

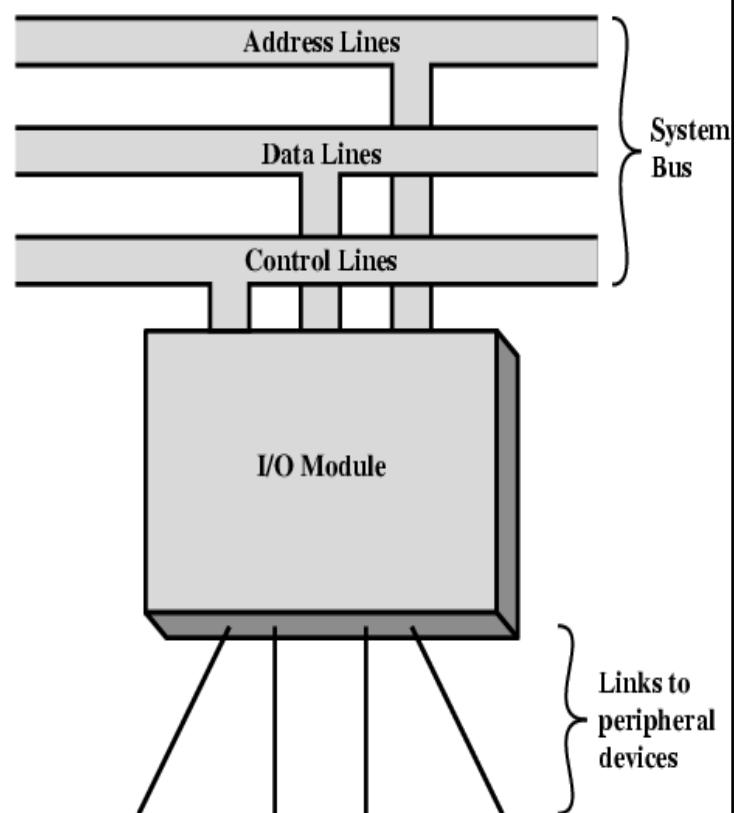
# Input/Output



- Grande varietà di periferiche
  - gestiscono quantità di dati differenti
  - a velocità diverse
  - in formati diversi
- Tutti più lenti della CPU e della RAM
- Necessità di avere moduli di I/O

