
Interruzioni

Nuovo Corso di Calcolatori Elettronici

Dipartimento di Informatica e Sistemistica
Università degli Studi di Napoli "Federico II"

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Fonti

- B. Fadini, C. Savy, Fondamenti di Informatica II, Liguori Ed., pagg.: 379 - 389
- Fonti alternative:
 - » "Introduzione all'architettura dei calcolatori" – V.C. Hamacker, Z.G. Vranesic, S.G. Zaky – Mac Graw Hill
 - ◆ Cap.4
 - » "Microcomputer Architecture and Programming" - J. F. Wakerly - John Wiley and Sons, Inc.
 - ◆ Cap. 11
 - » Manuale Motorola

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



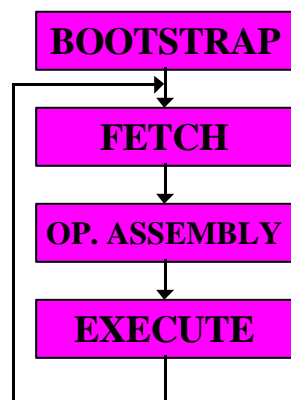
Roadmap

- Le interruzioni
- La fase di interrupt nel ciclo del processore
- Le cause di interruzione
- Le fasi di un'interruzione
- Ripristino del programma
- Latenza
- Identificazione di dispositivi: soluzioni
- Gestione delle priorità

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il ciclo del processore semplificato



- Se il ciclo del processore fosse effettivamente quello mostrato in figura, sorgerebbero alcuni problemi, come per esempio:
 - » un'applicazione "prepotente" potrebbe impadronirsi della risorsa processore senza mai lasciarla;
 - » non ci sarebbe modo di rimuovere forzatamente un'applicazione che entri per errore in un ciclo infinito;
 - » il sistema operativo, in generale, avrebbe un controllo limitato sul sistema.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



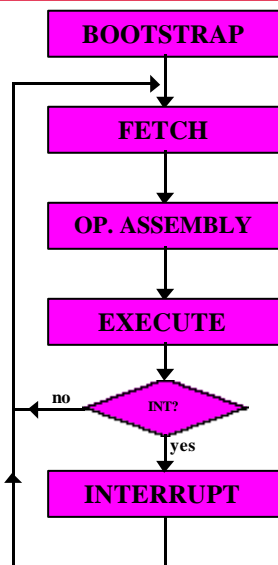
INTERRUPT : esempio

- In un programma che prevede operazioni di I/O, nel momento in cui inizia la comunicazione col dispositivo il programma entra in un ciclo di attesa nel quale controlla ripetutamente lo stato del dispositivo. Durante tale periodo il processore non esegue alcuna operazione utile. In linea di principio il processore può passare a fare " *altre cose*", ma perché ciò sia realizzabile è necessario un meccanismo per far sì che il dispositivo avverta il processore quando è pronto. Questo segnale è detto **interrupt**. Questo esempio mette in mostra come (eliminando il controllo continuo da parte del processore sul dispositivo) il processore mentre attende il compimento delle operazioni di I/O può anche eseguire altre funzioni: l'utilizzo degli interrupt permette di eliminare i periodi di attesa.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il ciclo del processore esteso al meccanismo delle interruzioni.



- La soluzione comunemente adottata consiste nel permettere al "supervisore" di prendere il controllo del processore al termine di ciascun ciclo.
- Questo avviene esclusivamente nel caso in cui si verificano eventi "eccezionali", di solito *asincroni* con l'esecuzione del programma correntemente in corso.
- In assenza di tali eventi l'elaborazione procede nella maniera consueta.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



La fase INTERRUPT (1/2)

INTERRUPT

- La fase di INTERRUPT viene eseguita nel caso in cui il segnale INT è alto.
- Questo evento è sintomatico del fatto che alcuni eventi sono "pendenti" e devono essere "serviti".
- Gli eventi possono essere di natura diversa e possono essere generati da diverse cause.
- L'interruzione rappresenta il "servizio" che provvede a gestire questi eventi: ciascuna delle cause di interruzione richiede al sistema specifiche azioni elaborative (conteggio del tempo, segnalare un errore, rendere disponibile una risorsa, ...).
- Al termine del servizio, il programma interrotto viene ripreso, se tale ripresa è compatibile con la stessa interruzione.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



La fase INTERRUPT (2/2)

INTERRUPT

- Durante questa fase del ciclo del processore, comunque, non viene eseguito un programma. Se INT è alto si innesca il sistema per saltare alla particolare ISR che verrà eseguita (come tutti i programmi) nel normale ciclo del processore.
- Per eseguire un programma (*software*), infatti, sarebbe necessario trovarsi all'interno del ciclo principale e muoversi tra le fasi di *fetch* ed *execute*.
- Ciò che avviene nella fase di *interrupt* consiste invece in una serie di meccanismi *hardware* che "preparano" il processore a gestire l'interruzione.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



