



Gestione dell'Input / Output

Prof. Alberto Borghese
Dipartimento di Informatica
alberto.borghese@unimi.it

Università degli Studi di Milano

Riferimento Patterson: 6.3, 6.4 e 6.6



Sommario

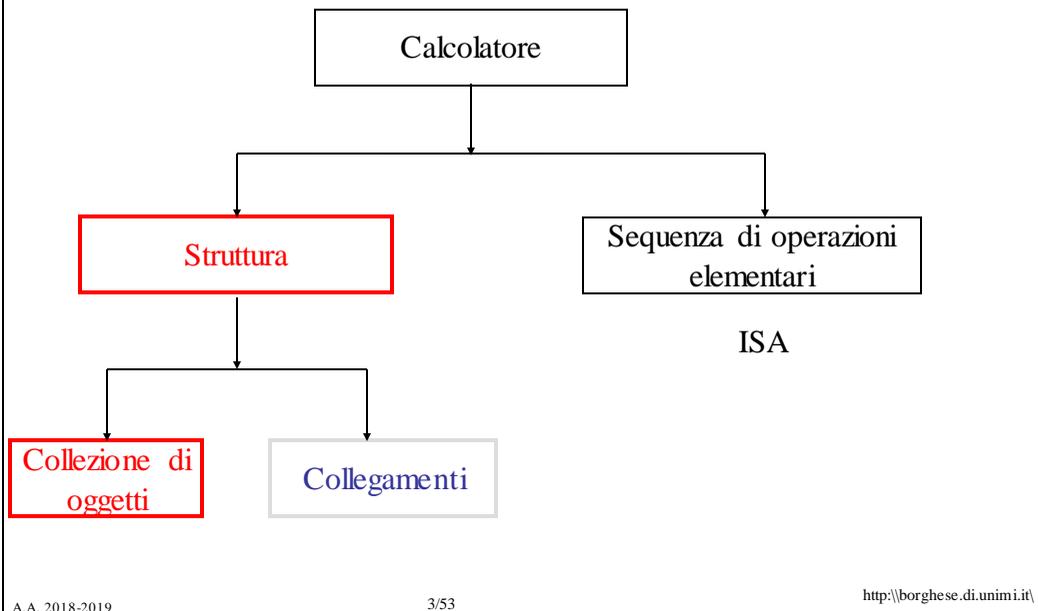
I collegamenti tra CPU e gli altri componenti

Trasferimento tra I/O e CPU

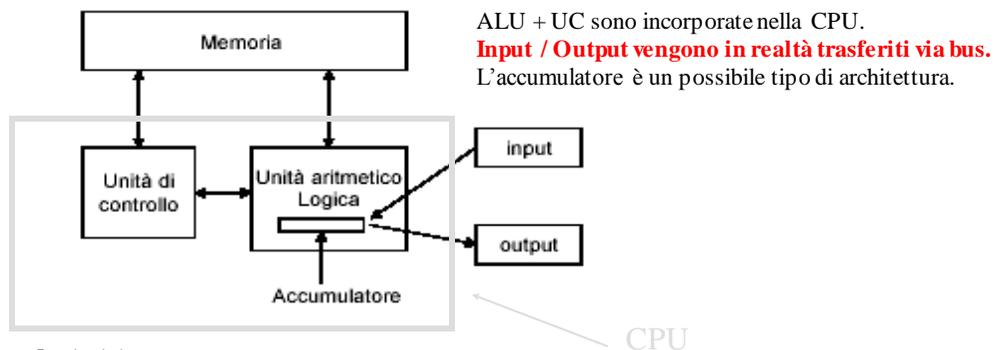
Dischi



Descrizione di un elaboratore



Architettura di Von Neumann



I principi:

- I dati e le istruzioni sono memorizzate in una memoria read/write.
- Il contenuto della memoria può essere recuperato in base alla sua posizione, e non è funzione del tipo di dato.
- L'esecuzione procede sequenzialmente da un'istruzione alla seguente.
- Già vista e modificata (evoluzione nel tempo).



I/O



E' la parte più complessa di un'architettura per la sua variabilità

Dispositivi eterogenei per:

velocità di trasferimento.

latenze.

sincronismi.

modalità di interazione (con l'uomo o con una macchina)



Dispositivi di I/O - classificazione

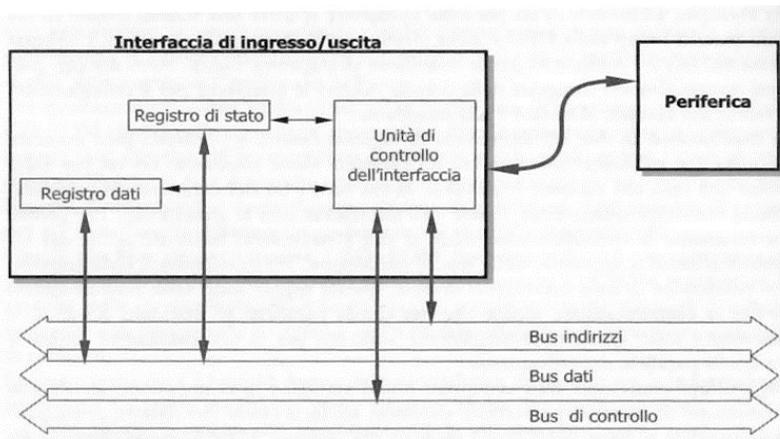


| Device | Behavior | Partner | Data rate (Mbit/sec) |
|----------------------|-----------------|---------|----------------------|
| Keyboard | Input | Human | 0.0001 |
| Mouse | Input | Human | 0.0038 |
| Voice Input | Input | Human | 0.2640 |
| Sound Input | Input | Machine | 3.0000 |
| Scanner | Input | Human | 3.2000 |
| Voice output | Output | Human | 0.2640 |
| Sound output | Output | Human | 8.0000 |
| Laser printer | Output | Human | 3.2000 |
| Graphics display | Output | Human | 800.0000–8000.0000 |
| Cable modem | Input or output | Machine | 0.1280–6.0000 |
| Network/LAN | Input or output | Machine | 100.0000–10000.0000 |
| Network/wireless LAN | Input or output | Machine | 11.0000–54.0000 |
| Optical disk | Storage | Machine | 80.0000–220.0000 |
| Magnetic tape | Storage | Machine | 5.0000–120.0000 |
| Flash memory | Storage | Machine | 32.0000–200.0000 |
| Magnetic disk | Storage | Machine | 800.0000–3000.0000 |

+ **latenza**



Modalità trasferimento dati



Parallela (bus PCI, bus processore-memoria).

