SO, Computer e Programmi Utente

Sistemi Operativi

Antonino Staiano
Email: antonino.staiano@uniparthenope.it

Introduzione

- Principi fondamentali delle funzioni di un SO
- Il Computer
- Interazione del SO con il Computer e i programmi utente

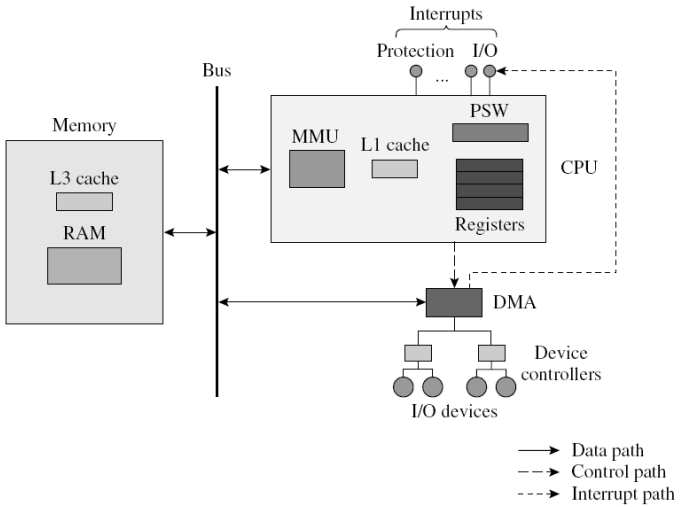
Principi Fondamentali

- Il *kernel* del SO è l'insieme di routine che costituiscono il nucleo del SO
  - Implementa le funzioni di controllo
  - Fornisce un insieme di servizi per i programmi utente
  - Si trova in memoria durante il funzionamento di un SO
- Un'*interruzione* (***interrupt***) dirotta la CPU verso l'esecuzione di codice kernel
- Un interrupt software è usato dai programmi per comunicare le loro richieste al kernel

Principi Fondamentali SO (cont.)

- Il kernel deve assicurare che non ci siano mutue interferenze fra i programmi degli utenti e tra un programma utente e il SO
- La CPU ha due modalità operative:
  - Modalità kernel
    - La CPU può eseguire tutte le istruzioni
    - Il kernel opera con la CPU in tale modalità in modo da poter controllare le operazioni del computer
  - Modalità utente
    - La CPU non può eseguire le istruzioni che possono interferire con altri programmi o con il SO se usata in modo indiscriminato
    - CPU lavora in tale modalità per eseguire i programmi utente

Il Computer



Il Computer (cont.)

- La CPU
- Unità di gestione della memoria (MMU)
- Gerarchia della memoria
- Input/Output
- Interrupt

La CPU

- Sono visibili due caratteristiche della CPU ai programmi utente o al SO:
  - *Registri general-purpose* (**GPR**)
    - Chiamati anche registri accessibili ai programmi
    - Mantengono dati, indirizzi, valori, o stack pointer durante l'esecuzione di un programma
  - *Registri di controllo*
    - Contengono informazioni che controllano o influenzano le operazioni della CPU
    - L'insieme dei registri di controllo è chiamato *Program Status Word* (**PSW**)
      - Ciascun registro di controllo del PSW è riferito come un *campo* del PSW

Campi Principali del PSW

Program counter (PC)	Condition code (CC)	Mode (M)	Memory protection information (MPI)	Interrupt mask (IM)	Interrupt code (IC)

Field	Description
Program counter	Contains address of the next instruction to be executed.
Condition code (flags)	Indicates some characteristics of the result of the last arithmetic or logical instruction, e.g., whether the result of an arithmetic instruction was < 0, $= 0$, or > 0. This code is used in execution of a conditional branch instruction.
Mode	Indicates whether the CPU is executing in kernel mode or user mode. We assume a single-bit field with the value 0 to indicate that the CPU is in kernel mode and 1 to indicate that it is in user mode.
Memory protection information	Memory protection information for the currently executing program. This field consists of subfields that contain the *base register* and *size register*.
Interrupt mask	Indicates which interrupts are enabled (that is, which interrupts can occur at present) and which ones are masked off.
Interrupt code	Describes the condition or event that caused the last interrupt. This code is used by an interrupt servicing routine.

