



Le interconnessioni tra i dispositivi

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
borgese@dsi.unimi.it

Università degli Studi di Milano

Riferimento Patterson: 6.1-6.3-6.5

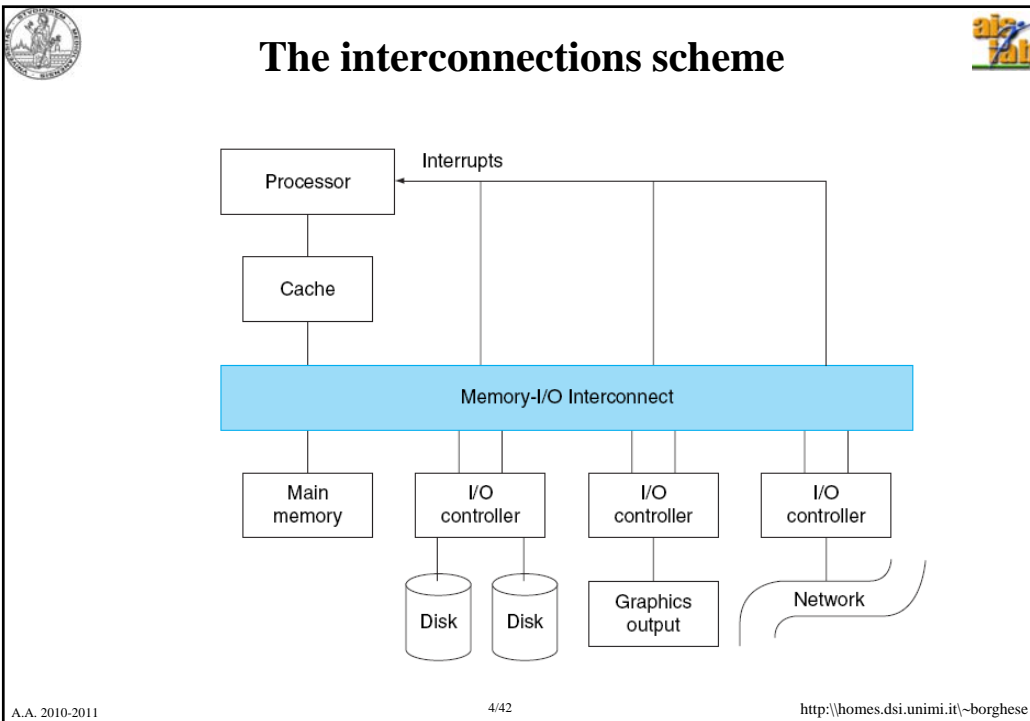
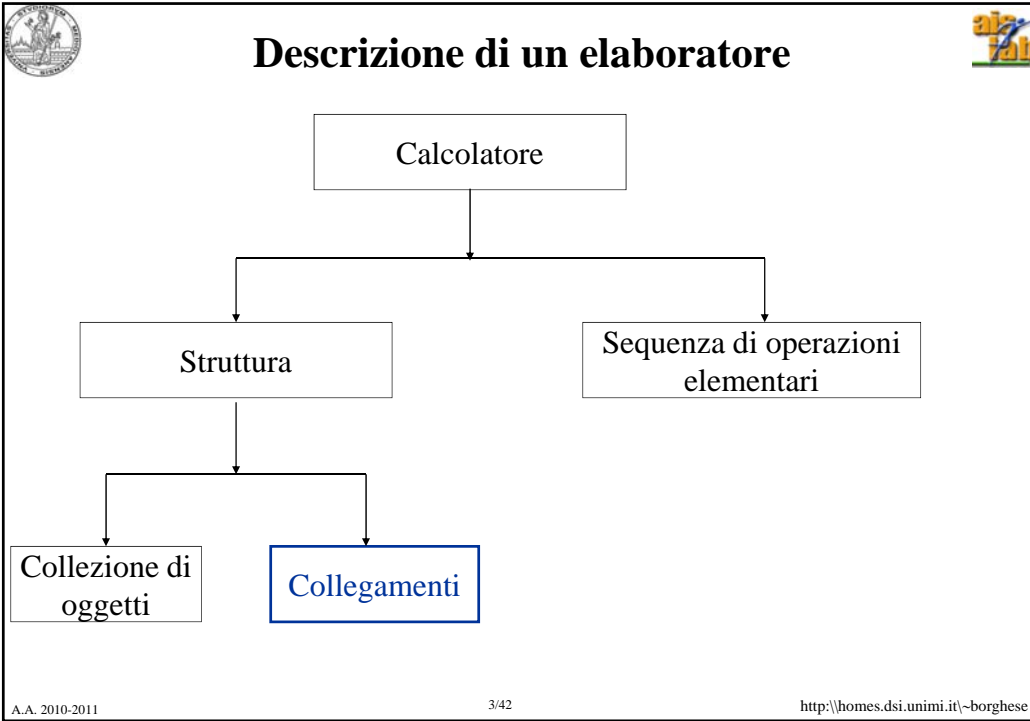