Seriale (RS232, RS432). 1 bit alla volta.

Seriale ad alta velocità: USB, Firewire (IEEE 1394), PCIe.

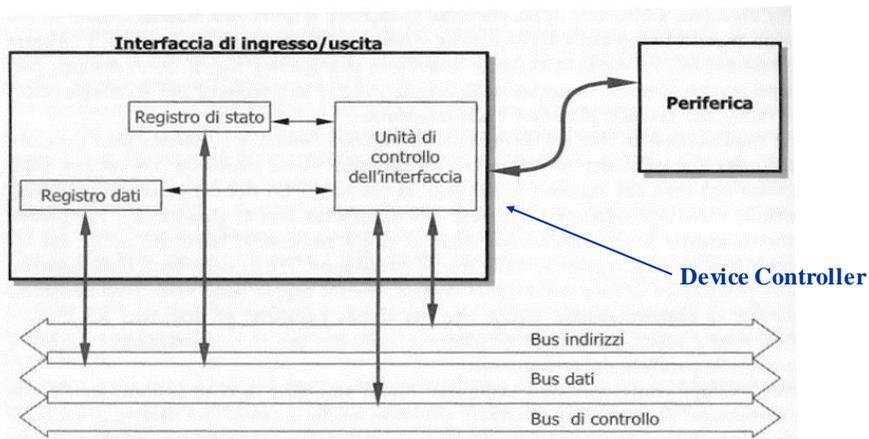


Bus & buffer

- I dispositivi sono collegati al bus tramite **porte**.
- I dispositivi collegati al bus variano in termini di velocità dell'esecuzione delle operazioni \Rightarrow necessario un meccanismo di sincronizzazione per garantire il trasferimento efficiente delle informazioni sul bus.
- Tipicamente all'interno delle unità che utilizzano il bus sono presenti dei **registri di buffer** per mantenere l'informazione durante i trasferimenti e non vincolarsi alla velocità del dispositivo più lento connesso al bus.
- All'interno dell'ampiezza di banda massima si può:
 - ◆ **Aumentare la velocità di trasferimento**. Buffer grossi.
 - ◆ **Ridurre i tempi di risposta (latenza)**. Buffer piccoli.



Porta I/O



Segnali di controllo: *Busy, Ack, Interrupt...*

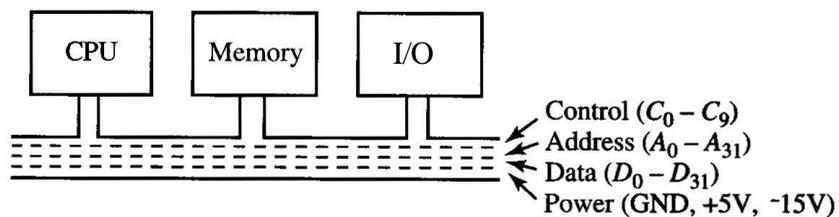
Registri:

- Dati (registro o buffer di memoria)
- Stato: situazione della periferica (idle, busy, down....) e comando in esecuzione.

Il driver agisce inviando alla periferica comandi opportuni e dati.



Il bus (connessione a cammino comune)



Pathway che connette tutti i dispositivi in modo bidirezionale (a partire dal PDP-8, omnibus, 1965). Trasferimento parallelo dei dati. Connessioni bidirezionali.

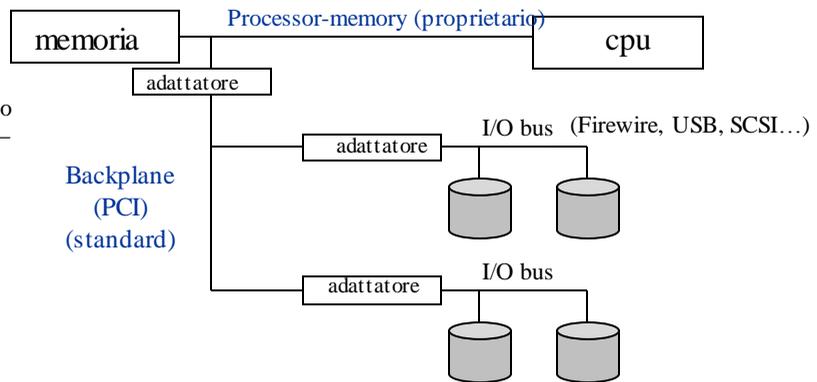
Principali vantaggi della struttura a bus singolo:

- elevata flessibilità
- bassi costi.
- Problema: i dispositivi non possono trasmettere contemporaneamente.

Le architetture contengono uno o più bus che collegano questi tre componenti.



La gerarchia dei bus



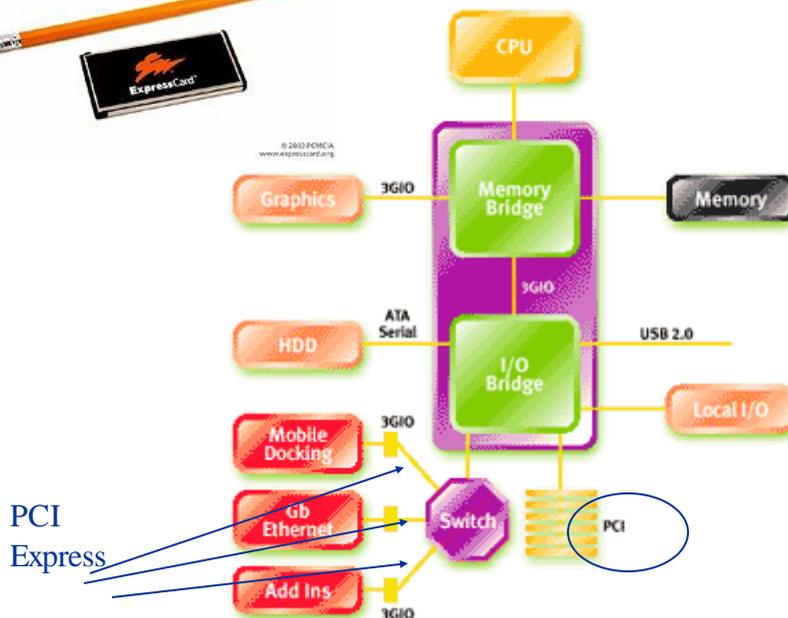
Gli adattatori sono dispositivi attivi – bridges.

