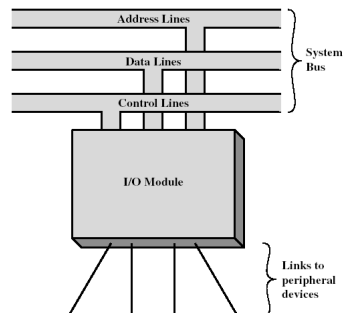


Input/Output

Tecniche di I/O

- Ci sono tre tecniche di I/O:
 - I/O programmato
 - I/O guidato dalle interruzioni
 - Accesso diretto alla memoria (DMA)
- Un modulo di I/O si interfaccia al bus di sistema e controlla una o più periferiche



Funzionalità dei moduli di I/O

- Le funzioni ed i requisiti principali di un modulo di I/O possono essere raggruppati nelle seguenti categorie:
 - Controllo e temporizzazione
 - Comunicazione con il processore
 - Comunicazione con il dispositivo
 - Bufferizzazione dei dati
 - Rilevamento di errori

Controllo e temporizzazione

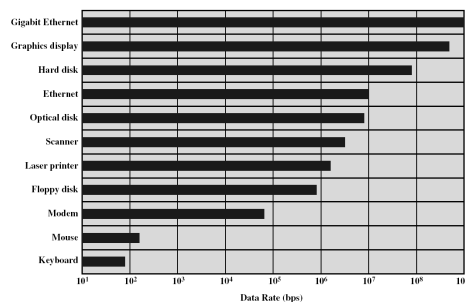
- Il processore può decidere di comunicare con uno o più dispositivi di I/O in qualunque momento
- Funzionalità di controllo e temporizzazione sono necessarie per coordinare il flusso di traffico tra le risorse interne e i dispositivi esterni
- Esempio di lettura da dispositivo:
 - Il processore interroga il modulo di I/O per verificare lo stato del dispositivo
 - Il modulo di I/O restituisce lo stato del dispositivo
 - Il processore richiede il trasferimento dati con un comando indirizzato al modulo di I/O
 - Il modulo ottiene i dati dal dispositivo
 - I dati sono trasferiti dal modulo alla CPU

Comunicazione con processore e dispositivo di I/O

- La comunicazione tra modulo di I/O e processore avviene nei seguenti passi:
 - Decodifica del comando: il modulo di I/O accetta comandi dal processore generalmente inviati come segnali sul bus di controllo
 - Dati: i dati sono scambiati tra modulo di I/O e processore attraverso il bus dati
 - Resoconto di stato: essendo un dispositivo di I/O molto più lento del processore è necessario che sia in grado di comunicare il suo stato (busy o ready ed eventuali condizioni di errore)
 - Riconoscimento dell'indirizzo: il modulo di I/O deve essere in grado di riconoscere l'indirizzo di ogni dispositivo controllato
 - Invio di caratteri di controllo al dispositivo

Bufferizzazione dei dati

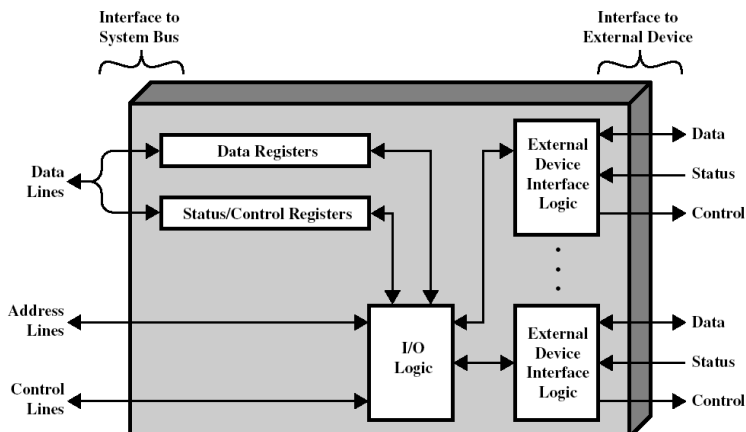
- La differenza di velocità tra processore/memoria e dispositivo di I/O impone che i dati trasferiti al dispositivo vengano temporaneamente memorizzati dal modulo di I/O
- Analogamente, i dati trasferiti dal dispositivo sono bufferizzati per evitare di rallentare il trasferimento verso processore/memoria



Rilevamento di errori

- Gli errori che un modulo di I/O deve essere in grado di rilevare (e riportare al processore) possono essere divisi in due classi:
 - Malfunzionamenti meccanici ed elettrici (blocco della carta, traccia disco rovinata, ecc.)
 - Alterazione delle informazioni durante il trasferimento da modulo a dispositivo o viceversa. Per il controllo di tali errori vengono usati codici (bit di parità, CRC, ecc.)

