Lezione 5:

La gestione dell'I/O

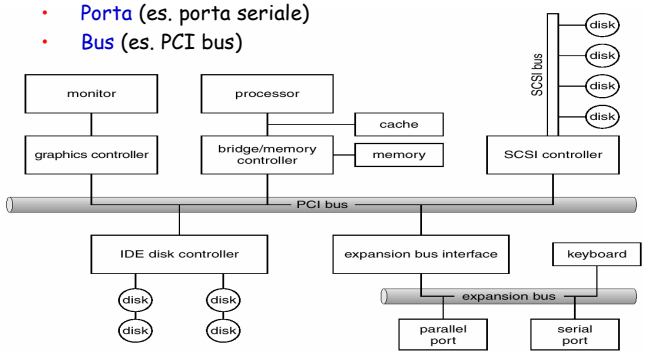
- Hardware di I/O
- Software di I/O
- Sottosistema per l'I/O del kernel
- Prestazioni

dispositivi di I/O

- I dispositivi di I/O sono l'interfaccia del calcolatore verso il mondo esterno
- Esiste una incredibile varietà di dispositivi di I/O.
  - Dispositivi classici di I/O
    (tastiera, schermo, mouse)
  - Dispositivi di memorizzazione
    (Dischi rigidi, nastri, CDROM)
  - Dispositivi di rete
    (Scheda di rete, radio, linea seriale)
  - Dispositivi multimediali
    (Videocam, microfono,)
  - ...

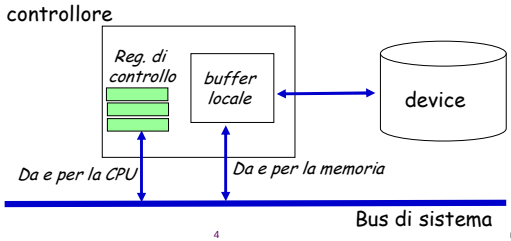
Metodi di accesso

- Il sistema operativo deve fornire metodi d'accesso uniformi per accedere a tali dispositivi in maniera semplice ed efficiente
- Concetti comuni:
  - Porta (es. porta seriale)
  - Bus (es. PCI bus)



Controllori dei dispositivi di I/O

- Ogni dispositivo e' governato da un *controller*
- Ogni *controller* ha
  - un suo buffer locale.
  - Registri di controllo
- Le operazioni di I/O avvengono dal dispositivo verso i buffer locali e viceversa.
- La CPU sposta i dati tra la memoria e il buffer

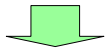


Esempio

- 3 registri di controllo
  - status (read only) indica lo stato del dispositivo
    - 1 dispositivo occupato
    - 0 dispositivo pronto ad eseguire un comando
  - command (write) indica il comando da eseguire
    - Read / write
  - control (write) indica che un comando e' pronto nel registro command
    - 1 comando da eseguire presente nel registro command
    - 0 nessun comando da eseguire
- 2 buffer
  - Di input
  - Di output
- L'insieme delle regole per il coordinamento tra CPU e controller e' chiamato negoziazione (handshaking)

Esempio di negoziazione per l'output

1. La CPU interroga il registro status fino a che status==0
2. La CPU definisce il registro command = write e trasferisce i dati dalla memoria al buffer di output
3. La CPU pone il registro control=1
4. Il controller pone status=1
5. Il controller esamina il registro command e trasferisce i dati dal buffer al dispositivo
6. Il controller pone command=0 e status=0



ATTESA ATTIVA nel passo 1

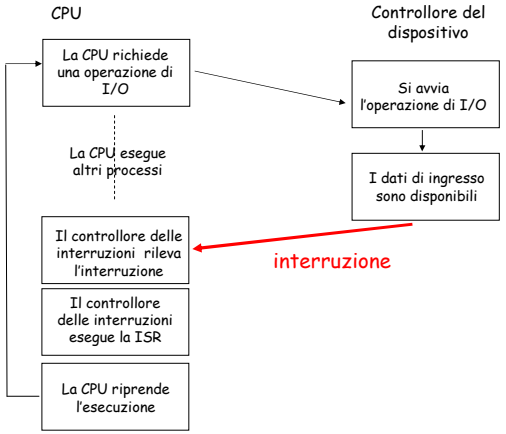
Attesa attiva (polling)