Backplane
(PCI)
(standard)

Processore-Memoria (cache): lunghezza ridotta, molto veloci, tipicamente sincroni.

Back-plane: notevole lunghezza, molti device connessi.

I/O: servono per far coesistere la memoria, il processore e i dispositivi di I/O su di un unico bus.
Tipicamente asincroni.



PCI
Express

Struttura gerarchica –
gerarchia + stella



Il PCI Express



| Caratteristica | PCI | PCI Express |
|---|---|--|
| Tipo di bus | Back-plane | Versatile (back-plane, I/O) |
| Ampiezza di base del bus (numero di segnali per i dati) | 32-64 (collegamento bidirezionale) | 4 + 4 (In/Out) (collegamento monodirezionale) |
| Numero di dispositivi master | molti | 1 |
| Temporizzazione | Sincrono 33-66Mhz | Sincronizzato: 2.5GHz |
| Modalità di funzionamento | Sincrona parallela (1,024Mbit/s – 4,096Mbit/s) | Sincrona seriale (2,5Gbit/s) |
| Ampiezza di banda di picco teorica | 133-512MB/s (PCI64) | 300MB/s per direzione |
| Ampiezza di banda stimata raggiungibile per bus di base | 80MB/s | 1,200MB/s (x 4 linee) |
| Massimo numero di dispositivi | 1024 (32 dispositivi per segmento) | 1 |
| Massima lunghezza del bus | 0,5 metri | 0,5 metri |
| #Mbyte / s / linea | 4-16 | 75 + 75 |
| Nome dello standard | PCI | PCI - Express |

bottleneck

A.A. 2018-2019

13/53

ghese.di.unimi.it



La comunicazione tra i componenti

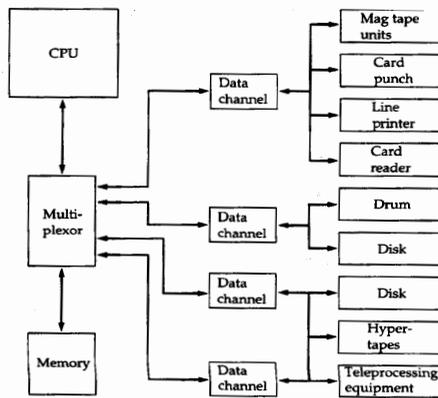


Figure 2.5 An IBM 7094 Configuration

Switch centralizzato (multiplexor)

Architettura a nodo comune (a bus) (cf. bus PCI)

Programma di "canale"

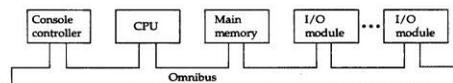


Figure 2.9 PDP-8 Bus Structure

A.A. 2018-2019

14/53



| | Intel 5000P chip set | Intel 975X chip set | AMD 580X CrossFire |
|--|------------------------|---------------------------|---------------------------|
| Target segment | Server | Performance PC | Server/Performance PC |
| Front Side Bus (64 bit) | 1066/1333 MHz | 800/1066 MHz | — |
| Memory controller hub ("north bridge") | | | |
| Product name | Blackbird 5000P MCH | 975X MCH | |
| Pins | 1432 | 1202 | |
| Memory type, speed | DDR2 FBDIMM 667/533 | DDR2 800/667/533 | |
| Memory buses, widths | 4 x 72 | 1 x 72 | |
| Number of DIMMs, DRAM/DIMM | 16, 1 GB/2 GB/4 GB | 4, 1 GB/2 GB | |
| Maximum memory capacity | 64 GB | 8 GB | |
| Memory error correction available? | Yes | No | |
| PCIe/External Graphics Interface | 1 PCIe x16 or 2 PCIe x | 1 PCIe x16 or 2 PCIe x8 | |
| South bridge interface | PCIe x8, ESI | PCIe x8 | |
| I/O controller hub ("south bridge") | | | |
| Product name | 6321 ESB | ICH7 | 580X CrossFire |
| Package size, pins | 1284 | 652 | 549 |
| PCI-bus: width, speed | Two 64-bit, 133 MHz | 32-bit, 33 MHz, 6 masters | — |
| PCI Express ports | Three PCIe x4 | | Two PCIe x16, Four PCI x1 |
| Ethernet MAC controller, interface | — | 1000/100/10 Mbit | — |
| USB 2.0 ports, controllers | 6 | 8 | 10 |
| ATA ports, speed | One 100 | Two 100 | One 133 |
| Serial ATA ports | 6 | 2 | 4 |
| AC-97 audio controller, interface | — | Yes | Yes |
| I/O management | SMBus 2.0, GPIO | SMBus 2.0, GPIO | ASF 2.0, GPIO |

I bridge



Arbitraggio del bus



Protocollo La comunicazione su bus deve essere regolata attraverso un **protocollo di comunicazione**.

Viene introdotto il concetto di **bus master (padrone del bus)**, il cui scopo è quello di controllare l'accesso al bus.

L'architettura più semplice è quella che prevede un unico bus master (il processore) in cui tutte le comunicazioni vengono mediate dal processore stesso.

Questo può creare un collo di bottiglia. Ad esempio nel caso di trasferimento di dati da I/O a memoria.

Si utilizza allora un'architettura con più dispositivi master.

In questo caso occorre definire e rispettare una policy che coordini i vari dispositivi bus master. Questa policy si chiama di **arbitraggio** del bus. Un solo dispositivo alla volta può essere master, tutti gli altri ascoltano.

Questo è il principale inconveniente dei bus a nodo comune.



Protocollo di arbitraggio



Occorre stabilire quale master autorizzare all'utilizzo del bus.

