

Sistemi Operativi 1

AA 2018/2019

Il Sistema di I/O

Introduzione

- Hardware di I/O
- Interfacce di I/O
- Software di I/O

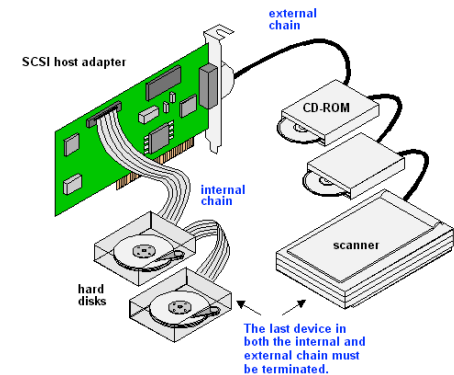
Sotto-sistema di I/O

- Insieme di metodi per controllare i dispositivi di I/O
- Obiettivo:
 - Fornire ai processi utente un'interfaccia efficiente e indipendente dai dispositivi
- Interazione tra due componenti:
 - I/O hardware
 - I/O software (S.O.)

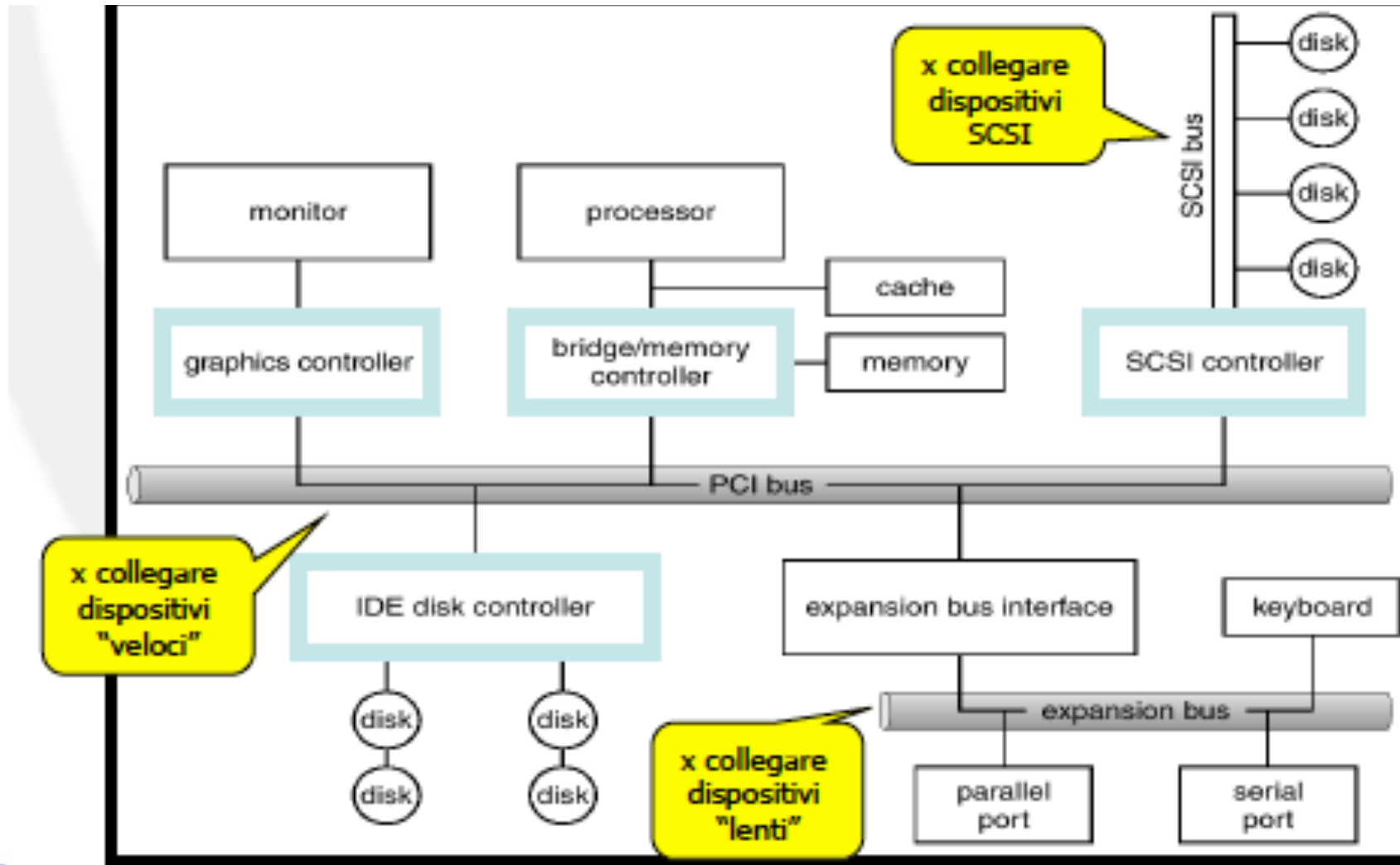
HARDWARE DI I/O

I/O Hardware

- Numerosi ed eterogenei dispositivi di I/O per:
 - Memorizzazione (dischi, nastri, ...)
 - Trasmissione (modem, schede di rete, ...)
 - interazione uomo-macchina (tastiera, monitor, ...)
- Distinzione tra:
 - Dispositivi (parte non elettronica)
 - Controllori dei dispositivi (parte elettronica)
- Concetti comuni:
 - Porta (punto di connessione)
 - Bus (daisy chain o condiviso, set di fili e relativo protocollo)
 - Controllore (agisce su porta, bus o dispositivi)



Tipica struttura di un bus per PC



Controllore dei dispositivi

- Parte elettronica di un dispositivo
 - Detto anche **device controller**
- Connesso tramite bus al resto del sistema
- Associato ad un indirizzo
- Contiene registri per comandare il dispositivo
 - Registro(i) di stato
 - per capire se il comando è stato eseguito, se c'è stato un errore, se i dati sono pronti per essere letti, ...
 - Registro di controllo
 - per inviare comandi al dispositivo
 - Buffer (uno o più) per la “conversione” dei dati



