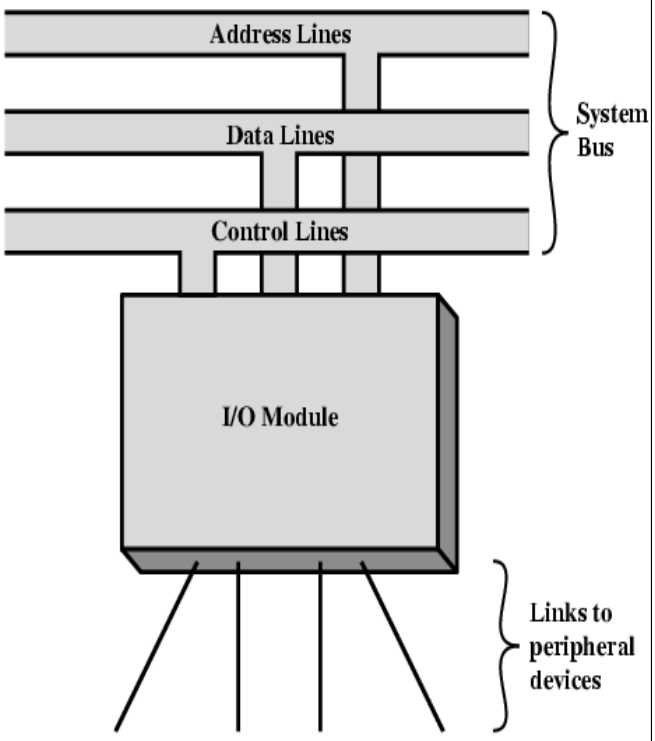
Input/Output

- 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

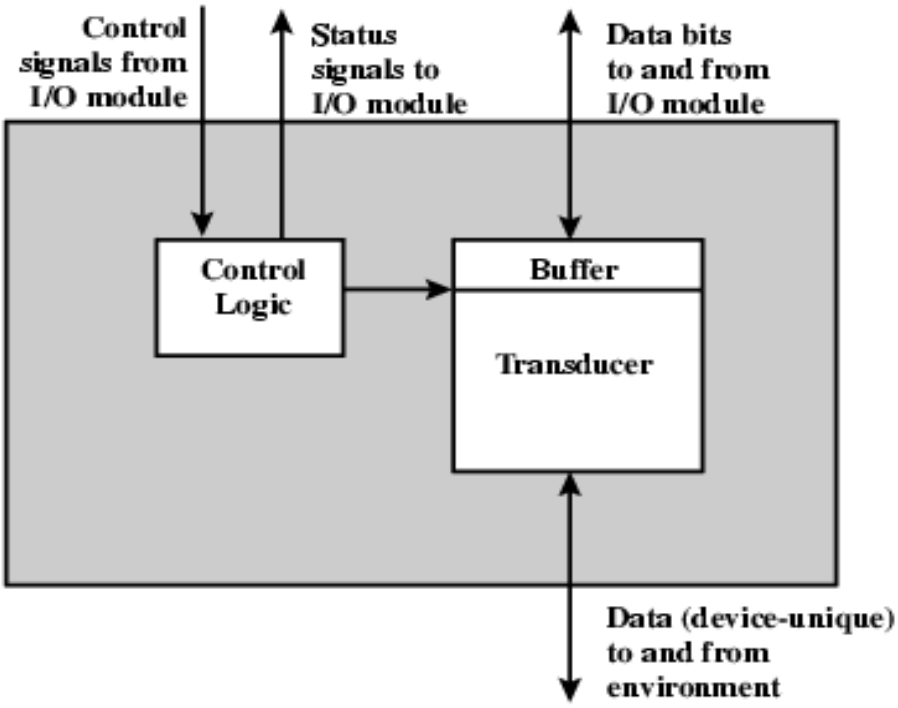


Dispositivi Esterni

- 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

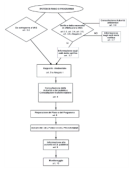
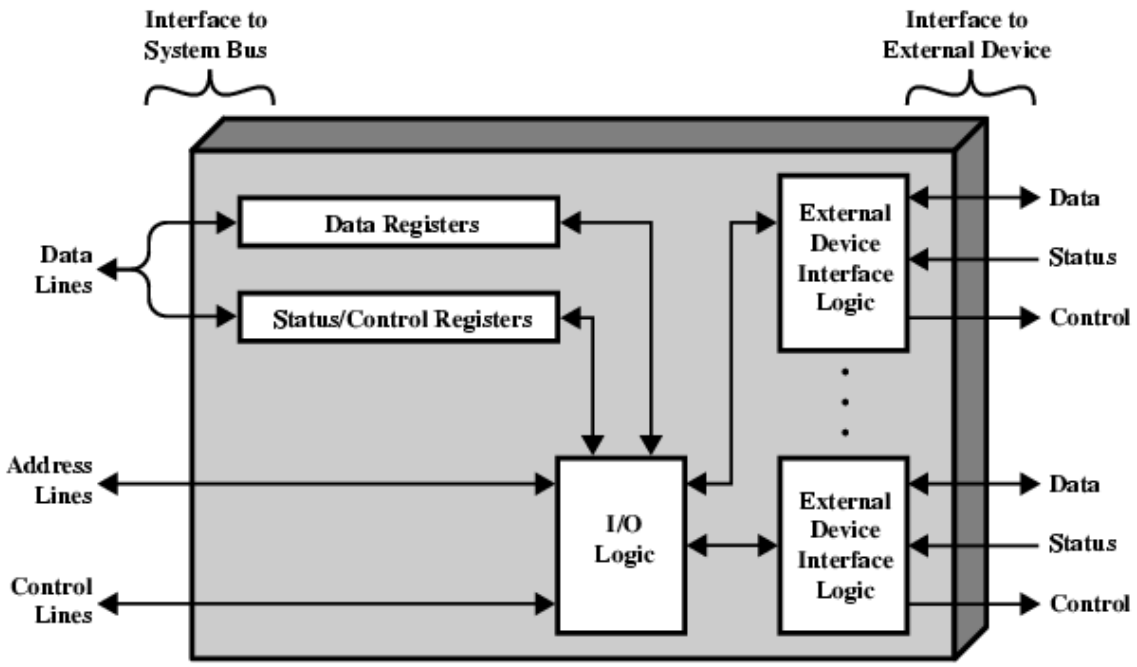


Diagramma modulo I/O



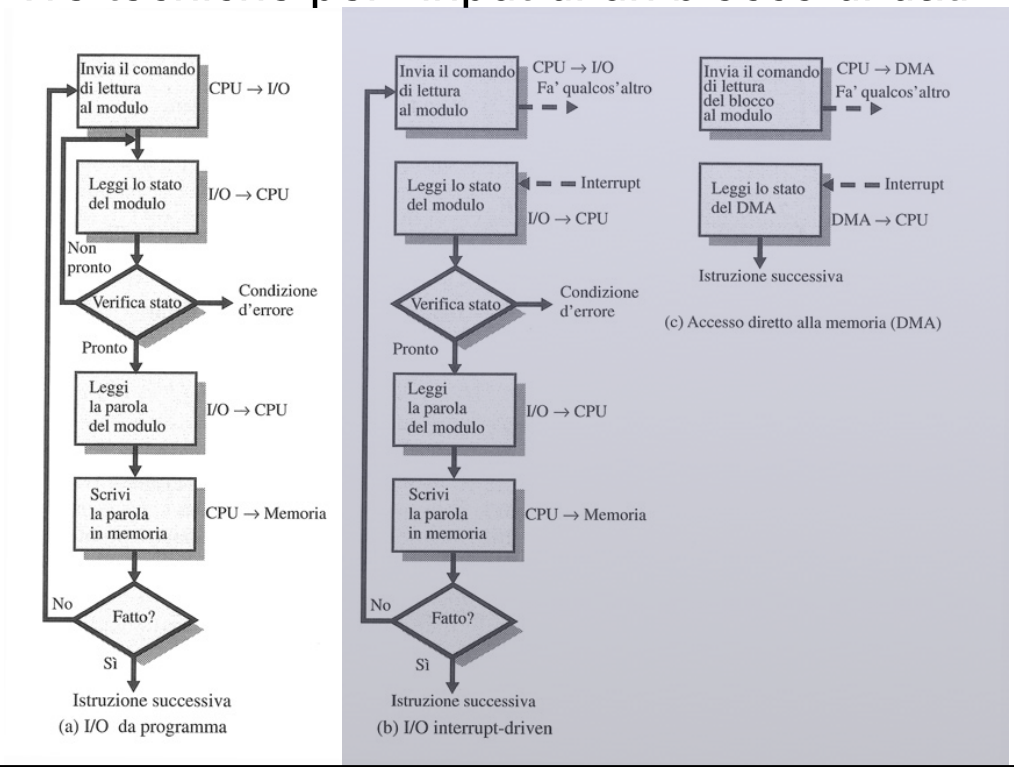
Caratteristiche modulo I/O

- 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



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

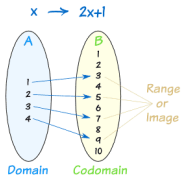
Comandi I/O

- CPU invia indirizzo
  - che identifica modulo (& dispositivo se >1 per modulo)
- CPU invia commando
  - 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

