Calcolatori Elettronici
Sistema di I/O

Ing. Gestionale e delle Telecomunicazioni
A.A. 2009/10

Gabriele Cecchetti - Anna Lina Ruscelli

Sistema di Ingresso e Uscita

- Sommario:
  - Generalità
  - Interfaccia di I/O
  - Controllo di programma
  - Interruzione
  - Gestione di Interruzione nei sistemi Pentium
  - Accesso diretto alla memoria (DMA)
- Riferimenti
  - C. Hamacher, “Introduzione all’architettura del Calcolatore”, cap. 8.

Alcune nozioni fondamentali
Interfaccia di periferica
Gestione di I/O a controllo di programma

GENERALITÀ

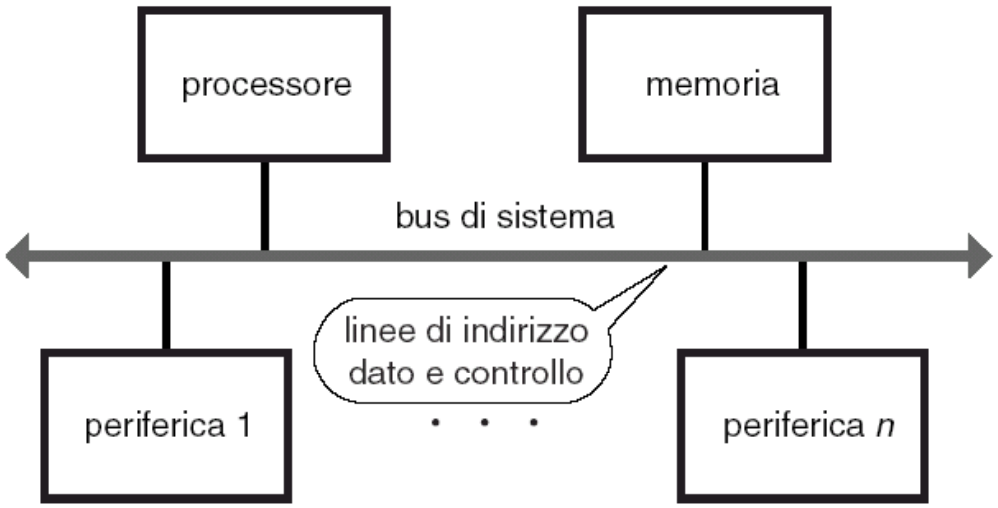
Operazioni di I/O

- Le unità di periferia (**periferiche**), o dispositivi di I/O, servono al calcolatore per scambiare dati con l'ambiente.
- La periferica può essere:
  - **letta**: il processore ottiene da essa un dato
  - **scritta**: il processore le invia un dato
- Le operazioni di I/O non sono sostanzialmente diverse da quelle di memoria (lettura e scrittura di parola).
- Si possono scambiare singoli byte (o parole), o blocchi di byte (o di parole), con la periferica.
- L'operazione di scambio di un blocco di byte viene scomposta in una sequenza di operazioni di byte.

Struttura del Bus

- Le unità periferiche del calcolatore sono collegate al processore tramite il bus.
- Il **bus** è un fascio di collegamenti (fili) e contiene tre gruppi di linee:
  - **di indirizzo**
  - **di dato**
  - e **di controllo** (linee in numero vario)
- Se il bus è unico, viene spesso chiamato ***bus di sistema*** (*system bus*).

Schema di Collegamento



Periferiche collegate al bus di sistema

Funzionamento del Bus

- In generale, l'operazione di I/O tra processore e periferica (cioè interfaccia) si svolge nel modo seguente:
  - il processore emette l'indirizzo della periferica da leggere o scrivere tramite il bus indirizzi, e il comando (di lettura o scrittura) tramite il bus di controllo; tutte le periferiche vedono i contenuti di entrambi i bus,
  - le periferica interessata rileva l'indirizzo come proprio, si attiva e interpreta il comando, quindi:
    - se l'operazione è di **lettura**, invia il dato al processore tramite il bus dei dati
    - se l'operazione è di **scrittura**, riceve il dato dal processore tramite il bus dei dati (si suppone che il processore invii il dato dopo l'indirizzo).
- Ci possono essere altri segnali di controllo scambiati per sincronizzare processore e periferica.

Interfaccia di Ingresso-Uscita

- Più precisamente, la periferica **non** è collegata direttamente al bus, bensì all'**interfaccia** (o porta) **di ingresso-uscita** (o di I/O), la quale a sua volta è collegata direttamente al bus del calcolatore.
- Ogni interfaccia è associata ad un indirizzo (o a un intervallo di indirizzi consecutivi), che la identificano univocamente tra tutte le altre interfacce.
- L'interfaccia risponde all'indirizzo inviato dal processore tramite le linee di indirizzo e scambia dati con il processore tramite le linee di dato.
- Il bus di controllo serve per scambiare comandi e risposte (tra interfaccia e processore), e segnali di sincronizzazione di varia natura.

Schema di Indirizzamento (1/2)

- Gli indirizzi di interfaccia di I/O possono essere assegnati in due modi differenti:
  - Spazio di indirizzamento di I/O **unificato** con quello di memoria:
    - alcuni indirizzi di memoria sono assegnati alle interfacce (e non sono più usabili come memoria)
  - Spazio di indirizzamento di I/O **separato** da quello di memoria:
    - gli indirizzi di I/O sono indipendenti da quelli di memoria e in aggiunta a questi
- La maggior parte dei calcolatori attuali ha spazi di indirizzamento di memoria e di I/O unificati.

Schema di Indirizzamento (2/2)

- Se gli spazi di indirizzamento di I/O e memoria sono unificati, le istruzioni macchina che lavorano in memoria (caricamento e memorizzazione) possono **anche** leggere e scrivere da e verso le interfacce di I/O:
  - gestione **unificata** di memoria e I/O
- Altrimenti il processore deve disporre di istruzioni macchina **specifiche** per le operazioni di I/O (istruzioni IN e OUT):
  - gestione **separata** di memoria e I/O

Schema di Indirizzamento e SO

- Se il calcolatore è dotato di sistema operativo con meccanismo di memoria virtuale e paginazione, il SO stesso può gestire una o più pagine come se fossero assegnate a periferiche.
- In tale modo lo spazio di memoria virtuale risulta suddiviso naturalmente in indirizzi di memoria vera e propria e in indirizzi di periferica.
- Naturalmente il sistema deve contenere un'unità di gestione della memoria (MMU) capace di realizzare tale funzionalità di virtualizzazione.