- Interrogare occasionalmente un dispositivo è in sé un'operazione efficiente
  - Leggi il registro status
  - Test sul contenuto
  - Salto ad un altro punto del codice se status=0
- tale tecnica diviene però inefficiente se le ripetute interrogazioni trovano raramente un dispositivo pronto per il servizio mentre altre utili elaborazioni attendono la CPU.

L'alternativa: le interruzioni

- I controllori dei dispositivi, gli errori e le chiamate di sistema generano segnali d'interruzione per comunicare alla CPU un evento asincrono
- Le interruzioni vengono generate dai dispositivi e recapitate alla CPU attraverso la linea delle richieste delle interruzioni
- Dipendono dall'architettura e possono avere nomi differenti
- Esempio: architettura IA32 (Intel Pentium)
  - Interrupt (se generati da dispositivi hardware)
  - Exceptions (se generati da errori o da programmi in esecuzione)

Ciclo di I/O basato sulle interruzioni



Accesso diretto alla memoria

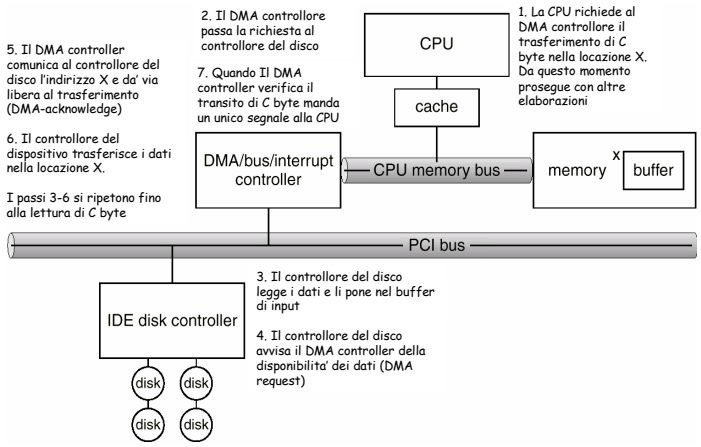
La dimensione di buffer dei controllori e' di pochi byte.

Quando devono essere trasferiti molti dati vengono generate molte interruzioni alla CPU

La gestione dell'I/O da e per la memoria e' di solito delegato ad una unita' separata dalla CPU chiamata
Controllore dell'Accesso Diretto alla Memoria (DMA controller)

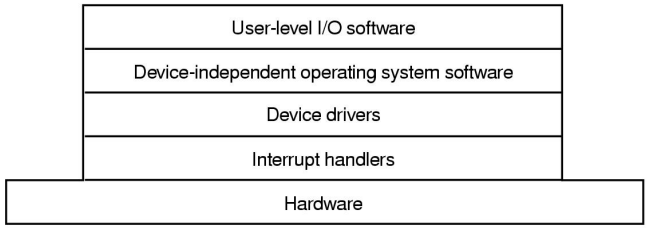
Il controllore DMA agisce direttamente sul bus della memoria, esegue il trasferimento senza l'aiuto della CPU e genera una sola interruzione

Passi di un trasferimento DMA



I livelli del software di I/O

Benche' l'organizzazione del software che gestisce l'I/O dipende dal sistema operativo e' in genere possibile individuare alcune componenti fondamentali