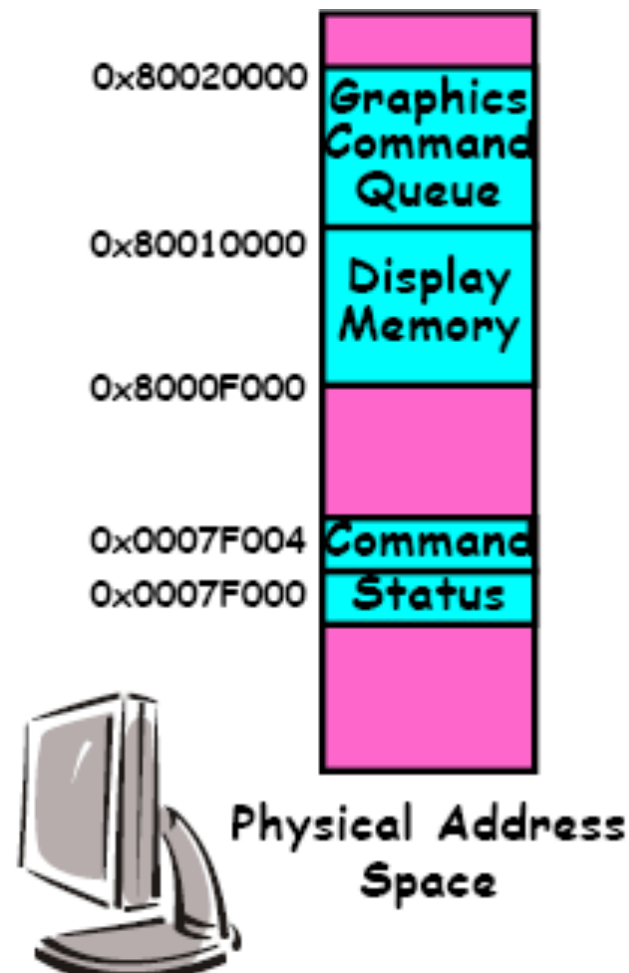
Controller dei dispositivi

- Come avviene accesso ai registri?
 - Dispositivi mappati in memoria (memory-mapped)
 - I registri sono visti nello spazio di indirizzamento della memoria
 - Accesso ai registri tramite le istruzioni di accesso alla memoria
 - Necessario disabilitare la cache
 - Permettono di scrivere driver in linguaggio ad alto livello
 - Nessuno speciale accorgimento per la protezione
 - Sufficiente allocare lo spazio di indirizzamento di I/O fuori dallo spazio utente
 - Dispositivi mappati su I/O (I/O-mapped)
 - Accesso ai registri tramite istruzioni specifiche
 - Nessun problema di gestione della cache
 - Soluzioni ibride (es.: Pentium)
 - Registri → I/O mapped
 - Buffer → memory mapped



Esempio di memory-mapped

- Controllore del display
 - L'HW mappa i registri e la memoria del display nello spazio di indirizzi fisico
 - Scrive nella mem (frame buffer) I cambi che avvengono sullo schermo. Addr: 0x80001000-0X8000F000
 - Scrive le descrizioni grafiche nell'area della coda di comandi
 - Es, un set di triangoli di una scena. Addr: 0x80002000-0X8001FFFF
 - Scrivere nel registro comando potrebbe causare all'HW grafico del display di fare qualcosa
 - Es. Rendere la scena appena scritta (Addr: 0x0007F004)



Accesso ai dispositivi di I/O

- Opzioni
 - Polling
 - Interrupt (I/O Programmato)
 - DMA (Direct memory access)

Polling

- Determina lo stato del dispositivo mediante lettura ripetuta del busy-bit del registro di status
 - Es.: Quando il busy bit è a 0 il comando viene scritto nel registro di controllo e il command-ready bit del registro di status viene posto a 1, quindi l'operazione di I/O viene eseguita
 - Ciclo di attesa attiva → spreco di CPU

Ciclo Busy-wait per attendere l'I/O dal dispositivo

H Busy bit = 0?

H Command register = write e data-out=[byte]

H Command-ready=1

C Command-ready=1 ? Then busy-bit=1

C Read command-register, read data-out, execute I/O

C Command-ready=0 and busy-bit=0



Interrupt

- Dispositivo di I/O “avverte” la CPU tramite un segnale su una connessione fisica
- Problematiche gestite da controllore di interrupt (circuiti)
 - Interrupt possono essere mascherabili
 - Interrupt ignorabile durante l’esecuzione di istruzioni critiche
 - Interrupt sono numerati
 - Valore = indice in una tabella (vettore di interrupt)
 - Vettore di interrupt
 - “Consegna” l’interruzione al gestore corrispondente
 - Basato su priorità
 - Interrupt multipli sono ordinati
 - Interrupt con priorità + alta prelaiono quelli con priorità + bassa

