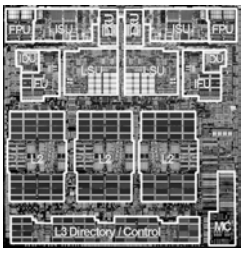
Università degli Studi
di Cassino



Corso di Calcolatori Elettronici II

Il Sistema di bus

Anno Accademico 2004/2005

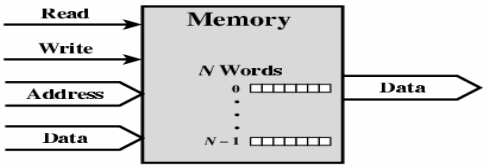
Francesco Tortorella

Connettiamo tutto !

- Sistema di elaborazione:
  struttura formata da unità diverse (CPU, moduli di memoria, moduli di I/O) collegate e comunicanti
- Ogni unità presenta tipi di connessioni diverse
- Esaminiamo
  - Un modulo di memoria
  - Un modulo di Input/Output
  - La CPU

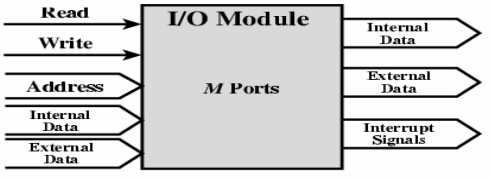
Connessione del modulo di memoria

- Riceve ed invia dati
- Riceve indirizzi di registri
- Riceve segnali di controllo
  - Read
  - Write
  - Tempificazione



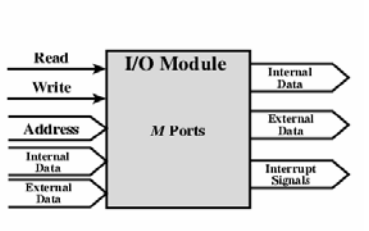
Connessione del modulo di I/O (1)

- Simile al modulo di memoria
- Output
  - Riceve dati dal sistema
  - Invia dati alla periferica
- Input
  - Riceve dati dalla periferica
  - Invia dati al sistema



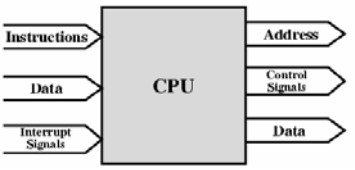
Connessione del modulo di I/O (2)

- Riceve segnali di controllo dal sistema
- Invia segnali di controllo alla periferica
- Riceve indirizzi dal sistema (p.es. per identificare la periferica)
- Invia al sistema segnali di richiesta di interruzione



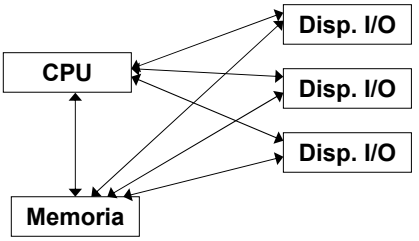
Connessione della CPU

- **Legge** istruzioni e dati
- Scrive dati
- Invia segnali di controllo alle altre unità
- Riceve (e gestisce) richieste di interruzione



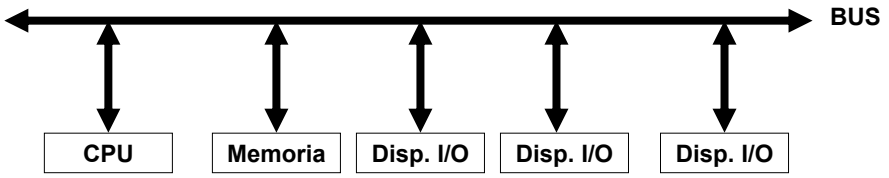
Come si realizza la connessione ?

- Una prima soluzione potrebbe essere quella di collegare i vari componenti con connessioni punto-punto, ma ciò comporta:
  - complessità dell'hardware
  - costo realizzativo
  - struttura estremamente rigida



Una soluzione efficiente: il bus

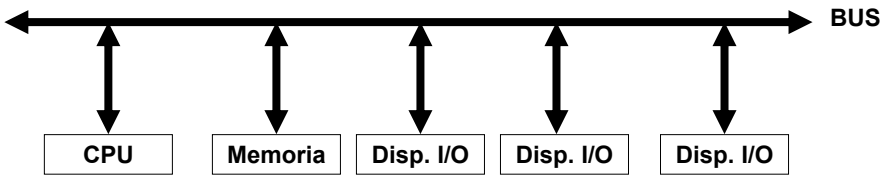
Alternativa: utilizzare un canale di comunicazione condiviso dai vari componenti