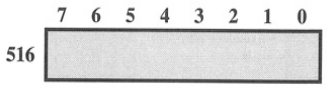
- Nell' I/O da programma il trasferimento dati è molto simile all'accesso alla memoria (dal punto di vista della CPU)
- Ad ogni dispositivo viene assegnato un identificatore unico
- I comandi di CPU riferiscono tale identificatore (indirizzo)

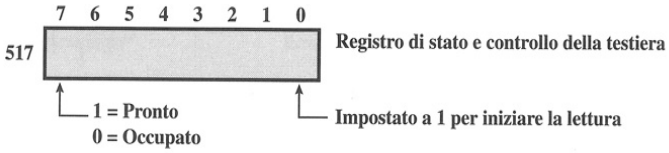
I/O Mapping



- 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 memoria 'normalè accedono anche le 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 di utente (*privilegi*)
    - Con la tecnica della memoria segmentata, *più* aree di I/O possono mappare sul *medesimo* spazio di indirizzamento fisico

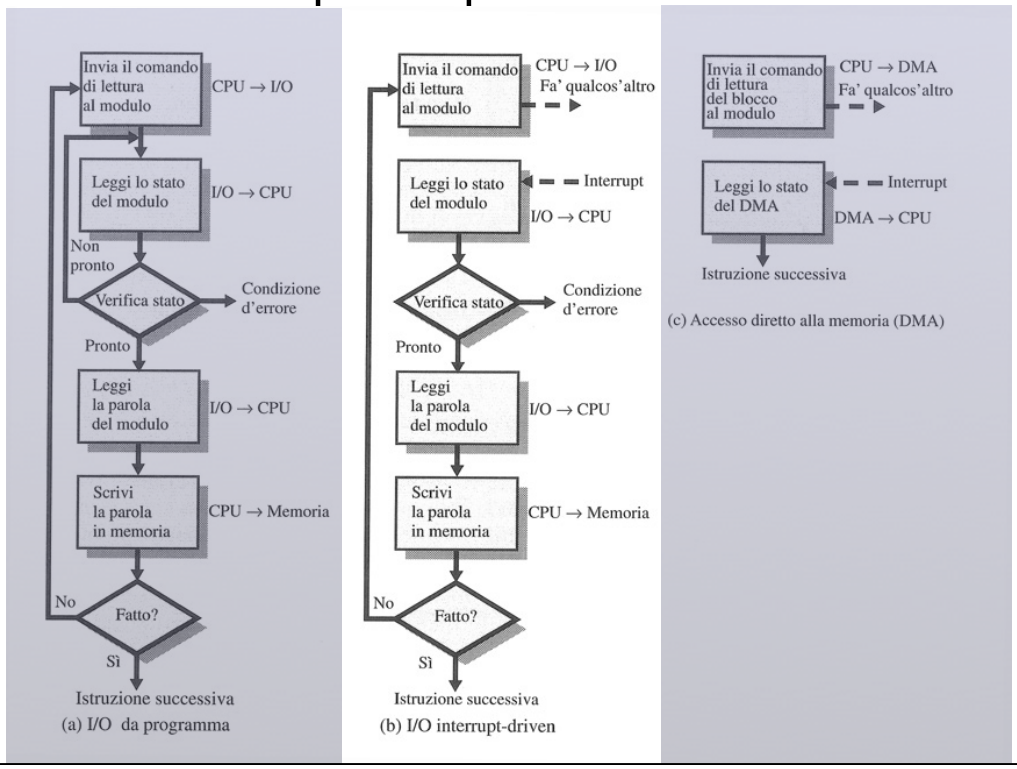
Confronto fra I/O memory mapped e separato

