Università di Udine — Facoltà di Scienze MM.FF.NN.

Laurea in Informatica — A.A. 2005/06

Trasparenze del Corso di *Sistemi Operativi*

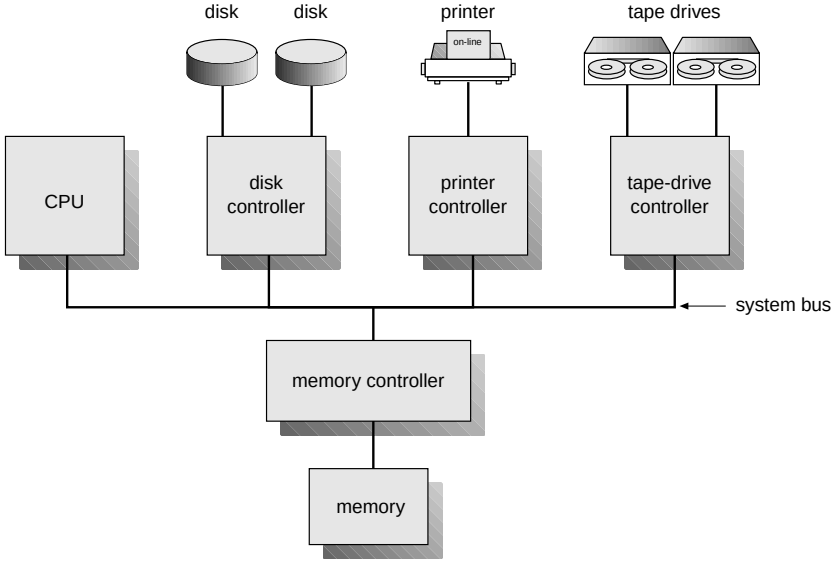
Ivan Scagnetto

Università di Udine

Struttura dei Sistemi di Calcolo

- Operazioni dei sistemi di calcolo
- Struttura dell'I/O
- Struttura della memoria
- Gerarchia delle memorie
- Protezione hardware
- Invocazione del Sistema Operativo

Architettura dei calcolatori



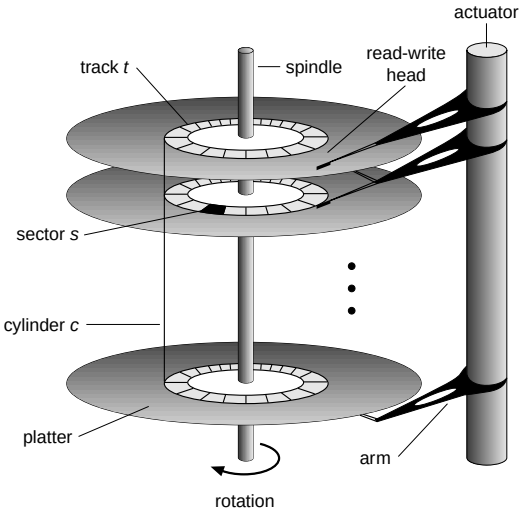
Struttura della Memoria

- Memoria principale – la memoria che la CPU può accedere direttamente.
- Memoria secondaria – estensione della memoria principale che fornisce una memoria non volatile (e solitamente più grande)

Dischi magnetici

piatti di metallo rigido ricoperti di materiale ferromagnetico, in rotazione

- la superficie del disco è logicamente divisa in *tracce*, che sono sottodivise in *settori*
- Il *controller dei dischi* determina l'interazione logica tra il dispositivo ed il calcolatore