Struttura dell'interfaccia
Componenti interni
Esempio di interfaccia di terminale
Tecniche di ingresso-uscita

INTERFACCIA DI I/O

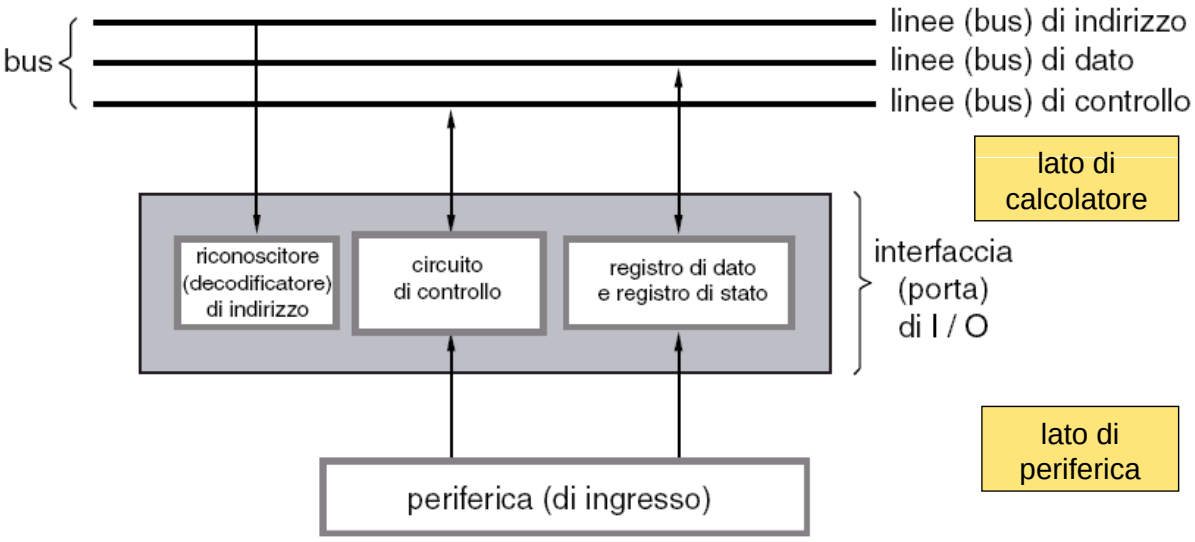
Struttura dell'Interfaccia di I/O (1/2)

- L'interfaccia di I/O ha struttura varia, ma si possono distinguere quattro componenti fondamentali:
  - **Riconoscitore di indirizzo**:
    - esamina l'indirizzo corrente sulle linee di indirizzo e segnala se sia quello (o uno di quelli) assegnato all'interfaccia, nel quale caso l'interfaccia si attiva
    - è specifico per l'interfaccia, o quanto meno programmabile
  - **Registro di stato**:
    - contiene alcuni bit che indicano lo stato di funzionamento della periferica:
      - dato pronto per il processore, attesa di nuovo dato da parte del processore, errore, ecc
    - serve per scambiare tra interfaccia e processore varie informazioni di stato e controllo

Struttura dell'Interfaccia di I/O (2/2)

- **Registro di dato**:
  - serve per scambiare il dato con il processore:
    - operazione di lettura: la periferica deposita il dato nel registro, da dove il processore lo va a leggere
    - operazione di scrittura: il processore deposita il dato nel registro, da dove la periferica lo va ad acquisire
  - se l'interfaccia ha più indirizzi assegnati, può avere altrettanti registri di dato e di stato
- **Circuito di controllo**:
  - serve per controllare, coordinare e sincronizzare il funzionamento dell'interfaccia
  - a seconda dell'interfaccia può essere semplice o molto complesso e ricco di funzionalità

Schema di Interfaccia di I/O



Componenti fondamentali dell'interfaccia di I/O (qui sola lettura)

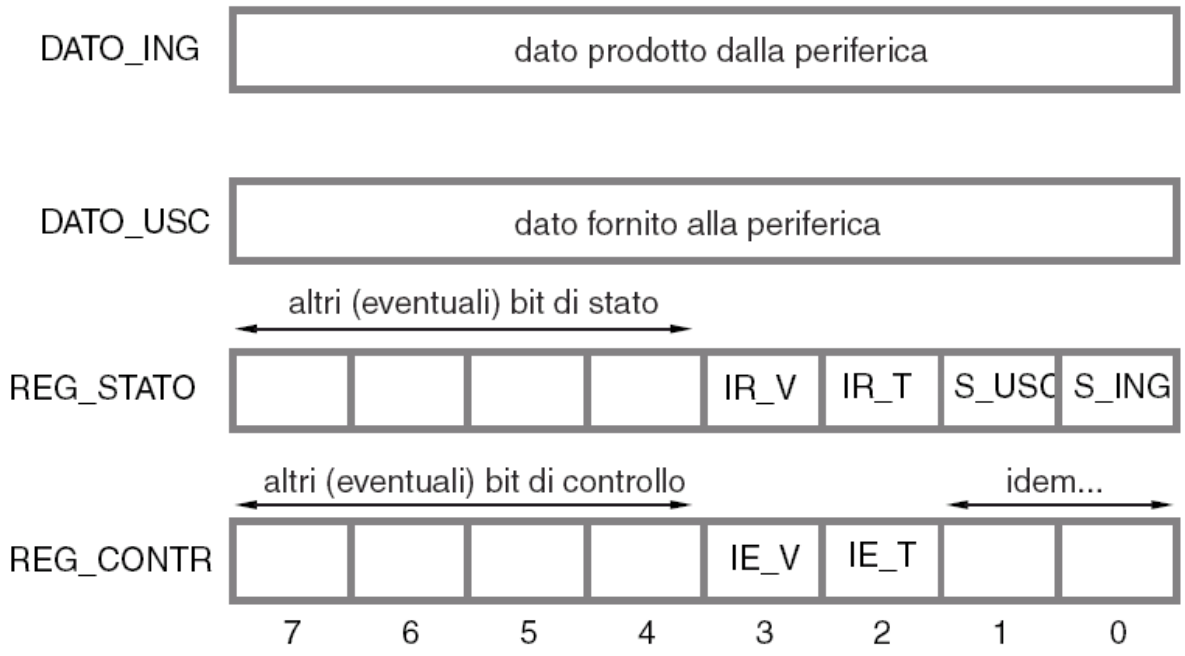
Esempio – Porta del Terminale

- Il terminale è l'insieme di:
  - tastiera: dispositivo di ingresso di caratteri
  - schermo (monitor): dispositivo di uscita
    - a carattere (finestra di terminale)
    - grafico (più o meno sofisticato)
- Qui si considera il terminale con uscita a caratteri, la versione più semplice.
- Il terminale ha un'**interfaccia di lettura e scrittura**:
  - la tastiera invia un carattere (codice ASCII del tasto premuto) al processore
  - il processore invia un carattere (codice ASCII) allo schermo, da visualizzare nella posizione corrente

Registri di Dato e di Stato

- La porta ha **quattro** indirizzi assegnati:
  - **DATO_ING**: indirizzo del registro di dato dove la tastiera mette il codice ASCII del tasto premuto, per il processore
  - **DATO_USC**: indirizzo del registro di dato dove il processore mette il codice ASCII del carattere da visualizzare, per lo schermo
  - **REG_STATO**: indirizzo del registro di stato tramite cui processore e terminale si scambiano informazioni di funzionamento (dato pronto o no, e simili)
  - **REG_CONTR**: indirizzo del registro di stato contenente i bit di controllo del meccanismo di interruzione (vedi di seguito)
- I registri di dato nell'esempio sono da 8 bit ciascuno (un byte).
- I registri di stato e di controllo sono da 8 bit, ma solo alcuni bit di essi sono realmente utilizzati.
- I registri potrebbero essere anche da 16, 32, ecc, bit.