“Procedura” di Servizio dell’Interrupt

- Una procedura di servizio dell’interrupt può essere vista come una normale procedura (sottoprogramma). Una differenza importante è che un sottoprogramma esegue una funzione richiesta dal programma chiamante, mentre in generale una procedura di servizio degli interrupt può non avere nulla in comune con il programma che è in esecuzione al momento della ricezione del segnale di interrupt.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Le cause di interruzione

- Una interruzione può essere causata da necessità di natura diversa. Per esempio:
 - » **interruzione periodica** (p.es. ogni 10ms) per permettere al sistema operativo di computare il tempo speso da una applicazione e di cedere eventualmente la risorsa processore ad un’altra applicazione (multiprogrammazione);
 - » **interruzione di I/O**: una periferica di I/O che informa su un suo particolare stato (p.es. pronta a ricevere dati) al fine di sincronizzarsi con il processore;
 - » **interruzione per errori nel programma** correntemente eseguito (p. es. overflow, esecuzione di un’istruzione inesistente o privilegiata, etc.); tali interruzioni vengono anche chiamate *traps*;
 - » **interruzione per guasti** al sistema rivelati da apposite sonde;
 - » **interruzione per riportare l’intero sistema** in uno stato noto (reset);
 - » **interruzioni programmate (software interrupt)** generate da un programma che voglia accedere una risorsa condivisa (es. periferiche di I/O) e per questo chiede la mediazione del sistema operativo. Sono le uniche interruzioni ad essere *sincrone* con il programma correntemente in esecuzione.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Abilitazione delle interruzioni

- Una **causa di interruzione** non provoca di per sé una interruzione ma soltanto una **richiesta di interruzione**.
- Affinché l'interruzione sia attiva è necessario che essa sia abilitata: in questo modo è possibile implementare particolari strategie per le quali si inibiscono alcune tipologie di interruzioni.
- Tale filosofia porta al **Modello fondamentale di sistema delle interruzioni**.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Modello fondamentale di sistema delle interruzioni

- Il modello si basa su due registri speciali ed un flip-flop:
 - » Registro **Richieste di Interruzioni RI**, che memorizza le richieste di interruzioni;
 - » Registro **Maschera delle interruzioni M**, che abilita singolarmente le richieste
 - » **Flip-Flop di Abilitazione generale, AG**, che abilita il sistema nel suo complesso.
- Ciascun evento I (causa di interruzione) posiziona il flip-flop RI_i del registro RI (richieste) ed è abilitato dal corrispondente bit M_i della maschera. Tutto il sistema è poi abilitato da AG. Il segnale INT è pertanto:

$$INT = AG \cdot \sum_i RI_i \cdot M_i$$

- ove la sommatoria è da intendersi in senso booleano.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Fasi del processo delle interruzioni

- Esecuzione normale
 - Servizio dell'interruzione
 - ◆ Salvataggio del contesto (hardware)
 - ◆ Identificazione del device o della causa di interruzione
 - ◆ Salto all'entry point della Interrupt Service Routine (ISR)
 - ◆ Salvataggio del contesto (software)
 - ◆ Servizio dell'interruzione
 - ◆ Ripristino del contesto (software)
 - ◆ Ripristino del contesto (hardware)
- Esecuzione normale

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il ripristino del programma

- Una interruzione potrebbe eseguire un'elaborazione B completamente indipendente da quella A correntemente in corso sul processore, interrompendola.
- La gestione delle interruzioni deve quindi anche provvedere a mettere A in condizioni di continuare successivamente senza "accorgersi" di nulla.
- Sorge la necessità di salvare (prima) e ripristinare (dopo) lo stato del programma che viene di volta in volta interrotto.
- In questo modo A può continuare la sua elaborazione senza risentire in alcun modo del servizio dell'interruzione, a parte il ritardo di tempo.
- Le informazioni che devono essere salvate e ripristinate comprendono di solito il PC, i flag dei codici di condizione e il contenuto di qualsiasi registro che sia usato sia dal programma che dalla routine di gestione dell'interruzione.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il salvataggio dello stato

- L'operazione di salvataggio può essere svolta in parte o completamente in *hardware* o in *software*.
- Una esigenza comune resta comunque quella di salvare lo stato indispensabile poiché il salvataggio richiede trasferimenti di dati, eventualmente da e verso la memoria.
- Data la frequenza con cui le interruzioni vengono prodotte, questo rappresenta quindi un carico aggiuntivo che deve essere ridotto al minimo.
- Il salvataggio dello stato incrementa inoltre il ritardo tra l'istante di ricezione della richiesta di interruzione e l'istante in cui inizia l'esecuzione della routine di interrupt (ISR).
- Questo tempo viene detto **latenza di interrupt**.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il salvataggio dello stato : fase "hardware"