Sommario

Le interconnessioni

La gestione delle interconnessioni

La gestione dell'I/O





I/O



Dispositivi eterogenei per:

velocità di trasferimento.

latenze.

sincronismi.

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



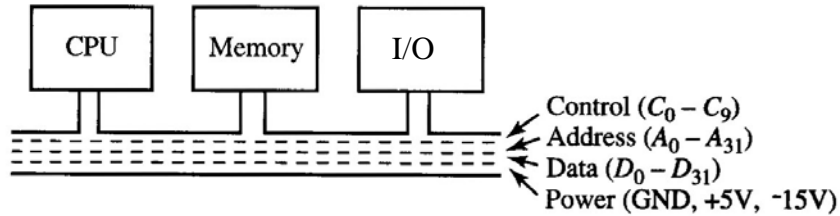
Dispositivi di I/O - classificazione



| 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 |



Il bus (connessione a cammino comune)



Pathway che connette tutti i dispositivi in modo bidirezionale (a partire dal PDP-8, omnibus, 1965).

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.

A.A. 2010-2011

7/42

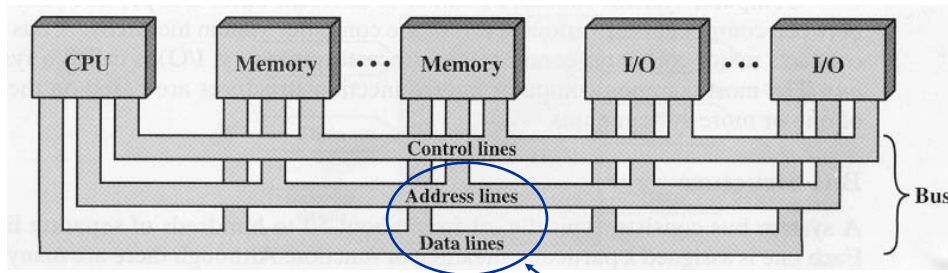
<http://homes.dsi.unimi.it/~borghese>



La struttura del bus



3 gruppi funzionali: dati, indirizzi e segnali di controllo.



Data lines: **ampiezza del bus** = ampiezza della parola dati.

Address lines: **capacità di indirizzamento** (memoria principale + I/O).

Control lines: **comandi e stato dei dispositivi**.

Problema: quale dispositivo può trasferire sul bus in caso di richieste multiple? Chi decide?

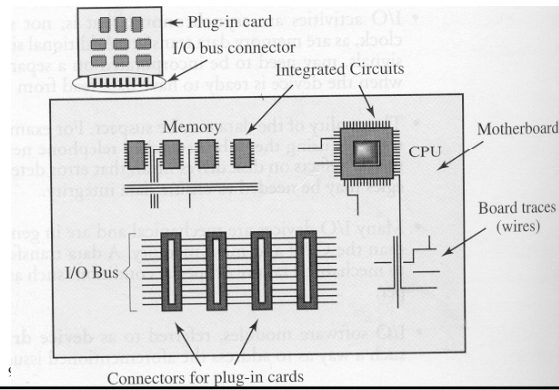
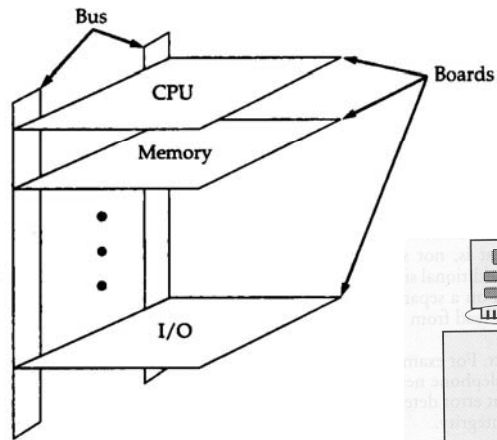
A.A. 2010-2011