Gestore delle interruzioni

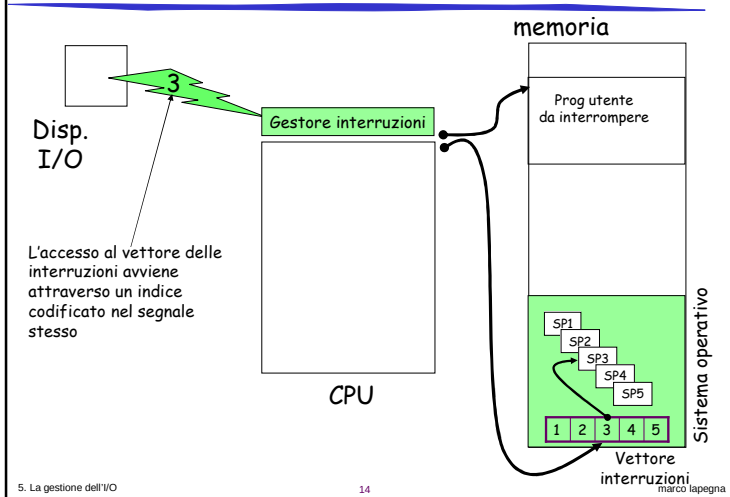
Problemi:

- Necessita' di differire le interruzioni nel caso di elaborazioni critiche
- Necessita' di un meccanismo efficiente e rapido per determinare quale dispositivo ha generato l'interruzione
- Necessita' di priorita' tra le varie interruzioni

Soluzione:

- La gestione e'affidata ad un dispositivo separato dalla CPU chiamato controllore delle interruzioni
- Per rendere efficiente la gestione delle interruzioni si usa di solito una sola Interrupt Service Routine (ISR) che gestisce una tabella di puntatori contenente gli indirizzi delle varie procedure di servizio (vettore delle interruzioni)

Gestione delle interruzioni



Compiti del gestore delle interruzioni

- Un dispositivo attiva una comunicazione elettrica con il controllore delle interruzioni della CPU
- La CPU interrompe l'esecuzione dell'istruzione corrente
- salva lo stato del programma in esecuzione (dati e registri della CPU).
- trasferisce il controllo ad una *interrupt service routine* (ISR) il cui codice si trova in una prefissata zona della memoria
- Le ISR (Interrupt Service Routine) eseguono il codice specifico per gestire il segnale appena giunto (ad es. trasferisce il dato dal buffer locale del controller alla memoria)
- Al termine dell'esecuzione della ISR, il S.O. ripristina lo stato del programma e riprende l'esecuzione

Vettore delle interruzioni della CPU intel Pentium

vector number	description	
0	divide error	errori
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	
19Ð31	(Intel reserved, do not use)	
32Ð255	maskable interrupts	dispositivi

Driver dei dispositivi

- Problema: differenze tra i vari controllori di dispositivo:
  - Es: i controller dei dischi possono essere molto diversi tra loro (# tracce, #settori,…)
- Il software che governa i dispositivi e rende uniforme l'accesso e' detto driver del dispositivo
- I driver hanno il compito di isolare il sottosistema di I/O del nucleo dai controllori dei dispositivi
  - Es: se si sostituisce un disco e' necessario sostituire solo il relativo driver senza modificare il resto del sistema operativo
- Poiche' ogni sistema operativo ha le sue convenzioni, ogni dispositivo necessita di un driver differente per ogni sistema operativo. Tale driver e' in genere scritto dal produttore del dispositivo