La CPU (cont.)

- La CPU può operare in due modalità:
  - *Modalità kernel*
    - Può eseguire *istruzioni privilegiate*
    - Il SO pone la CPU in modalità kernel quando sta eseguendo istruzioni nel kernel
  - *Modalità utente*
    - Non può eseguire istruzioni privilegiate
    - Il SO pone la CPU in modalità utente mentre esegue programmi utente
- Il campo *Mode* (**M**) del PSW contiene 0 se la CPU è nella modalità kernel (privilegiata) e 1 se è nella modalità utente

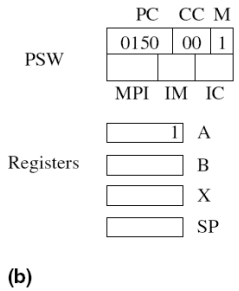
Stato della CPU

- GPR e PSW contengono tutte le informazioni necessarie per sapere cosa sta facendo la CPU
  - Stato della CPU
- Il kernel salva lo stato della CPU quando toglie la CPU ad un programma in esecuzione
  - Quando il programma deve riprendere, ricarica lo stato salvato della CPU nei GPR e PSW

Esempio: Stato della CPU (cont.)

Address		Instruction	
0142		MOVE	A, ALPHA
0146		COMPARE	A, 1
0150		BEQ	NEXT
		...	
0192	NEXT		
		...	
0210	ALPHA	DCL_CONST	1

(a)



(b)

(a) Listato di un programma *assembly* che mostra l'indirizzo associato ad ogni istruzione o dato.

(b) Stato della CPU dopo l'esecuzione dell'istruzione `COMPARE`.

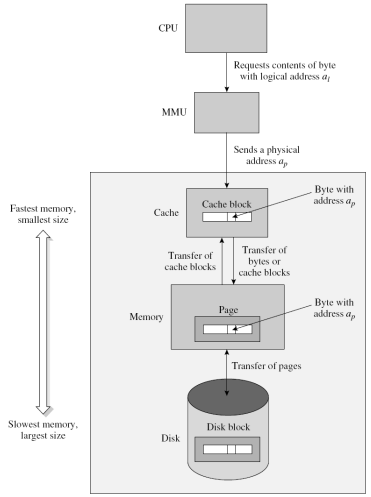
Unità di Gestione della Memoria (MMU)

- La *memoria virtuale* è un'illusione di una memoria più grande di una memoria della memoria reale di un computer
  - Implementata usando l'allocazione di memoria non contigua e l'MMU
    - La CPU passa l'indirizzo di un dato o dell'istruzione, usato nell'istruzione corrente, alla MMU
      - E' chiamato *indirizzo logico*
    - La MMU traduce l'indirizzo logico in un *indirizzo fisico*

Gerarchia della Memoria

- La *gerarchia della memoria* fornisce una memoria ampia e veloce, a basso prezzo
  - E' una disposizione di numerose memorie con diverse velocità di accesso e dimensioni
    - La CPU accede solo alla memoria più veloce, ovvero la *cache*
    - Se un byte richiesto non è presente nella memoria a cui si sta accedendo, esso viene caricato da una memoria più lenta
    - Il tempo di accesso effettivo dipende da quante volte si verifica tale situazione in una memoria più veloce

Funzionamento di una Gerarchia di Memoria



Gerarchia di Memoria (cont.)

- Quando la CPU esegue una ricerca nella cache, può verificarsi un successo (hit) o un fallimento (miss)
  - Lo hit ratio (**h**) della cache è la frazione di byte acceduti dalla CPU che comportano un successo della ricerca nella cache

- Temo accesso effettivo (gerarchia con cache e memoria)

$$t_{ema} = h \times t_{cache} + (1 - h) \times (t_{tra} + t_{cache})$$
$$= t_{cache} + (1 - h) \times t_{tra}$$

Dove

t_{ema} = tempo di accesso alla memoria effettivo,
t_{cache} = tempo di accesso alla cache, e
t_{tra} = tempo impiegato per trasferire un blocco della cache dalla memoria alla cache

Gerarchia di Memoria (cont.)

