

SISTEMI DI INPUT/OUTPUT

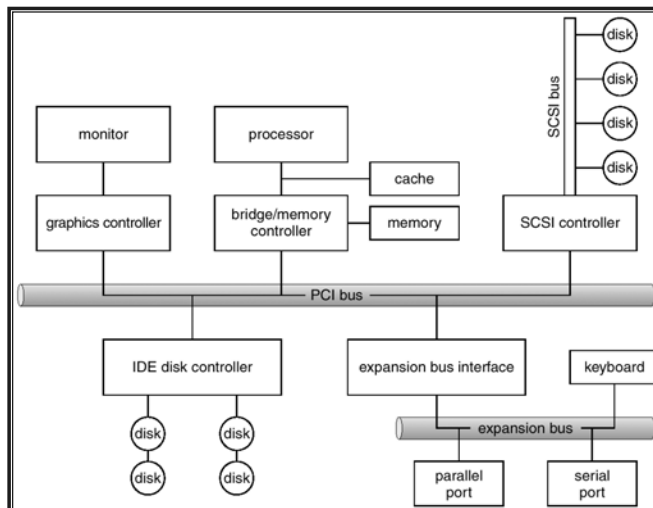
Sistemi I/O

- Hardware di I/O
- Interfaccia di I/O per le applicazioni
- Sottosistema per l'I/O del kernel
- Trasformazione delle richieste di I/O
- Stream
- Prestazioni

I/O Hardware

- Grande varietà di dispositivi di I/O → molti metodi di gestione.
- Concetti comuni:
 - Porte
 - Bus (*daisy chain* o accesso diretto condiviso)
 - Controller (adattatore host)
- I dispositivi hanno degli indirizzi che possono essere usati tramite due metodi:
 - Istruzioni di I/O
 - I/O mappato in memoria.

Struttura del Bus di un PC



Indirizzi dei dispositivi di I/O sui PC

I/O address range (hexadecimal)	device
000-00F	DMA controller
020-021	interrupt controller
040-043	timer
200-20F	game controller
2F8-2FF	serial port (secondary)
320-32F	hard-disk controller
378-37F	parallel port
3D0-3DF	graphics controller
3F0-3F7	diskette-drive controller
3F8-3FF	serial port (primary)

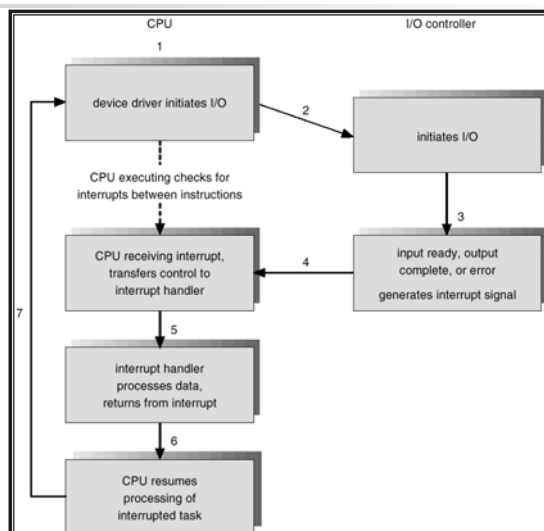
Interrogazione ciclica (Polling)

- L'interazione tra un controller e la CPU avviene secondo il modello produttore-consumatore.
- Per determinare lo stato del dispositivo si usano tre bit principali del registro di stato del dispositivo:
 - `command-ready`
 - `busy`
 - `error`
- La CPU esegue una interrogazione ciclica (*Busy-wait cycle*) ponendosi in attesa che il bit `busy` sia 0.
- Questo provoca una attesa attiva della CPU.

Interruzione (Interrupt)

- CPU ha una linea di richiesta di interrupt per ricevere un segnale dal dispositivo quando è libero. (attesa passiva)
- La CPU invoca una routine di gestione all'arrivo dell'interruzione.
- Esiste anche una linea degli interrupt "mascherabili" per evitare di ricevere interruzioni in particolari condizioni.
- Il vettore degli interrupt serve per invocare il gestore opportuno per ogni interrupt ricevuto.
 - Basato su priorità
 - alcuni interrupt non sono "mascherabili".
- Il meccanismo degli interrupt è anche usato per le eccezioni