Ad ogni dispositivo viene assegnata una *priorità*.

Il dispositivo a priorità maggiore può accedere prima al bus.

Meccanismo di accesso al bus diventa:

1. Richiesta del bus (*bus request*)
2. Assegnamento del bus (*bus grant*)

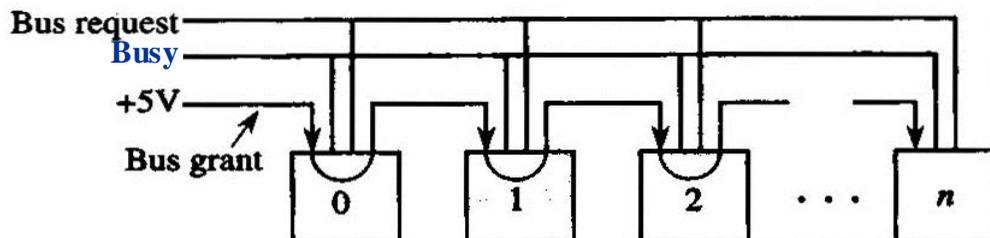
Problema è assicurare una *fairness*.

Compromesso tra *fairness* e priorità.

Un arbitro si preoccupa quindi di gestire *bus request* e *bus grant*.



Arbitraggio distribuito in Daisy Chain



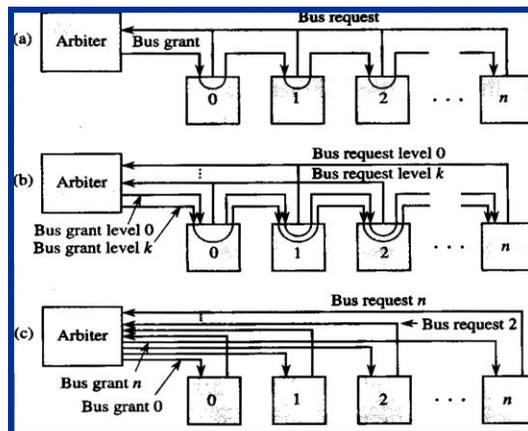
Viene introdotto il segnale di Busy.



Schemi di arbitraggio centralizzati



Arbitraggio
In Daisy Chain



- Arbitraggio centralizzato parallelo non scala con il numero di dispositivi.
- Nell'arbitraggio centralizzato con priorità, un dispositivo elabora un solo segnale di grant, gli altri segnali di grant vengono fatti passare inalterati.



Arbitraggio distribuito con autoselezione



In questo schema un dispositivo che vuole prendere il controllo del bus, deve:

- 1) Inviare il segnale di richiesta del bus.
- 2) Scrivere sul bus il codice che lo identifica.
- 3) Controllare se il bus è libero.
- 4) Se il bus non è occupato ma ci sono richieste contemporanee di bus, controllare il codice dei dispositivi che hanno fatto richiesta.
- 5) Occupare il bus se è libero o i dispositivi hanno priorità minore: inviare 0 sulla linea di bus grant ai dispositivi successivi, asserisce la linea di busy (il bus è occupato), altrimenti non fare nulla. *Ciascun dispositivo è arbitro.*
- 6) Deassertire la richiesta del bus.

Problema: **rilevamento delle collisioni**. Occorre prevedere un segnale di ricevuto.



Sommario



I collegamenti tra CPU e gli altri componenti

Trasferimento tra I/O e CPU

Dischi



Modalità di trasferimento dati



Controllo da programma:
con polling.

Mediante interrupt:
con DMA.



Funzionamento di un driver



Funzione dei driver:

- Controllano l'operato dei device controller.
 - Gestiscono lo scambio dei dati dal controller (registro dati) e la memoria.
1. CPU richiede alla periferica (controller della periferica) l'esecuzione di un'operazione di read o di write.
 2. I dati coinvolti nell'operazione devono essere trasferiti da e verso la memoria centrale.

Per potere eseguire un'operazione di read / write, occorre spesso una serie di operazioni sul dispositivo, che vengono eseguite attraverso il controller.

Esempio: una stampante ha 1 registro dati ed 1 registro di stato. Il registro di stato contiene il bit done, che viene impostato a 1 quando il carattere è stato stampato; ed il bit error che, indica se ci sono problemi. Il processore deve controllare che non ci siano errori e che il bit di done sia stato settato ad 1 prima di inviare un altro dato.

Per potere inviare i comandi al controller, occorre prima avere individuato il controller giusto!!
A ciascun dispositivo viene dato uno o più numeri -> indirizzo personalizzato.

2 modalità:

- Memory-mapped
- Istruzioni speciali di I/O



Istruzioni speciali di I/O



Istruzioni appartenente alla ISA che indirizzano direttamente il dispositivo (i registri del dispositivo):

- Numero del dispositivo
- Parola di comando (o indirizzo della parola che contiene il comando)

Sul bus è possibile inviare il numero del dispositivo su un insieme di linee dedicate.

Ci saranno linee dedicate anche ai segnali di controllo (read / write).

I dati viaggeranno sulle linee dedicate ai dati.

Rendendo le istruzioni illegali al di fuori del kernel mode del processore, i programmi utenti non accedono direttamente ai device controller.

Esempio di architetture di questo tipo: Intel IA-32, IBM370.

```
out 70h, ax    # Trasferisci alla porta di I/O 70, il contenuto di AX
```



Indirizzamento memory-mapped

- I registri del device controller sono considerati come celle di memoria RAM.
- I loro indirizzi saranno diversi da quelli delle celle di memoria RAM effettive.
- Il processore esegue operazioni di I/O come se fossero operazioni di lettura/scrittura in memoria.

Esempio:

```
sw $s0, indirizzo
```

```
lw $s0, indirizzo
```

dove l'indirizzo è al di fuori dallo spazio fisico della memoria.

- I controller ascoltano tutti i segnali in transito sul bus (*bus snooping*) e si attivano solamente quando riconoscono sul bus indirizzi, l'indirizzo corrispondente alla propria locazione di memoria.
- Gli indirizzi riservati ai registri del controller fanno di solito riferimento alla porzione di memoria riservata al SO e non accessibile quindi al programma utente.
- I programmi utente devono quindi passare dal SO per accedere a questi indirizzi riservati (**modalità kernel**) e quindi effettuare operazioni di I/O. Questo è quanto viene fatto ricorrendo alle System Call.