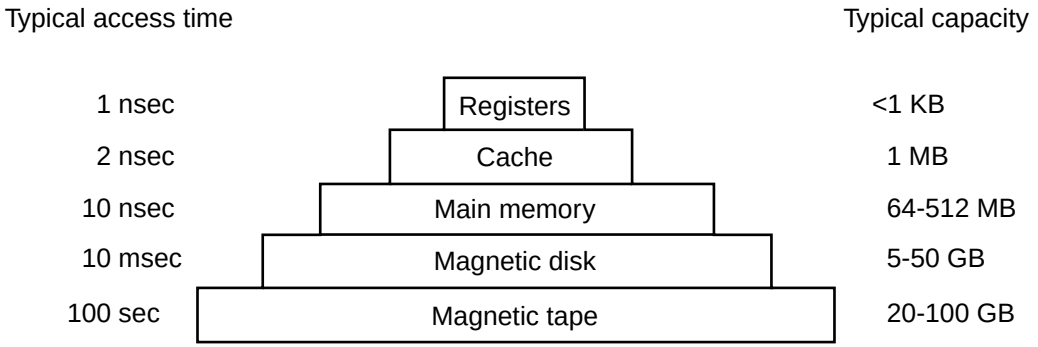


Gerarchia della Memoria

I sistemi di memorizzazione sono organizzati gerarchicamente, secondo

- velocità
- costo
- volatilità

Caching – duplicare i dati più frequentemente usati di una memoria, in una memoria più veloce. La memoria principale può essere vista come una cache per la memoria secondaria.

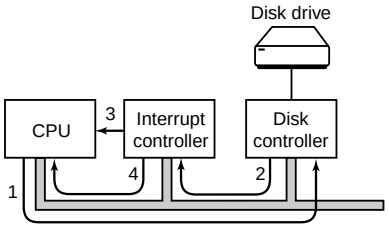


Operazioni dei sistemi di calcolo

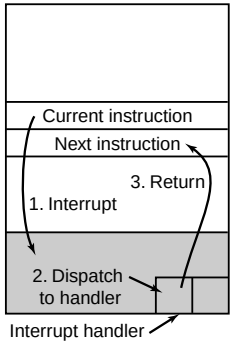
- I dispositivi di I/O e la CPU possono funzionare concorrentemente
- Ogni controller di dispositivo gestisce un particolare tipo di dispositivo.
- Ogni controller ha un buffer locale
- La CPU muove dati da/per la memoria principale per/da i buffer locali dei controller
- l'I/O avviene tra il dispositivo e il buffer locale del controller
- Il controller informa la CPU quando ha terminato la sua operazione, generando un *interrupt*.

Schema comune degli Interrupt

- Gli interrupt trasferiscono il controllo alla routine di servizio dell'interrupt. Due modi:
  - *polling*
  - *vettore di interruzioni:* contiene gli indirizzi delle routine di servizio.



(a)



(b)

Gestione dell'interrupt

- L'hardware salva l'indirizzo dell'istruzione interrotta (p.e., sullo stack).
- Il S.O. preserva lo stato della CPU salvando registri e program counter in apposite strutture dati.
- Per ogni tipo di interrupt, uno specifico segmento di codice determina cosa deve essere fatto.
- Terminata la gestione dell'interrupt, lo stato della CPU viene riesumato e l'esecuzione del codice interrotto viene ripresa.
- Interrupt in arrivo sono *disabilitati* mentre un altro interrupt viene gestito, per evitare che vadano perduti.
- Un *trap* è un interrupt generato da software, causato o da un errore o da una esplicita richiesta dell'utente (istruzioni `TRAP`, `SVC`).
- Un sistema operativo è *guidato da interrupt*

I/O sincrono

I/O sincrono: dopo che l'I/O è partito, il controllo ritorna al programma utente solo dopo che l'I/O è stato completato

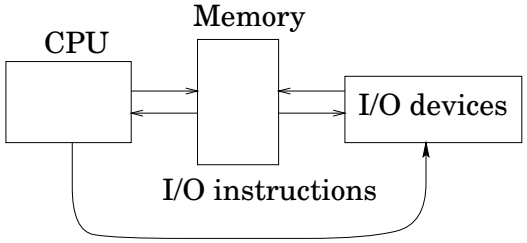
- l'istruzione **wait** blocca la CPU fino alla prossima interruzione
- oppure, un ciclo di attesa (*busy wait*); accede alla memoria
- al più una richiesta di I/O è eseguita alla volta; non ci sono I/O paralleli

I/O asincrono

I/O asincrono: dopo che l'I/O è partito, il controllo ritorna al programma utente senza aspettare che l'I/O venga completato

- *chiamata di sistema* (*System call*) – richiede al sistema operativo di sospendere il processo in attesa del completamento dell'I/O.
- una *tabella dei dispositivi* mantiene tipo, indirizzo e stato di ogni dispositivo di I/O.
- Il sistema operativo accede alla tabella dei dispositivi per determinare lo stato e per mantenere le informazioni relative agli interrupt.