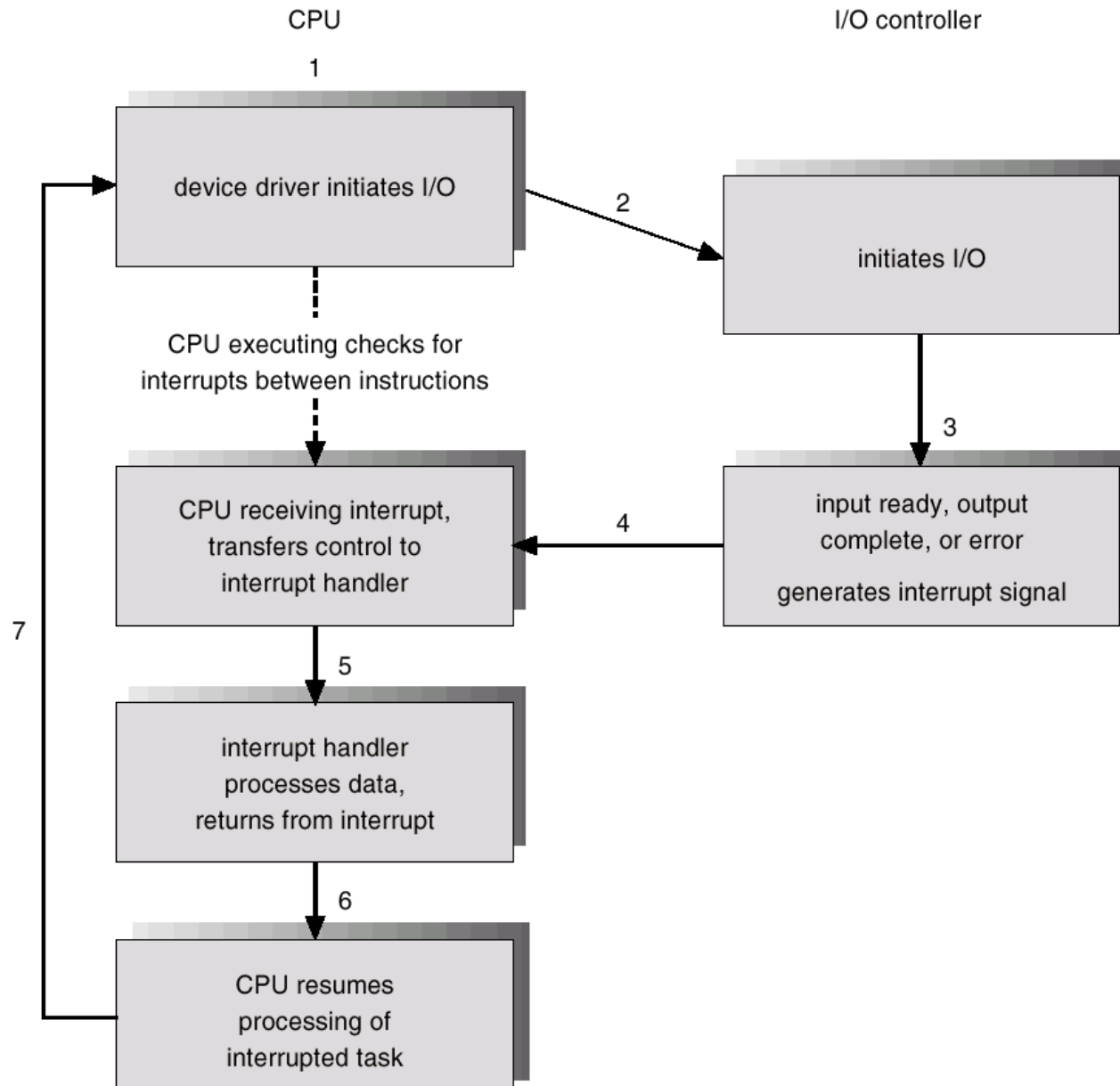


Vettore delle interruzioni della CPU (es. Pentium)

indice del vettore	descrizione
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19-31	(Intel reserved, do not use)
32-255	maskable interrupts