Driver dei dispositivi

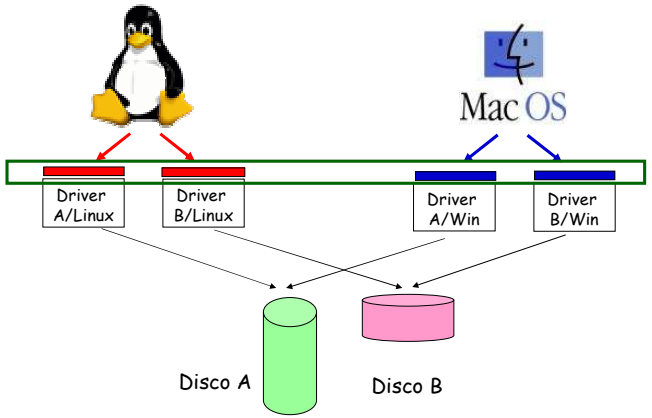
- 2 categorie di driver dei dispositivi
  - Dispositivi a blocco (ad es. dischi, fotocamere, cdrom)
  - Dispositivi a carattere (ad es. tastiera, terminali, stampanti, interfacce di rete)
- Funzioni principali dei driver
  - Accettare richieste di I/O dagli strati superiori del s.o.
  - Effettuare controlli sui dati
  - Gestire le differenti velocita' dei dispositivi (protocollo prod. cons.)
  - Garantire l'alimentazione, rilevare la presenza del dispositivo (es. hot swap dei dispositivi USB)

Sw di I/O indipendente dai dispositivi

- I driver dei dispositivi esportano le funzionalita' dei dispositivi di I/O verso il sistema operativo
- I driver sono strettamente legati al dispositivo, ma ogni s.o. utilizza un disco sempre allo stesso modo (ad es. differenti metodi di memorizzazione di un file)
- Problema: portabilita' delle applicazioni su calcolatori con dispositivi di I/O differenti:
  - Es: l'accesso ad un file dovrebbe essere sempre lo stesso indipendentemente dal tipo di disco su cui e' memorizzato

Necessita' di un livello software che renda standard l'accesso ai dispositivi

esempio



Compiti del sw di I/O indipendente dai dispositivi

- Interfacciamento uniforme
  - (rendere i dispositivi quanto piu' simili possibile)
- Bufferizzazione
  - (contribuire a gestire le differenti velocita' dei dispositivi)
- Gestire e riportare errori
  - (es. lettura da unita' di output e viceversa)
- Allocare e rilasciare risorse
  - (Garantire la mutua esclusione nell'accesso di alcuni dispositivi)

Sw di I/O a livello utente

- Librerie utente per l'I/O
  - (es. le chiamate di sistema scanf e printf)
- software per la gestione dei file speciali associati ai dispositivi
  - (es. spooling)

spooling

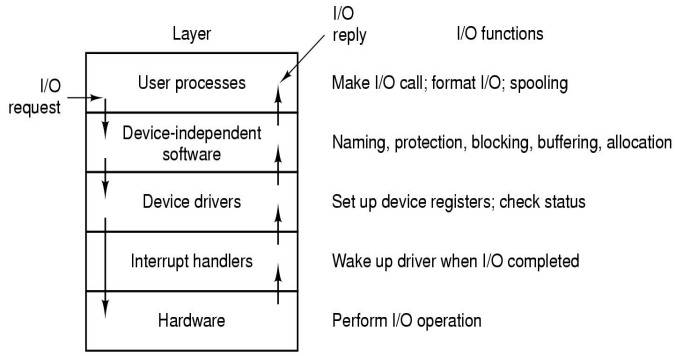
In molti s.o. (es. UNIX) i dispositivi sono associati a file speciali

Esempio: la directory /dev contiene tali file speciali

Problema : se un processo accede al file speciale associato ad una stampante e non lo rilascia per un lungo periodo, la stampante non puo' essere utilizzata da altri processi

Soluzione: si copia il file da stampare in una directory di spool che viene gestita da un apposito processo (demone) che e' l'unico ad avere diritti sul file speciale associato alla stampante

Riepilogo di una istruzione di I/O



Semantica dell' I/O

- Un dispositivo puo' essere utilizzato in molte maniere differenti
  - I/O bloccante o non bloccante
  - I/O bufferizzato o non bufferizzato
  - I/O sincrono o asincrono
- In genere il sistema operativo cerca di supportare quante più maniere possibile per ciascun dispositivo.
- La metodologia è in genere definita all'atto dell'apertura del dispositivo.