- 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

I/O Interrupt Driven

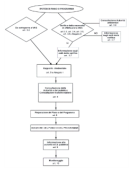
- Evita l'attesa da parte della CPU
- Nessun controllo ripetuto dello stato del dispositivo da parte della CPU
- Il modulo di I/O interrompe la CPU quando è pronto

Tre tecniche per l'input di un blocco di dati

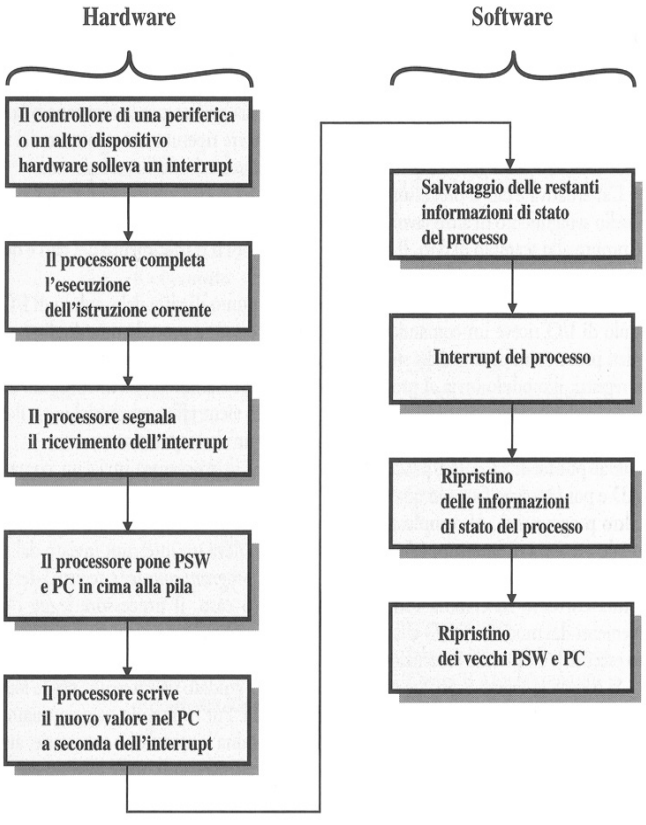


I/O Interrupt Driven Operazioni Base

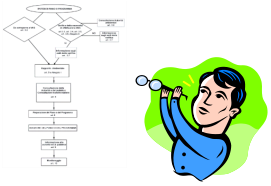
- CPU rilascia comando di lettura
- Modulo I/O ottiene i dati dalla periferica mentre la CPU svolge altro lavoro
- Modulo I/O interrompe la CPU
- CPU richiede i dati al modulo I/O
- Modulo I/O trasferisce i dati



Semplice elaborazione delle interruzioni

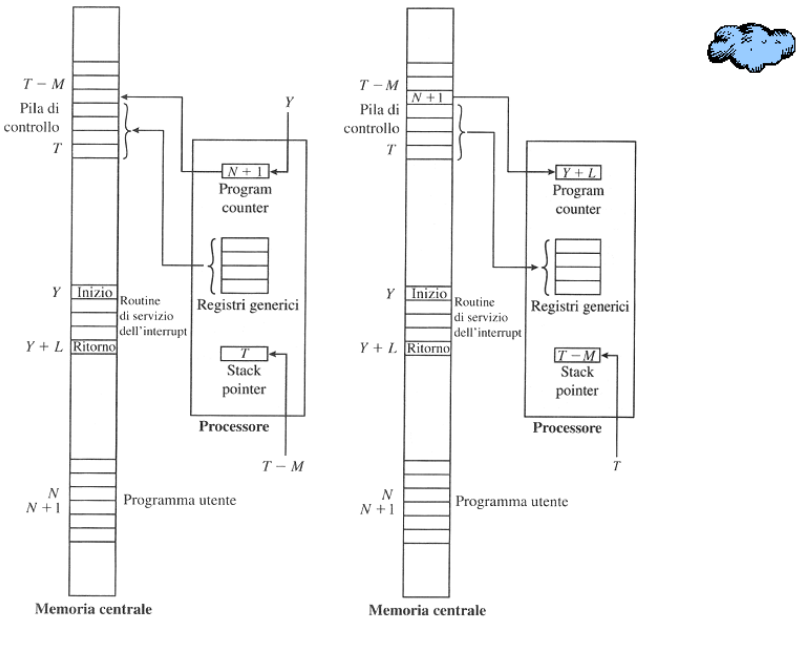


Punto di Vista della CPU



- Rilascia comando di lettura
- Esegue altro lavoro
- Controlla se c'è interruzione alla fine di ogni ciclo di istruzione (ciclo fetch/execute con trattamento delle interruzioni)
- Se interruzione presente:
  - Salva contesto (PC e registri)
  - Interruzione del processo corrente e elaborazione interrupt
    - Lettura dati da modulo I/O e scrittura in memoria