Terminale – Registri Dato, Stato, Controllo



Terminale – Bit di Stato

- I bit del registro di stato REG_STATO hanno i significati seguenti:
  - **S_ING o stato di ingresso**:
    - va automaticamente a 1 non appena la tastiera ha depositato nel registro DATO_ING il codice ASCII del tasto premuto
    - va automaticamente a 0 non appena il processore ha letto il contenuto del registro DATO_ING
  - **S_USC o stato di uscita**:
    - va automaticamente a 1 non appena il processore ha scritto nel registro DATO_USC il carattere da visualizzare
    - va automaticamente a 0 non appena lo schermo ha acquisito il contenuto del registro DATO_USC
- Il circuito di controllo della porta aggiorna in modo automatico i due bit S_ING e S_USC.
- I bit di stato IR_V e IR_T servono per l'interruzione.

Bit di Controllo di Interruzione

- Alcuni bit (nel registro di stato e di controllo) servono per gestire ordinatamente il meccanismo di interruzione:
  - **IR_T** (*interrupt request di tastiera*), va automaticamente a 1 quando la tastiera chiede interruzione (a seguito di pressione di un tasto), e in seguito viene riportato a 0 dal processore
  - **IR_V**, (*interrupt request di video)* idem, ma vale per lo schermo
  - **IE_T** (*interrupt enable di tastiera*), se il processore lo pone a 1 la tastiera ha il permesso di chiedere interruzione, se lo pone a 0 non ha il permesso (serve per ammettere o inibire l'interruzione)
  - **IE_V** (*interrupt enable di video)*, idem, ma vale per lo schermo
- Questi bit sono usati quando il terminale è gestito in modalità di interruzione (vedi di seguito).

Tecniche di Gestione di I/O

- **Controllo di programma**:
  - l'operazione di lettura o scrittura di periferica è interamente gestita dal processore
  - è una tecnica puramente SW
- **Interruzione** (o interrupt):
  - parte dell'operazione è assegnata direttamente all'interfaccia
  - è una tecnica mista HW/SW
- **Accesso diretto alla memoria** (*Direct Memory Access* o DMA):
  - quasi tutta l'operazione è svolta in HW (tranne la fase iniziale)
  - necessita di un apposito controllore di DMA, un dispositivo HW specializzato (integrato nell'interfaccia o nel processore o separato)

Concetto di controllo di programma

Ciclo di controllo

CONTROLLO DI PROGRAMMA

Controllo di Programma

- Il programma esamina (si dice “monitora”) periodicamente o con continuità l’interfaccia, controllandone i bit di stato.
- Se le condizioni indicate dai bit di stato sono opportune, legge o scrive i registri di dato, come necessario.
- Eventualmente aggiorna i bit di stato (e di controllo), in base all’esito dell’operazione.
- Il programma è ciclico ed ha il compito di gestire la periferica.

Esempio - Controllo del Terminale

- Il programma esamina ciclicamente (monitora) lo stato della tastiera.
- Se questa ha pronto un codice di tasto, il programma:
  - lo legge e deposita in un buffer di memoria apposito
  - lo scrive nel registro di dato dello schermo affinché venga visualizzato
  - verifica se il carattere sia di fine linea:
    - se sì termina invocando una routine di elaborazione della linea
    - altrimenti, torna a monitorare la tastiera …
- Con qualche dettaglio aggiunto, vedi la codifica.

Vantaggi e Svantaggi del Controllo di Programma

- **Vantaggi** :
  - semplice da realizzare e collaudare
  - economico (solo SW, niente HW aggiuntivo)
- **Svantaggi**
  - **impegna costantemente** o quasi **il processore**
  - non concede tempo ad altre attività
  - presuppone che lo stato della periferica sia sempre noto e ben rappresentato nella porta e che non muti tanto rapidamente da non essere monitorabile periodicamente o con continuità
  - per ogni dato scambiato esegue diverse istruzioni macchina e pertanto è un metodo di gestione relativamente lento
- È adatto a periferiche semplici da gestire, che abbiano pochi dati da scambiate e caratterizzate da comportamento prevedibile e regolare.

Concetto di interruzione
Richiesta – identificazione - salto a routine di servizio - priorità

INTERRUZIONE

Interruzione (1/2)

- La tecnica di interruzione prevede che:
  - la periferica possa richiedere **autonomamente** un servizio (di ingresso-uscita) al processore
  - il processore reagisca sospendendo l'attività corrente e passi a eseguire il servizio richiesto dalla periferica
  - al termine del servizio, il processore riprenda la sua attività precedentemente interrotta
- La periferica (o, meglio, la sua interfaccia) richiede il servizio mandando al processore un segnale di interruzione (**interrupt request**).

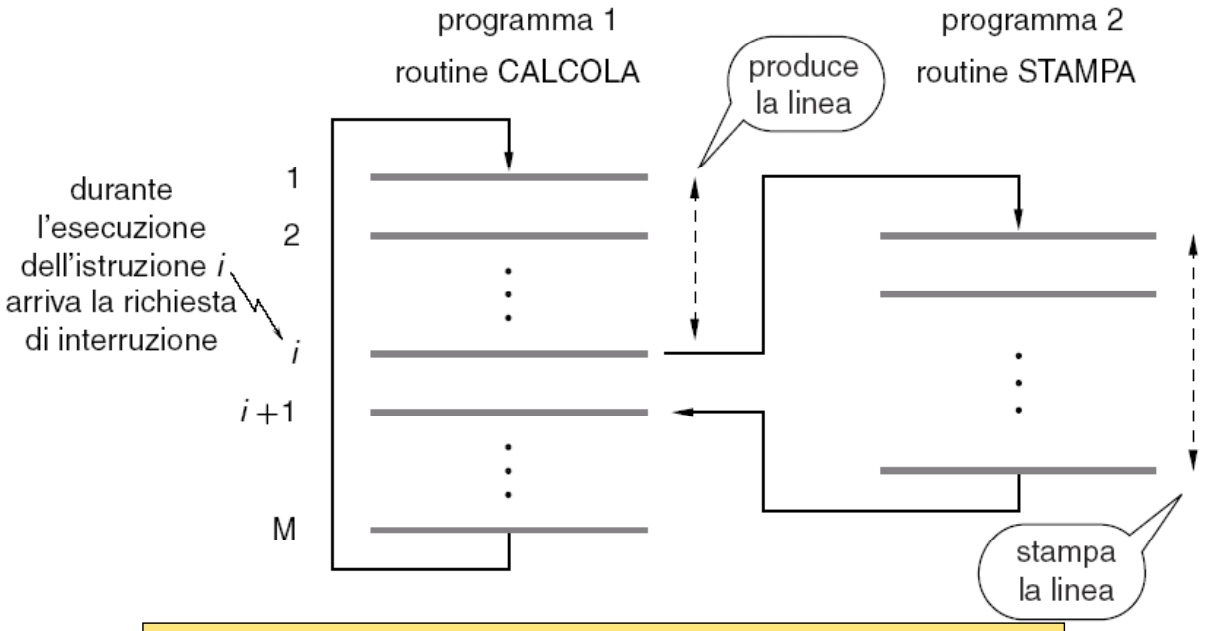
Interruzione (2/2)