Struttura di un modulo di I/O



Modulo di I/O: nomenclatura

- Un modulo di I/O incaricato di gestire la maggior parte dell'elaborazione, offrendo al processore un'interfaccia ad alto livello, è detto solitamente *canale di I/O* o *processore di I/O*
- Moduli più primitivi che richiedono controlli dettagliati da parte della CPU sono detti *controllori di I/O* o *controllori di dispositivo*

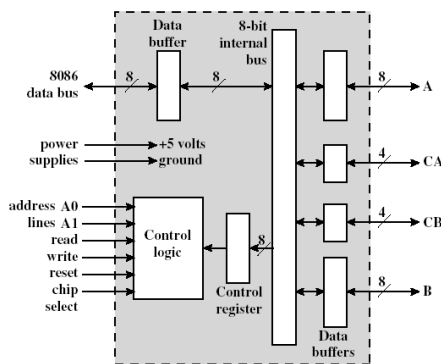
Indirizzamento dei dispositivi

- La comunicazione tra il processore (o la memoria) ed il dispositivo avviene attraverso i registri (o porte).
- Questi sono accessibili tramite il bus di sistema, al pari delle celle di memoria, in quanto a ciascuno di essi è associato un indirizzo.
- La loro connessione al bus può avvenire secondo 2 modalità:
 - memory-mapped I/O
 - isolated I/O, o I/O-mapped I/O

Memory-mapped I/O

- I registri dei dispositivi di I/O sono connessi come le normali celle di memoria.
- Lo spazio di indirizzamento per la memoria è quindi ridotto.
- Si può fare accesso ai registri delle periferiche utilizzando tutte le istruzioni ed i modi di indirizzamento utilizzabili per accedere alla memoria.
- È la soluzione adottata dal 68000.

Periferica programmabile 82C55A

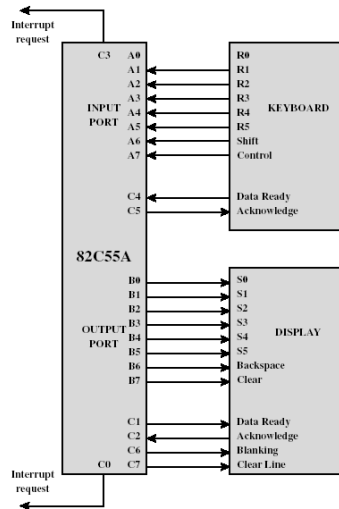


(a) Block diagram

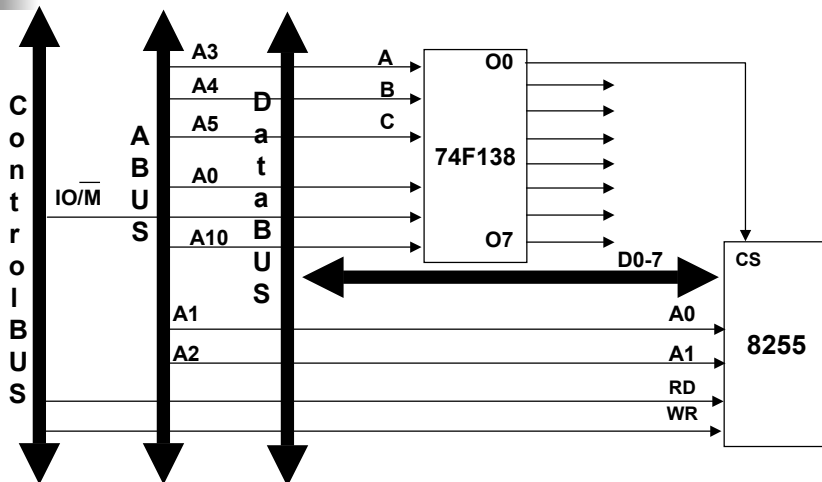
PA3	1	40	PA4
PA2	2	39	PA5
PA1	3	38	PA6
PA0	4	37	PA7
Read	5	36	Write
Chip select	6	35	Reset
Ground	7	34	D0
A1	8	33	D1
A0	9	32	D2
PC7	10	31	D3
PC6	11	30	D4
PC5	12	29	D5
PC4	13	28	D6
PC3	14	27	D7
PC2	15	26	V
PC1	16	25	PB7
PC0	17	24	PB6
PB0	18	23	PB5
PB1	19	22	PB4
PB2	20	21	PB3