Ciclo di I/O basato su interrupt



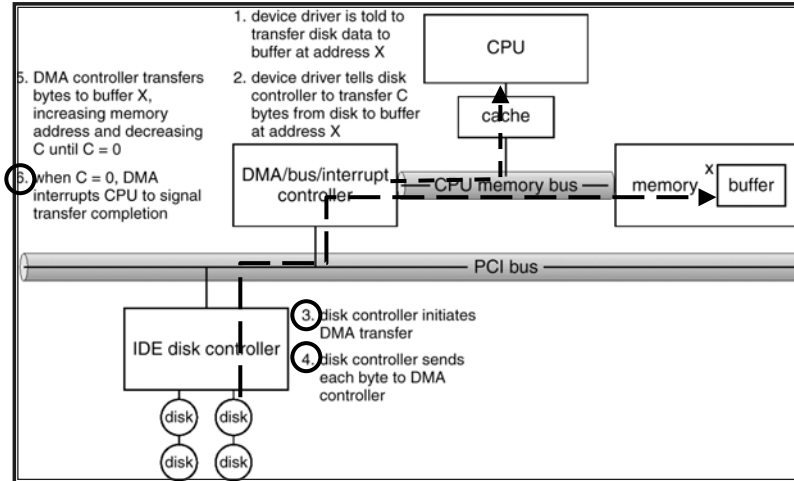
Vettore degli interrupt dell' Intel Pentium

vector number	description
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
19D31	(Intel reserved, do not use)
32D255	maskable interrupts

Direct Memory Access (DMA)

- DMA: processore che ha lo scopo di effettuare il controllo nel trasferimento di dati.
- Usato per liberare la CPU dai compiti di controllo.
- Richiede un controller DMA.
- L'interazione (*handshaking*) tra il controller DMA e il controller di un dispositivo realizza il trasferimento.

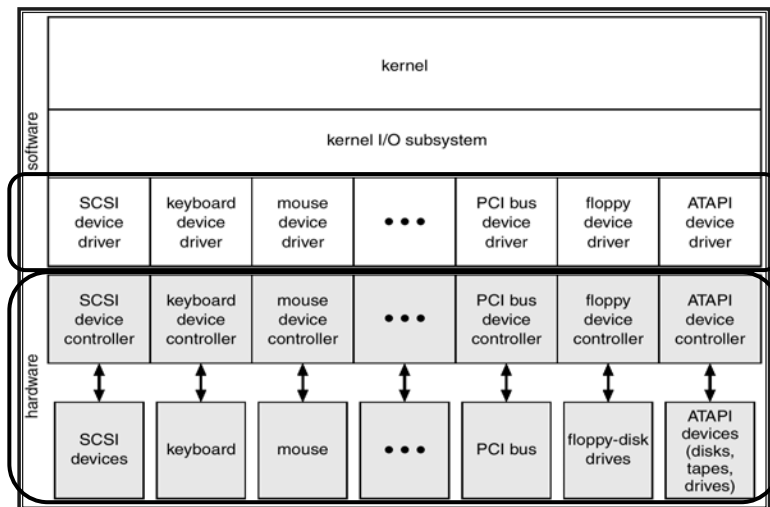
Passi di un trasferimento con DMA



Interfaccia di I/O per le Applicazioni

- Le system call di I/O offrono una interfaccia uniforme ai diversi comportamenti dei dispositivi.
- I driver dei dispositivi (*device-driver*) sono la parte del sistema operativo che permette di usare l'I/O in maniera uniforme (nascondendo le differenze).
- I dispositivi variano in molti aspetti
 - a caratteri o a blocchi,
 - ad accesso sequenziale o diretto,
 - condivisi o dedicati
 - con diverse velocità
 - lettura, scrittura o entrambi.

Struttura del kernel per I/O



Sistemi Operativi

10.13

D. Talia - UNICAL

Caratteristiche dei dispositivi di I/O

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 readDwrite	CD-ROM graphics controller disk

Sistemi Operativi

10.14

D. Talia - UNICAL

Dispositivi a caratteri o a blocchi

- I dispositivi a blocchi includono i dischi:
 - operazioni: lettura, scrittura, posizionamento
 - I/O di basso livello e accesso al file system
 - accesso ai file mappato in memoria.

- I dispositivi a caratteri includono la tastiera, il mouse, le porte seriali:
 - operazioni: `get`, `put`
 - si possono costruire operazioni più strutturate: su linee, su buffer di caratteri.

Dispositivi di rete

- Diversi dai dispositivi a caratteri o a blocchi.

- Unix e Windows NT/9//2000 includono una *socket interface*
 - Separa i protocolli di rete dalle operazioni di rete
 - Include l'operazione di `select` per controllare più porte.

- Esistono diversi molti modi di uso dei dispositivi di rete (*pipes*, *code FIFO*, *stream*, *mailbox*)

Clock e Timer

- Forniscono l'ora corrente, il tempo trascorso da un dato evento, il timer per indicare quando avviare un'operazione.
- Sistemi *timer programmabili* permettono di avviare operazioni o interrupt in dati intervalli temporali.
- In UNIX la system call `ioctl` nelle sue diverse forme permette di gestire operazioni con timer.