8/42

<http://homes.dsi.unimi.it/~borghese>



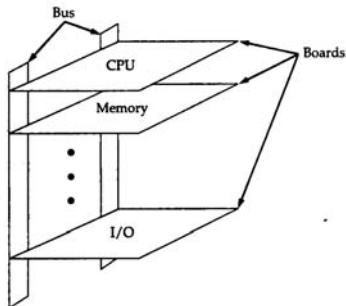
Struttura fisica del bus



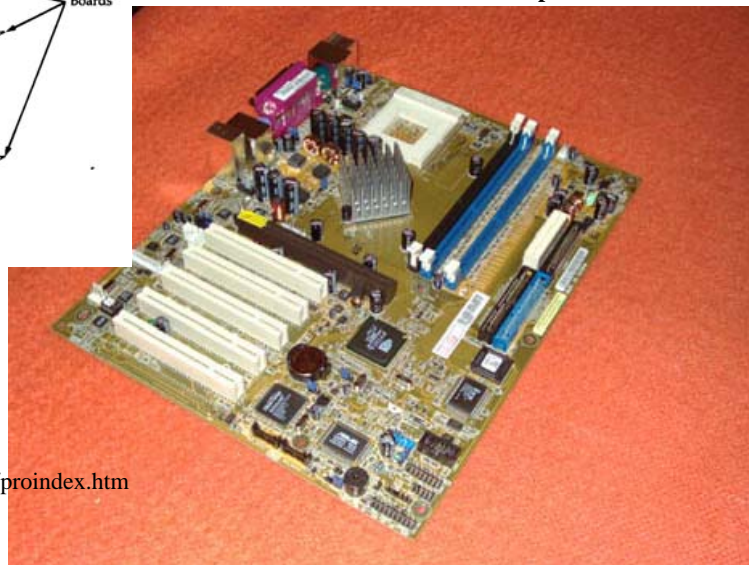
A.A. 2010-2011



Esempio di mother board



Asus A7N8X Deluxe Specifications

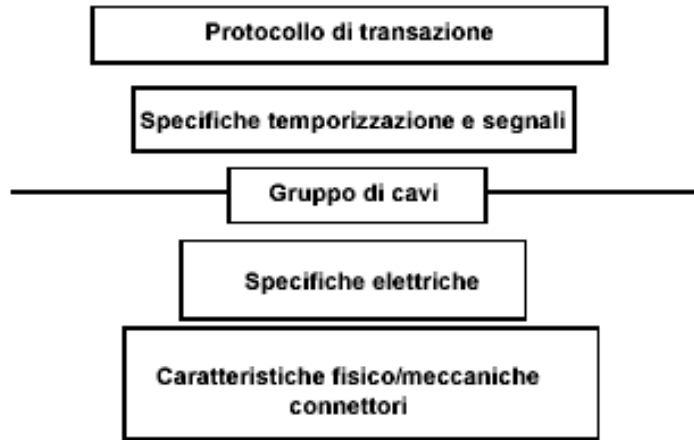


<http://www.asus.it/products/proindex.htm>

A.A. 2010-2011



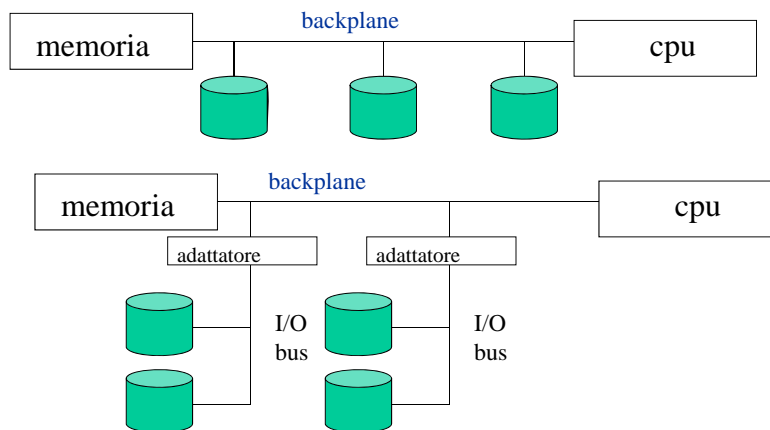
Livelli di definizione



Transazione su bus: sequenza di operazioni che partono da una richiesta, e si concludono con il trasferimento di dati.