(b) Pin layout

Periferica programmabile 82C55A



Periferica programmabile 82C55A



Connessione memory mapped

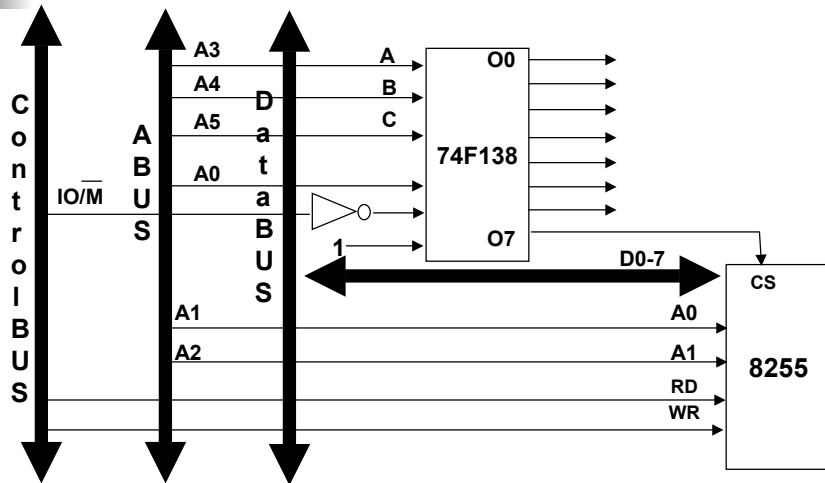
Indirizzi delle porte dell'82C55A: memory mapped

- Gli indirizzi dei registri dell'82C55A sono i seguenti:
 - porta A: 0400h
 - porta B: 0402h
 - porta C: 0404h
 - Controllo: 0406h

Isolated I/O

- Gli spazi di indirizzamento per la memoria e per le porte di I/O sono separati, e sono attivati alternativamente da appositi segnali (ad esempio IO/M nell'8086).
- Per accedere alle porte di I/O si devono utilizzare apposite istruzioni (IN e OUT).

Periferica programmabile 82C55A



Connessione isolated I/O

Indirizzi delle porte dell'82C55A: isolated I/O

- Gli indirizzi dei registri dell'82C55A sono I seguenti:
 - porta A: 0038h
 - porta B: 003Ah
 - porta C: 003Ch
 - Controllo: 003Eh

I/O programmato

- I dati sono scambiati direttamente tra il processore ed il modulo di I/O
- Un programma assume il controllo diretto dell'operazione di I/O:
 - Rileva lo stato del dispositivo
 - Invio del segnale di lettura/scrittura
 - Trasferimento dati
- Quando il processore assegna un comando al modulo di I/O deve aspettare fino al completamento dell'operazione:
 - Spreco di tempo

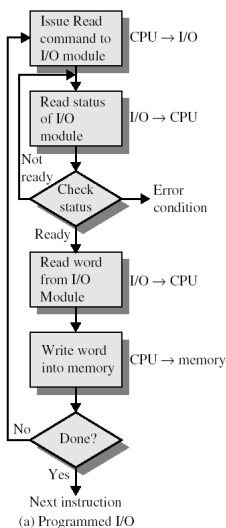
I/O programmato

- Quando il processore esegue un programma ed incontra un'istruzione legata all'I/O la esegue emettendo un comando diretto al modulo di I/O
- Il modulo realizza l'azione richiesta ed imposta i bit opportuni nel registro di stato:
 - Non avverte/interrompe il processore
- Il processore deve controllare periodicamente lo stato del dispositivo

I/O programmato

- Per eseguire un'operazione di I/O il processore invia ad un indirizzo, che indica un particolare modulo di I/O ed un dispositivo, un comando:
 - Comando di controllo: usato per attivare la periferica e dirle cosa fare
 - Test: usato per verificare lo stato del dispositivo
 - Lettura
 - Scrittura