Direct Memory Access (DMA)



- Usata per dispositivi in grado di trasferire dati a velocità prossime a quelle della memoria
- I controller trasferiscono blocchi di dati dal buffer locale direttamente alla memoria, senza intervento della CPU.
- Viene generato un solo interrupt per blocco, invece di uno per ogni byte trasferito.

Protezione hardware

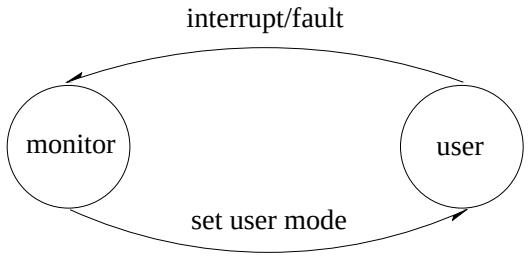
- Funzionamento in dual-mode
- Protezione dell'I/O
- Protezione della Memoria
- Protezione della CPU

Funzionamento Dual-Mode

- La condivisione di risorse di sistema richiede che il sistema operativo assicuri che un programma scorretto non possa portare altri programmi (corretti) a funzionare non correttamente.
- L'hardware deve fornire un supporto per differenziare almeno tra due modi di funzionamento
  1. *User mode* – la CPU sta eseguendo codice di un utente
  2. *Monitor mode* (anche *supervisor mode*, *system mode*, *kernel mode*) – la CPU sta eseguendo codice del sistema operativo

Funzionamento Dual-Mode (Cont.)

- La CPU ha un *Mode bit* che indica in quale modo si trova: supervisor (0) o user (1).
- Quando avviene un interrupt, l'hardware passa automaticamente in modo supervisore



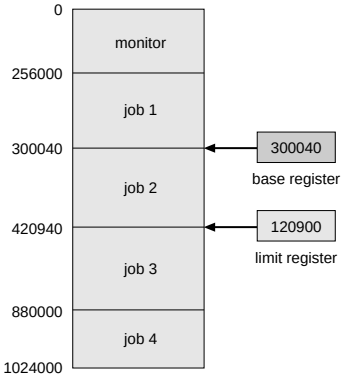
- Le *istruzioni privilegiate* possono essere eseguite solamente in modo supervisore

Protezione dell'I/O

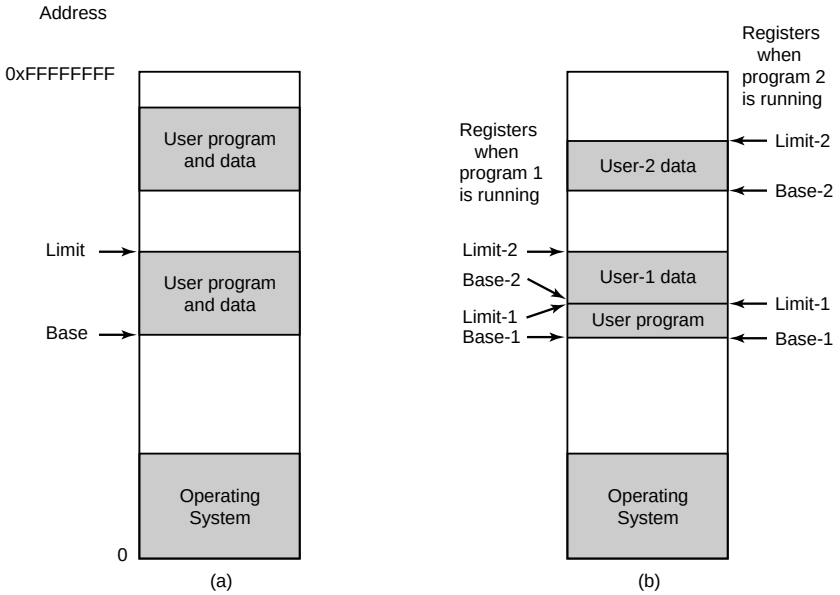
- Tutte le istruzioni di I/O sono privilegiate
- Si deve assicurare che un programma utente non possa mai passare in modo supervisore (per esempio, andando a scrivere nel vettore delle interruzioni)

