SCHEDA DI AUTOVALUTAZIONE

Autovalutazione delle conoscenze e delle abilità che hai raggiunto alla fine del capitolo: metti un segno di spunta ✔ vicino alle voci che ritieni di aver acquisito.

Conoscenze di base

- ☐ Definizione di automa
- ☐ Grafo di Mealy e di Moore
- ☐ Proprietà degli automi
- ☐ Elementi di memoria
- ☐ Stati equivalenti di un automa
- ☐ Automa minimo equivalente

Conoscenze di approfondimento

- ☐ Circuito sincrono e asincrono
- ☐ Sollecitazione di circuiti a livelli e a impulsi sia in modalità sincrona e sia in modalità asincrona
- ☐ Macchina di Turing
- ☐ Relazione tra macchina di Turing e Algoritmo
- ☐ Tesi di Church-Turing

Abilità generali

- ☐ Progettare semplici automi
- ☐ Rappresentare un automa mediante il grafo di Mealy e di Moore
- ☐ Rappresentare un automa mediante il modello matematico
- ☐ Trasformare di un automa di Mealy in Moore e viceversa
- ☐ Calcolare eventuali stati equivalenti di un automa
- ☐ Calcolare l'automa minimo equivalente
- ☐ Rappresentare l'automa minimo equivalente
- ☐ Realizzare reti di memoria tramite i più noti flip flop
- ☐ Sintetizzare semplici automi sia in base al modello di Mealy e sia in base al modello di Moore utilizzando i più diffusi di flip flop
- ☐ Sintetizzare un automa minimo equivalente utilizzando i più diffusi flip flop

Abilità avanzate

- ☐ Individuare quale tipo di segnale utilizzare per pilotare i circuiti
- ☐ Progettare MdT elementari che implementano semplici algoritmi
- ☐ Essere in grado di comprendere che cosa è una funzione calcolabile e non calcolabile

Domande per la preparazione alla prova orale pag. 467-468
Soluzioni dei quesiti pag. 472

Implementazioni fisiche

Part 2

CAPITOLO 5

Architetture

In questo capitolo avrai una visione generale di un sistema di elaborazione. Saprai distinguere tra bus di sistema e bus di espansione e ne imparerai le funzioni e i servizi standard. Sarai in grado di collegare e gestire le memorie elettroniche, ottiche e magnetiche. Conoscerai le tecnologie delle stampanti e le modalità per interfacciarle al sistema. Infine apprenderai la funzione di chipset e BIOS nel sistema di elaborazione.

Modello di sistema di elaborazione
Modello di un calcolatore convenzionale
Le memorie elettroniche
Bus
- **Bus standard e bus ISA**
- **Bus locali e bus VESA**

Plug & Play
Bus PCI
Porta AGP
PCI-express
- **Bus PCMCIA**
- **Bus SCSI**

Lo standard USB
Il bus IEEE 1394
La porta seriale
La porta parallela
Stampanti e memory card
- **Protocollo PICTBRIDGE**

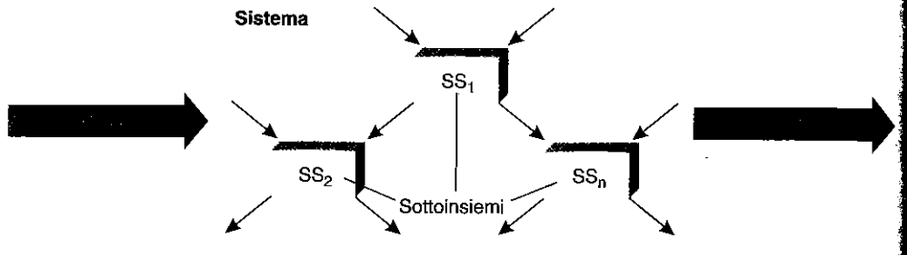
La porta IDE-EIDE e gestione dei dischi fissi
Bus SATA e gestione dei dischi
- **Accodamento nativo dei comandi**
- **Modalità di trasferimento dei dati per i dischi**
- **Algoritmi di traslazione nella gestione dei dischi**
- **Sistema RAID per organizzare più dischi fissi**
- **Dispositivi di storage in rete**