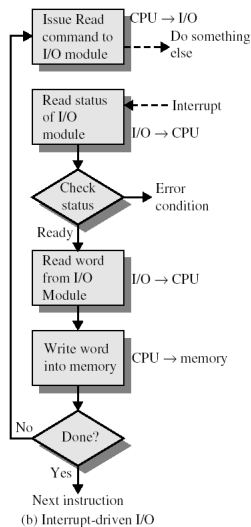
I/O programmato: esempio di lettura



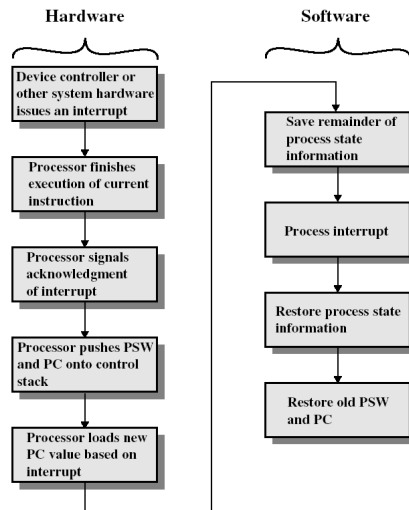
I/O con interruzioni

- Il problema dell'I/O programmato è che la CPU deve aspettare affinché il modulo di I/O non sia pronto:
 - Durante l'attesa deve interrogare continuamente lo stato del dispositivo
- Un'alternativa è quella di fare in modo che la CPU emetta il comando di I/O e poi prosegua con le altre istruzioni:
 - Sarà il modulo di I/O ad interrompere la CPU quando sarà pronto per lo scambio dei dati

I/O con interruzioni: esempio di lettura



Gestione delle interruzioni



I/O con interruzioni: problemi

- Poiché in un sistema di elaborazione sono, in generale, presenti moduli di I/O multipli, come può il processore riconoscere quale dispositivo ha generato l'interruzione?
- Se si sono verificate "contemporaneamente" diverse interruzioni, in base a quale criterio il processore decide quale elaborare?

Identificazione di un dispositivo

- L'identificazione di un dispositivo può avvenire nei seguenti modi:
 - Linee di interruzioni multiple
 - Interrogazione via software
 - Daisy chain (interrogazione via hardware)
 - Arbitraggio del bus

Linee di interruzioni multiple

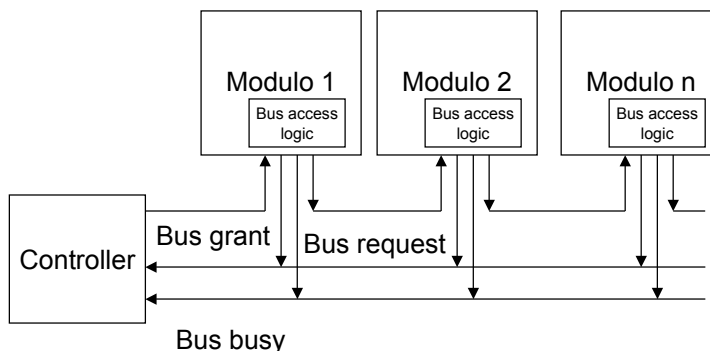
- Una linea di interruzione è dedicata ad ogni modulo di I/O presente nel sistema:
 - Difficilmente realizzabile dal punto di vista hw
 - Richiede troppi pin per la CPU

Interrogazione via software

- Quando il processore rileva un'interruzione, esegue una procedura che si occupa di consultare ogni modulo di I/O per determinare quale ha provocato l'interruzione:
 - Spreco di tempo proporzionale al numero di moduli di I/O presenti nel sistema

Interrogazione via hardware

- L'utilizzo di una struttura daisy chain permette di interrogare via hw i vari moduli



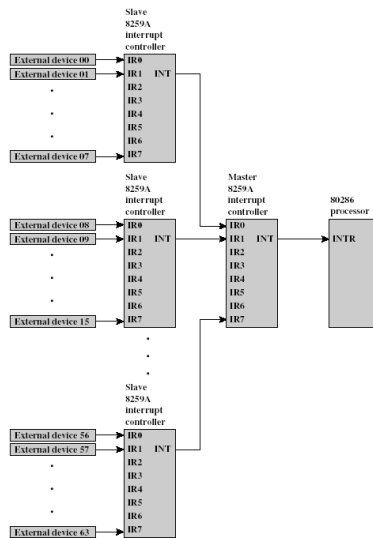
Arbitraggio del bus

- In questa tecnica, il modulo deve prima prendere il controllo del bus e poi può inviare un segnale di interruzione:
 - Un solo modulo alla volta può inviare richieste