Vantaggi:

- **Basso costo:**
  lo stesso canale di comunicazione è utilizzato in modi diversi per diverse esigenze
- **Versatilità:**
  facile aggiungere o rimuovere dispositivi e periferiche
- **Semplicità di progetto:**
  è possibile operare una decomposizione sul progetto dell'architettura

Svantaggi

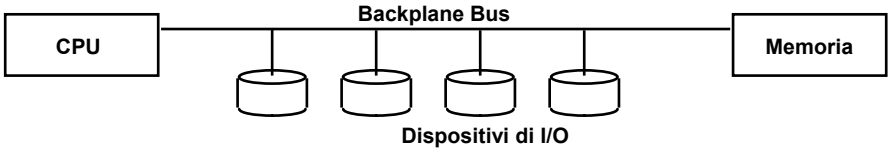


- **velocità limitata:**
  limita il throughput dell'I/O e della memoria
- **La velocità sul bus è limitata da fattori fisici**:
  - la lunghezza del bus
  - il numero di dispositivi presenti
  - la necessità di supportare un insieme di dispositivi con latenze e velocità di trasferimento che variano notevolmente

Tipi di bus

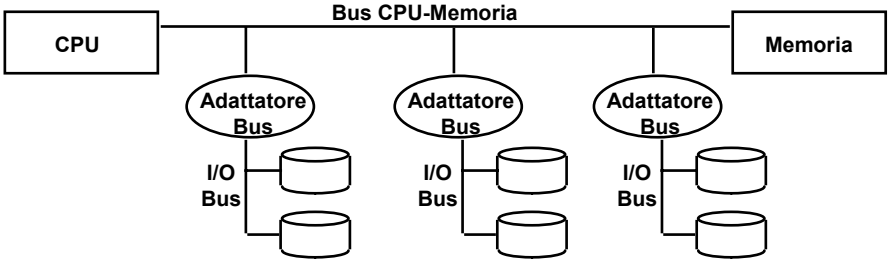
- **Bus CPU-Memoria** (specifici)
  - corti ed ad alta velocità
  - progettati solo per il memory system (massimizzano la banda passante tra memoria e processore)
  - sono connessi direttamente al processore
  - ottimizzati per il trasferimento a blocchi della cache
- **I/O Bus** (industry standard)
  - di solito lunghi e più lenti
  - devono adattarsi a un vasto insieme di dispositivi I/O
  - si connettono al processor-memory bus o al backplane bus
- **Backplane Bus** (standard o proprietario)
  - backplane: una struttura di interconnessione nello chassis
  - coesistono processori, memorie, e dispositivi di I/O
  - vantaggio di costo: un bus per tutti i componenti

Sistema a bus singolo



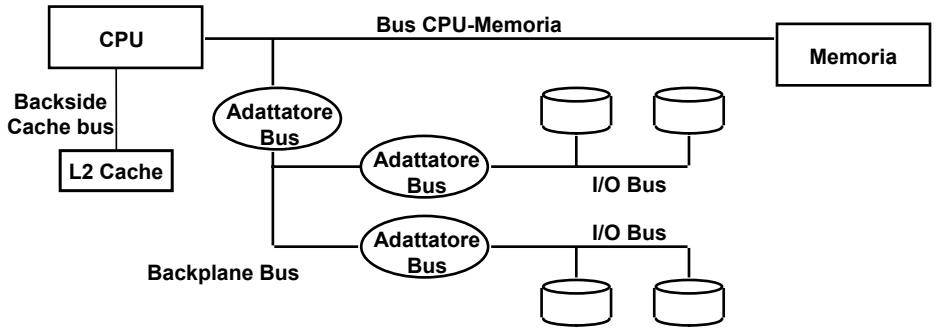
- Un solo bus (il backplane bus) è usato per la comunicazione tra processore, memoria e dispositivi di I/O
- Vantaggi: semplice ed economico
- Svantaggi: lento, il bus può diventare il principale collo di bottiglia del sistema
- Esempio: IBM PC - AT