- Il servizio richiesto dalla periferica tramite interruzione si chiama **routine di servizio di interruzione** (Interrupt Service Routine, ISR).
- La routine di servizio di interruzione e' simile ad una routine ordinaria (procedura o funzione), ma il meccanismo di attivazione è diverso.
- La routine di servizio può essere chiamata ed attivata in un punto qualunque del programma corrente, in modo imprevedibile e senza preavviso.
- Invece la routine ordinaria viene chiamata e attivata da istruzioni macchina apposite (chiamata a routine o call), collocate in punti definiti del programma e noti a priori.

Esempio - Stampa

- Il programma ciclico CALCOLA (esso stesso una routine, di tipo ordinario) produce una serie di linee di testo (caratteri) da stampare.
- Ogniqualvolta la stampante ha finito di emettere la linea corrente, manda al processore una richiesta di interruzione per avere nuovi dati.
- La richiesta chiama e attiva la routine di servizio di interruzione STAMPA, la quale trasferisce la nuova riga di caratteri alla stampante e poi termina.
- Terminata la routine STAMPA, la routine CALCOLA riprende da dove si era sospesa.

Interruzione e Servizio



Relazione tra programma corrente e routine di servizio

Particolarità

- L'arrivo del segnale di richiesta di interruzione è imprevedibile:
  - l'istruzione macchina raggiunta dal segnale di richiesta viene comunque portata a termine
  - subito dopo l'esecuzione viene passata alla routine di servizio, che svolge le sue attività di comunicazione con la periferica
  - quando la routine di servizio termina, il programma interrotto riprende dall'istruzione macchina consecutiva a quella che era stata interessata dall'interruzione (già portata a termine prima)
- Nota bene: l'interruzione vera e propria è il segnale di richiesta; il trasferimento di caratteri alla stampante avviene tramite una porta di ingresso-uscita (nell'esempio di sola uscita) del tutto ordinaria.

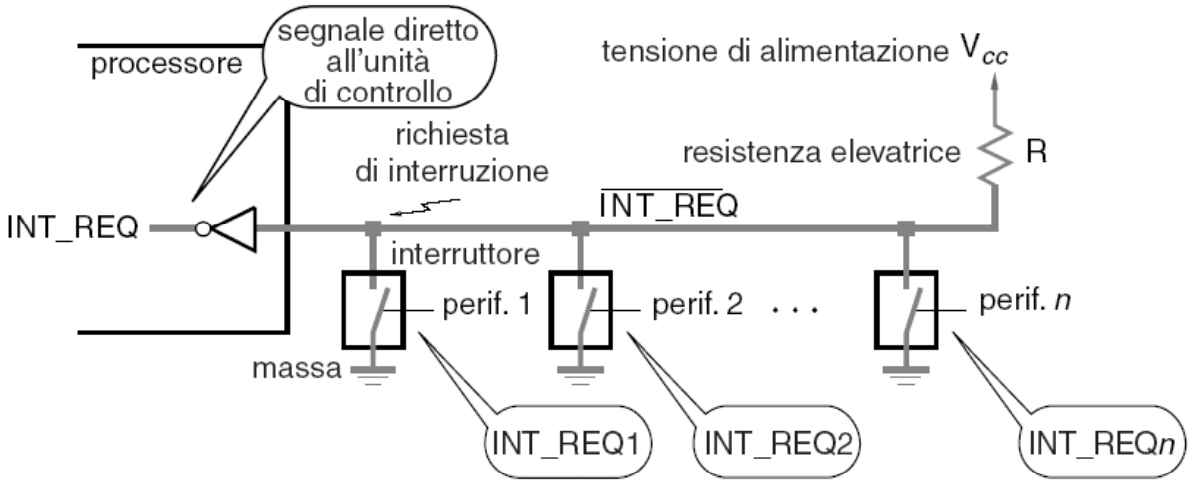
Rientro da Interruzione

- Per gestire il meccanismo di rientro al programma corrente interrotto, si usa la pila (***stack***), come nel caso di routine ordinaria.
- Quando l'esecuzione passa alla routine di servizio di interruzione, l'indirizzo di rientro al programma corrente (nell'esempio l'indirizzo $i + 1$), viene memorizzato nello stack.
- La routine di servizio termina con un'istruzione macchina apposita, chiamata **RETI** (return from interrupt), la quale recupera l'indirizzo della istruzione da eseguire dallo stack e lo carica nel contatore di programma.

Segnale di Interruzione

- Il bus di controllo del calcolatore contiene un segnale di richiesta di interruzione:
  - interrupt request (**INT_REQ**)
- Tutte le periferiche sono collegate alla linea di INT_REQ e possono attivarla (per esempio portandola a 0) per richiedere il servizio.
- Tale meccanismo di segnalazione funziona correttamente se c'è una sola richiesta per volta.
- Di seguito si mostrerà come rilassare tale vincolo, di per sé troppo restrittivo.

Segnale di Interruzione



Meccanismo di richiesta di interruzione (versione semplice)

Problematiche del meccanismo di Interruzione

- In generale al calcolatore sono collegate due o più periferiche, e tutte o molte potrebbero servirsi del meccanismo di interruzione.
- Inoltre una stessa periferica potrebbe avere bisogno di due o più routine di servizio, con compiti differenziati a seconda della richiesta.
- Infine le periferiche sono indipendenti, in linea di principio non sincronizzate e potrebbero richiedere interruzione **simultaneamente** o a distanza di tempo molto ravvicinata.