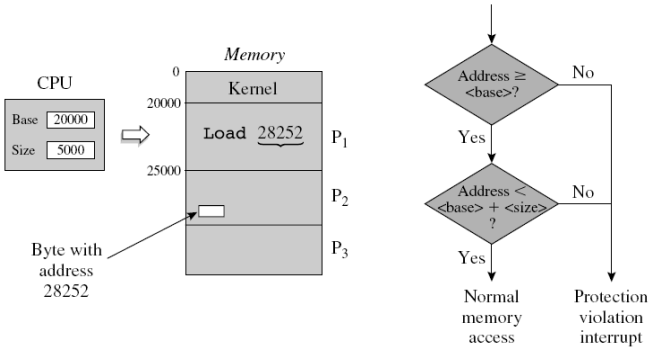
- Il funzionamento della memoria è analogo al funzionamento della cache
  - Blocchi di byte (*pagine*) sono trasferiti dal disco alla memoria o dalla memoria al disco
  - Ma,
    - La gestione della memoria ed il trasferimento dei blocchi tra memoria e disco sono eseguiti dal SW
    - Nella cache, il trasferimento è eseguito dallo HW
- La gerarchia di memoria che comprende MMU, memoria e il disco è chiamata *memoria virtuale*

Gerarchia di Memoria

- La *protezione della memoria* è implementata controllando se un indirizzo di memoria usato da un programma si trova fuori dall'area di memoria ad esso allocata
  - Registri di controllo usati:
    - *Base* e *size* (chiamato anche limite)
      - Indirizzo del primo byte = <base>
      - Indirizzo dell'ultimo byte = <base> + <size> - 1

Esempio: Fondamenti di Protezione della Memoria

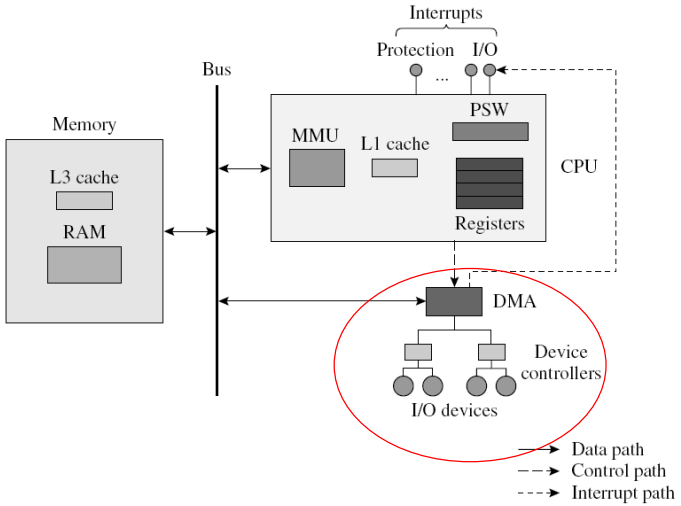
- L'esecuzione di una istruzione `load` causa una violazione della protezione



Input/Output

I/O mode	Description
Programmed I/O	Data transfer between the I/O device and memory takes place through the CPU. The CPU cannot execute any other instructions while an I/O operation is in progress.
Interrupt I/O	The CPU is free to execute other instructions after executing the I/O instruction. However, an interrupt is raised when a data byte is to be transferred between the I/O device and memory, and the CPU executes an interrupt servicing routine, which performs transfer of the byte. This sequence of operations is repeated until all bytes get transferred.
Direct memory access (DMA)-based I/O	Data transfer between the I/O device and memory takes place directly over the bus. The CPU is not involved in data transfer. The DMA controller raises an interrupt when transfer of all bytes is complete.