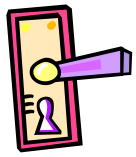
Cambiamento in Memoria e Registri per un Interrupt



(a) L'interrupt avviene dopo un'istruzione all'indirizzo N

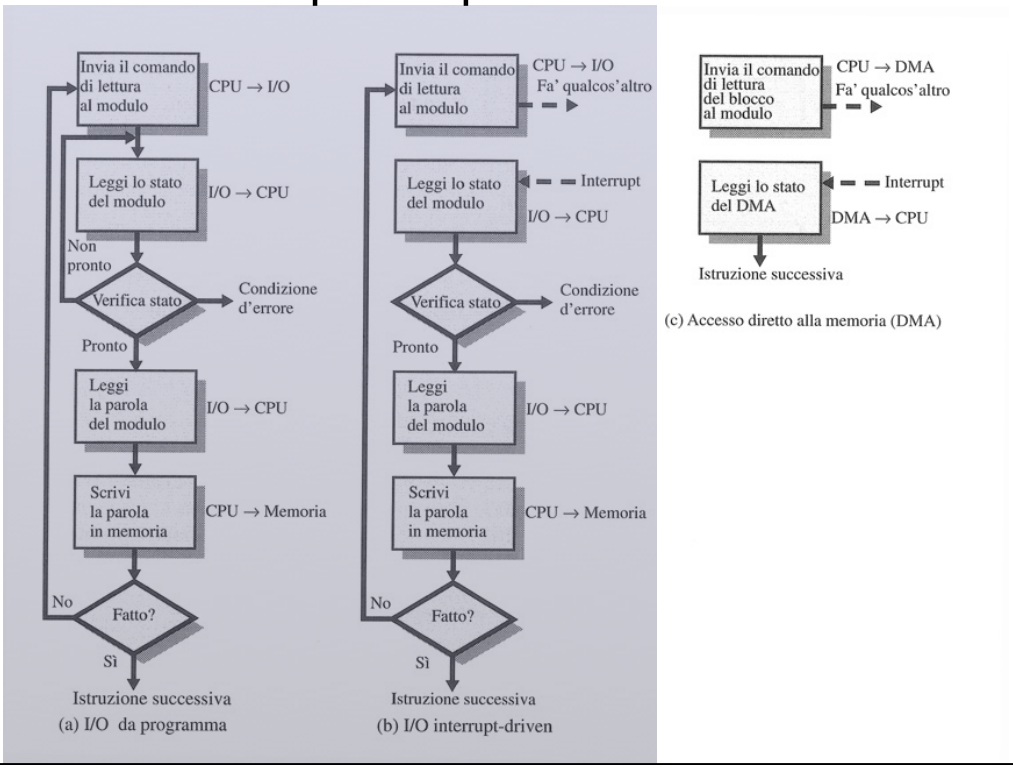
(b) Ritorno dall'interrupt

Accesso Diretto alla Memoria (Direct Memory Access)



- Sia I/O da programma che interrupt driven richiedono l'intervento attivo della CPU
  - Il tasso di trasferimento dei dati è limitato
  - CPU è impegnata in tali operazioni e non può svolgere altre attività per lei più specifiche
- DMA riduce l'intervento della CPU al minimo necessario