I/O a controllo di programma

E.g. chiamata `syscall` per la stampa di una stringa.

La periferica ha un ruolo passivo. Il processore esegue tutto il lavoro.

Svantaggio: La CPU dopo avere predisposto il controller all'esecuzione dell'I/O si ferma e si mette ad interrogare il registro di stato della periferica in attesa che il **ready bit** assuma un determinato valore. Stato *busy waiting* o *spin lock*.

```
begin
```

```
1. Predisponi i registri del controller ad effettuare una  
operazione di lettura.
```

```
2. While (ready-bit == 0) do;           // spin lock (o busy waiting)
```

```
3. Carica il dato acquisito;
```

```
end;
```



Esempio: Receiver (tastiera)



NB i dispositivi vengono indirizzati tramite gli indirizzi "alti".

```
.text
.globl main
main:
    li $t0, 0x8000 0000 # indirizzo del receiver control register (2Gbyte)
    li $t2, 0x8000 0004 # indirizzo del receiver data register

# Ciclo di lettura di un carattere
ciclo:  lw $t1, 0($t0)           # Contenuto del registro di controllo
        beq $t1, $zero, ciclo  # if ($t1 != 0) esci

        lw $a0, 0($t2)         # Caricamento del dato in a0

        li $v0, 10             # exit
        syscall
```



I/O a gestione di programma - costo



Ipotesi:

- 1) Tastiera gestita a controllo di programma che opera a 0,01Kbyte/s.
- 2) Frequenza di clock: 50Mhz.

Determinare il tempo in cui verrebbe effettivamente utilizzata la CPU per trasferire 1 word, tenendo conto che ci vogliono 20 cicli di clock per trasferire ogni byte.

$$T = 4 \text{ [byte]} / 10 \text{ [byte]} / [\text{s}] = 0,4\text{s}$$

$$\#\text{cicli_clock} = 50 * 10^6 \text{ [#cicli]} / [\text{s}] * 0,4 \text{ [s]} = 20,0 * 10^6 \text{ [#cicli]}$$

Invece ne utilizza solo $20 * 4 = 80$ per trasferire i dati.

$$\%\text{Sfruttamento della CPU è: } (80 / 20,0 * 10^6) * 100 = 0,0004\%$$

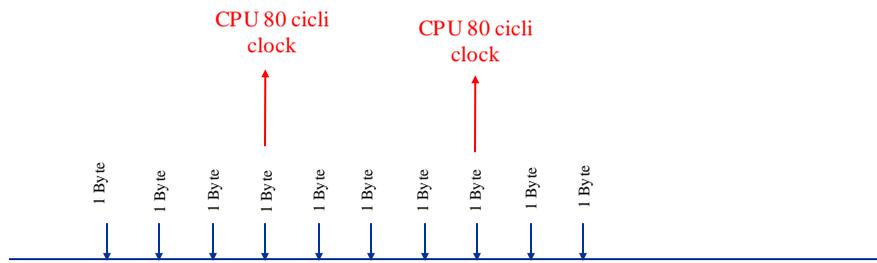


Visualizzazione temporale



Hard-disk:

Trasferimento di 1 word
50 Kbyte/sec
Costo del trasferimento 80 cicli di clock



I/O a gestione di programma - costo



Ipotesi:

- 1) Hard-disk1 che opera a 50Kbyte/s.
- 2) Hard-disk2 che lavora a 2MByte/s.
- 3) Frequenza di clock: 100Mhz.

Determinare la percentuale di tempo in cui verrebbe effettivamente utilizzata la CPU per trasferire 1 word, tenendo conto che ci vogliono 20 cicli di clock per ogni byte (approssimare $1k = 1,000$).

Hard-disk1: $4 \text{ byte} / 50k\text{byte/s} = 80\mu\text{s}$ fase attiva =>
 $100 \times 10^6 / \text{\#cicli_clock/s} * 80 \times 10^{-6} \text{ (tempo_trasferimento in \#cicli)} =$
 $8000 \text{\#cicli_clock} \Rightarrow \% \text{sfruttamento} = (20 * 4) / 8 \times 10^3 = 1\%$

Hard-disk2: $4 \text{ byte} / 2M\text{byte/s} = 2\mu\text{s}$ fase attiva =>
 $50 \times 10^6 / \text{\#cicli_clock/s} * 2 \times 10^{-6} \text{ (tempo_trasferimento in \#cicli)} =$
 $100 \text{ cicli_clock} \Rightarrow \% \text{sfruttamento} = (20 * 4) / 100 = 80\%$.

Spin-lock è un tempo dedicato all'attesa.



Polling



Interrogazione del registro di stato della periferica.

Ciclo di polling: durante un ciclo di **busy-waiting** su un dispositivo si esegue il **polling** sugli altri dispositivi di I/O.

Quando una periferica necessita di un qualche intervento, si soddisfa la richiesta e si prosegue il ciclo di polling sugli altri I/O.

```
// Leggi dato da perif_x
begin
a.  Predisponi i registri dei controller ad eseguire una read;
b.  if(ready_bit (perif_1) == 1) servi perif_1; #Esempio: Mouse
    if (ready_bit (perif_2) == 1) servi perif_2; #Esempio: Hard disk1
    if (ready_bit (perif_3) == 1) servi perif_3; #Esempio: Hard disk2
    ....
    if (ready_bit (perif_n) == 1) servi perif_n;
    UpdateFunctions; #Programma di gestione in funzione della
                    #situazione delle periferiche (sistemi di controllo)
    goto b;
end;
```



I/O a gestione di programma – costo polling



Ipotesi:

- 1) Costo del polling (# cicli di clock per un'operazione di polling, costituita da trasferimento del controllo alla procedura di polling, accesso al dispositivo di I/O, trasferimento dati e ritorno al programma utente): 400 cicli di clock. Se non si trasferiscono dati il costo è considerato trascurabile.
- 2) Frequenza di clock: 500Mhz
- 3) Parola di 4 byte.