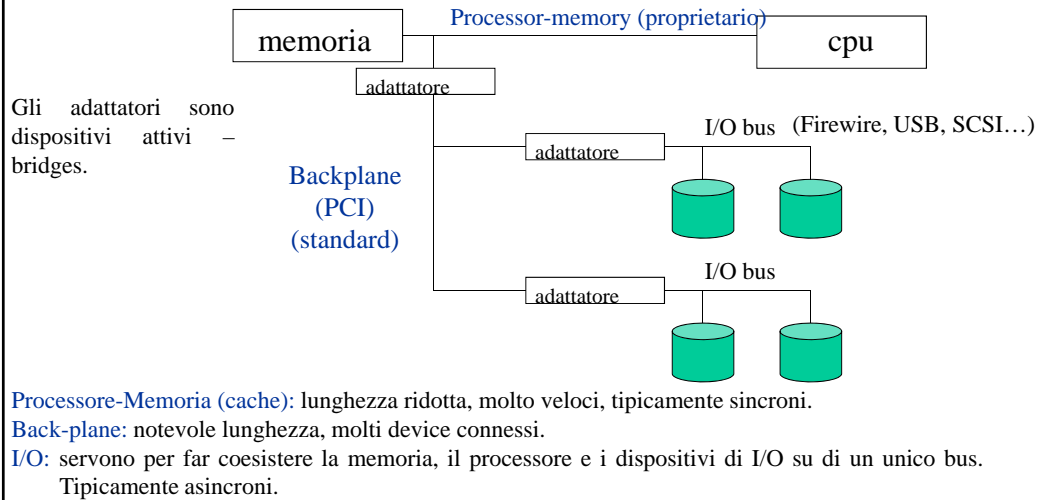


Tipologie di bus – vecchio stile





La gerarchia dei bus



La transazione su bus

Invio dell'indirizzo
Lettura / scrittura nel dispositivo di I/O

Esistono due schemi principali di comunicazione su di un bus (di operare una transazione):

- Sincrono
- Asincrono



I bus



| Caratteristica | PCI | SCSI |
|---|------------------------------------|--------------------------------|
| Tipo di bus | Back-plane | I/O |
| Ampiezza di base del bus (numero di segnali per i dati) | 32-64 | 8-32 |
| Numero di dispositivi master | molti | molti |
| Temporizzazione | Sincrono 33-66Mhz | Asincrono o sincrono (5-20Mhz) |
| Ampiezza di banda di picco teorica | 133-512MB/s (PCI64) | 5-40MB/s |
| Ampiezza di banda stimata raggiungibile per bus di base | 80MB/s | 2,5-40MB/s |
| Massimo numero di dispositivi | 1024 (32 dispositivi per segmento) | 7-31 |
| Massima lunghezza del bus | 0,5 metri | 25 metri |
| Nome dello standard | PCI | ANSI X3.131 |

A.A. 2010-2011

15/42

<http://homes.dsi.unimi.it/~borghese>



Caratteristiche di bus asincroni



| Caratteristiche | Firewire (1394) | USB 2.0 |
|-------------------------------|---|---|
| Tipo di BUS | I/O | I/O |
| Ampiezza bus (numero segnali) | 4 | 2 |
| Clock | Asincrono / Sincrono | Asincrono |
| Picco di velocità | 50Mbyte/s (Firewire 400) 100Mbyte/s (Firewire 800) | 0.2Mbyte/s (low speed) 1.5Mbyte/s (full speed) 60Mbyte/s (high speed) |
| Numero massimo di device | 63 | 127 |
| Nome standard | IEEE 1394a, 1394b | USE Implementors Forum |
| Lunghezza massima | 4.5m | 5m |

Bus seriali ad alta velocità, punto a punto, con commutazione.