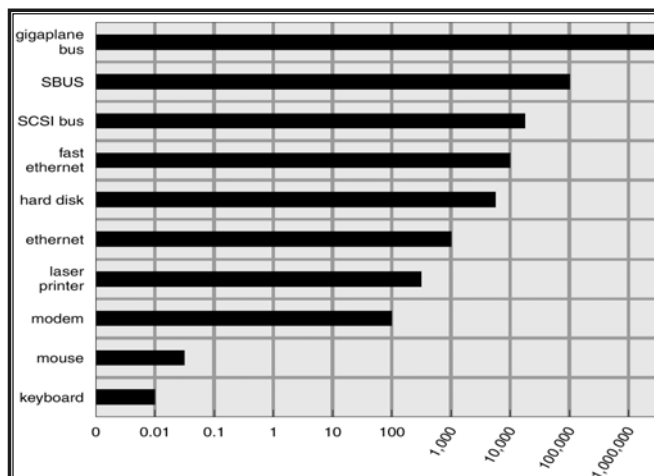
I/O Bloccante e non Bloccante

- **I/O bloccante** - il processo si sospende finché l'I/O non viene completato
 - facile da usare e capire
 - non opportuno in alcuni casi.
- **I/O non bloccante** - la chiamata di I/O ritorna appena possibile.
 - interfaccia utente, copia dei dati (*buffered I/O*)
 - implementato tramite multi-threading
 - ritorna velocemente con il numero di byte trasferiti.
- **I/O Asincrono** - il processo esegue mentre si svolge l'I/O
 - difficile da usare
 - il sottosistema di I/O segnala al processo il completamento dell'I/O.

Sottosistema per l'I/O del Kernel

- Il kernel offre un insieme di servizi per le operazioni di I/O.
- **Scheduling**
 - Alcune richieste di I/O sono ordinate tramite le codi dei dispositivi.
 - Alcuni S.O. usano delle politiche *fair* (eque).
- **Buffering**: memorizza dati in memoria centrale durante il loro trasferimento tra dispositivi:
 - tra dispositivi con velocità diverse
 - tra dispositivi con dimensioni di memoria diversa
 - per realizzare la "semantica di copia"

Velocità di trasfer. dei dispositivi sul Sun Enterprise



Sottosistema per l'I/O del Kernel

- **Caching** : memoria veloce per tenere copie di dati.
 - Differente dal buffering (è solo una copia).
 - Importante per le prestazioni.

- **Spooling** : mantiene l'output di un dispositivo.
 - Se il dispositivo può servire una richiesta per volta.
 - Esempio: una stampante.

- **Prenotazione dei dispositivi** : fornisce accesso esclusivo ad un dispositivo.
 - System call allocazione e deallocazione
 - controllare eventuale deadlock.

Gestione degli errori

- I sistemi operativi devono poter gestire situazioni di errore come ad esempio errori di lettura, scrittura o di dispositivo non disponibile.

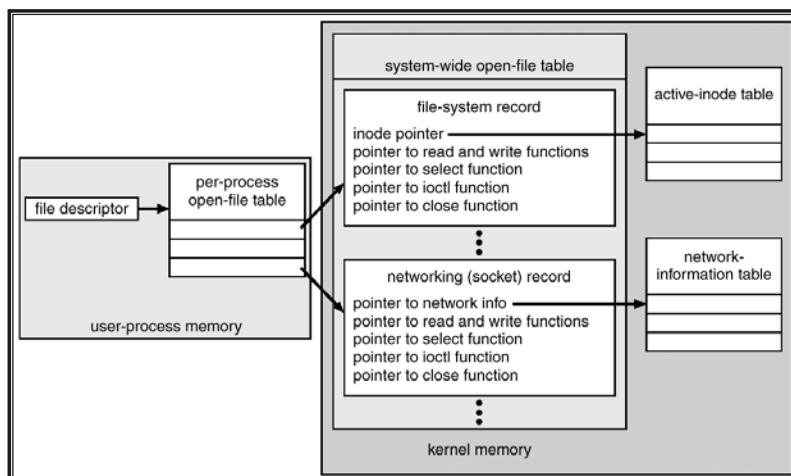
- La maggior parte in caso di errore ritornano un codice di errore (***errno*** in UNIX) che indica il malfunzionamento avvenuto.

- Gli errori verificatosi vengono raccolti in un log di sistema.

Strutture dati del Kernel

- Il kernel mantiene informazioni di stato dei dispositivi di I/O incluso la tabella dei file aperti, le connessioni di rete, lo stato dei dispositivi a carattere, ecc.
- Molte strutture dati sono necessarie per i buffer, l'allocazione della memoria, i blocchi liberi, ecc.
- Alcuni S.O.usano un approccio object oriented nella gestione dei dispositivi e usano lo scambio di messaggi per effettuare operazioni di I/O.