Sistema a due bus



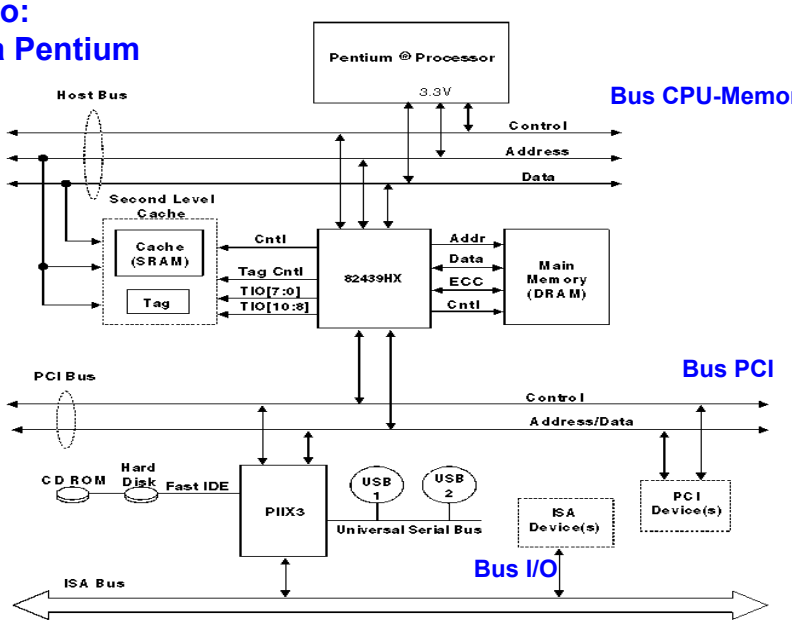
- I bus di I/O sono connessi al bus CPU-memoria tramite “adattatori” :
  - Bus CPU-memoria: usato principalmente per il traffico di memoria
  - Bus I/O: forniscono slot di espansione per i dispositivi di I/O
- Esempio: Apple Macintosh-II
  - NuBus: Processore, memoria, e pochi, ben selezionati dispositivi
  - SCSI Bus: il resto dei dispositivi di I/O

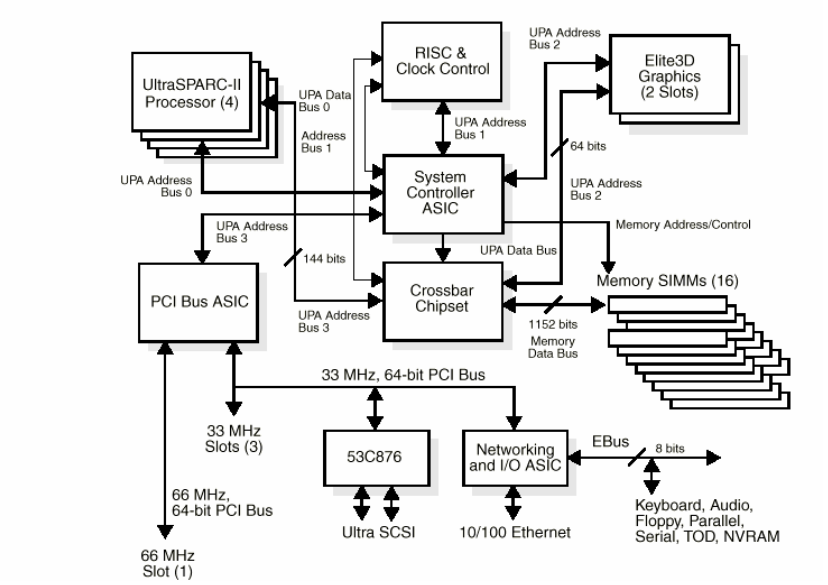
Sistema a tre bus



- Un backplane bus è connesso al bus CPU-memoria
  - il bus CPU-memoria è usato per il traffico di memoria
  - i bus di I/O sono connessi al backplane bus
- Vantaggio: il bus del processore è molto più scarico (c'è un'unica connessione)

Esempio: sistema Pentium



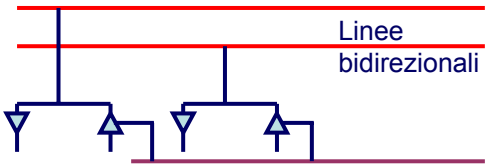


Organizzazione di un bus

- Linee di Controllo:
  - trasportano segnali di richiesta e di acknowledgment
  - indicano che tipo di transazione si sta svolgendo
- Linee Indirizzi:
  - trasportano gli indirizzi dei dati da trasferire
- Linee Dati:
  - trasportano i dati da trasferire (anche comandi complessi per alcune periferiche)

Collegamento al bus

- Il collegamento delle periferiche al bus avviene di solito tramite un dispositivo intermedio (*driver* del bus) che:
  - fornisce potenza sufficiente per pilotare le linee del bus
  - isola l'uscita del dispositivo quando questa è inattiva
- Sugli ingressi dei dispositivi si pone un *latch*, che realizza il buffering dei dati.
- Il collegamento a linee bidirezionali del bus avviene tramite un *transceiver*



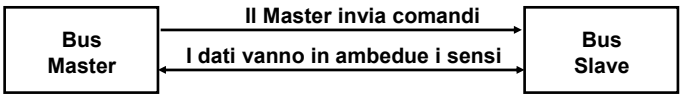
Organizzazione della comunicazione sul bus

La comunicazione sul bus non è paritetica.