A.A. 2010-2011

16/42

<http://homes.dsi.unimi.it/~borghese>



Il PCI Express



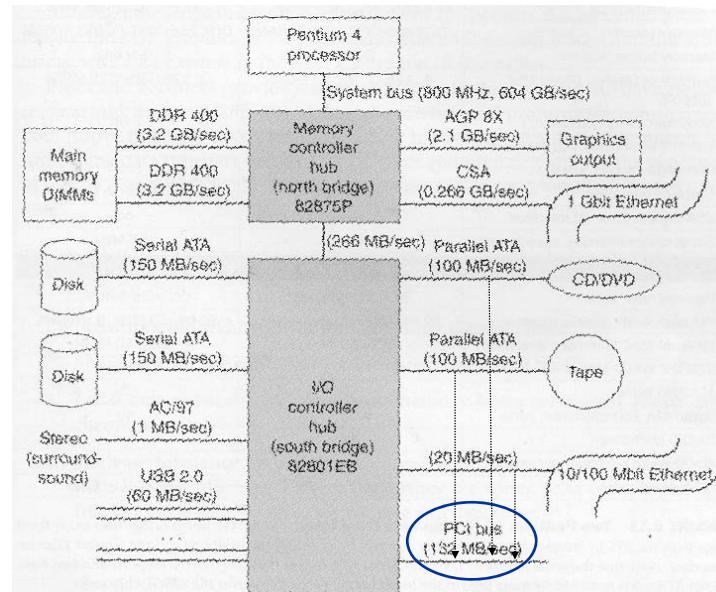
| Caratteristica | PCI | PCI Express |
|---|---|------------------------------|
| Tipo di bus | Back-plane | Versatile (back-plane, I/O) |
| Ampiezza di base del bus (numero di segnali per i dati) | 32-64 | 4 + 4 (In/Out) |
| Numero di dispositivi master | molti | 1 |
| Temporizzazione | Sincrono 33-66Mhz | 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 | 300MB/s |
| 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 |

A.A.

prghese

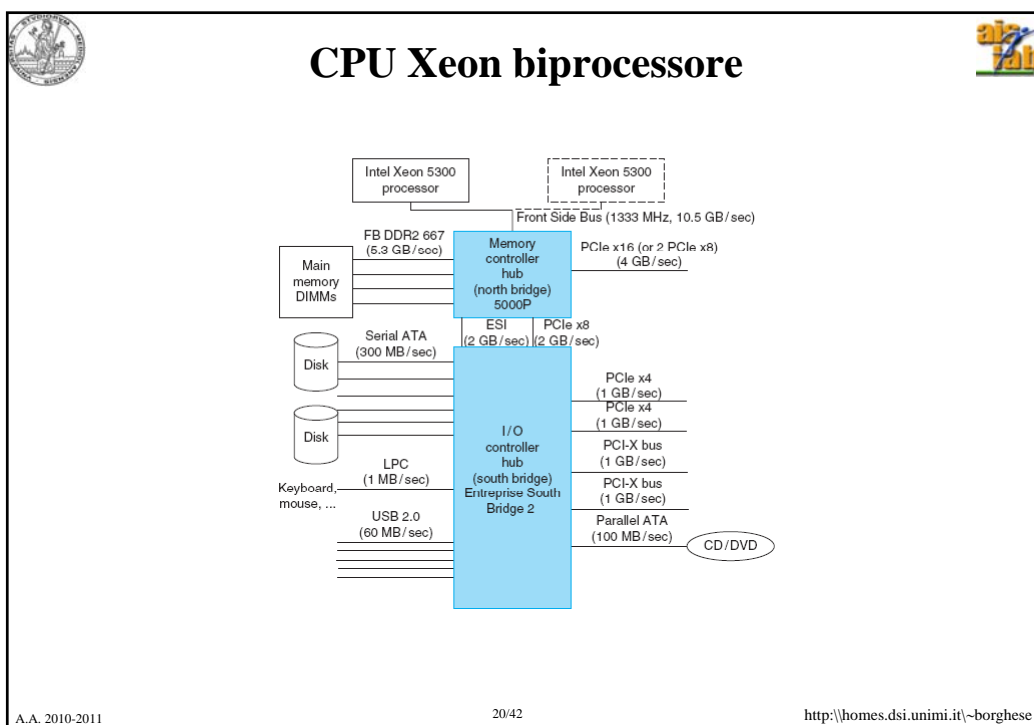
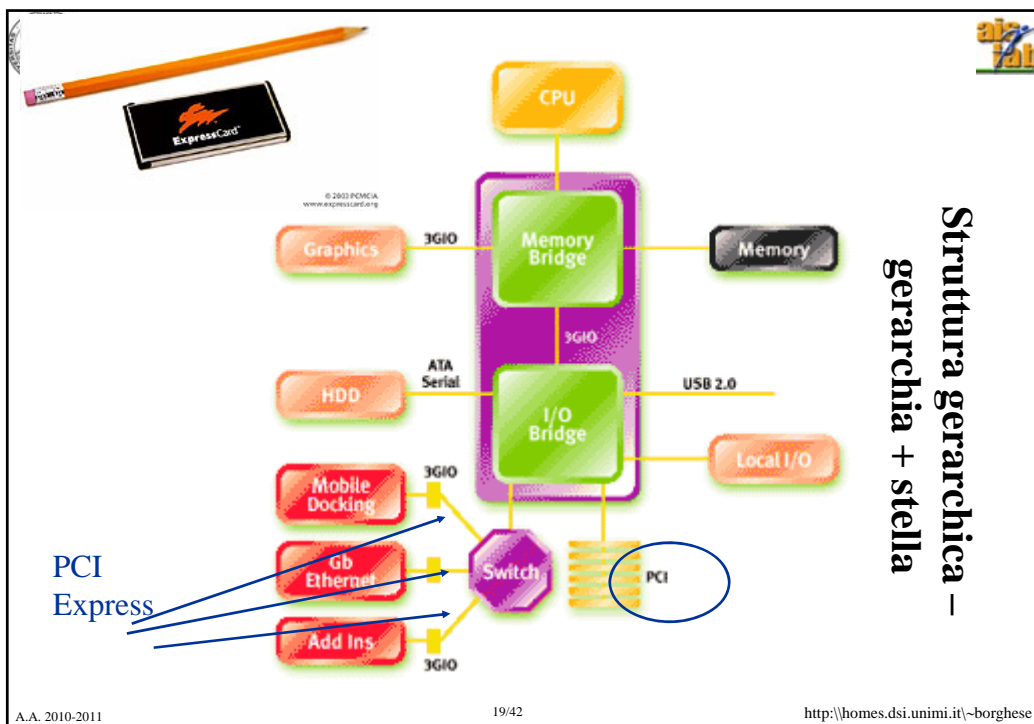