## Moduli di Input/Output

- Si interfacciano con
  - CPU e Memoria
  - Una o più periferiche

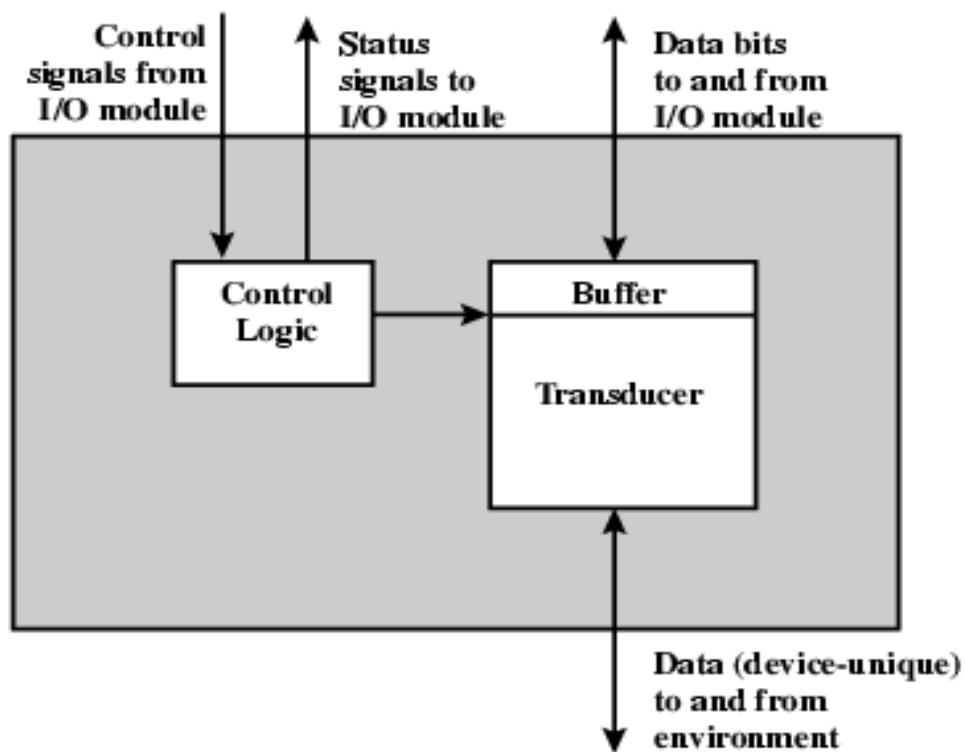


# Dispositivi Esterni

- Comprensibili dall'uomo
  - video, stampante, tastiera
- Comprensibili dalla macchina
  - Monitoraggio e controllo
- Comunicazione
  - Modem
  - Rete [Network Interface Card (NIC)]



## Schema di dispositivo esterno



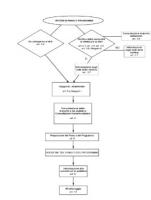
# Funzioni del modulo I/O



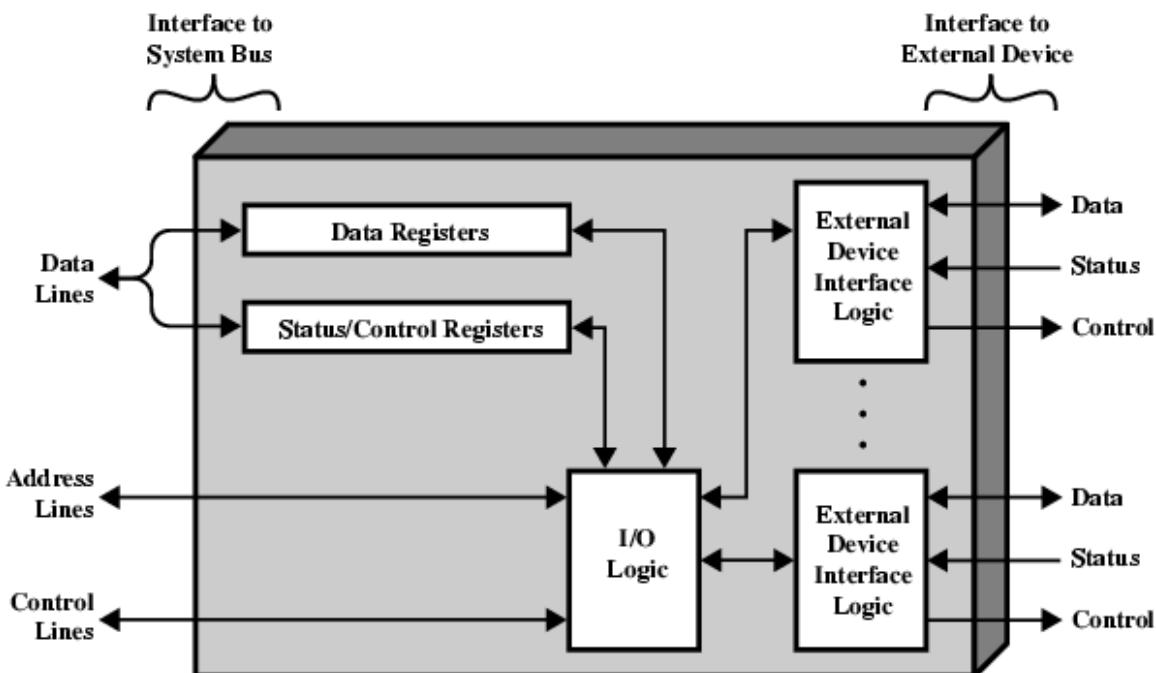
- Controllo & Temporizzazione
- Comunicazione con CPU
- Comunicazione con i dispositivi
- *Buffering* dei dati
- Rilevazione degli errori