Abilitazione e Disabilitazione

- Si può **abilitare e disabilitare** il meccanismo di interruzione, agendo dal lato del processore oppure della periferica.
- **Lato processore**:
  - un bit di stato (nel registro di stato SR) permette di rendere il processore sensibile o insensibile alla richiesta (se è in stato di insensibilità, il processore la ignora)
  - talvolta ci sono istruzioni macchina apposite per modificare tale bit: DI ed EI (disable / enable interrupt), o con nomi simili
- **Lato periferica**:
  - si può permettere o vietare alla periferica l'uso del meccanismo di interruzione, mediante bit appositi situati nel registro di controllo dell'interfaccia di I/O della periferica stessa
  - per esempio, vedi i bit IE_T e IE_V dell'interfaccia di terminale, nel registro di controllo REG_CONTR (esempio precedente)
  - tali bit vanno modificati da parte del processore, con istruzioni macchina di ingresso-uscita scrivendo nel registro di controllo

Annidamento di Servizio

- Usando la pila come struttura dati per gestire gli indirizzi di rientro da routine di servizio, si possono avere servizi annidati.
- Spetta al programmatore decidere se una routine di servizio vada eseguita con interruzione disabilitata (dal lato di processore) oppure abilitata.
- Se la routine di servizio non deve essere interrompibile, basta disabilitare il meccanismo di interruzione all'inizio della routine (con un'istruzione DI) e riabilitarlo subito prima del rientro (con un'istruzione EI).
- Spesso, il meccanismo di interruzione viene disabilitato in modo automatico non appena parte la routine (ciò è attinente soprattutto al rapporto con il sistema operativo).

Identificazione della Periferica (1/2)

- Di solito il calcolatore deve erogare uno tra diversi servizi di interruzione possibili, secondo quale sia la periferica che ha richiesto l'interruzione e che va servita.
- Ciascun servizio può essere svolto da una routine di servizio differente.
- Oppure più servizi possono essere raggruppati in una sola routine, che ne esegue in modo selettivo uno solo a ogni attivazione.

Identificazione della Periferica (2/2)

- Per decidere quale servizio svolgere a seguito della richiesta, ci sono tre metodi fondamentali:
  - **una sola linea INT_REQ** e salto a indirizzo di memoria fisso di routine di servizio (la quale è unica)
  - **due o più linee INT_REQ distinte**, ciascuna associata a una specifica routine di servizio collocata ad un indirizzo di memoria specifico
  - una sola linea INT_REQ e meccanismo di interruzione vettorizzata (***vectored interrupt***)
- Tali metodi non sono sempre esclusivi e non pochi processori li possono usare tutti e tre.

1) Salto a Indirizzo Fisso

- Accettando la richiesta di interruzione, il processore salta ad un indirizzo di memoria prefissato, dove inizia la routine di servizio (che è **unica**).
- La routine scandisce (polling) le porte di I/O di tutte le periferiche (o di un gruppo), esaminando i bit di stato che segnalano quale di esse abbia fatto la richiesta.
- Nota bene: si suppone che la periferica mandi la richiesta sulla linea INT_REQ e simultaneamente attivi un bit apposito nel registro di stato della sua interfaccia.
- Identificata la periferica interrompente, la routine ne disattiva il bit di stato, esegue il servizio specifico per la periferica in questione e poi termina.
- Nell'esempio del terminale, tali bit di stato sono IR_T e IR_V (per tastiera e schermo), in REG_STATO.

2) Linee Multiple di Richiesta

- **Ciascuna** periferica dispone di una sua linea di richiesta INT_REQ **specifica** ed eventualmente di una linea di conferma INT_ACK.
- Ogni linea è associata a un indirizzo di memoria prefissato di salto, dove si trova la routine di servizio corrispondente.
- Naturalmente il numero di periferiche è limitato dal numero di linee, che di solito è relativamente modesto, ≤ 8 o circa.

3) Interruzione Vettorizzata

- A ogni periferica è associato un **codice identificativo univoco** (di solito di pochi bit, ≤ 8).
- La periferica manda la richiesta su INT_REQ.
- All'arrivo della conferma su INT_ACK (nota che la richiesta potrebbe essere rimasta pendente a lungo), la periferica manda al processore (via bus dei dati) il codice di identificazione.
- Il processore riceve il codice e lo usa **per derivare l'indirizzo di memoria effettivo** dove è collocata la routine di servizio corrispondente.

Vettore di Interruzione

- Il metodo più comune per derivare l'indirizzo effettivo di routine di servizio a partire dal codice di identificazione di periferica, è il seguente:
  - gli indirizzi iniziali di tutte le routine di servizio sono elencati in una tabella (un array) di indirizzi (di fatto, sono numeri interi)
  - il codice identificativo funziona come indice della tabella, e punta all'elemento contenente l'indirizzo di routine da usare.
- Gli indirizzi di routine si chiamano **vettori di interruzione**.
- La tabella dei vettori sta in una regione di memoria a ciò designata (di solito essa parte dalla cella iniziale di memoria), e va inizializzata correttamente (di solito se ne occupa il S.O.).
- Insieme al vettore di interruzione, in alcuni casi può anche essere associata una parola di stato.

Confronto delle tecniche di identificazione delle periferiche

- Saltare a indirizzo fisso ha costo limitato, ma scandire le periferiche può dare luogo a ritardi.
- Associare un indirizzo ad ogni linea di richiesta è più flessibile, ma il numero delle linee non può essere molto elevato.
- Il meccanismo di vettorizzazione è il metodo più raffinato e costoso, è una tecnica versatile e facilmente riconfigurabile anche in corso di esecuzione.
- Di fatto il metodo di vettorizzazione è molto diffuso, tranne che in processori assai semplici.

Schema di Priorità

- Quando ci sono numerose periferiche collegate in interruzione, per **gestire in modo ordinato** i permessi di interruzione annidati e la simultaneità di richiesta occorre prevedere uno schema di priorità.
- Secondo lo schema di priorità, il processore ascolta e serve determinate richieste di interruzione con preferenza rispetto ad altre.
- Esistono varie soluzioni a seconda del numero di linee di controllo da aggiungere al bus.