Pentium IV



A.A. 2010-2011

lsi.unimi.it/~borghese





| | 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
b
r
i
d
g
e

A.A. 2010-2011 21/42 http://homes.dsi.unimi.it/~borghese

Sommarrio

I bus

La gestione dei bus

La gestione dell'I/O

A.A. 2010-2011 22/42 http://homes.dsi.unimi.it/~borghese

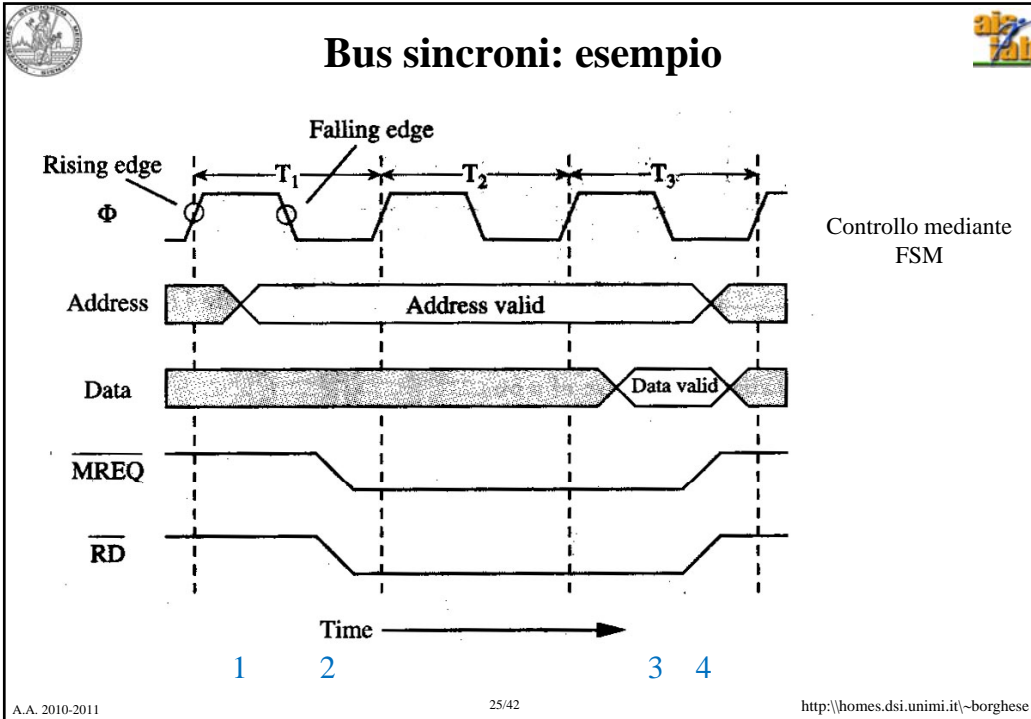


Bus sincroni



- The le linee di controllo è presente la linea che porta il segnale di clock (**bus clock**). Esiste un protocollo di comunicazione scandito dai cicli di clock, in generale diverso (ma sincronizzato) da quello della CPU.
- Questo tipo di protocollo permette di ottenere bus molto veloci.
- *Svantaggi:*
 - ◆ Ogni device deve essere sincronizzato.
 - ◆ Lunghezza limitata (per evitare che i ritardi nei fronti dovuti alla propagazione producano disallineamenti, clock skew).
 - ◆ Tutti i dispositivi devono potere lavorare alla frequenza imposta dal bus clock.
- I bus processor-memory sono spesso sincroni in quanto:
 - ◆ hanno dimensioni ridotte.
 - ◆ hanno pochi elementi connessi.

Ciclo di bus (*bus cycle*): numero di cicli per effettuare una transazione: tipicamente da 2 a 5 cicli di bus clock.



- ### Bus asincroni
- Un bus asincrono **non** è dotato di clock.
 - La comunicazione tra due parti avviene mediante un protocollo di **handshaking**.
(!MSYN) -> Job -> (!SSYN) -> (MSYN) -> (SSYN)