Bus Master: ha la capacità di controllare il bus, inizia la transazione

Bus Slave: modulo attivato in seguito alla transazione

Un dispositivo sul bus può fare da solo master (CPU), da solo slave (memoria) o da master/slave (DMA controller)



La comunicazione tra master e slave avviene in base ad una sequenza precisa di azioni realizzate dai partner (transazione).

Il **Protocollo di Comunicazione del Bus** precisa la sequenza di eventi e le specifiche di tempificazione con cui avviene la transazione.

Transazione di bus

- Una transazione di bus include due parti:
  - invio del comando (e dell'indirizzo) – request
  - trasferimento dei dati – action
- Il master inizia la transazione:
  - inviando il comando (e l'indirizzo)
- Lo slave (identificato dall'indirizzo) risponde:
  - inviando i dati al master se richiesti
  - ricevendo i dati dal Master se richiesto
- L'operazione può essere realizzata con due diversi tipi di tempificazione:
  - tempificazione sincrona
  - tempificazione asincrona

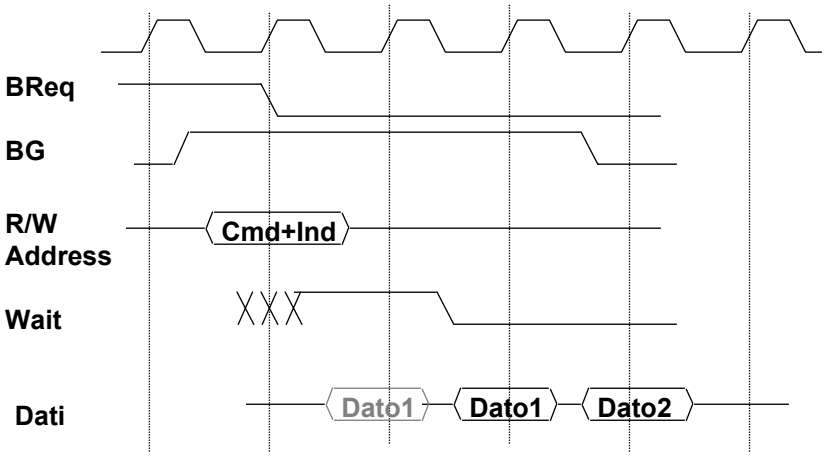
Bus sincrono

- Le operazioni sul bus sono tempificate da un segnale di sincronizzazione (clock)
- Gli istanti in cui sono validi i segnali sulle linee sono fissati dal clock (tipicamente sul fronte di salita o di discesa)
- In relazione al clock, si definiscono anche la durata dei segnali e gli intervalli temporali di separazione dei segnali (tipicamente in termini di periodi o semiperiodi del clock)
- **Vantaggi:**

  semplicità di progettazione del sistema (circuiti sincroni più facili da progettare degli asincroni)
- **Svantaggi:**

  la velocità del bus deve adeguarsi ai dispositivi più lenti

Tempificazione di un bus sincrono



Bus asincrono

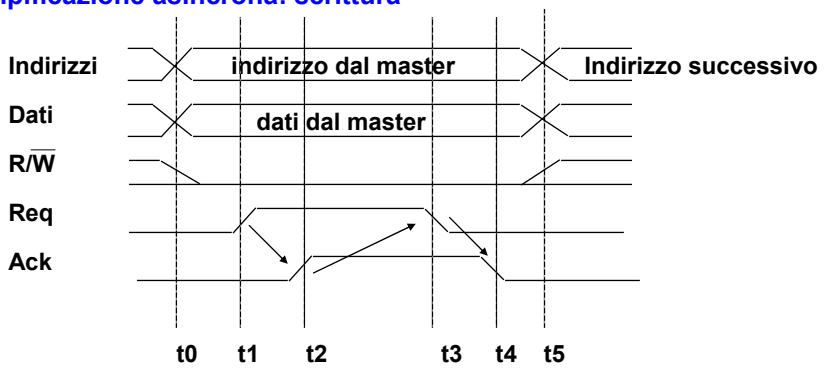
- Nei bus asincroni manca un segnale di clock principale.
- Le operazioni avvengono sulla base di un colloquio (handshake) tra master e slave basato su apposite linee di sincronizzazione (DATA READY, DATA REQUEST, ACK, ecc.)
- Il tempo impiegato dalle singole operazioni è legato esclusivamente alle velocità esibite dai dispositivi coinvolti.
- **Vantaggi:**