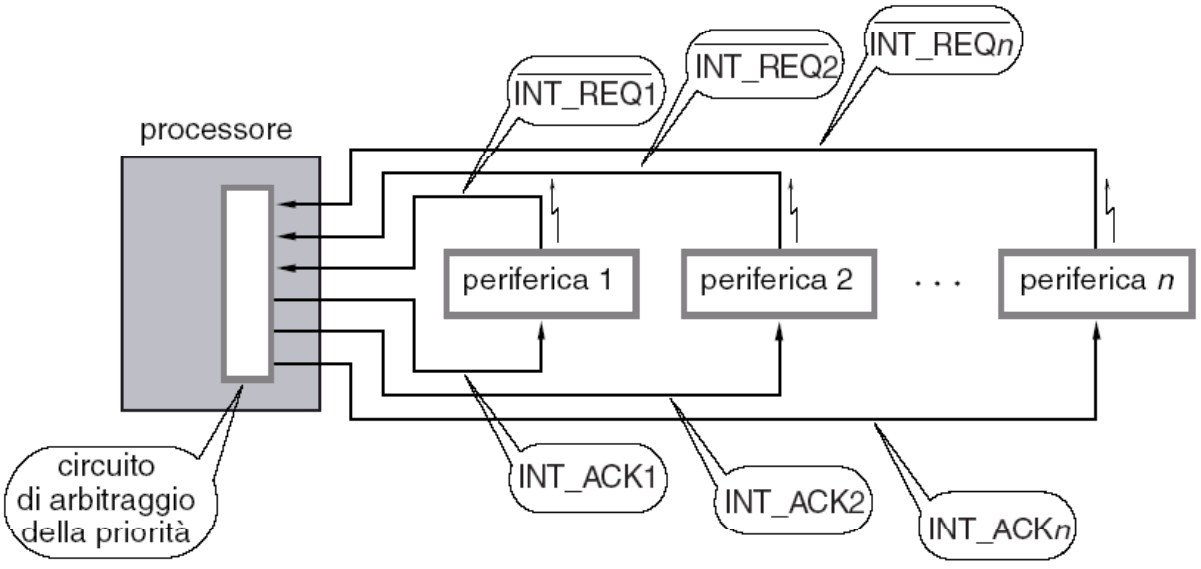
Conferma di Interruzione

- Per gestire la priorità occorre comunque un meccanismo di interruzione basato su una coppia di segnali (linee nel bus di controllo):
  - **INT_REQ**, per richiedere il servizio (come si fa senza priorità)
  - **INT_ACK** (interrupt acknowledge o conferma di interruzione, emessa dal processore), per dare conferma dell'avvenuto inizio del servizio
- Nell'intervallo di tempo tra l'invio della richiesta di interruzione e l'arrivo della conferma (che potrebbe tardare anche a lungo), si dice che l'interruzione è pendente (**pending interrupt**).

Controllore di Interruzione

- Come specificato, ogni periferica ha una coppia di fili (linee):
  - INT_REQ, per richiedere il servizio di interruzione
  - INT_ACK, per confermare l'inizio del servizio
- Richieste e conferme confluiscono nel **circuito di arbitraggio di priorità** (o controllore di interruzione), che le ordina in priorità secondo uno schema fisso oppure programmabile.
- Il processore programma all'inizio il controllore, il quale in seguito opera in modo autonomo.
- Spesso il controllore è integrato nel processore.

Schema di Priorità - Controllore

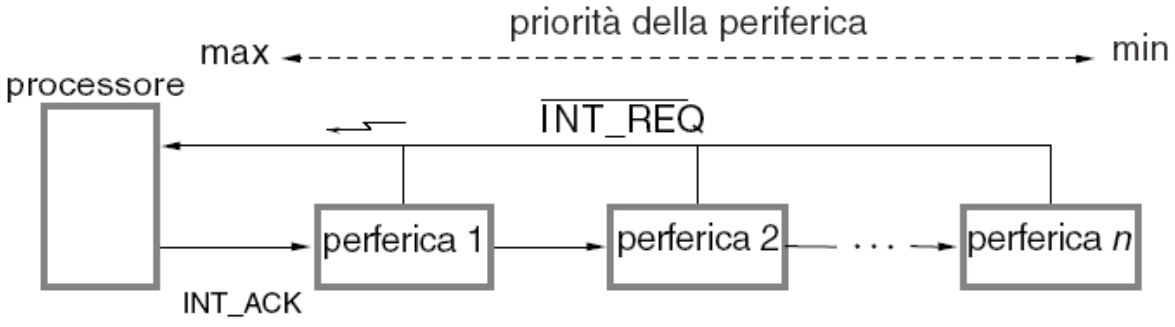


Soluzione con coppia di fili INT_REQ e INT_ACK per ciascuna periferica e con circuito di arbitraggio della priorità (controllore di interruzione)

Collegamento a Festone

- Le periferiche sono collegate a festone (*daisy chain*) e **l'ordine lineare di collegamento lungo il festone dà luogo alla priorità**.
- La linea di richiesta INT_REQ è unica e comune a tutte le periferiche. Ognuna di esse può attivarla.
- La linea di conferma INT_ACK è unica ma attraversa le periferiche, le quali possono decidere se inoltrare la conferma alla periferica successiva o intercettarla e tenerla per sé.
- Di norma il segnale di conferma si propaga lungo la catena di periferiche.
- Quando una periferica richiede una interruzione, attende la conferma e, quando la riceve, la intercetta.

Schema di Priorità – Collegamento a Festone (daisy chain)

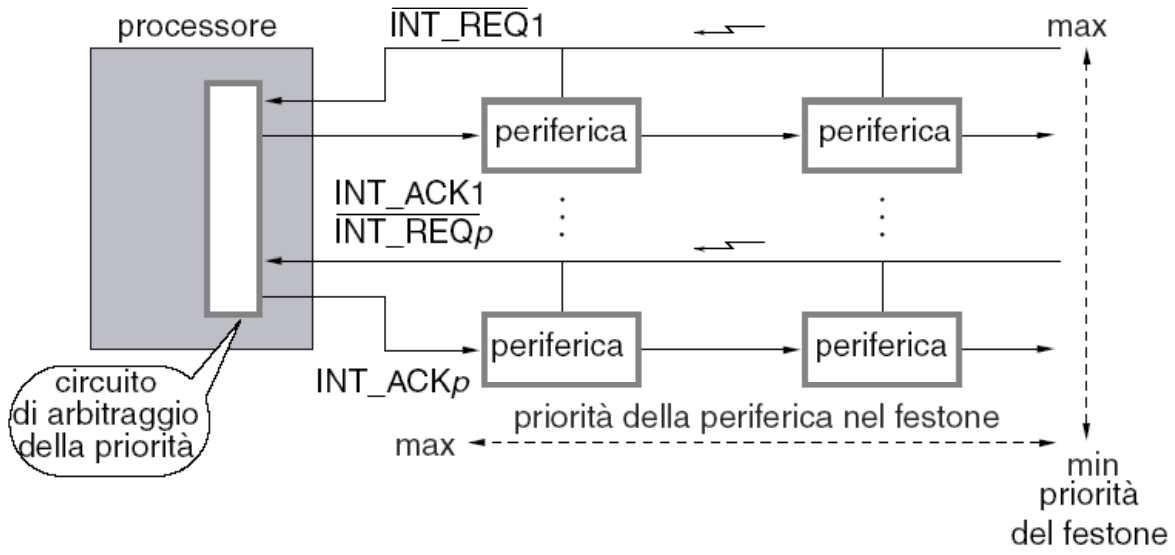


La linea di richiesta (INT_REQ) è unica: il processore emette il segnale di conferma (INT_ACK) non appena riceve una richiesta, **il segnale si propaga lungo la catena di periferiche e la periferica interrompente lo intercetta**, senza passarlo alla periferica successiva

Festone Multiplo

- Il **festone singolo** dà luogo a uno schema di priorità puramente lineare.
- Per avere flessibilità maggiore si possono prevedere due o più festoni, **a priorità differente**.
- La priorità della periferica dipende in primo luogo dal festone cui essa è collegata e, in secondo luogo, dalla posizione che essa occupa nel festone.
- Si può collegare una stessa periferica a più festoni, per darle modo di mandare richieste differenziate secondo il livello di priorità.