 - I bus asincroni possono avere lunghezza elevata per connettere molti dispositivi.
 - Sono efficienti quando i tempi di esecuzione delle varie periferiche variano molto tra loro.
 - Spesso i bus di I/O sono asincroni.
- A.A. 2010-2011 26/42 http://homes.dsi.unimi.it/~borghese



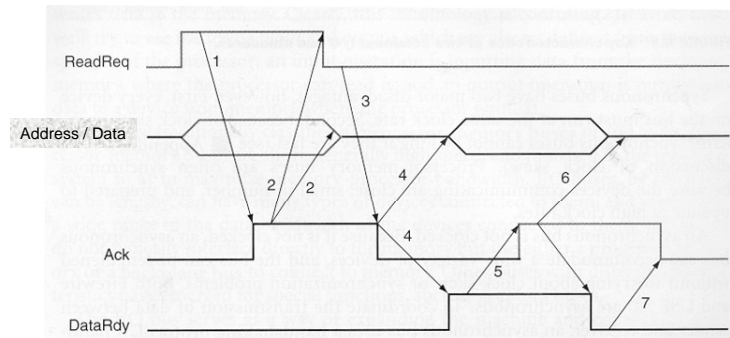
Esempio di handshaking



ReadReq: segnale di controllo di MemoryRead (MREQ+RD). In corrispondenza di questo segnale l'**indirizzo** della parola di memoria viene inviato sul bus dati.

Data Rdy: viene utilizzato per indicare che la parola è pronta sulle linee di dato. Questo segnale viene inviato dalla memoria quando il dato è disponibile in uscita dalla memoria.

Ack: viene utilizzato come risposta ai 2 segnali precedenti. Feed-back (letto indirizzo, letto / scritto il dato).



I segnali di ReadReq e DataReady rimangono alti fino a quando il ricevente (la memoria) non ha visto il comando e letto / scritto i dati corrispondenti.

A.A. 2010-2011

27/42

<http://homes.dsi.unimi.it/~borghese>



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.

A.A. 2010-2011

28/42

<http://homes.dsi.unimi.it/~borghese>



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*)

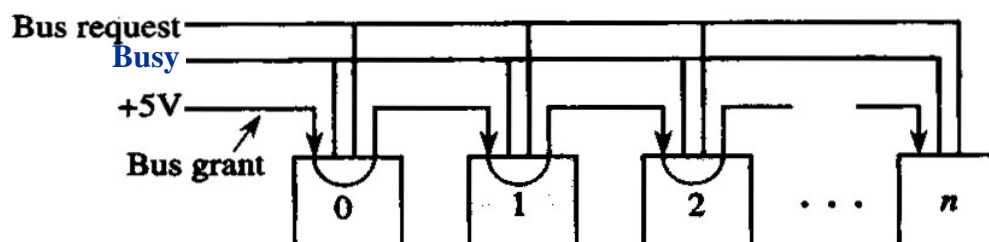
Problema è assicurare una *fairness*.

Compromesso tra *fairness* e priorità.