  possibilità di connettere periferiche con velocità diverse

- **Svantaggi:**

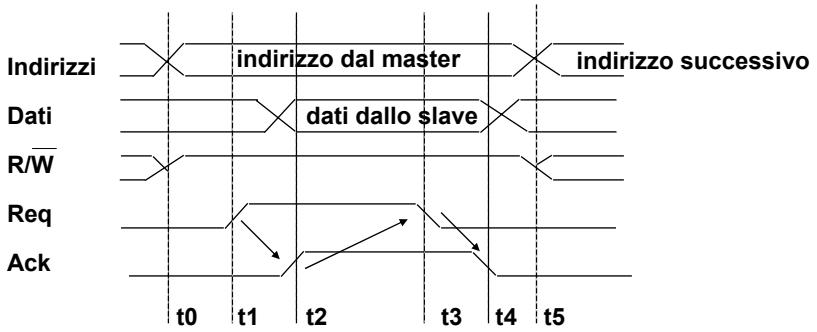
  al tempo necessario per una transazione bisogna aggiungere il tempo dell'handshake

Tempificazione asincrona: scrittura



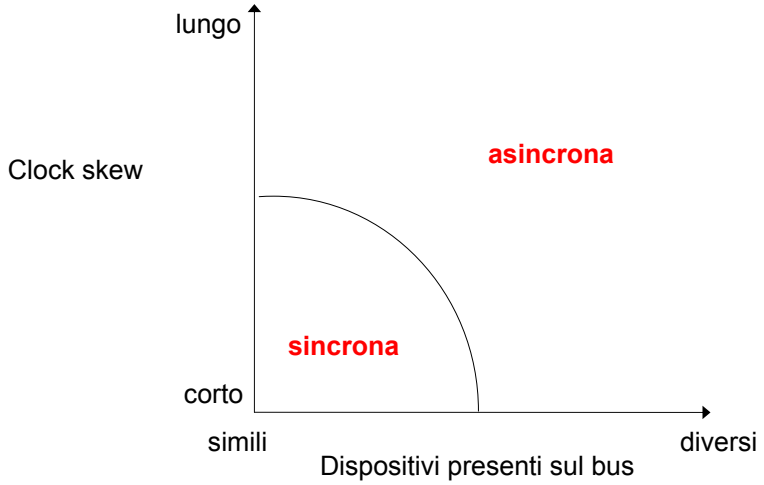
t0 : Il master predispone indirizzo, direzione del trasferimento, dati.
Attende un certo intervallo di tempo (necessario perché gli slaves leggano l'indirizzo)
t1: Il master asserisce la linea di richiesta (REQ)
t2: Lo slave asserisce ack (dato ricevuto)
t3: Il master abbassa la linea di richiesta
t4: Lo slave abbassa l'ack

Tempificazione asincrona: lettura



t0 : Il master predispone indirizzo e direzione del trasferimento
Attende un certo intervallo di tempo (necessario perché gli slaves leggano l'indirizzo)
t1: Il master asserisce la linea di richiesta (REQ)
t2: Lo slave asserisce ack (dato disponibile)
t3: Il master abbassa la linea di richiesta (dato ricevuto)
t4: Lo slave abbassa l'ack

Tempificazione sincrona o asincrona ?

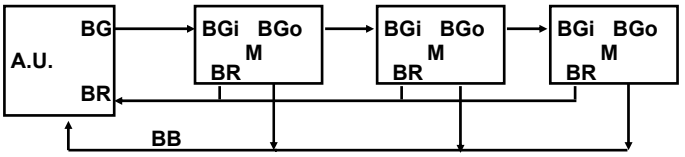


Arbitraggio del bus

- Per iniziare una qualunque transazione, il master deve ottenere il controllo del bus.
- Nel caso ci siano più master a richiedere il bus, bisogna decidere a quale di questi assegnarlo tramite un *arbitraggio del bus*, gestito da un apposito dispositivo (*Arbitration Unit* o *arbitro di bus*).
- Il dispositivo che intende controllare il bus segnala un *bus request* all'arbitro di bus; il controllo viene eventualmente concesso tramite un *bus grant* (molti bus request unico bus grant).
- A valle del bus grant, il dispositivo (master) può iniziare la transazione.
- La tecnica di arbitraggio deve bilanciare due fattori:
  - *priority*
  - *fairness*
- Tre principali tecniche di arbitraggio:
  - *daisy chain*
  - *independent requesting*
  - *polling*