I/O bloccante e non bloccante

- Bloccante: il processo passa nella coda dei processi in attesa. Al termine dell'I/O il processo passa nella coda dei processi pronti
  - codice più facilmente comprensibile
  - insufficiente per alcune necessità
- Non bloccante: sovrappone elaborazione e I/O
  - si realizza attraverso il multithreading
  - restituisce rapidamente il controllo dell'applicazione fornendo un parametro che indica quanti byte di dati sono stati trasferiti

I/O sincrono e asincrono

- **sincrono**: il processo attende che l'operazione sia completata
  - Le chiamate di sistema per l'I/O che conosciamo sono tutte sincrone.
- **asincrono**: il processo prosegue mentre avviene l'I/O.
  - Di difficile utilizzo
  - Il sottosistema di I/O segnala al processo il completamento dell'I/O.
  - L'utente registra apposite funzioni da eseguire quando l'I/O è completato.
  - Chiamate di sistema apposite: ***aio_read(), aio_write()***

Differenze tra i dispositivi

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 readÐwrite	CD-ROM graphics controller disk

Sottosistema per l'I/O

- Molti servizi relativi all'I/O sono offerti dal sottosistema di I/O del nucleo:
  - Scheduling
  - Memorizzazione transitoria
  - Caching
  - Code (*Spooling*)
  - Gestione degli errori

Sottosistema per l'I/O

- Scheduling
  - fare lo scheduling di un insieme di richieste di I/O significa stabilire un ordine d'esecuzione efficace;
  - l'ordine in cui si verificano le chiamate del sistema delle applicazioni è raramente la scelta migliore.
  - Di solito si usa una coda per ogni dispositivo

- Memorizzazione transitoria:
  - un buffer (memoria di transito) è un'area di memoria che contiene dati mentre vengono trasferiti tra due dispositivi o fra un'applicazione e un dispositivo
  - necessità di gestire la differenza di velocità dei dispositivi (es. prod. cons.)
  - gestione dei dispositivi che trasferiscono dati in blocchi di dimensioni diverse
  - realizzazione della "semantica delle copie"

Sottosistema per l'I/O

- Cache:
  - regione di memoria veloce per copie di dati
  - sempre solo copia di informazioni già memorizzate
  - migliora l'efficienza

- Code (*spooling*):
  - memoria di transito contenente dati per un dispositivo che non può accettare flussi di dati intercalati quando il dispositivo può gestire solo una richiesta alla volta (es.: una stampante)

- Uso esclusivo dei dispositivi
  - Un processo può accedere a un dispositivo che non sia già attivo riservandosene l'uso e restituendolo al sistema quando non ne ha più bisogno
  - Le appilcazioni hanno la responsabilità di evitare situazioni di stallo

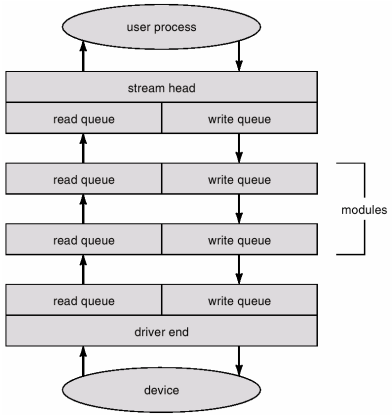
Sottosistema per l'I/O

- Gestione degli errori
  - Esempi: sovraccarico della rete, dispositivo difettoso,…
  - Un sistema operativo che usa la protezione della memoria può proteggersi da molti tipi di errori dovuti ai dispositivi o alle applicazioni ➔ difficile il blocco del sistema
  - Di norma, una chiamata del sistema per l'I/O riporta un bit (o un flag) d'informazione sullo stato d'esecuzione della chiamata (es. UNIX usa la variabile errno)
  - Molti dispositivi SCSI mantengono alcune pagine di informazioni sugli errori avvenuti; queste pagine possono essere richieste dalla macchina, ma ciò accade raramente