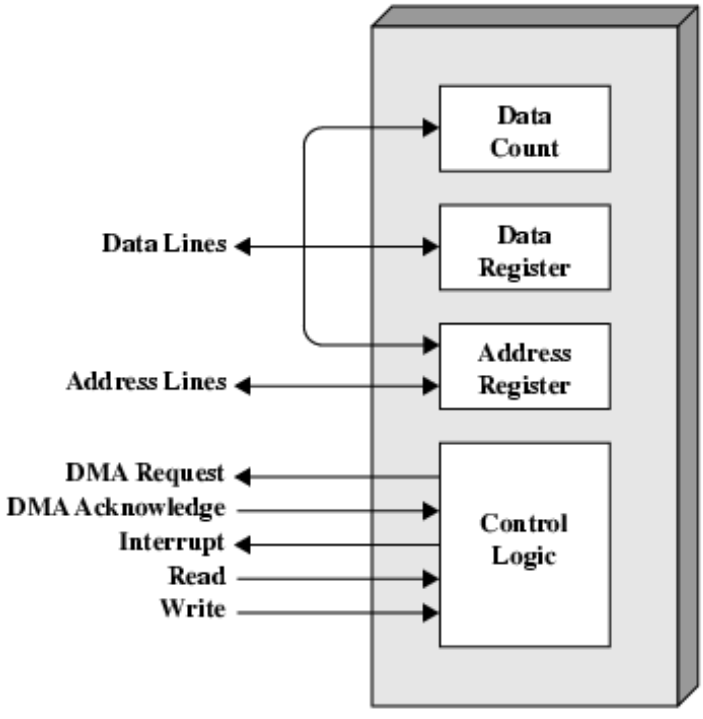
Tre tecniche per l'input di un blocco di dati



Funzione del DMA

- Modulo (hardware) addizionale, connesso al bus
- Il controllore DMA sostituisce la CPU per la maggior parte delle attività di I/O

Diagramma di un tipico modulo DMA



Operazioni DMA

- CPU comunica al controllore DMA:
  - lettura/scrittura
  - indirizzo dispositivo
  - indirizzo iniziale in memoria del blocco dati coinvolto nell'operazione (da dove leggere o dove scrivere i dati)
  - quantità di dati da trasferire
- CPU prosegue eseguendo altre attività
- Il controllore DMA si occupa del trasferimento dei dati ("colloquia" direttamente con la memoria centrale)
- Il controllore DMA invia un interrupt alla CPU quando ha terminato il trasferimento

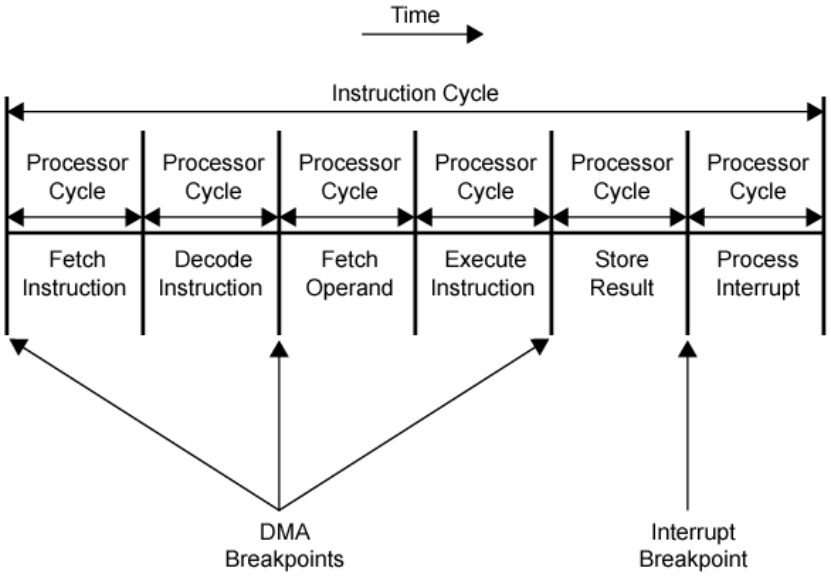
Trasferimento dati DMA

- Il *DMA controller* può accedere al canale dati in uno di due modi
  - **Una parola alla volta**, sottraendo di tanto in tanto alla CPU il controllo sul canale (*cycle stealing*)
  - **Per blocchi**, prendendo possesso del canale per una serie di trasferimenti (*burst mode*)
  - La CPU è bloccata in entrambi i casi, ma il *burst mode* è **più efficace** perché l'acquisizione del canale è onerosa

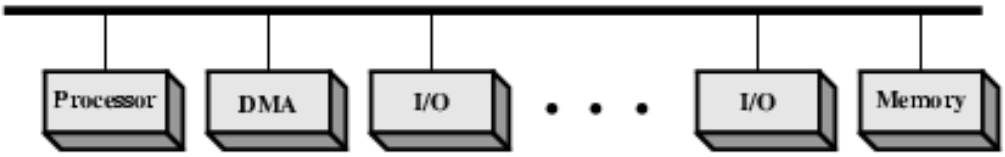
Trasferimento dati DMA
(*cycle stealing*)