Ciclo di I/O basato su interrupt

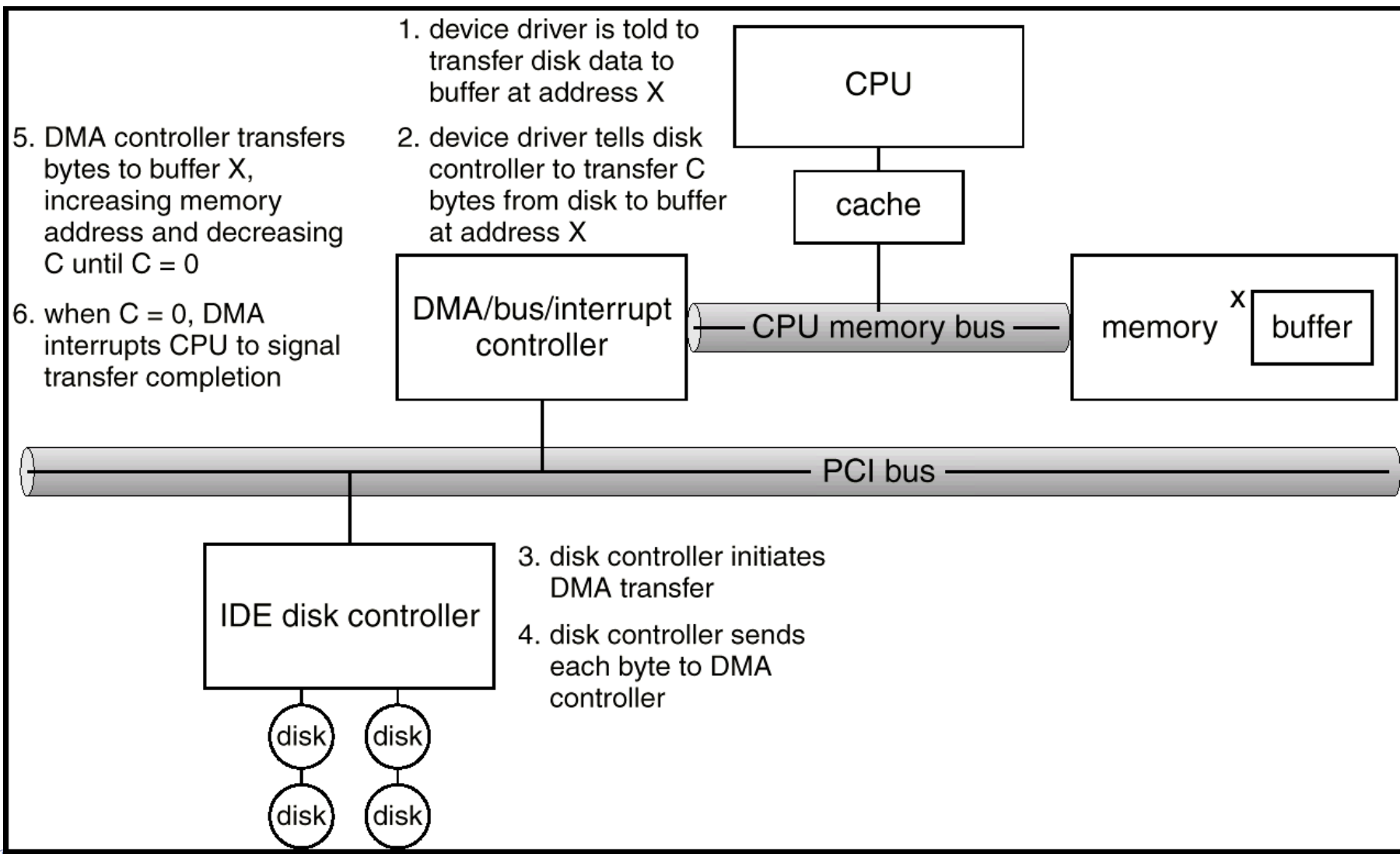


DMA – Direct Memory Access

- Pensato per evitare I/O programmato (PIO) per grandi spostamenti di dati
 - Usare CPU per controllare registro di stato del controllore e trasferire pochi byte alla volta è uno spreco
- Richiede esplicito HW (DMA controller)
- Basato sull'idea di bypassare la CPU per trasferire dati direttamente tra I/O e memoria
 - Per effettuare un trasferimento la CPU invia al DMAC :
 - L'indirizzo di partenza dei blocchi/memoria da trasferire
 - L'indirizzo di destinazione
 - Il numero di byte da trasferire
 - La direzione del trasferimento
 - Il DMA controller gestisce il trasferimento, comunicando con il controllore del dispositivo mentre la CPU effettua altre operazioni
 - Il DMA controller interrompe la CPU al termine del trasferimento



