necessario che il programma continui l'esecuzione dal punto in si era interrotto

10.2.2.2 Interrupt Controller (IC)

Un IC è un sistema che gestisce le richieste di interrupt delle periferiche: riceve le richieste delle periferiche e pilota il segnale di interrupt verso la CPU

10.2.2.3 Latenza di interrupt

La latenza di interrupt è il tempo che intercorre tra la richiesta di interrupt e l'avvio della ISR corrispondente

10.2.2.4 Disabilitazione degli interrupt

Se una CPU sta svolgendo un'operazione essenziale (come una ISR) è necessario evitare una richiesta di interrupt interrompa il processore; per evitare ciò in un registro della CPU viene memorizzato un bit di abilitazione degli interrupt (se settato non la CPU non può essere interrotta)

10.2.2.5 Identificazione della periferica

Per identificare la periferica che ha inviato una richiesta di interrupt si utilizzano le seguenti soluzioni:

- linee di interrupt multiple: la CPU possiede diversi segnali per le richieste di interrupt
- polling: esiste un solo segnale per le richieste di interrupt e, quando arriva la richiesta di interrupt, la CPU scandisce le parole di stato delle periferiche per individuare quella che ha inviato la richiesta
- interrupt vettorizzato
  - quando la CPU è pronta a servire una richiesta di interrupt invia un segnale di Interrupt Acknowledge
  - l'IC carica sul bus dati un codice di identificazione
  - il codice viene utilizzato dalla CPU come indice per accedere all'Interupt Vector Table (IVT), un vettore contenente gli indirizzi delle ISR

10.2.2.6 Priorità di interrupt

Nei sistemi complessi esiste una gerarchia di interrupt in cui ogni ISR può essere interrotta solo da una richiesta di interrupt avente priorità superiore

10.2.2.7 Procedura di interrupt

La procedura di interrupt consiste nelle seguenti operazioni:

- una periferica invia la richiesta di interrupt
- l'IC inoltra la richiesta verso la CPU
- la CPU attiva il segnale di Interrupt Acknowledge
- l'IC invia alla CPU il codice della periferica

- la CPU salva il PC e il registro di stato (per poter riprendere l'esecuzione delle operazioni che stava eseguendo)
- la CPU estrae dalla IVT l'indirizzo della ISR
- la CPU esegue la ISR
- la CPU riprende l'esecuzione del programma interrotto

10.2.2.8 Eccezioni

Le eccezioni sono eventi in grado di interrompere l'esecuzione di un programma:

- interrupt
- segnali di errore
- segnali per il debug
- eccezioni di privilegio (quando si tenta di fare accesso ad un'area di memoria o eseguire operazioni senza averne il diritto)

10.2.3 Direct Memory Access (DMA)

Il DMA è un meccanismo di gestione delle periferiche utilizzato per spostare grandi moli di dati da una periferica alla memoria

10.2.3.1 DMA Controller

Il DMA Controller è un modulo che esegue il trasferimento dei dati da una periferica ad una locazione di memoria; è in grado di arbitrare tra richieste di DMA contemporanee

10.2.3.2 Procedura di trasferimento

La procedura di trasferimento dei dati di una periferica in DMA consiste nelle seguenti operazioni:

- la CPU carica nei registri IOAR e DC del DMA Controller l'indirizzo dell'area di memoria da trasferire e della direzione del trasferimento
- il DMA Controller riceve una richiesta di trasferimento da parte di una periferica
- il DMA Controller invia un segnale di DMA Request alla CPU
- quando la CPU riceve il segnale di DMA request rilascia il bus e attiva il segnale di DMA Acknowledge
- il DMA Controller comincia il trasferimento parola per parola; dopo il trasferimento di ogni parola il DMA Controller aggiorna i registri IOAR e DC
- quando DC arriva a zero termina il trasferimento
- il DMA Controller invia una richiesta di interrupt alla CPU
- la CPU disattiva il DMA Acknowledge e riprende il controllo del bus

10.2.3.3 Modi di funzionamento

10.2.3.3.1 Trasferimento a blocchi Il DMA Controller mantiene il controllo del bus per tutto il tempo richiesto per trasferire un blocco di dati (la CPU è bloccata per tutta la durata del trasferimento)

10.2.3.3.2 Trasferimento con cycle stealing Il DMA Controller trasferisce i dati in piccoli blocchi; alla fine di ogni blocco il DMA Controller rilascia il bus e subito dopo inoltra una nuova richiesta alla CPU

10.2.3.3.3 Trasferimento in transparent DMA Il DMA Controller rileva quando la CPU non utilizza il bus ed esegue il trasferimento solo in quei periodi di tempo

Capitolo 11

Bus

11.1 Introduzione

11.1.1 Bus

Un bus è un sistema elettronico condiviso che interconnette due o più dispositivi

11.1.2 Transceiver

Un transceiver è il sistema di interfaccia usato dai dispositivi per accedere al bus; un transceiver è composto da:

- un driver tri-state: per pilotare (scrivere) le linee del bus
- un receiver: per leggere i valori presenti nel bus