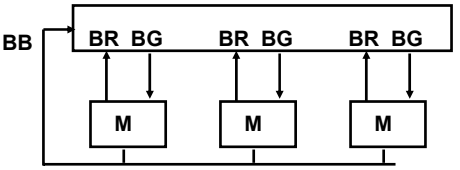
} Arbitraggio centralizzato

Daisy Chain



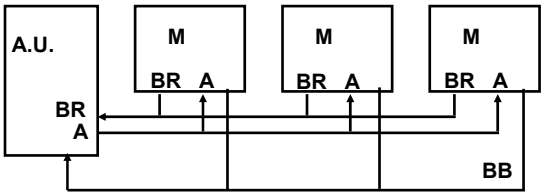
- Tutti i possibili masters sono connessi (wired OR) su un unica linea di BUS REQUEST
- In presenza di un segnale di BUS REQUEST attivo, l'Arbitration Unit controlla il BUS BUSY e, se inattivo, asserice BUS GRANT
- I master sono collegati a catena (chain) sulla linea BUS GRANT. Il segnale viene propagato lungo la catena a partire dal primo device verso l'ultimo. La priorità è automaticamente stabilita in base all'ordine dei dispositivi sulla catena
- Vantaggi: semplicità Svantaggi: rigidità

Independent Requesting



- C'è una coppia BUS REQUEST-BUS GRANT per ogni device
- L'identificazione del device è immediata e la politica di arbitraggio si realizza gestendo le linee di BUS GRANT
- Vantaggi: Politica di arbitraggio facilmente programmabile; estrema velocità di gestione; indipendenza tra i device
- Svantaggi: necessarie 2n+1 linee di controllo per gestire n device.

Polling



- La linea BUS GRANT è sostituita da N linee tramite le quali l'A.U. interroga i singoli device in sequenza.
- In presenza di una richiesta, l'A.U. verifica BUS BUSY e, se inattivo, emette sulle linee di polling a turno l'identificativo dei vari device.
- Il primo device "attivo" il cui codice appare sulle linee di polling, asserisce BUS BUSY, facendo arrestare il processo di interrogazione

Tecniche di arbitraggio distribuito

Oltre alle tecniche precedenti, che richiedono un'unità di arbitraggio centralizzata, esistono altre tecniche di arbitraggio distribuito, in cui l'assegnazione del bus non viene decisa da un arbitro esterno, ma in base ad un confronto tra i vari master che fanno richiesta del bus:

- **Self selection**
- **Collision detection**

Confronto sincrono/asincrono

- Operazione di lettura in memoria di una parola da 32 bit
  - Tempo per la lettura in memoria: 200 ns
- Bus sincrono
  - Periodo di clock: 50 ns
  - Un periodo di clock per ogni trasmissione sul bus
- Bus asincrono
  - 40 ns per ogni trasmissione di handshaking

Bus sincrono

- 50 ns per inviare indirizzo alla memoria
- 200 ns per la lettura in memoria
- 50 ns per inviare il dato
- 300 ns per l'intera transazione
- Banda del bus:
  - 32 bit / 300 ns = 13.3 Mbyte/sec

Bus asincrono

- Master prepara indirizzo e R/W
- 40 ns master asserisce REQ
- 200 ns per la lettura in memoria
- slave asserisce ACK
- 40 ns master abbassa REQ
- 40 ns slave abbassa ACK
- Banda del bus:
  - 32 bit / 320 ns = 10 Mbyte / sec

Tecniche per aumentare la banda del bus

Oltre alla scelta della tecnica di tempificazione, altri parametri influiscono sulla velocità di trasferimento del bus:

parallelismo del data bus	aumento del numero di linee
linee dati ed indirizzi separate	aumento del numero di linee
trasferimento dati a blocchi	accresciuto tempo di risposta
split transaction	tempi di transazione più lunghi