Strutture dati del Kernel per l'I/O in UNIX



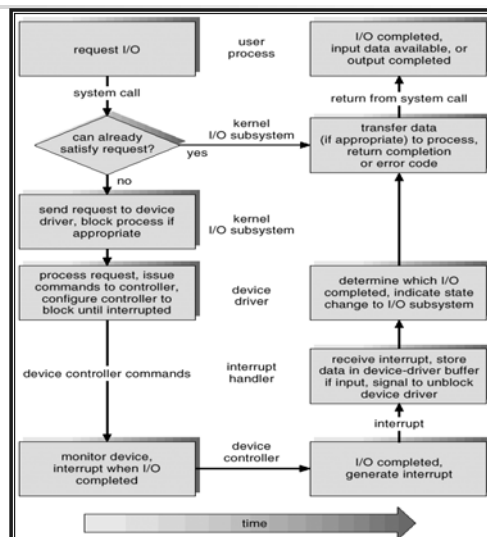
Dalle richieste di I/O alle operazioni hardware

■ Esempio:

Consideriamo un processo che esegue la lettura di un file dal disco. I passi da svolgere sono i seguenti:

- Determina il dispositivo che contiene il file
- Traduce il nome del file in quello di un dispositivo
- Legge fisicamente i dati dal disco nel buffer
- Rende i dati disponibili al processo
- Ritorna il controllo al processo.

Schema di esecuzione di una richiesta di I/O



STREAMS

■ **STREAM**

In UNIX è una canale di comunicazione full-duplex tra un processo utente e un dispositivo.

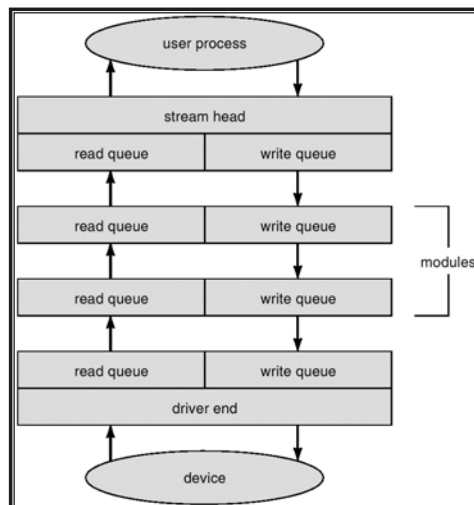
■ Una STREAM consiste di:

- **stream head** : interfaccia con il processo utente;
- **driver end** : interfaccia con il dispositivo;
- zero o più moduli stream tra il processo e il driver del dispositivo.

■ Ogni modulo contiene una **read queue** e una **write queue**

■ Lo scambio messaggi è usato per comunicare tra le code.

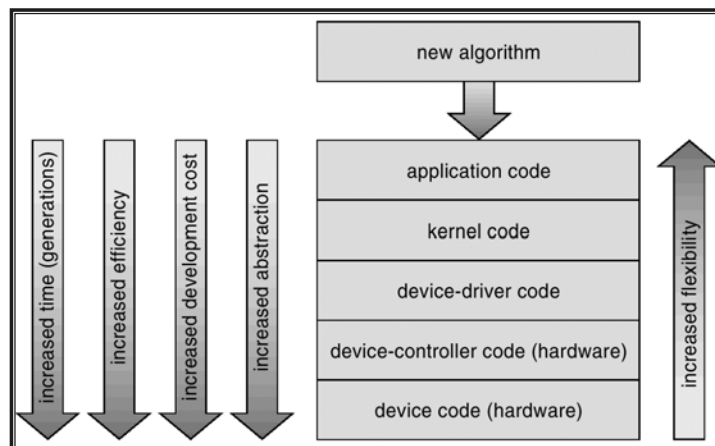
Struttura degli STREAMS



Migliorare le prestazioni

- Per un'esecuzione efficiente delle operazioni di I/O è necessario:
 - ridurre il numero di context switch
 - ridurre la copia di dati
 - ridurre gli interrupt usando trasferimenti di grandi quantità di dati in un'unica operazione, controller intelligenti, polling.
 - usare il DMA
 - bilanciare il carico di CPU, memoria, bus, e sistema di I/O.

Successione delle realizzazione dei servizi di I/O



Domande

- Spiegare la differenza tra l'interazione tramite polling e l'interruzione tramite interrupt tra CPU e dispositivi di I/O.
- Quali sono i benefici che derivano dalla disponibilità del DMA ?
- Spiegare il concetto di spooling e dove questo viene usato.
- Cosa è uno Stream in Unix e quali sono i benefici per le applicazioni che effettuano operazioni su dispositivi di I/O?