11.1.3 Architettura

L'architettura di un bus può essere:

- a bus singolo
- a bus multiplo (per prestazioni elevate o quando si devono connettere classi di dispositivi con caratteristiche diverse)

11.2 Unità connesse al bus

11.2.1 Unità master

Un'unità master (generalmente la CPU) è l'unità connessa al bus che inizia ogni procedura di trasferimento dati e sceglie l'unità con cui cominicare

11.2.2 Unità slave

Un'unità slave (memorie e periferiche) è un'unità connessa al bus che risponde ai comandi dell'unità master

11.3 Tempistiche

11.3.1 Bus sincroni

Un bus sincrono è un bus in cui le unità ad esso collegate utilizzano la stessa frequenza di clock (imposta dal dispositivo più lento) e ogni unità di dato viene trasferita in un periodo di clock

11.3.1.1 Ciclo di wait

Un ciclo di wait è un meccanismo che introduce occasionalmente dei cicli di clock aggiuntivi per permettere allo slave di ottenere più tempo per eseguire l'operazione richiesta

11.3.2 Bus asincroni

Un bus asincrono è un bus in cui ogni operazione di comunicazione può avere una sua velocità determinata da appositi segnali di controllo (massima flessibilità)

11.4 Segnali del bus

11.4.1 Segnali del bus

Un bus può trasportare tre tipi di segnali:

- segnali di dato (di solito un multiplo di 8)
- segnali di indirizzo: identificano lo slave
- segnali di controllo: forniscono informazioni di: stato, temporizzazione, tipo

11.4.2 Multibus: ABUS, DBUS, CBUS

Un multibus è un bus in cui ogni linea trasporta o un segnale di dato (DBUS) o un segnale di indirizzo (ABUS) o un segnale di controllo (CBUS)

11.4.2.1 Ciclo di lettura

Il ciclo di lettura di un multibus consiste nelle seguenti fasi:

- il master carica gli indirizzi sull'ABUS e attiva opportuni segnali di controllo
- lo slave carica i dati sul DBUS
- lo slave attiva XACK
- dopo la lettura il master disattiva i segnali di controllo e libera l'ABUS
- lo slave disattiva XACK e libera il DBUS

11.4.3 Bus multiplexato

Un bus multiplexato è un bus le cui linee trasportano in tempi diversi segnali di tipo diverso

11.5 Arbitraggio

11.5.1 Arbitraggio di un bus

L'arbitraggio di un bus è un meccanismo che designa quale unità sarà il nuovo master del bus quando più unità inviano contemporaneamente richiesta di diventare master

11.5.2 Arbitraggio distribuito con bus SCSI

Un bus SCSI è un bus che possiede 8 linee $DB(i)$ utilizzate sia per il trasferimento dati sia per l'arbitraggio:

- durante l'arbitraggio ogni linea è associata a un dispositivo ($DB(7)$ ha la priorità massima)
- quando la linea BSY diventa inattiva ogni dispositivo connesso al bus alza la rispettiva linea $DB(i)$; il dispositivo avente priorità massima diventa il nuovo master del bus

11.5.3 Arbitraggio centralizzato

11.5.3.1 Daisy chaining

Un arbitraggio centralizzato con daisy chaining funziona nel seguente modo:

- un'unità fa richiesta del bus (BUS REQUEST) e attende che il bus si liberi (BUS BUSY)
- l'arbitro attiva il segnale BUS GRANT
- ogni unità:
  - se ha fatto richiesta del bus: attiva BUS BUSY e diventa master del bus
  - se non ha fatto richiesta del bus: attiva BUS GRANT verso l'unità a valle

Vantaggi: richiede solo tre segnali di controllo
Svantaggi: non permette modifiche di priorità; non è tollerante ai guasti

11.5.3.2 Polling

Un arbitraggio centralizzato con polling funziona nel seguente modo:

- un'unità fa richiesta del bus (BUS REQUEST) e attende che il bus si liberi (BUS BUSY)
- l'arbitro scandisce secondo un ordine di priorità tutte le unità collegate mettendo sulle linee POLL COUNTER l'ID di ogni unità
- quando viene indirizzata un'unità che aveva fatto richiesta essa attiva il segnale BUS BUSY (diventando master del bus) e l'arbitro interrompe la scansione

Vantaggi: le priorità possono essere cambiate modificando la sequenza di scansione; è tollerante a guasti
Svantaggi: richiede $2 + \log n$ segnali di controllo per gestire n unità

11.5.3.3 Richieste indipendenti

Un arbitraggio centralizzato con richieste indipendenti funziona nel seguente modo:

- un'unità i-esima fa richiesta del bus (BUS REQUEST) e attende che il bus si liberi (BUS BUSY)
- l'arbitro concede il bus alla j-esima unità tra quelle che hanno fatto richiesta avente priorità massima
- la j-esima unità diventa master del bus e attiva il segnale di BUS BUSY

Vantaggi: le priorità sono modificabili e dipendono dagli algoritmi implementati dall'arbitro; è tollerante ai guasti
Svantaggi: richiede $2n + 1$ segnali di controllo per gestire n unità