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



Arbitraggio distribuito in Daisy Chain



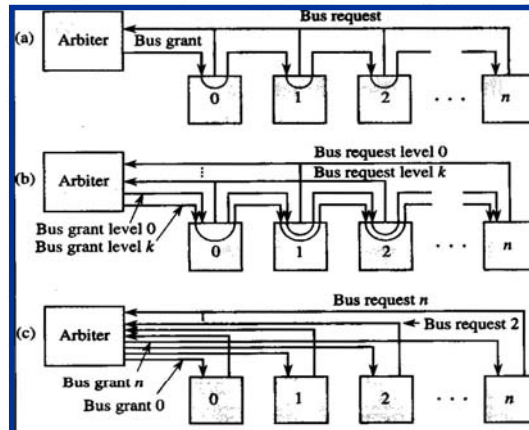
Viene introdotto il segnale di Busy.



Schemi di arbitraggio centralizzati



Arbitraggio
In Daisy Chain



Arbitraggio
Centralizzato
Con priorità

Arbitraggio
Centralizzato
Parallelo

- 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) Deasserire la richiesta del bus.

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



Sommario



I bus

La gestione dei bus

La gestione dell'I/O



I/O



Dispositivi eterogenei per:

velocità di trasferimento.

latenze.

sincronismi.

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

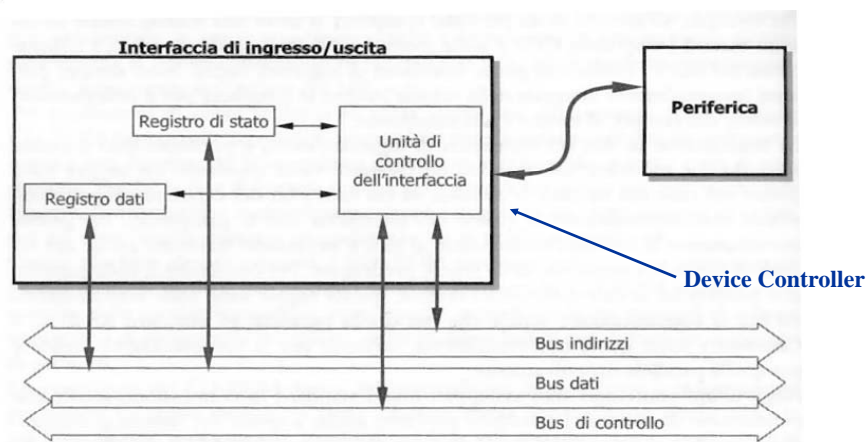


Bus & buffer

- 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



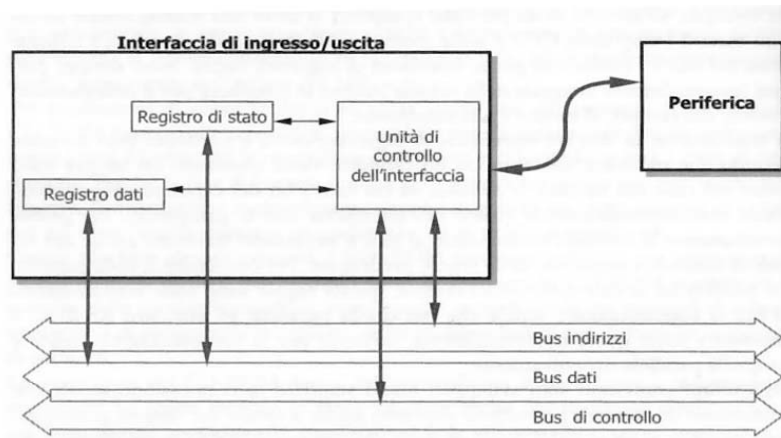
Registri:

- Dati
- Stato: situazione della periferica (idle, busy, down....) e comando in esecuzione.

Il driver agisce inviando alla periferica comandi opportuni e dati.



Modalità trasferimento dati



Parallela (centronics). 1 byte alla volta.

Parallela (bus PCI, bus processore-memoria).

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

Nuovi standard sono USB, Firewire (IEEE 1394) e Bluetooth / 802.11 / 802.15 (wireless).



Funzionamento di un driver

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.



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.
- 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.



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
        andi $t1, $t1, 0x1    # if $t1 == 1 esci
        beq $t1, $zero, ciclo

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

        li $v0, 10          # exit
        syscall
```



Sommario



Il bus ed il protocollo di trasferimento

Tipologie di bus

La gestione dell'I/O