Controllore delle interruzioni

- Molto spesso, i processori hanno una sola linea per la richiesta delle interruzioni (INTR) e una sola linea per la conferma dell'avvenuta ricezione dell'interruzione (INTA)
- Un dispositivo esterno alla CPU, controllore delle interruzioni, è solitamente incaricato di gestire/arbitrare moduli di I/O multipli

82C59A



82C59A

- L'82C59A riceve le richieste dai moduli collegati, determina quale ha la priorità maggiore e avverte il processore attivando la linea INTR
- Il processore risponde attivando la linea INTA
- Il controller pone sul bus di dati l'opportuno vettore di informazioni che permette alla CPU di eseguire la routine di interrupt opportuna
- L'82C59A è programmabile e la CPU può definire lo schema di priorità da assegnare ai vari moduli

8086/88: interrupt

- L'8086/8088 implementa 256 tipi di interrupt, suddivisi in 4 gruppi:
 - interrupt hardware esterni
 - interrupt software
 - interrupt interni
 - interrupt non mascherabili.

Priorità

- In ordine di priorità decrescente si hanno:
 - interrupt interni
 - interrupt non mascherabili
 - interrupt software
 - interrupt hardware esterni.
- Solo interrupt di un gruppo con priorità più elevata possono interrompere una routine di servizio dell'interrupt.
- All'interno di un gruppo ogni interrupt ha un numero; gli interrupt con numero inferiore hanno priorità maggiore.
- Il numero assegnato agli interrupt interni e a quello non mascherabile non può essere modificato.

Vector table

- Contiene i puntatori alle routine di servizio di ciascuno dei 256 tipi di interrupt possibili.
- Si trova in memoria agli indirizzi più bassi, da 00000H a 003FEH.
- Ad ogni tipo di interrupt sono associati 4 byte:
 - i primi 2 byte contengono l'offset a cui si trova la procedura all'interno del segmento
 - i 2 byte successivi contengono l'indicazione del segmento di codice in cui si trova la procedura.
- L'indirizzo assoluto del primo dei 4 byte associati ad un interrupt n è pari a $n*4$.

Abilitazione/disabilitazione degli interrupt

- Il flag IF permette di abilitare/disabilitare gli interrupt hardware esterni.
- IF può essere modificato via software attraverso le istruzioni CLI e STI.
- IF viene automaticamente resettato all'attivazione di una routine di servizio dell'interrupt.

Interrupt non mascherabili

- Sono attivati tramite il pin NMI.
- Non sono mascherabili usando il flag IF.
- Sono positive edge triggered. Per essere rilevati, il segnale NMI deve restare attivo per 2 colpi di clock consecutivi.
- Ad essi corrisponde il vettore di interrupt all'indirizzo 0008H.
- Sono utilizzati in connessione con eventi quali la caduta di alimentazione o un errore di lettura in memoria.

Interrupt interni

- Divisione per Zero: quando il quoziente di una DIV o IDIV non può essere rappresentato nel campo destinazione, viene attivata la routine corrispondente al vettore di interrupt in 0000H
- Overflow: la istruzione INTO causa l'attivazione della routine di interrupt corrispondente a 0010H se OF vale 1
- Single Step: se il flag TF vale 1 al termine di ogni istruzione viene attivata la routine di interrupt corrispondente a 0004H
- Breakpoint: se l'istruzione INT non ha parametri, viene codificata su un solo byte e causa l'attivazione della routine di interrupt corrispondente a 000CH.

Mappa degli interrupt

Interrupt	0	→	Divide Error
	1	→	Single Step
	2	→	NonMaskable Interrupt
	3	→	Breakpoint
	4	→	Overflow
	5 ÷ 31	→	Reserved
	32 ÷ 255	→	Liberi