- Nel modello fondamentale si considera ridotto al minimo l'intervento *hardware* (quest'ultimo deve realizzare il salto alla ISR ed il salvataggio del valore corrente di PC per consentire la futura ripresa del programma interrotto):
 - » Detto SAVE il registro in cui viene memorizzato PC e START l'indirizzo d'inizio della ISR (START_i nel caso di interrupt vettorizzati) si ha:
SAVE:=PC;
PC:=START;
AG:=0;
- La disabilitazione delle interruzioni viene effettuata al fine di evitare, che prima ancora che il processo delle interruzioni inizi, possa innestarsi un'ulteriore interruzione.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Il salvataggio dello stato : fase “software”

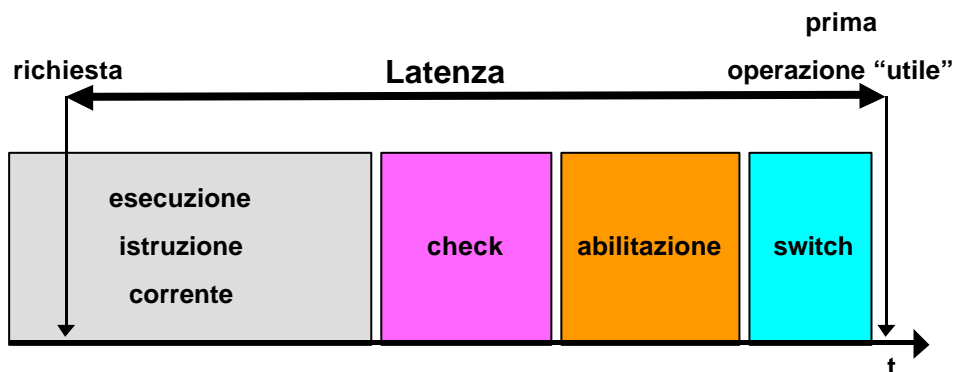
- Le altre informazioni necessarie alla corretta ripresa del programma sospeso vengono salvate dalle istruzioni che si trovano all’inizio della ISR e ripristinate dalle istruzioni che stanno alla fine.
- La parte di “salvataggio/ripristino software” realizzato dalla ISR riguarda:
 - » Salvataggio dei registri per la ripresa del programma interrotto
 - » Individuare la causa di interruzione
 - » Servire l’interruzione
 - » Ripristinare lo stato dei registri
 - » Riprendere il programma interrotto oppure saltare all’esecuzione di un altro programma

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Latenza di un’interruzione

- È il tempo massimo che intercorre tra la richiesta di attenzione e l’effettivo servizio dell’interruzione



DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Nesting delle Interruzioni

- Durante il servizio di una interruzione è possibile (in linea di principio) che divenga attiva una nuova interruzione: viene allora interrotta l'interruzione e servita la nuova causa.
- Tale situazione prende il nome di **Nesting delle Interruzioni** e viene gestita con tecniche LIFO: l'ultimo programma ad essere stato interrotto è il primo ad essere ripreso.
- In questo caso si pone il problema della **priorità**:
 - » Quale tra due cause verificatesi simultaneamente deve essere servita per prima.
 - » Quale causa può interrompere il servizio di un'altra causa.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Identificazione dei dispositivi (1/2)

- Se ci sono più dispositivi, il processore deve essere in grado di identificare il dispositivo che ha generato l'interruzione, poiché probabilmente diverse azioni dovranno essere intraprese a seconda del particolare dispositivo.
- Il dispositivo X può richiedere una interruzione mentre si sta servendo un altro interrupt, oppure più dispositivi possono richiedere interrupt esattamente nello stesso istante.
- I dispositivi hanno una linea comune attraverso la quale segnalano richieste di interruzioni (INT).
- Quando INT è alto si pone comunque il problema di identificare da quale dispositivo è partita la richiesta.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Identificazione dei dispositivi (2/2)

- Esistono diverse soluzioni a questo problema
- Tutte le soluzioni impiegano un misto di hardware e di software
- Tutte le soluzioni dipendono fortemente sia dall'architettura del sistema che da quella del processore

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



La soluzione a registri di stato

- Una possibile soluzione consiste nel dotare ogni dispositivo di un **registro di stato**.
- Quando un dispositivo richiede un'interruzione inizializza un bit nel registro di stato, il bit di richiesta di interrupt (**Interrupt Request, IRQ**).
- La procedura di servizio inizia interrogando tutti i dispositivi in un certo ordine e, non appena trova un bit alto, fa partire la corrispondente routine di interrupt.
- Questa interrogazione ciclica (**polling**) è semplice da realizzare, ma ha lo svantaggio di richiedere un certo tempo per interrogare anche i dispositivi che non hanno invocato alcun servizio.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