- Il controllore DMA prende possesso del bus per un ciclo
- Trasferisce una parola (word) di dati
- Non è una interruzione
  - La CPU non cambia contesto
- La CPU rimane “sospesa” proprio nel momento prima che acceda al bus
  - Ad esempio, prima del caricamento di un dato e/o operando o di una scrittura
- Rallenta la CPU ma non così tanto come nel caso in cui sia la CPU stessa ad occuparsi del trasferimento dati

Breakpoint di DMA e di Interrupt durante un ciclo di istruzione

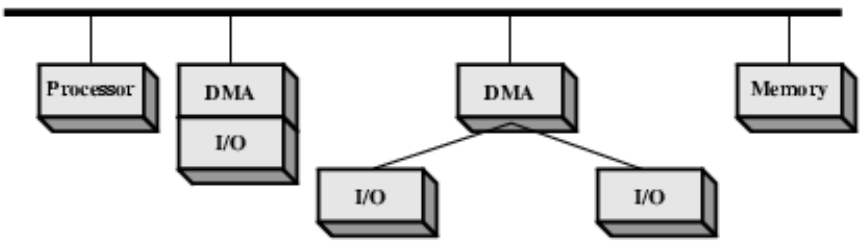


Configurazioni DMA



- Bus singolo, controller DMA isolato
- Ogni trasferimento usa il bus due volte
  - da I/O a DMA e poi da DMA alla memoria
- CPU perde il possesso del bus due volte

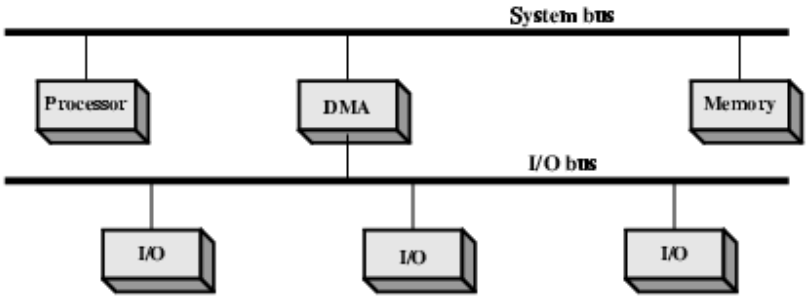
Configurazioni DMA



(b) Single-bus, integrated DMA-I/O

- Bus singolo, controllore DMA integrato con I/O
- Può controllare più di un dispositivo
- Ogni trasferimento usa il bus una volta
  - da DMA a memoria
- CPU perde il controllo del bus una sola volta

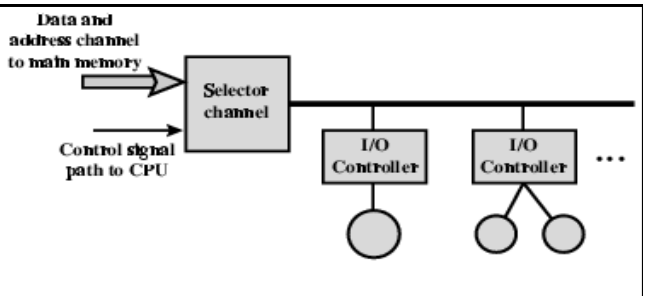
Configurazioni DMA



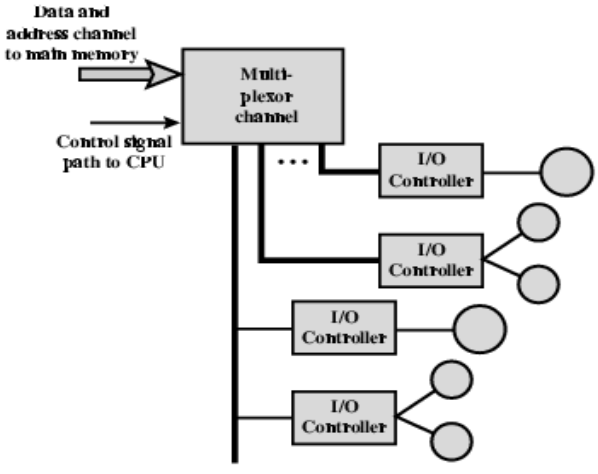
(c) I/O bus

- Bus di I/O separato
- DMA necessita di una sola interfaccia I/O
- Ogni trasferimento usa il bus di sistema una sola volta
  - da DMA a memoria
- CPU perde il controllo del bus una sola volta

Canali I/O



(a) Selector



(b) Multiplexor