Opzioni di progettazione per il bus

Opzione	*Prestazioni elevate*	*Basso costo*
Parallelismo Bus	Linee indirizzi e linee dati separate	Linee indirizzi e linee dati multiplexate
Parallelismo Dati	ampio -> più veloce (es., 64 bits)	limitato -> meno costoso (es, 8-16 bits)
Dimensione del trasferimento	Trasferire più words ha minore overhead	Trasferire una singola word è più semplice
Bus master	Multiplo (arbitraggio necessario)	Singolo (nessun arbitraggio)
Split transaction?	Si — pacchetti Request- Reply separati forniscono maggiore banda (richiede più master)	No — una connessione continua è più economica ed ha minore latenza
Tempificazione	Sincrono	Asincrono

Esempi di bus

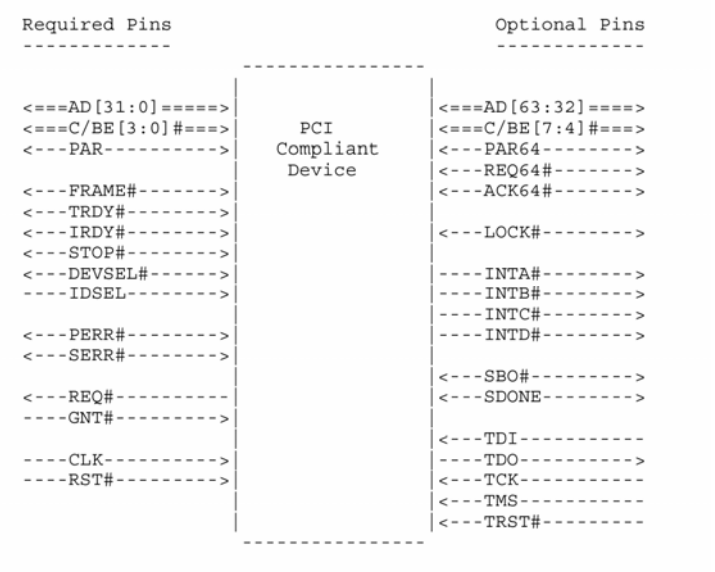
Bus	**SCSI-1**	**PCI**
Tempificazione	asincr. - sincr. (10 MHz)	sincr. (33-66 MHz)
Linee ind./dati	multiplex	multiplex
Parallelismo dati	8,32	32,64
Master	multiplo	multiplo
Arbitraggio	self-selection	ind. request.
Banda max teor.	5-40 MB/s (sincr.)	133-512 MB/s
# max dispositivi	7,31	1024 (con più segmenti 32 per segmento)

PCI Bus

- Peripheral Component Interconnection
- architettura definita da Intel e rilasciata al pubblico dominio
- 32 o 64 bit
- arbitraggio sincrono centralizzato e *nascosto*
- 49 linee obbligatorie + 51 facoltative

PCI

Descrizione dei segnali



PCI Bus Lines (obbligatorie)

- **Linee di sistema**
  - CLK attivo sul fronte di salita
  - RST# reset del device PCI
- **Linee indirizzi & dati**
  - AD[31:0] 32 linee time-mux per indirizzi/dati
  - C/BE[3:0]# Bus Control (*address phase*)/Byte Enable (*data phase*)
- **Linee per la gestione del protocollo**
  - FRAME#
  - IRDY#
  - TRDY#
  - STOP#
  - LOCK#
  - IDSEL
  - DEVSEL#

PCI Bus Lines (obbligatorie)

- **Linee per l'arbitraggio**
  linee non condivise (ogni master ha la sua coppia)
  connessione diretta con l'arbitro di bus
  - REQ#
  - GNT#
- **Linee per la segnalazione di errori**
  - PERR# Parity error
  - SERR# System error
- **Linee di interruzione**
  connesse in wired OR
  - INTA# INTB# INTC# INTD#

PCI Bus Lines (opzionali)

- **Linee per il supporto alle attività di cache**

- **Linee per l'estensione a 64 bit**
  - AD[63:32] 32 linee indirizzi/dati addizionali time multiplexed
  - C/BE[7:4] 4 linee controllo/byte enable addizionali
  - REQ64# richiesta del master di un trasferimento a 64 bit
  - ACK64# accettazione dello slave del trasferimento a 64 bit
  - PAR64 linee per il controllo di parità per la MS word

- **JTAG/Boundary Scan**
  - linee per procedure di testing

Comandi PCI

L'attività del bus si manifesta sotto forma di transazioni tra il master e lo slave. Il master determina il tipo di transazione su opportune linee (C/BE appartenenti al gruppo indirizzi/dati) durante la fase di invio dell'indirizzo.