## Passi di I/O (versione semplificata)

- CPU interroga il modulo I/O sullo stato del dispositivo connesso
- Il modulo I/O restituisce lo stato del dispositivo
- Se dispositivo pronto a trasmettere, CPU richiede il trasferimento dei dati, tramite comando a modulo I/O
- Il modulo I/O ottiene una unità di dati dal dispositivo esterno
- Il modulo I/O trasferisce i dati alla CPU



# Diagramma modulo I/O



## Caratteristiche modulo I/O



- Nascondere o rivelare le proprietà del dispositivo alla CPU
- Supportare dispositivi singoli o multipli
- Controllare le funzioni del dispositivo o lasciare il controllo alla CPU
- Caratteristiche Sistema Operativo
  - Ad esempio, Unix tratta tutto quello che può come se fosse un file

# Tecniche di gestione Input/Output

## ■ I/O da programma

Programmed I/O



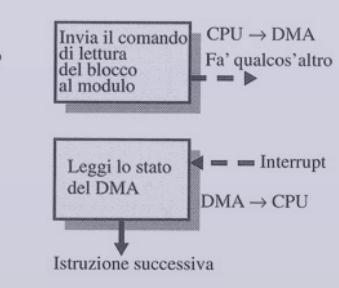
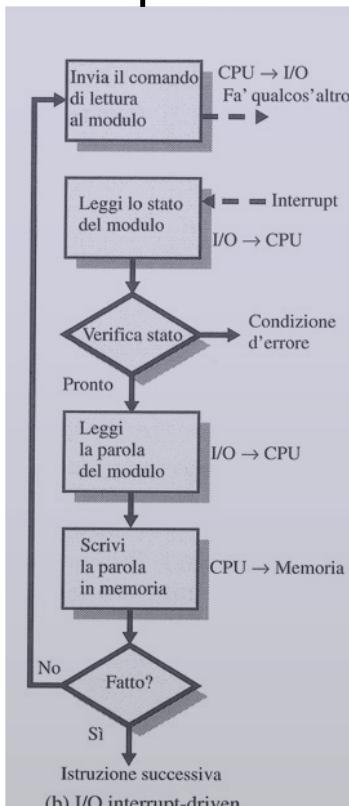
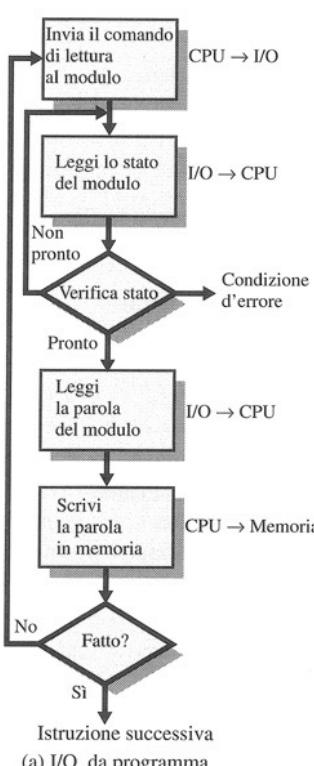
## ■ I/O guidato da interrupt

Interrupt Driven I/O

## ■ Accesso Diretto alla Memoria

Direct Memory Access (DMA)

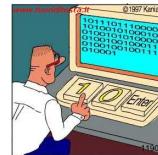
## Tre tecniche per l'input di un blocco di dati



(b) I/O interrupt-driven

(c) Accesso diretto alla memoria (DMA)

# I/O da programma



- CPU ha il controllo diretto sull' I/O
  - Controllo stato dispositivo
  - Comandi lettura/scrittura
  - Trasferimento dati
- CPU aspetta che il modulo I/O completi l'operazione
- Spreca tempo di CPU

## I/O da programma- dettaglio



- CPU richiede operazione I/O
- Modulo I/O esegue operazione
- Modulo I/O setta bit di stato
- CPU controlla bit di stato periodicamente
- Modulo I/O non informa direttamente CPU
- Modulo I/O non interrompe CPU
- CPU può attendere o fare altro e controllare più tardi