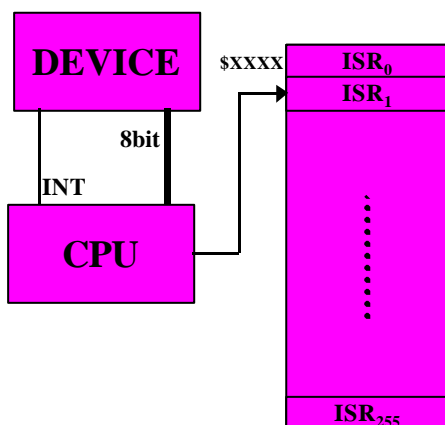
Gli interrupt vettorizzati

- Un approccio alternativo consiste nel prevedere che sia il dispositivo stesso a fornire un proprio identificativo all'atto di una richiesta: è il dispositivo che richiede l'interrupt ad identificarsi.
- L'identificativo serve inoltre a calcolare l'indirizzo della routine di interruzione che deve essere invocata, rendendo il meccanismo molto efficiente.
- Il numero di bit utilizzati pone il limite massimo sul numero di diversi dispositivi che possono essere riconosciuti.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



La soluzione del M68000



- Il processore M68000 utilizza il meccanismo degli **interrupt vettorizzati**.
- In memoria sono presenti 256 locazioni consecutive dette *vettori di interrupt*.
- Ciascuna di queste locazioni contiene l'indirizzo di una ISR.
- Quando un dispositivo richiede un'interrupt, invia al processore un numero di 8 bit che rappresenta il *vettore di interrupt* da utilizzare.

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Interrupt Annidati: priorità

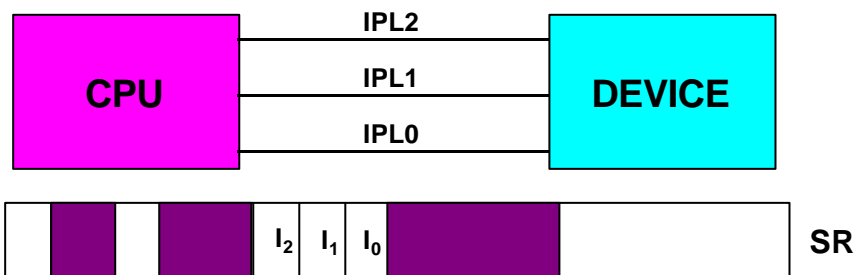
- Una richiesta di interrupt proveniente da un dispositivo ad alta priorità deve essere accettata anche mentre il processore sta servendo un'altra richiesta inviata da un dispositivo a bassa priorità (es. clock).
- Durante l'esecuzione di una ISR vengono accettate le richieste di interrupt solo da dispositivi a priorità più alta rispetto a quella corrente.
- La priorità corrente è mantenuta nel processore: il livello di priorità del processore rappresenta la priorità del programma che è in fase di esecuzione.
- Quando il processore accetta richieste a priorità più elevata imposta la sua priorità al valore di priorità della richiesta appena accettata.
- Nel 68000 il valore corrente di priorità è registrato nei bit priorità dell'SR

DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Gestione delle priorità

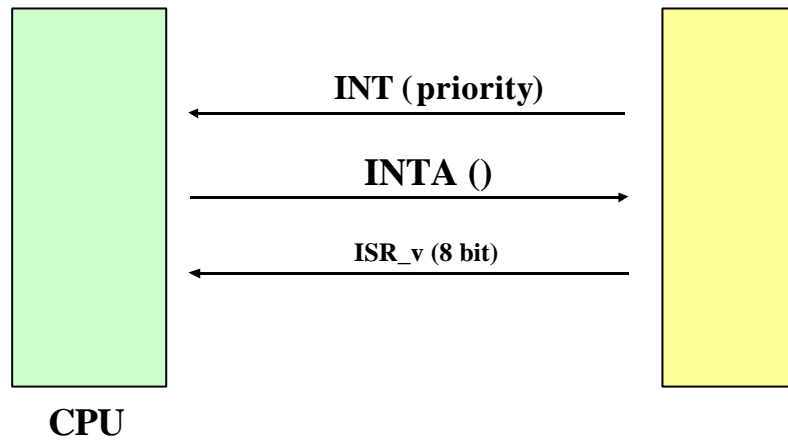
- Problemi:
 - » Mascheramento
 - » Abilitazione
- Soluzione del 68K:
 - » Interrupt Priority Level
 - » Processor Priority Level
 - » Le interruzioni a priorità 7 non sono mascherabili (nonmaskable interrupt)



DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli



Interrupt 68000



DIS - Dipartimento di Informatica e Sistemistica - Università di Napoli