Dischi ottici
USB Flash Drive
BIOS

1. Modello di sistema di elaborazione

Dal punto di vista **strutturale** *(hardware)*, si può dare la seguente definizione:

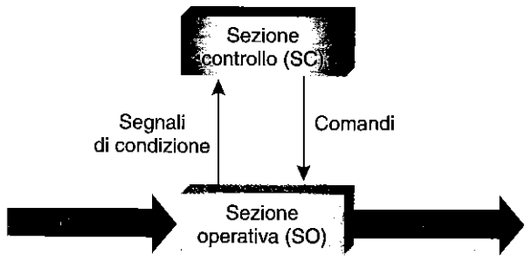
Il **sistema di elaborazione** è l'interconnessione di un insieme di unità (sottosistemi) concorrenti e interagenti direttamente o indirettamente, al fine di fornire un prefissato numero di servizi.



- I sottosistemi sono *concorrenti* quando le attività vengono svolte nello stesso tempo: più genericamente l'attività di un sottosistema comincia prima che termini l'attività dell'altro. Due sistemi concorrenti non interagenti si dicono *disgiunti*.
- Due sottosistemi interagiscono in modo *indiretto* quando sono logicamente indipendenti, ma utilizzano le stesse risorse e queste sono insufficienti. Per esempio in un supermercato i clienti fanno la spesa in modo indipendente l'uno dall'altro, però sono costretti ad interagire (fare la coda) indirettamente perché il numero delle casse aperte, di solito, è inferiore al numero dei clienti che devono pagare.
- Due sottosistemi interagiscono in modo *diretto* quando dipendono l'uno dall'altro per proseguire la loro attività. Per esempio, c'è interazione diretta tra il cliente e la cassiera: il cliente non paga se prima non gli viene comunicato l'importo e la cassiera non emette lo scontrino se non incassa il corrispettivo della spesa.

Vedere un sistema costituito da insieme di sottosistemi consente di utilizzare una metodologia di progettazione di tipo *top-down* simile a quella utilizzata nella ricerca degli algoritmi risolutivi dei problemi informatici.

Dal punto di vista **funzionale** (*software*), un sistema di elaborazione può essere rappresentato mediante l'interconnessione di due sottosistemi (*macchine sequenziali*) dette *Sezione di Controllo* (**SC**) e *Sezione Operativa* (**SO**).

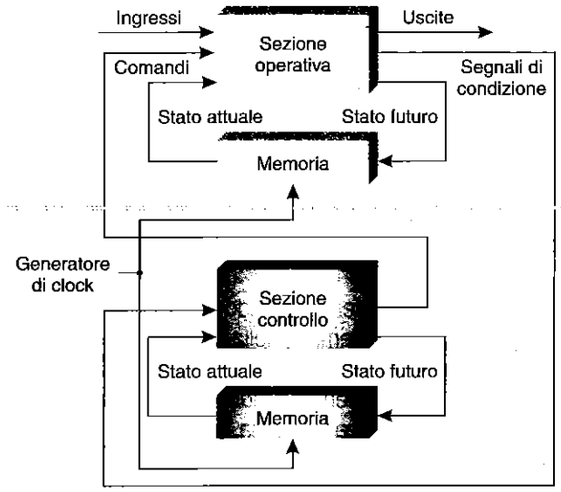


La figura illustra l'interconnessione tra la sezione controllo e la sezione operativa: le istruzioni fornite tengono conto dello stato della sezione operativa, cioè prima di dare un comando la sezione di controllo deve tenere conto della condizione in cui si trova la sezione operativa.