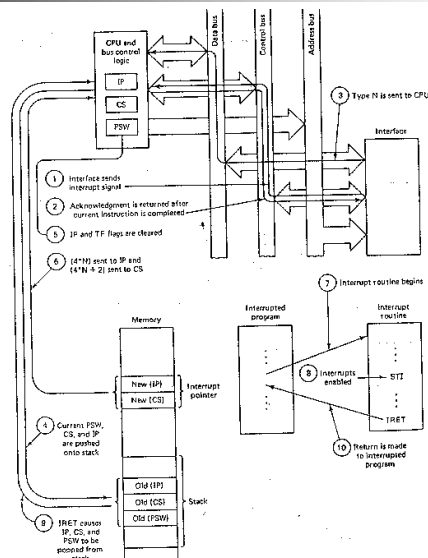
8086: protocollo di interrupt

- Un dispositivo esterno invia una richiesta di interrupt sul pin INTR (*level triggered*)
- Durante l'ultimo periodo di clock di una istruzione la CPU rileva la presenza di un valore 1 su INTR
- La CPU invia un impulso su INTA per segnalare che l'interrupt è stato rilevato
- La CPU invia un secondo impulso su INTA per chiedere al dispositivo di scrivere sul data bus il numero corrispondente al tipo dell'interrupt
- La CPU legge dal data bus il tipo n dell'interrupt (1 byte)

8086: protocollo di interrupt

- La CPU salva nello stack il valore del registro dei flag (PSW) e l'indirizzo di ritorno (registri CS e IP)
- La CPU azzerava i flag IF e TF per disabilitare gli interrupt hardware esterni ed il *trap mode*.
- La CPU accede all'elemento (4*n)-esimo nella Interrupt Vector Table
- Viene attivata la corrispondente procedura di Interrupt.

8086: protocollo di interrupt

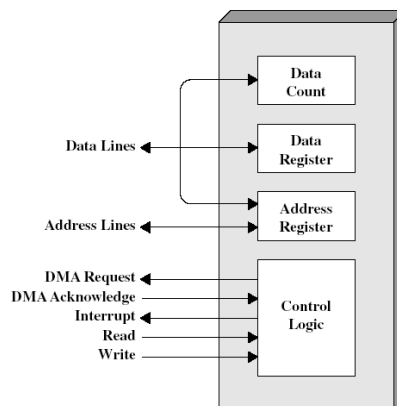


Accesso diretto alla memoria

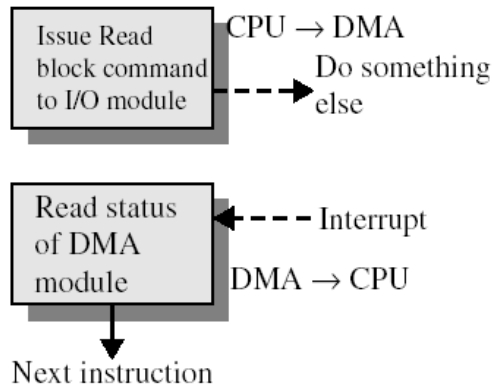
- L'I/O con interruzioni, nonostante sia più efficiente dell'I/O programmato, richiede ancora l'intervento della CPU nel trasferimento dei dati:
 - La velocità di trasferimento dei dati è limitata dalla velocità con cui il processore può controllare e servire il dispositivo
 - Il processore è vincolato dalla gestione del trasferimento
- Nel caso debbano essere trasferite grandi quantità di dati è necessaria una tecnica più efficiente:
 - DMA: Direct Memory Access

DMA

- Il trasferimento in DMA utilizza un ulteriore modulo sul bus di sistema:
 - Simula il comportamento del processore
 - Controlla il bus di sistema
- Il DMA deve usare il bus in mutua esclusione con la CPU



DMA: esempio di lettura



DMA: modi di funzionamento

- Il trasferimento dei dati in DMA può avvenire in vari modi:
 - trasferimento a blocchi (burst transfer)
 - trasferimento con cycle stealing
 - trasferimento in transparent DMA.

Trasferimento a blocchi

- Prevede che il DMA Controller, una volta acquisito il controllo del bus, lo mantenga per tutto il tempo richiesto per trasferire un blocco di dati.
- In tal modo il trasferimento avviene alla massima velocità ma la CPU è bloccata per tutta la durata del trasferimento.
- Il trasferimento a blocchi è importante per periferiche quali i dischi magnetici, dove il trasferimento del blocco non può essere interrotto

Trasferimento cycle stealing

- Il DMA Controller trasferisce i dati in piccoli blocchi occupando il bus per periodi limitati di tempo.
- In tal modo la velocità di trasferimento è minore ma la CPU non è bloccata per periodi troppo lunghi.

Trasferimento in transparent DMA

- Il DMA Controller è in grado di rilevare quando la CPU non utilizza il bus, e solo in quei periodi esegue il trasferimento dei dati.
- In tal modo la CPU non è praticamente rallentata dal DMA Controller.