07. Gestione dell'Input_Output

Introduzione e Moduli I/O

I sistemi di elaborazione devono gestire una grande varietà di periferiche che differiscono per quantità di dati trattati, velocità operative e formati utilizzati. Tutte le periferiche sono significativamente più lente della CPU e della RAM, rendendo necessaria l'introduzione di moduli di I/O che fungono da interfaccia tra CPU/memoria e i dispositivi esterni.

I dispositivi esterni si classificano in:

- **Comprensibili dall'uomo**: video, stampante, tastiera
- **Comprensibili dalla macchina**: monitoraggio e controllo
- **Comunicazione**: modem, schede di rete (NIC)

Ogni dispositivo esterno presenta una struttura interna composta da logica di controllo, buffer e trasduttore per la conversione dei segnali.

Funzioni e Struttura del Modulo I/O

Le funzioni principali del modulo I/O sono:

- Controllo e temporizzazione
- Comunicazione con la CPU
- Comunicazione con i dispositivi
- Buffering dei dati
- Rilevazione degli errori

Il flusso operativo semplificato prevede che la CPU interroghi il modulo sullo stato del dispositivo, il modulo restituisca tale stato, e se il dispositivo è pronto la CPU richieda il trasferimento dati attraverso il modulo.

Il modulo I/O può nascondere o rivelare le proprietà del dispositivo alla CPU, supportare dispositivi singoli o multipli, e controllare le funzioni del dispositivo oppure delegare il controllo alla CPU. Unix, ad esempio, tratta tutti i dispositivi come file.

Tecniche di Gestione I/O

Esistono tre tecniche fondamentali:

- **I/O da programma** (Programmed I/O)
- **I/O guidato da interrupt** (Interrupt Driven I/O)
- **Accesso Diretto alla Memoria** (DMA)

I/O da Programma

La CPU mantiene il controllo diretto sull'I/O gestendo stato del dispositivo, comandi di lettura/scrittura e trasferimento dati. La CPU attende il completamento dell'operazione da parte del modulo I/O, con conseguente spreco di tempo di elaborazione.

Caratteristiche operative:

- Il modulo I/O setta bit di stato
- La CPU controlla periodicamente tali bit
- Il modulo non informa direttamente la CPU né la interrompe

- La CPU può attendere o svolgere altre operazioni e controllare successivamente

I comandi I/O prevedono l'invio di un indirizzo identificativo del modulo e del dispositivo, seguito da:

- **Comandi di controllo**: configurazione del dispositivo (es. velocità disco)
- **Comandi di test**: verifica dello stato (alimentazione, errori)
- **Comandi di lettura/scrittura**: trasferimento dati tramite buffer

Nell'I/O da programma ogni dispositivo riceve un identificatore unico utilizzato come indirizzo nei comandi.

I/O Mapping

I/O Memory-Mapped

Dispositivi e memoria condividono lo stesso spazio di indirizzamento; le operazioni I/O appaiono come normali letture/scritture in memoria, senza necessità di comandi speciali.

Vantaggi:

- Utilizzo delle normali istruzioni di accesso alla memoria
- Possibilità di scrivere driver in linguaggi ad alto livello
- Protezione più agevole tramite privilegi

Svantaggi:

- Incompatibilità con la cache (il dato rilevante risiede sempre nella memoria del dispositivo)
- Incompatibilità con architetture a bus multipli (necessità di filtraggio degli indirizzi)

I/O Separato (Isolated)

- Spazi di indirizzamento distinti per I/O e memoria
- Richiede linee di selezione tra I/O e memoria
- Comandi speciali con insieme limitato di istruzioni

I/O Interrupt Driven

Questa tecnica evita l'attesa attiva da parte della CPU ed elimina il controllo ripetuto dello stato del dispositivo. Il modulo I/O interrompe la CPU quando è pronto.

Flusso operativo:

1. La CPU rilascia un comando di lettura
2. Il modulo I/O ottiene i dati dalla periferica mentre la CPU svolge altro lavoro
3. Il modulo interrompe la CPU
4. La CPU richiede i dati al modulo I/O
5. Il modulo trasferisce i dati alla CPU

Dal punto di vista della CPU, dopo aver rilasciato il comando esegue altro lavoro e controlla le interruzioni alla fine di ogni ciclo di istruzione. Se presente un'interruzione:

- Salva il contesto (PC e registri)
- Interrompe il processo corrente
- Gestisce l'interrupt leggendo i dati dal modulo I/O e scrivendoli in memoria

---

Direct Memory Access (DMA)

Sia l'I/O da programma che l'interrupt driven richiedono l'intervento attivo della CPU, limitando il tasso di trasferimento e impegnando la CPU in operazioni non specifiche. Il DMA riduce questo intervento al minimo necessario attraverso un modulo hardware addizionale connesso al bus.

Operazioni DMA

La CPU comunica al controllore DMA:

- Tipo di operazione (lettura/scrittura)
- Indirizzo del dispositivo
- Indirizzo iniziale in memoria del blocco dati
- Quantità di dati da trasferire

La CPU prosegue con altre attività mentre il controllore DMA gestisce il trasferimento dialogando direttamente con la memoria centrale. Al termine, il DMA invia un interrupt alla CPU.

Modalità di trasferimento

Il DMA controller può accedere al canale dati in due modi:

- **Cycle stealing**: una parola alla volta, sottraendo periodicamente il controllo del bus alla CPU. Il controllore prende possesso del bus per un solo ciclo, trasferisce una word senza che la CPU cambi contesto; la CPU rimane sospesa solo nel momento prima dell'accesso al bus.
- **Burst mode**: per blocchi, prendendo possesso del canale per una serie di trasferimenti. Risulta più efficiente poiché l'acquisizione del canale è onerosa.

In entrambi i casi la CPU è bloccata, ma il DMA rallenta meno rispetto alla gestione diretta del trasferimento da parte della CPU.

Configurazioni DMA

Configurazione	Uso del bus per trasferimento	Perdita controllo CPU
Bus singolo, controller isolato	Due volte (I/O→DMA, DMA→memoria)	Due volte
Bus singolo, controller integrato con I/O	Una volta (DMA→memoria)	Una volta
Bus di I/O separato	Una volta (DMA→memoria)	Una volta

La configurazione con controller integrato può gestire più dispositivi. La configurazione con bus separato richiede una sola interfaccia I/O per il DMA.