Trasferimento con DMA



INTERFACCIA DI I/O

Caratteristiche dei dispositivi di I/O

- I dispositivi di I/O si differenziano per molti aspetti

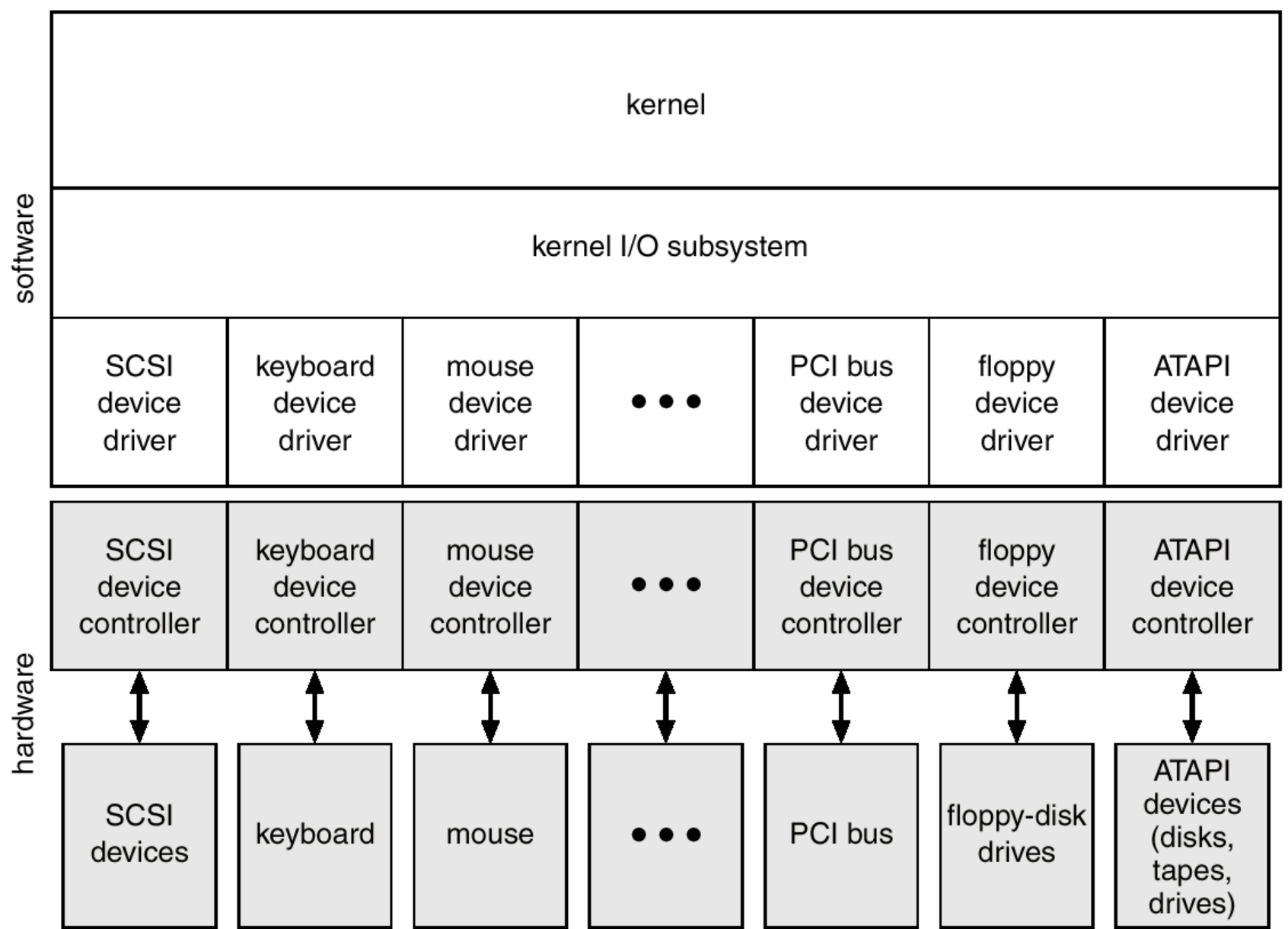
aspect	variation	example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
sharing	dedicated sharable	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only read&write	CD-ROM graphics controller disk