Direct Memory Access



Interruzioni

- Un evento è una situazione che richiede l'attenzione del SO
- Il progettista del computer associa un *interrupt* ad ogni evento
  - Lo scopo è di riportare l'occorrenza dell'evento al SO permettendogli di eseguire delle azioni per la sua gestione
- L'azione dell'interrupt salva lo stato della CPU e carica i nuovi contenuti nei PSW e GPR
  - La CPU inizia ad eseguire istruzioni di una routine di servizio interruzioni (*interrupt servicing routine - ISR*) nel kernel
- Ad ogni interrupt è associata una priorità
  - Se occorrono più interrupt nel medesimo istante, la CPU servirà l'interrupt con priorità maggiore
  - Gli altri interrupt restano *pendenti* fino al momento in cui sono selezionati per essere serviti

Interruzioni (cont.)

Class	Description
I/O interrupt	Caused by conditions like I/O completion and malfunctioning of I/O devices.
Timer interrupt	Raised at fixed intervals or when a specified interval of time elapses.
Program interrupt	(1) Caused by exceptional conditions that arise during the execution of an instruction, e.g., arithmetic exceptions like overflow, addressing exceptions, and memory protection violations. (2) Caused by execution of a special instruction called the *software interrupt instruction*, whose sole purpose is to cause an interrupt.

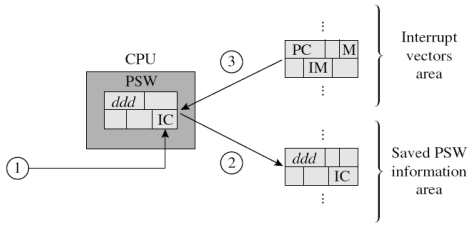
Codice Interruzione

- Quando si verifica un interrupt di un tipo, lo HW imposta un codice d'interrupt nel campo IC del PSW
  - Indica che si è verificato un interrupt di quel tipo
  - Utile per conoscere la causa dell'interrupt
- I codici dipendono dall'architettura
  - Interrupt di I/O: codice -> indirizzo periferica
  - L'istruzione per generare un interrupt SW (SI) ha un operando intero usato come codice dell'interrupt

Mascheramento Interrupt

- Il campo IM (interrupt mask) del PSW indica a quali interrupt è consentito di verificarsi in quel momento
- IM può contenere un intero m per cui solo interrupt con priorità maggiore di m possono verificarsi
  - Oppure, contiene una stringa di bit, dove ogni bit indica se uno specifico tipo di interrupt è abilitato o meno a verificarsi
  - Un interrupt non abilitato è detto mascherato o disabilitato
  - Il verificarsi di un interrupt disabilitato non viene perso
    - Pendente fino a che non è abilitato e rilevato

Azione di un'Interruzione



Step	Description
1. Set interrupt code	The interrupt hardware forms a code describing the cause of the interrupt. This code is stored in the *interrupt code* (IC) field of the PSW.
2. Save the PSW	The PSW is copied into the *saved PSW information* area. In some computers, this action also saves the general-purpose registers.
3. Load interrupt vector	The interrupt vector corresponding to the interrupt class is accessed. Information from the interrupt vector is loaded into the corresponding fields of the PSW. This action switches the CPU to the appropriate interrupt servicing routine of the kernel.

Interazione del SO con il Computer e i Programmi Utente

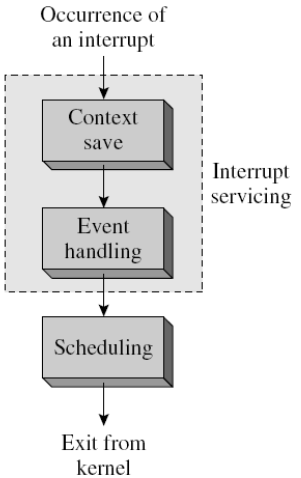
- Il SO interagisce con il computer per
  - Conoscere informazioni sugli eventi, in modo da poterli servire
  - Ripristinare lo stato della CPU per riprendere l'esecuzione di un programma dopo aver servito un interrupt
- I programmi hanno bisogno di usare i servizi del SO per scopi quali iniziare un'operazione di I/O
  - Il metodo che causa un interrupt e passa la richiesta al SO è noto come *Chiamata di sistema* (***System call***)
- Discuteremo:
  - Controllare l'esecuzione dei programmi
  - Servire un interrupt
  - Chiamate di sistema