Determinare l'impatto del polling per 3 dispositivi diversi:

- A) Mouse. Deve essere interrogato almeno 30 volte al secondo per non perdere alcun movimento dell'utente.
- B) Hard disk. Trasferisce dati al processore in parole da 16 bit ad una velocità di 50 Kbyte/s.
- C) Hard disk. Trasferisce dati al processore in blocchi di 4 parole e può trasferire 4 Mbyte/s.

Supponiamo che il costo del trasferimento per la CPU sia dovuto principalmente alle operazioni di preparazione e di richiesta di bus, mentre il costo per la CPU dovuto al trasferimento tramite il bus sia trascurabile (ad esempio perchè viene utilizzata la modalità *burst* delle DRAM).



I/O a gestione di programma - costo



Mouse:

(Per ogni accesso trasferisco 2 byte: x, y)
Occorrono quindi 30 accessi/s \rightarrow transfer rate = 60Byte/s
In termini di cicli di clock: $30 \times 400 = 12,000$ cicli_clock/s
 $12,000 / 500,000,000 = 0,000024s \Rightarrow 0,0024\%$
Piccolo impatto sulle prestazioni.

Hard disk1:

Per ogni accesso possiamo trasferire (half word).
Occorrono quindi 25k accessi/s.
In termini di cicli di clock: $25k \times 400 = 10M$ cicli_clock/s $\Rightarrow 2\%$
Medio impatto sulle prestazioni.

Hard disk2:

Per ogni accesso possiamo trasferire 16byte.
Occorrono quindi 250k accessi/s
In termini di cicli di clock: $250k \times 400 = 100M$ cicli_clock/s $\Rightarrow 20\%$
Alto impatto sulle prestazioni.

Ciclo di polling 22,0024% utilizzo CPU



I/O a controllo di programma



- I miglioramenti del polling rispetto al controllo di programma sono molto limitati.
- I problemi principali del polling (e dell'I/O a controllo di programma) sono:
 - Con periferiche lente, un eccessivo spreco del tempo di CPU, che per la maggior parte del tempo rimane occupata nel ciclo di busy waiting.
 - Con periferiche veloci, il lavoro svolto dalla CPU è quasi interamente dovuto all'effettivo trasferimento dei dati.

Il polling funziona bene per i sistemi embedded.

Nei dischi, si potrebbe attivare il polling quando ne è richiesto l'utilizzo.
Posizionamento delle testine (\Rightarrow spin lock)

Occorre disaccoppiare il trasferimento tra I/O e memoria e il funzionamento della CPU, per trasferimenti di grandi quantità di dati.



Interrupt



E' la periferica a segnalare al processore (su una linea del bus dedicata) di avere bisogno di attenzione.

La segnalazione viene chiamata *interrupt* perché interrompe il normale funzionamento del processore (*interrupt request*).

Quando il processore “se ne accorge” (fase di fetch), riceve un segnale di *interrupt acknowledge*.

Viene eseguita una procedura speciale, chiamata *procedura di risposta all'interrupt*.

Problema: Il programma utente deve potere procedere dal punto in cui è stato interrotto → *Salvataggio del contesto*.



Interrupt – esempio – comando print



1. Invio del comando print.
2. Se la periferica è in stato busy, CPU torna alla sua attività, scaricando sul registro di controllo la richiesta di output.
3. Quando la periferica diventa ready, viene inviato un interrupt.
4. Il programma di risposta all'interruzione, provvederà a trasferire alla periferica il dato che si vuole stampare.



Interrupt – esempio – comando print



1. Invio del comando print.
2. Se la periferica è in stato busy, CPU torna alla sua attività, scaricando sul registro di controllo la richiesta di output.
3. Quando la periferica diventa ready, viene inviato un interrupt.
4. Il programma di risposta all'interruzione, provvederà a trasferire alla periferica il dato che si vuole stampare.



DMA



Tra il momento in cui termina l'invio del comando al controller ed il momento in cui il dato è disponibile sul controller, la CPU può fare altro (tipicamente l'esecuzione di un altro programma).

Il meccanismo interrupt driven non svincola la CPU dal dovere eseguire le operazioni di trasferimento dati.

Per periferiche veloci, le operazioni di trasferimento dati occupano un tempo preponderante rispetto al tempo speso in spin lock.

Per evitare l'intervento della CPU nella fase di trasferimento dati, è stato introdotto il protocollo di trasferimento in Direct Memory Access (DMA).

Viene disaccoppiato il colloquio processore-Memoria dal colloquio IO-Memoria. Questo è reso più facile dalla struttura a bus gerarchici.

Il device controller che gestisce il trasferimento diventa **bus master**.



I passi della DMA

Il DMA controller è un processore specializzato nel trasferimento dati tra dispositivo di I/O e memoria centrale.

Per attivare il trasferimento viene richiesto alla CPU:

1. Spedire al DMA controller il tipo di operazione richiesta
2. Spedire al DMA controller l'indirizzo da cui iniziare a leggere/scrivere i dati.
3. Spedire al DMA controller il numero di byte riservati in memoria.

Per attivare il trasferimento al controller viene richiesta la corretta lettura dello stato della memoria e l'aggiornamento dell'indirizzo a cui trasferire il dato. *E' il controller che gestisce il trasferimento del singolo dato.*



Caratteristiche della DMA

La CPU si svincola completamente dall'esecuzione dell'operazione di I/O.

Il controller avvia l'operazione richiesta e trasferisce i dati da/verso memoria mentre la CPU sta facendo altro.

Dopo avere trasferito tutti i dati, il DMA invia un interrupt alla CPU per segnalare il completamento del trasferimento.

La CPU perciò controlla il (device) controller.



Interrupt – esempio – comando print



1. Invio del comando print.
2. Se la periferica è in stato busy, CPU torna alla sua attività, scaricando sul registro di controllo la richiesta di output.
3. Quando la periferica diventa ready, viene inviato un interrupt.
4. Il programma di risposta all'interruzione, provvederà a trasferire alla periferica il dato che si vuole stampare.