Priorità – Collegamento a Festone Multiplo



I due festoni sono a priorità differente
Lo **schema di priorità** complessivo e' **più complesso** di quello lineare

Priorità in Software (1/2)

- Il processore mantiene (nel registro di stato) un **livello di priorità**, cioè un numero (per esempio da 0 a 7, estremi inclusi).
- Ogni richiesta di interruzione è associata ad un livello di priorità ben definito (due o più richieste possono avere lo stesso livello).
- La richiesta di interruzione viene accettata e servita da parte del processore **se e solo se** il livello di priorità della richiesta è **maggiore o uguale a quello corrente** del processore.
- Quando il processore accetta la richiesta di interruzione, il suo livello di priorità diventa uguale a quello della richiesta accettata.

Priorità in Software (2/2)

- Quando la routine di servizio di interruzione termina, il livello di priorità del processore ritorna al valore originale, cioè a quello che valeva subito prima di accettare la richiesta.
- Il livello di priorità della richiesta è registrato nella tabella di interruzione insieme al vettore di interruzione corrispondente (il livello è un campo di bit contenuto nella parola di stato).
- Per salvare e ripristinare il livello di priorità corrente, il processore si serve della pila, esattamente come per l'indirizzo di rientro.

Esempio di Interruzione - Tastiera

- Esempio con **tastiera gestita con interruzione** (con uno qualsiasi dei tre metodi di identificazione di periferica).
- Il programma principale prepara in memoria un buffer per i caratteri in arrivo dalla tastiera, predispone il meccanismo di interruzione e lo attiva.
- Ogniqualvolta viene premuto un tasto, la tastiera ne registra il codice ASCII nel registro di dato della sua interfaccia di I/O e richiede l'interruzione.
- La routine di interruzione legge dalla porta della tastiera il codice ASCII del tasto premuto e lo scrive nel buffer di memoria predisposto dal processore.
- Quando viene premuto il tasto di fine linea, la routine disabilita il meccanismo di interruzione dal lato della tastiera (bit IE_T nel registro di controllo dell'interfaccia).

Vantaggi e Svantaggi dell' Interruzione (1/2)

- **Vantaggi** :
  - la periferica ha la possibilità di richiedere il servizio solo quando serve realmente
  - non implica, da parte del programmatore, la conoscenza dei tempi di azione caratteristici della periferica
  - lascia libero il processore di dedicarsi ad altro quando le periferiche non hanno bisogno di servizio
- **Svantaggi**:
  - ha una certa complessità circuitale, variabile a seconda della realizzazione
  - ha una certa complessità organizzativa, specialmente in sede di definizione dei livelli di priorità (se previsti)

Vantaggi e Svantaggi del Meccanismo di Interruzione (2/2)

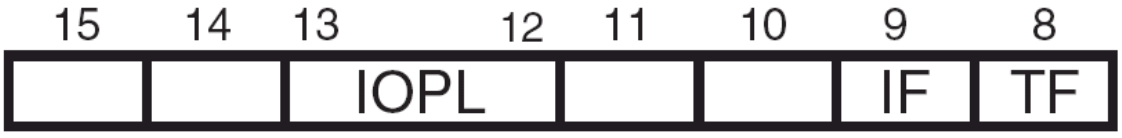
- È una tecnica adatta per gestire un insieme di **periferiche differenziate per modi e tempi di funzionamento**, con quantità mutevoli di dati da scambiare e caratterizzate da comportamento aleatorio e imprevedibile.
- In concreto è molto usata e rappresenta il metodo principale di interazione con il sistema operativo.

GESTIONE DI INTERRUZIONE NEI PROCESSORI PENTIUM

Esempio – Interruzione nell'IA-32

- I processori commerciali Intel Architecture a 32 bit (sia per dato sia per indirizzo), o IA-32 dispongono di meccanismo di interruzione vettorizzata.
- Per le particolarità si rimanda all'esempio del testo in cui le istruzioni macchina generiche sono trasposte con gli aggiustamenti necessari, in istruzioni IA-32 specifiche, che sono elencate.

Registro di Stato IA-32



La struttura del registro di stato IA-32 è piuttosto complessa (qui se ne illustra solo una parte senza i bit di esito). Esistono **quattro modalità di utente/sistema** (selezionabili tramite i due bit **IOPL**), che impattano in modo diverso sulla gestione di interruzione. In almeno uno dei quattro modi il bit **IF** (Interrupt Flag) permette di abilitare o disabilitare il meccanismo di interruzione dal lato di processore.

Concetto di DMA

Controllore di DMA

ACCESSO DIRETTO ALLA MEMORIA (DMA)

Accesso Diretto a Memoria

- L'accesso diretto alla memoria (*Direct Memory Access* o DMA), è una **tecnica hardware** specifica per periferiche capaci di trasferire blocchi di dati di grande dimensione, a velocità elevata e frequentemente.
- Per esempio è usata spesso da dischi (ad alta velocità), scheda di rete e altre periferiche funzionanti a velocità elevata.
- Si basa sull'uso di dispostivi ad hoc e di segnali di controllo specifici nel bus del calcolatore.

Funzionamento (1/2)

- La periferica manda una richiesta di trasferimento (in lettura o scrittura) a un dispositivo specifico, chiamato controllore di DMA (**DMA controller**).
- Il controllore di DMA inoltra la richiesta al processore tramite una linea apposita del bus del calcolatore (chiamata in genere **BUS_REQ**, bus request).
- Il processore recepisce la richiesta e sospende subito l'esecuzione lasciando libero il bus del calcolatore.
- **Il controllore di DMA acquisisce il controllo del bus** e **gestisce** il trasferimento del blocco di dati tra la periferica e un buffer di memoria designato allo scopo, a seconda della periferica interessata dall'operazione.

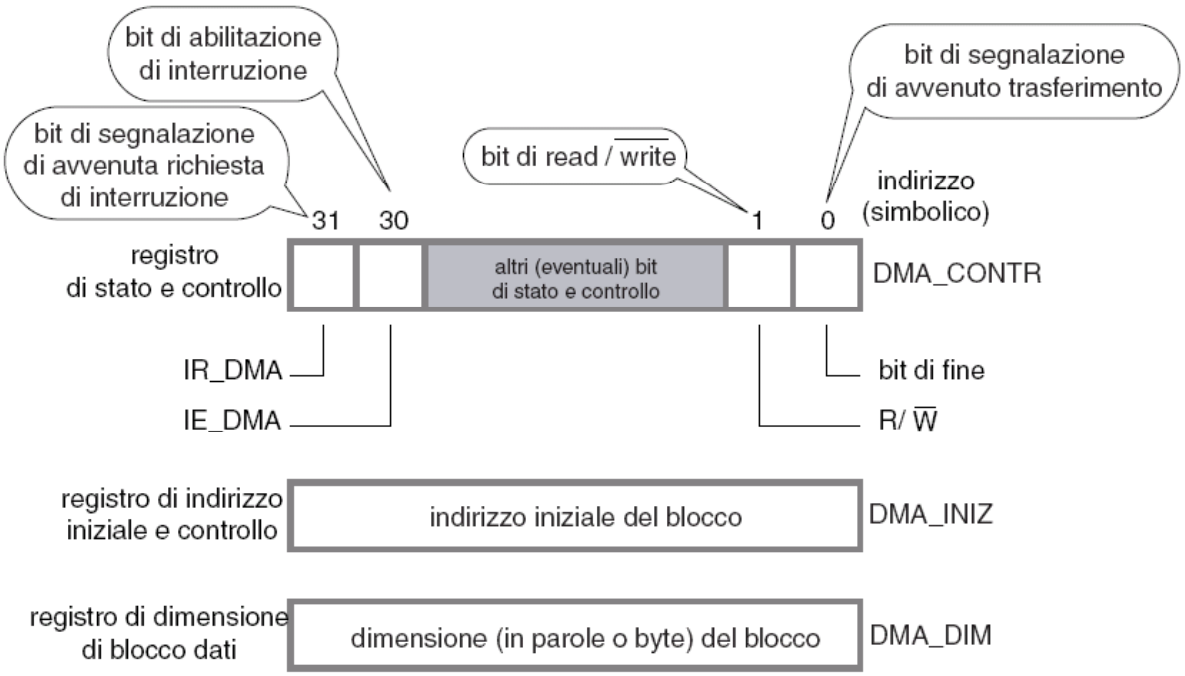
Funzionamento (2/2)