Controllare l'esecuzione dei programmi

1. Quando inizia un programma utente, il PSW dovrebbe contenere
   a. Il campo *Program counter* (PC)
   b. Il campo *Mode* (M) impostato a modalità utente (1)
   c. Il campo *Informazione di protezione della memoria* (MPI) con l'indirizzo di partenza in memoria e la dimensione del programma
   d. Il campo Maschera interrupt (IM) impostato con tutti gli interrupt abilitati
2. Quando un programma è interrotto, lo stato della CPU (PSW e GPR) viene salvato
   - *Tabella del programma* o *Process Control Block* (***PCB***)
3. Quando il programma è riavviato, è ripristinato il suo stato della CPU

Servizio dell'interruzione

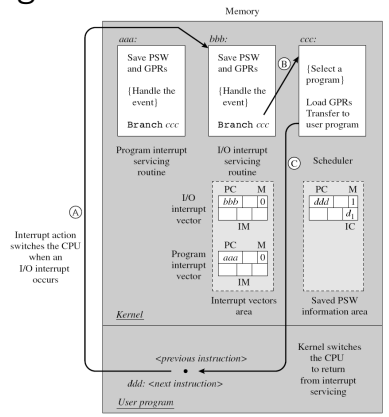
- Il kernel costruisce il vettore degli interrupt, per ogni classe di interrupt, in fase di boot
  - Per semplicità consideriamo lo stesso formato del PSW
    - PC
    - Modalità
    - MPI (0,dim memoria)
    - IM
- Il Context save salva lo stato della CPU del programma
- Lo scheduler seleziona un programma per l'esecuzione



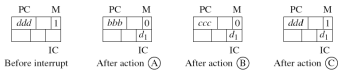
Servizio dell'interruzione (cont.)

Interrupt	Event handling action
Arithmetic exception	Abort the program.
Memory protection violation	Abort the program.
Software interrupt	Satisfy the program's request if possible; otherwise, note it for future action.
End of I/O operation	Find which program had initiated the I/O operation and note that it can now be considered for scheduling on the CPU. Initiate a pending I/O operation, if any, on the device.
Timer interrupt	(1) Update the time of the day. (2) Take appropriate action if a specified time interval has elapsed.

Servizio di interrupt di I/O e ritorno allo stesso programma utente

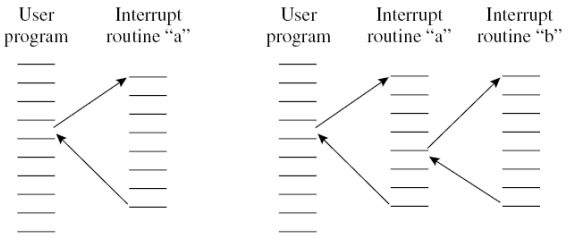


(a) Interrupt vectors and interrupt servicing routines.



(b) PSW contents at various times

Interrupt Annidati

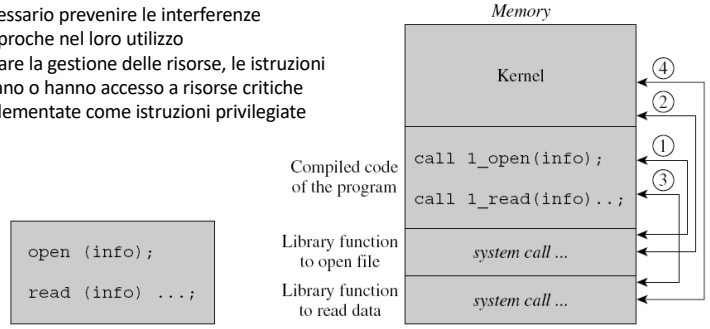


(a) (b)