Interfaccia di I/O

- Come è possibile trattare i differenti dispositivi in modo standard ?
 - Usando *abstraction*, *encapsulation* e *software layering* per “nascondere” le differenze al kernel del S.O.
 - Ovvero costruendo un’interfaccia comune
 - Insieme di funzioni standard
 - Le differenze sono incapsulate nei device driver
- Interfacce semplificano il lavoro del S.O.
 - Posso aggiungere nuovo HW senza modificare S.O.



Struttura a livelli dell'I/O



Dispositivi a blocco e a carattere

- Interfaccia dispositivi a blocchi
 - Memorizzano e trasferiscono dati in blocchi
 - Lettura/scrittura di un blocco indipendentemente dagli altri
 - Comandi tipo: read, write, seek
 - Dispositivi tipici: dischi
 - Memory-mapped I/O sfrutta block-device driver
- Interfaccia dispositivi a carattere
 - Memorizzano/trasferiscono stringhe di caratteri
 - No indirizzamento (no seek)
 - Comandi tipo: get, put
 - Dispositivi tipici: terminali, mouse, porte seriali



SOFTWARE DI I/O

Software di I/O

- Obiettivi
 - Indipendenza dal dispositivo
 - Notazione uniforme
 - Gestione degli errori
 - Gestione di varie opzioni di trasferimento (sincrono/asincrono, ...)
 - Prestazioni
- Organizzazione per livelli di astrazione
 1. Gestori degli interrupt
 2. Device driver
 3. SW del S.O. indipendente dal dispositivo
 4. Programmi utente

Software di I/O

- Gestori degli interrupt
 - Astratti il più possibile dal resto del S.O.
 - Bloccaggio/sbloccaggio processi (semafori, segnali, ...)
- Device driver
 - Obiettivo: tradurre le richieste astratte del livello superiore in richieste device-dependent
 - Contengono tutto il codice device-dependent
 - Spesso condivisi per classi di dispositivi
 - Interagiscono con i controllori dei dispositivi
 - Tipicamente scritti in linguaggio macchina