Protezione della Memoria

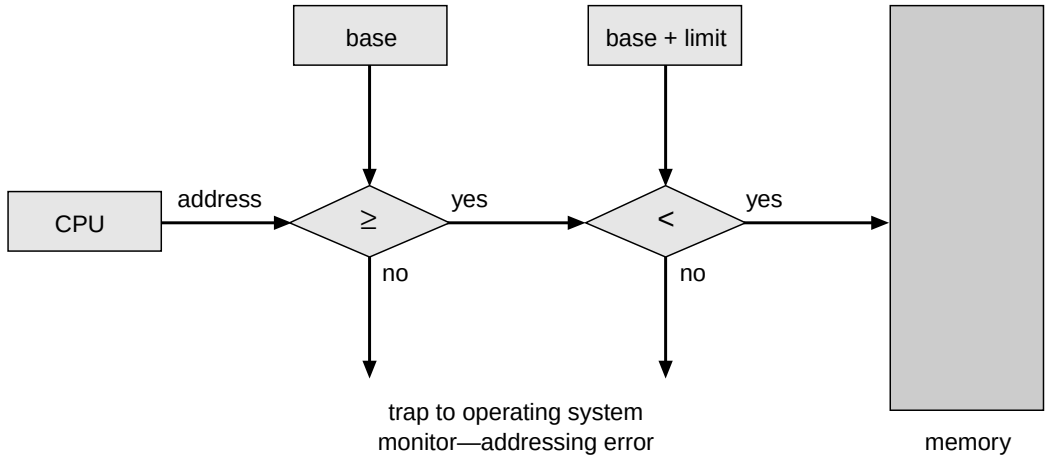
Si deve proteggere almeno il vettore delle interruzioni e le routine di gestione degli interrupt



- Per avere la protezione della memoria, si aggiungono due registri che determinano il range di indirizzi a cui un programma può accedere:
  registro base contiene il primo indirizzo fisico legale
  registro limite contiene la dimensione del range di memoria accessibile
- la memoria al di fuori di questo range è protetta



Protezione della Memoria (Cont.)



- Essendo eseguito in modo monitor, il sistema operativo ha libero accesso a tutta la memoria, sia di sistema sia utente
- Le istruzioni di caricamento dei registri base e limite sono privilegiate

Protezione della CPU

- il *Timer* interrompe la computazione dopo periodi prefissati, per assicurare che periodicamente il sistema operativo riprenda il controllo
  - Il timer viene decrementato ad ogni *tick* del clock (1/50 di secondo, tipicamente)
  - Quanto il timer va a 0, avviene l'interrupt
- Il timer viene usato comunemente per implementare il time sharing
- Serve anche per mantenere la data e l'ora
- Il caricamento del timer è una istruzione privilegiata

Invocazione del sistema operativo

- Dato che le istruzioni di I/O sono privilegiate, come può il programma utente eseguire dell'I/O?
- Attraverso le *system call* – il metodo con cui un processo richiede un'azione da parte del sistema operativo
  - Solitamente sono un interrupt software (**trap**)
  - Il controllo passa attraverso il vettore di interrupt alla routine di servizio della trap nel sistema operativo, e il mode bit viene impostato a "monitor".
  - Il sistema operativo verifica che i parametri siano legali e corretti, esegue la richiesta, e ritorna il controllo all'istruzione che segue la system call.
  - Con l'istruzione di ritorno, il mode bit viene impostato a "user"