Sommario



I collegamenti tra CPU e gli altri componenti

Trasferimento tra I/O e CPU

Dischi



Dischi

- Consentono di memorizzare dati in modo non volatile.
 - ◆ Dischi magnetici
 - ◆ Dischi a stato solido
 - ◆ Dischi ottici

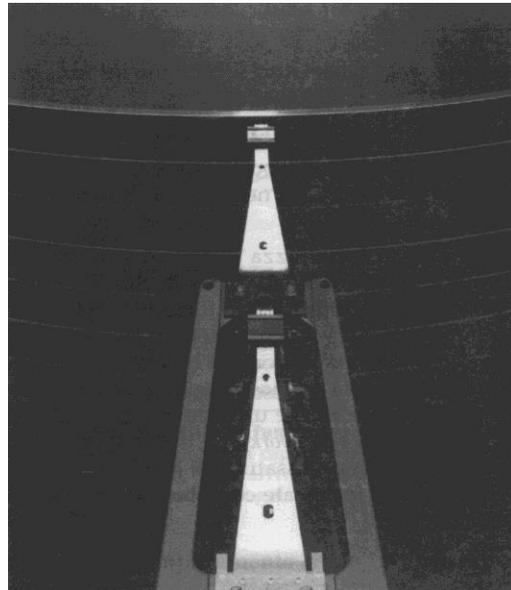
Dischi magnetici

- I dati sono letti/scritti mediante una testina.
- I dischi magnetici sono di due tipi principali:
 - ◆ hard disk
 - ◆ floppy disk (messi in commercio da Apple e Tandy nel 1978). In precedenza esistevano solamente le cassette magnetiche, a loro volta evoluzione dei nastri magnetici immessi sul mercato nel 1934 in Germania dalla IG Farben, ora Basf, per un magnetofono AEG.



Hard disk magnetico

- Costituiti da un insieme di piatti rotanti (da 1 fino a 25) ognuno con due facce, di diametro che va da 2.5cm a 10cm.
- La pila dei piatti viene fatta ruotare alla stessa velocità (5,400 – 15,000 rpm = revolutions per minute).
- Esiste una testina per ogni faccia.
- Le testine si muovono in modo solidale.
- L'insieme delle tracce di ugual posto su piatti diversi è chiamato **cilindro**.
- La quantità di dati che possono essere memorizzati per traccia dipende dalla qualità del disco.





Hard disk – lettura / scrittura

- Per leggere/scrivere informazioni sono necessari tre passi:
 - ◆ la testina deve essere posizionata sulla traccia corretta;
 - ◆ il settore corretto deve passare sotto la testina;
 - ◆ i dati devono essere letti o scritti.
- **Tempo di seek (ricerca):** tempo per muovere la testina sulla traccia corretta.
- **Tempo di rotazione:** tempo medio per raggiungere il settore da trasferire (tempo per 1/2 rotazione). Misurato in rpm (rounds per minute = giri al **minuto**).
- **Tempo di trasferimento:** tempo per trasferire l'informazione.
- A questi tempi va aggiunto il tempo per le operazioni del **controller**.



Hard Disk - Seagate Ceetah 18XL

Dimensioni: 101.6 x 146.1 x 25.4mm
Capacità: 18.2 Gbyte.

Forma: low-profile.
Default Buffer (cache) Size 4,096 Kbytes

Peso: 0.68kg.
Spindle Speed 10,000 RPM

Number of Discs (physical): 3
Total Cylinders: 14,384

Number of Heads (physical): 6
Bytes Per Sector: 512

Internal Transfer Rate (min-max): 284 Mbits/sec - 424 Mbits/sec
Formatted Int Transfer Rate (min-max) 26.6 MBytes/sec - 40.5 MBytes/sec
External (I/O) Transfer Rate (max): 200 MBytes/sec
Avg Formatted Transfer Rate: 35.5 MBytes/sec

Average Seek Time, Read-Write: 5.2-6msec typical
Track-to-Track Seek, Read-Write: 0.6-0.8msec typical
Average Latency: 2.99 msec

Typical Current (12VDC +/- 5%): 0.5 amps
Typical Current (5VDC +/- 5%): 0.8 amps
Idle Power (typ): 9.5 watts



Other disks



| Characteristics | Seagate ST33000655SS | Seagate ST31000340NS | Seagate ST973451SS | Seagate ST9160821AS |
|--|--------------------------------|--------------------------------|--------------------------------|--------------------------------|
| Disk diameter (inches) | 3.50 | 3.50 | 2.50 | 2.50 |
| Formatted data capacity (GB) | 147 | 1000 | 73 | 160 |
| Number of disk surfaces (heads) | 2 | 4 | 2 | 2 |
| Rotation speed (RPM) | 15,000 | 7200 | 15,000 | 5400 |
| Internal disk cache size (MB) | 16 | 32 | 16 | 8 |
| External interface, bandwidth (MB/sec) | SAS, 375 | SATA, 375 | SAS, 375 | SATA, 150 |
| Sustained transfer rate (MB/sec) | 73-125 | 105 | 79-112 | 44 |
| Minimum seek (read/write) (ms) | 0.2/0.4 | 0.8/1.0 | 0.2/0.4 | 1.5/2.0 |
| Average seek read/write (ms) | 3.5/4.0 | 8.5/9.5 | 2.9/3.3 | 12.5/13.0 |
| Mean time to failure (MTTF) (hours) | 1,400,000 @ 25°C | 1,200,000 @ 25°C | 1,600,000 @ 25°C | — |
| Annual failure rate (AFR) (percent) | 0.62% | 0.73% | 0.55% | — |
| Contact start-stop cycles | — | 50,000 | — | >600,000 |
| Warranty (years) | 5 | 5 | 5 | 5 |
| Nonrecoverable read errors per bits read | <1 sector per 10 ¹⁶ | <1 sector per 10 ¹⁵ | <1 sector per 10 ¹⁶ | <1 sector per 10 ¹⁴ |
| Temperature, shock (operating) | 5°-55°C, 60 G | 5°-55°C, 63 G | 5°-55°C, 60 G | 0°-60°C, 350 G |
| Size: dimensions (in.), weight (pounds) | 1.0" x 4.0" x 5.8", 1.5 lbs | 1.0" x 4.0" x 5.8", 1.4 lbs | 0.6" x 2.8" x 3.9", 0.5 lbs | 0.4" x 2.8" x 3.9", 0.2 lbs |
| Power: operating/idle/standby (watts) | 15/11/— | 11/8/1 | 8/5.8/— | 1.9/0.6/0.2 |
| GB/cu. in., GB/watt | 6 GB/cu.in., 10 GB/W | 43 GB/cu.in., 91 GB/W | 11 GB/cu.in., 9 GB/W | 37 GB/cu.in., 84 GB/W |
| Price in 2008, \$/GB | ~ \$250, ~ \$1.70/GB | ~ \$275, ~ \$0.30/GB | ~ \$350, ~ \$5.00/GB | ~ \$100, ~ \$0.60/GB |