- Non appena il blocco è stato trasferito, il controllore di DMA restituisce il controllo del bus e il processore, rimasto fermo per tutta la durata del trasferimento del blocco di dati, riprende l'attività sospesa.
- Per il processore la sospensione è del tutto trasparente e passa inosservata. Esso si può rendere conto del trasferimento avvenuto esaminando lo stato del controllore, cioè leggendone il registro di stato.
- Spesso però il controllore di DMA segnala che un blocco è stato trasferito mandando al processore una richiesta di interruzione, non appena il processore è ripartito.

Controllore di DMA (1/3)

- Per gestire il trasferimento del blocco il controllore di DMA contiene alcuni registri specifici:
  - **indirizzo iniziale del buffer di memoria**
  - **dimensione del buffer**, ovvero del blocco di dati da trasferire (di solito la dimensione è espressa in byte)
  - **registro di controllo**, contenente vari bit:
    - R / W, per specificare se l'operazione sia di lettura o scrittura
    - bit di fine, si attiva quando il blocco è interamente trasferito
    - IR_DMA, si attiva quando il controllore richiede interruzione
    - IE_DMA, serve per abilitare o disabilitare il meccanismo di interruzione (che è facoltativo) del lato del controllore
- Il processore all'inizio deve inizializzare registri e stato del controllore di DMA, poi il controllore funziona in modo autonomo (tranne quando va riconfigurato).

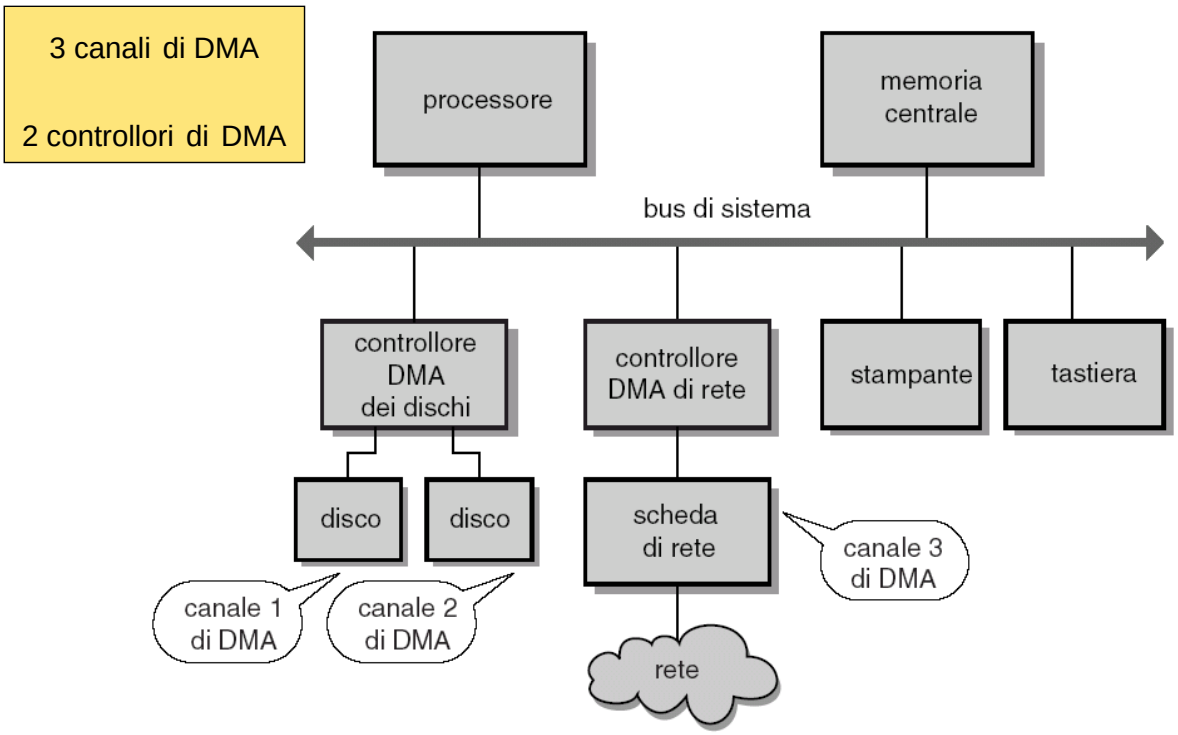
Controllore di DMA (2/3)



Controllore di DMA (3/3)

- Per il funzionamento del segnale di controllo BUS_REQ, si veda come funziona l'arbitraggio del bus.
- Il controllore di DMA può gestire più periferiche e, per ciascuna di esse, dispone di un gruppo di registri, programmato opportunamente (di solito dal SO).
- Ogni gruppo di registri va sotto nome di **canale di DMA** ed è legato a una periferica specifica.
- Il controllore di DMA può essere un dispositivo fisicamente separato dal processore ma spesso è realizzato sullo stesso componente integrato.
- Il calcolatore può disporre di un certo numero di canali di DMA, dedicati alle periferiche capaci e veloci.

Canali di DMA



Vantaggi e Svantaggi del DMA

- **Vantaggi**:
  - gestisce in modo efficiente le periferiche capaci e veloci
  - interferisce con l'attività normale del processore (tenendolo fermo) solo per il tempo **minimo** strettamente necessario per trasferire il blocco di dati
- **Svantaggi**:
  - ha una notevole complessità circuitale, dovuta al controllore
- Il DMA è comunque una tecnica molto usata per le periferiche capaci (molti dati) e veloci.
- Poiché spesso l' avvenuto trasferimento viene segnalato al processore tramite interruzione, il DMA si presenta più come un raffinamento della tecnica di interruzione che come un metodo a sé stante.