STREAM

- **STREAM**: e' una connessione bidirezionale tra un driver dispositivo e un processo utente.
- uno STREAM consiste di:
  - un elemento iniziale d'interfaccia per il processo utente (*STREAM head*)
  - un elemento terminale che controlla il dispositivo (*driver end*)
  - un certo numero di moduli intermedi fra questi due estremi.
- Tutti questi elementi possiedono una coppia di code, una di lettura e una di scrittura.
- Per il trasferimento dei dati tra le due code, si usa uno schema a scambio di messaggi.

STREAM



Performance

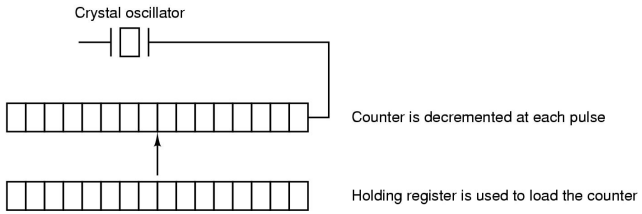
- L'I/O è uno tra i principali fattori che influiscono sulle prestazioni di un sistema:
  - richiede un notevole impegno di CPU per l'esecuzione del codice del driver e per uno scheduling equo ed efficiente
  - i risultanti cambi di contesto sfruttano fino in fondo la CPU e le sue memorie cache
  - copia dei dati
  - traffico di rete

Miglioramento delle performance

- Per migliorare l'efficienza dell'I/O si possono applicare diversi principi:
  - ridurre il numero dei cambi di contesto
  - ridurre il numero di copiature dei dati
  - ridurre la frequenza delle interruzioni tramite il trasferimento di grandi quantità di dati in un'unica soluzione, l'uso di controllori intelligenti e mediante l'interrogazione ciclica
  - uso di controllori DMA intelligenti
  - equilibrare le prestazioni della CPU, del sottosistema per la gestione della memoria, del bus e dell'I/O

Clock o timer

- Ogni calcolatore ha un orologio interno chiamato clock o timer
- 3 componenti
  - Cristallo al quarzo che vibra ad una fissata frequenza
  - Contatore che viene decrementato da ogni impulso del cristallo
  - Registro di inizializzazione del contatore



clock

- Il contatore viene inizializzato dal registro di caricamento
- Quando il contatore si azzera viene emessa una interruzione

La frequenza delle interruzioni viene determinata dal software
(dal valore del registro di caricamento)

Esempio: con un cristallo da 500 MHz e registro di 32 bit. ➔ il contatore viene decrementato ogni 2 ns

Interruzioni ogni 2 ns (registro di caricamento =1
Interruzioni ogni 8,6 sec (registro di caricamento = 2^{32})

Principali compiti del sw dei clock

- Mantenere ora e data in appositi registri di 32 o 64 bit anche quando si spegne il calcolatore
  - Contando i tic dal 1 gennaio 1970 (UNIX) o dal 1 gennaio 1980 (Windows)
- Evitare il monopolio della CPU da parte di un processo
- Tenere la contabilita' dell'uso delle risorse
  - Altri registri
- Ritardare segnali e interruzioni nel caso di processi critici

Terminali

- E' una delle interfacce utente principali
- 3 tipi di terminali
  - Terminali RS-232. Dispongono di tastiera e schermo. Comunicano con il s.o. un bit alla volta. Sono orientati a carattere
  - Display grafici. Hanno bisogno di una CPU e di memoria locale. Devono gestire il software per le finestre.
  - Terminali di rete. Utilizzati per accedere a risorse remote

Sw per terminali

- Dipende dal tipo di teminale
  - Semplice bufferizzazione di linee per i terminali RS-232
  - Librerie sofisticate per la gestione delle finestre nei terminali grafici (es. X-windows)