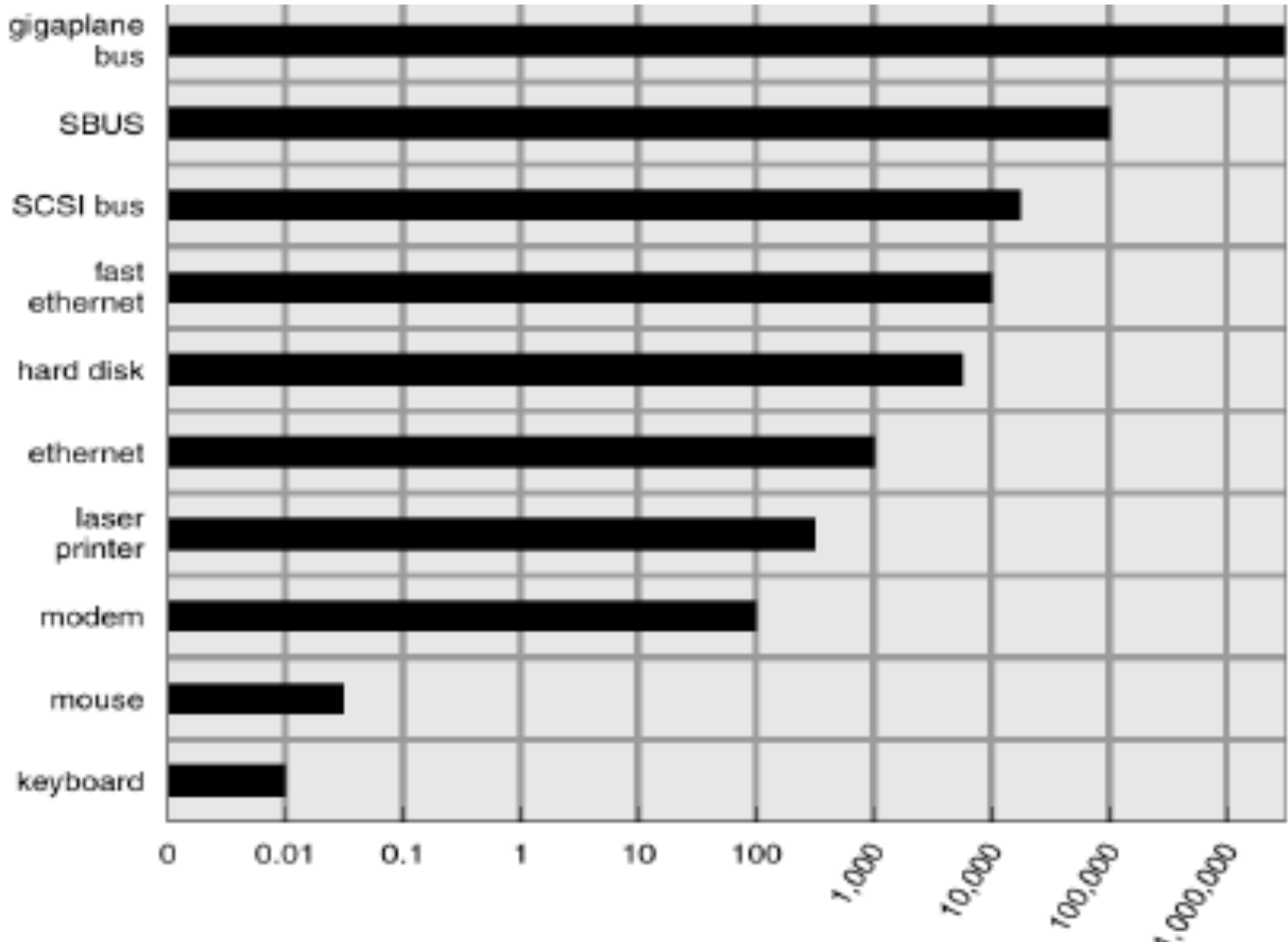


Software di I/O

- SW del S.O. indipendente dal dispositivo
 - Funzioni principali
 - Definizione di interfacce uniformi
 - Naming
 - Come individuare un device, e gestione nomi
 - Protezione
 - Tutte le primitive di I/O sono privilegiate
 - Buffering (Memorizzazione dei dati durante un trasferimento)
 - per gestire differenti velocità
 - per gestire differenti dimensioni (es.: pacchetti sulla rete vengono riassemblati)
 - Definizione della dimensione del blocco
 - Allocazione e rilascio dei dispositivi
 - Gestione errori
 - Tipicamente device-dependent, ma non il loro trattamento



Velocità di trasferimento



Buffering è
giustificato

Software di I/O

- Spooling: gestione di I/O dedicato (non condivisibile, es. stampante)
 - Più processi possono voler scrivere sulla stampante contemporaneamente, ma le stampe non devono essere interfogliate
 - Dati da processare sono “parcheggiati” in una directory (spooling directory)
 - Un processo di sistema (spooler) è l’unico autorizzato ad accedere alla stampante
 - Periodicamente, lo spooler si occupa di stampare i dati nella spooling directory
- Programmi utente
 - Tipicamente system call per l’accesso ai dispositivi



Ciclo di vita di una richiesta di I/O