- Due approcci per il servizio di due interrupt annidati:
  - Disabilitare gli interrupt annidati mediante il mascheramento
  - Gestire gli interrupt annidati – kernel prelazionabile

Chiamate di Sistema

- Un programma usa le risorse del computer come le periferiche di I/O. Tuttavia, le risorse sono condivise tra i programmi utente
  - necessario prevenire le interferenze reciproche nel loro utilizzo
- Per facilitare la gestione delle risorse, le istruzioni che allocano o hanno accesso a risorse critiche sono implementate come istruzioni privilegiate



(a) (b)

Definizione di System call: una richiesta che un programma fa al kernel attraverso un interrupt software

Esempio: una chiamata di sistema in un ipotetico SO

- La CPU fornisce l'istruzione SI per generare un interrupt software
- Il SO fornisce una system call per ottenere l'ora corrente
  - Il codice è 78: l'istruzione SI 78 causa un interrupt SW
- 78 è immesso nel campo IC del PSW prima che sia salvato
- Il vettore degli interrupt per il programma contiene *aaa* in PC
  - La CPU è commutata alla routine con indirizzo di partenza *aaa*
  - Trova che IC è 78 e determina che il programma ha bisogna dell'ora
    - L'orario è restituito in una locazione standard al programma, tipicamente un registro dati

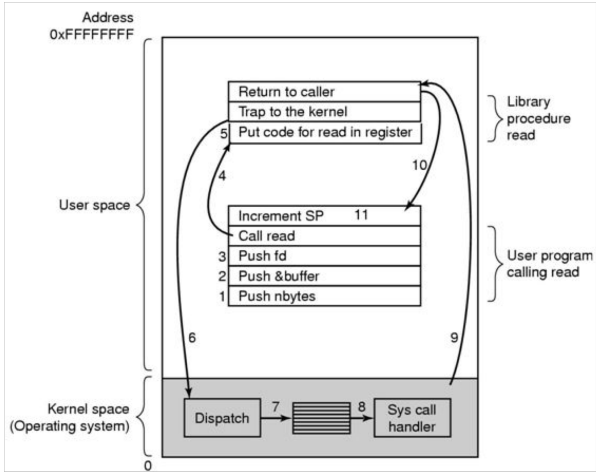
Chiamate di sistema (cont.)

- Alcune system call di **Linux**

Call number	Call name	Description
1	exit	Terminate execution of this program
3	read	Read data from a file
4	write	Write data into a file
5	open	Open a file
6	close	Close a file
7	waitpid	Wait for a program's execution to terminate
11	execve	Execute a program
12	chdir	Change working directory
14	chmod	Change file permissions
39	mkdir	Make a new directory
74	sethostname	Set hostname of the computer system
78	gettimeofday	Get time of day
79	settimeofday	Set time of day

Esempio: Esecuzione read

```
read (fd, &buff, nbytes)
```



Ricapitolando

- Un interrupt è un segnale speciale inviato alla CPU per indicare l'occorrenza di un evento
  - Trasferisce il controllo al SO
  - Una system call è un interrupt SW
    - Un programma la usa per richiedere servizi al SO
- I registri di controllo della CPU governano il loro funzionamento
  - Il Program status word (PSW) è una collezione di tali registri
- Il kernel del SO salva lo stato della CPU quando si verifica l'interrupt
  - Cioè, PSW e GPR
- La CPU ha due modalità operative controllate dal campo mode (M) del PSW
  - Modalità utente e modalità kernel

Ricapitolando (cont.)

- La gerarchia di memoria fornisce lo stesso effetto di una memoria veloce e ampia, a basso costo
  - Contiene:
    - Una memoria cache molto veloce e piccola
    - Una RAM più grande e lenta
    - Un disco
  - Il tempo di accesso effettivo dipende dal hit ratio della cache
  - Il sistema di I/O usa l'accesso diretto alla memoria (direct memory access – DMA) per consentire alla CPU ed al sistema di I/O di funzionare in modo indipendente