# Comandi I/O



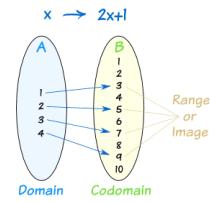
- CPU invia indirizzo
  - che identifica modulo (& dispositivo se >1 per modulo)
- CPU invia comando
  - di controllo – dire al modulo cosa fare
    - ad esempio, dare velocità al disco
  - di test – controlla lo stato
    - ad esempio, alimentazione? errore?
  - di lettura/scrittura
    - il modulo trasferisce i dati tramite buffer dal/verso il dispositivo

# Indirizzamento dispositivi I/O

- Nell' I/O da programma il trasferimento dati è molto simile all'accesso alla memoria (dal punto di vista della CPU)
- Ad ogni dispositivo viene assegnato un identificatore unico
- I comandi di CPU riferiscono tale identificatore (indirizzo)



# I/O Mapping



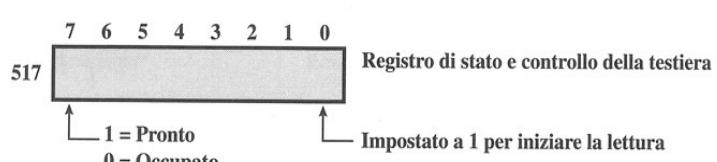
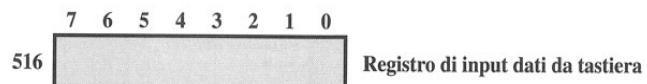
## ■ I/O memory-mapped

- Dispositivi e memoria condividono lo stesso spazio di indirizzamento
- I/O sembra proprio come lettura/scrittura di memoria
- Nessun comando speciale per I/O
  - Ampia varietà di comandi di accesso alla memoria disponibili

## ■ I/O separato (isolated)

- Spazi di indirizzamento separati
- Necessita di linee di selezione fra I/O e memoria
- Comandi speciali per I/O
  - Insieme limitato

Confronto fra  
I/O memory  
mapped  
e  
separato



INDIRIZZO	ISTRUZIONE	OPERANDO	COMMENTO
200	Load AC	"1"	Carica nell'accumulatore
	Store AC	517	Inizia la lettura della tastiera
202	Load AC	517	Legge il byte di stato
	Branch if Sign = 0	202	Sta in loop fino a quando è pronto
	Load AC	516	Carica un byte di dati

(a) I/O memory mapped

INDIRIZZO	ISTRUZIONE	OPERANDO	COMMENTO
200	Load I/O	5	Inizia la lettura della tastiera
201	Test I/O	5	Controlla il completamento
	Branch Not Ready	201	Sta in loop fino al completamento
	In	5	Carica un byte di dati

(b) I/O isolato

# Confronto fra I/O memory mapped e separato



- La tecnica *memory-mapped I/O* ha diversi vantaggi
  - Non necessita di istruzioni speciali
    - Le istruzioni che accedono alla memoria “normale” accedono anche alle aree di I/O
    - Il software di controllo di dispositivo può essere scritto interamente in linguaggi ad alto livello
  - Consente una più agevole protezione
    - è sufficiente nascondere le aree di I/O allo spazio di indirizzamento dell’utente (*privilegi*)
    - Con la tecnica della *memoria segmentata*, *più* aree di I/O possono mappare sul *medesimo* spazio di indirizzamento fisico

# Confronto fra I/O memory mapped e separato



- La tecnica *memory-mapped I/O* presenta anche alcuni svantaggi
  - Non si presta all’uso di cache
    - Il dato rilevante è *sempre e solo* nella memoria del dispositivo
    - Occorre disabilitare selettivamente la cache
  - Non è compatibile con architetture a bus multipli
    - I dispositivi di I/O non possono rispondere ad indirizzi emessi su bus non connessi
    - Occorre filtrare gli indirizzi emessi dalla CPU ed instradarli sul bus appropriato
      - Filtraggio a sorgente piuttosto che a destinazione