A.A. 2018-2019

47/53

<http://borghese.di.unim i.it>



Hard disk magnetico - prestazioni



- Tempo medio di seek: da 8 a 20 ms (può diminuire di più del 75% se si usano delle ottimizzazioni).
- Tempo medio di rotazione: da 2.8 ms a 5.6 ms.
- Tempo medio di trasferimento: 2/15 MB per secondo e oltre con cache.
- Tempo di controllo (utilizzato dalla logica del controller).

Qual è il tempo di lettura/scrittura di un settore di 512byte in un disco che ha velocità di rotazione di 7,200 rpm? Il tempo medio di seek è di 12ms, la velocità di trasferimento di 10Mbyte/s ed il tempo aggiuntivo richiesto dal controller è di 2ms.

$$12ms + (1/120 / 2) * 1000 ms + 0,5kbyte / 10Mbyte/s + 2ms = 12 + 4,2 + 0,05 + 2 = 18,25ms$$

Per un tempo di seek medio pari al 25% del tempo nominale ($t_{seek} = 3ms$)
 $3ms + 4,2ms + 0,5kbyte / 10Mbyte/s + 2ms = 9,25ms$

A.A. 2018-2019

48/53

<http://borghese.di.unim i.it>



RAID



RAID è un acronimo che sta per Redundant Array of Independent Disks (originariamente, 1988, stava per Redundant Array of Inexpensive Disks).

Ha queste caratteristiche:

- 1) RAID è un insieme, array, di dischi fisici visto dal sistema operativo come un drive logico singolo.
- 2) I dati vengono distribuiti attraverso i dispositivi fisici dell'array di dischi.
- 3) La capacità ridondante dei dischi viene utilizzata per memorizzare l'informazione di parità, che garantisce di potere recuperare i dati in casi di guasto (i guasti risultano più frequenti per la maggiore complessità dell'HW).



CD-ROM / DVD



- I CD-ROM sono basati sulla tecnologia laser per la memorizzazione delle informazioni. Vennero lanciati sul mercato nel 1982 da Philips e Sony per la registrazione di suoni.
- Memorizzano l'informazione codificata con incisioni di forme caratteristiche sulla superficie del disco.
- Un raggio laser colpisce la superficie del disco e viene da questa riflesso in modo diverso a seconda della forma della superficie colpita (superficie piatta, o rilievo rugoso che provoca scattering).
- Su CD-ROM è possibile immagazzinare informazione con una densità maggiore rispetto ai dischi magnetici.
- Un CD-ROM può memorizzare più di 650 MB di dati.

- Un DVD (Digital Video Disk) arriva a memorizzare 15.90 Gbyte (DVD-18). Esistono diversi dialetti e diversi formati HW: DVD-R, DVD+R.

- Un disco Blu-Ray (laser blu) arriva a 200 Gbyte ed è il dispositivo oggi più utilizzato.



Dischi a stato solido (Flash memory)



- EEPROM (memoria cancellabile elettronicamente: tempi di lettura e scrittura molto diversi)
- Used in portable devices and in hybrid systems

| Characteristics | Kingston SecureDigital (SD) SD4/8 GB | Transend Type I CompactFlash TS16GCF133 | RIDATA Solid State Disk 2.5 Inch SATA |
|--|--------------------------------------|---|---------------------------------------|
| Formatted data capacity (GB) | 8 | 16 | 32 |
| Bytes per sector | 512 | 512 | 512 |
| Data transfer rate (read/write MB/sec) | 4 | 20/18 | 68/50 |
| Power operating/standby (W) | 0.66/0.15 | 0.66/0.15 | 2.1/— |
| Size: height × width × depth (inches) | 0.94 × 1.26 × 0.08 | 1.43 × 1.68 × 0.13 | 0.35 × 2.75 × 4.00 |
| Weight in grams (454 grams/pound) | 2.5 | 11.4 | 52 |
| Mean time between failures (hours) | > 1,000,000 | > 1,000,000 | > 4,000,000 |
| GB/cu. in., GB/watt | 84 GB/cu.in., 12 GB/W | 51 GB/cu.in., 24 GB/W | 8 GB/cu.in., 16 GB/W |
| Best price (2008) | ~ \$30 | ~ \$70 | ~ \$300 |

A.A. 2018-2019

51/53

<http://borgese.di.unimi.it/>



Vantaggi e svantaggi



Pro

Latenza minore di 100-1000 volte dei dischi
 Consuma meno ed è più resistente agli urti.

Contra

Wear out (usura) → leveling (uniformità delle scritture)

Flash introdotte come memorie di boot per rendere il boot più veloce e nei dispositivi mobili quali MP3, telefonini...

| Characteristics | NOR Flash Memory | NAND Flash Memory |
|---------------------------------|------------------|---------------------|
| Typical use | BIOS memory | USB key |
| Minimum access size (bytes) | 512 bytes | 2048 bytes |
| Read time (microseconds) | 0.08 | 25 |
| Write time (microseconds) | 10.00 | 1500 to erase + 250 |
| Read bandwidth (MBytes/second) | 10 | 40 |
| Write bandwidth (MBytes/second) | 0.4 | 8 |
| Wearout (writes per cell) | 100,000 | 10,000 to 100,000 |
| Best price/GB (2008) | \$65 | \$4 |

Si va verso 1,000,000

A.A. 2018-2019



Sommario



I collegamenti tra CPU e gli altri componenti

Trasferimento tra I/O e CPU