I comandi sono:

- Avvenuta ricezione dell'interruzione
- Ciclo speciale
- Lettura da I/O
- Scrittura da I/O
- Lettura in memoria
- Lettura di linee multiple in memoria
- Scrittura in memoria
- Scrittura in memoria e validazione
- Lettura di configurazione
- Scrittura di configurazione
- Ciclo di indirizzo duale

PCI
Bus Command

Il contenuto delle linee C/BE nell'address phase definisce il tipo di transazione che si sta per eseguire sul bus

C/BE[3:0]#	Command Types
0000	Interrupt Acknowledge
0001	Special Cycle
0010	I/O Read
0011	I/O Write
0100	Reserved
0101	Reserved
0110	Memory Read
0111	Memory Write
1000	Reserved
1001	Reserved
1010	Configuration Read
1011	Configuration Write
1100	Memory Read Multiple
1101	Dual Address Cycle
1110	Memory Read Line
1111	Memory Write and Invalidate

La transazione PCI

La transazione parte con un *address phase*, segnalata dall'attivazione del segnale FRAME# ed iniziata in corrispondenza del successivo ciclo di clock con la scrittura di un indirizzo sulle linee AD[31:0].
Con il successivo ciclo di clock inizia il trasferimento, costituito da una o più *data phases,* in cui i dati sono trasferiti sempre sulle linee AD[31:0].

Nella terminologia PCI il master viene indicato con *initiator*, mentre lo slave si indica con *target.* Per precisare il tipo di transazione, il master setta opportune linee di controllo durante l'address phase.
Sia il master che lo slave possono inserire cicli di attesa tramite le linee IRDY# eTRDY#. I trasferimenti di dati validi si realizzano in corrispondenza di fronti del clock con le linee IRDY# e TRDY# entrambe asserite.

La transazione PCI

Le operazioni di I/O che accedono registri di device PCI richiedono tipicamente una sola data phase. Trasferimenti di blocchi da/verso la memoria consistono di molteplici data phases che leggono o scrivono molteplici locazioni di memoria.

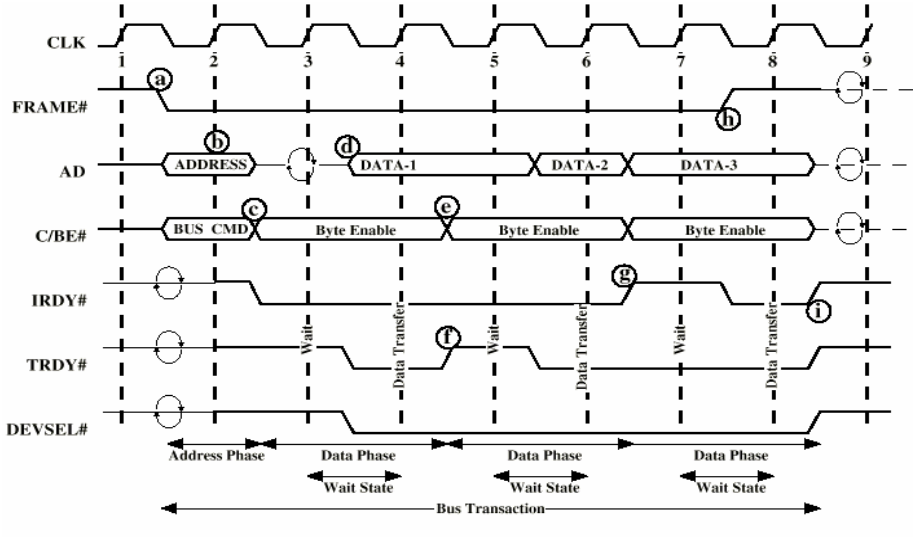
Sia il master che lo slave possono terminare una transazione in un istante qualunque. Il master completa la transazione deattivando il segnale FRAME# durante l'ultima data phase. Lo slave può terminare la transazione asserendo la linea STOP#. Quando il master riscontra un segnale di STOP# attivo, deve terminare la transazione corrente e iniziare una nuova fase di arbitraggio.

Se STOP# è attivato senza completare alcuna data phase, lo slave ha realizzato un *retry*. Se STOP# è attivato dopo una o più data phase, lo slave ha realizzato un *disconnect*.

Transazione di bus

- Acquisizione del bus da parte del master
- Fase indirizzo
  - precisa l'indirizzo dello slave
  - determina il tipo di transazione
- Una o più fasi dati
- Rilascio del bus

PCI Read Timing Diagram



PCI Bus Arbitration