La sezione operativa realizza l'**hardware**, mentre la sezione di controllo realizza il **software di base** del sistema di elaborazione:

sistema di elaborazione = hardware + software

Pertanto, utilizzando il modello della macchina sequenziale, si può rappresentare un sistema di calcolo nel seguente modo:



ATTIVITÀ DI VERIFICA

Rispondere ai quesiti di base n. 1, 2 pag. 278

2. Modello di calcolatore convenzionale

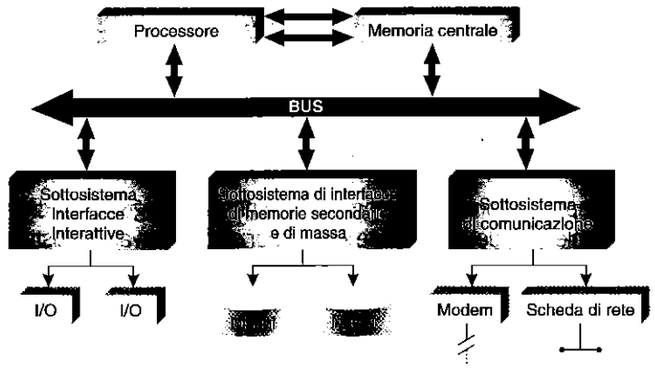
Un calcolatore convenzionale è un sistema di elaborazione nel quale deve essere eseguita un'istruzione per volta in ordine sequenziale, a meno che l'istruzione stessa non indichi una diversa sequenza.

Un sistema come questo si chiama anche modello di **Von Neumann**, dal nome del matematico che ne definì l'architettura nel 1952.

In un calcolatore convenzionale si possono individuare tre sottosistemi:

1. Sottosistema delle memorie
2. Sottosistema di calcolo
3. Sottosistema di input-output

che interagiscono tra loro, utilizzando per la comunicazione un sottosistema formato dal *bus*, al fine di eseguire il processo controllato dal programma utente.



• **Memoria principale**

La memoria centrale di elaborazione ha il compito di memorizzare i programmi e i dati che devono essere elaborati. Essa, su richiesta delle altre unità, esegue le funzioni di lettura e scrittura a livello di *parola macchina* che può essere il byte, la word e così via. Nei calcolatori convenzionali la memoria è costituita da una sola unità di elaborazione: in altre architetture l'unità può essere costituita da due unità sequenziali, come nei sistemi con *memoria cache*, di cui si parlerà nel seguito.

In alcuni sistemi di elaborazione che realizzano il *calcolo parallelo* oppure negli *array processor* si possono avere più unità indipendenti di memoria.

Il **calcolo parallelo** può essere eseguito da sistemi che collegano tanti calcolatori, ognuno dei quali esegue in modo indipendente un'istruzione e interagisce con gli altri calcolatori comunicando i risultati dell'elaborazione attraverso un canale. Sistemi di questo tipo si chiamano **transputer** (*transistor computer*), nel senso di sistemi che hanno un computer come elemento base, così come un circuito integrato ha come elemento base un transistor.

Per **array processor** si intende un sistema in grado di eseguire, contemporaneamente, la stessa istruzione su dati diversi. Per esempio, nella somma di due vettori si sommano gli elementi corrispondenti dei due vettori: la stessa operazione è fatta su dati diversi, quindi bisogna che ci siano diverse unità di memoria indipendenti e tante unità di calcolo, tutte uguali, che eseguono in contemporanea la stessa istruzione.

Le proprietà dell'unità di memoria sono:

- il **tempo di accesso**, cioè il tempo, dell'ordine dei nanosecondi, che intercorre tra la richiesta di un dato e la sua disponibilità.
- la **capacità di memorizzazione**, dell'ordine dei 512 MB o superiore.

■ Processore

Il compito del processore è di elaborare le istruzioni di un programma e interagire parzialmente o totalmente con le unità di I/O durante la loro comunicazione con la memoria.

Le funzione tipiche di un processore sono descritte dai passi della seguente procedura: essa indica le operazioni che devono essere ripetute per ciascuna istruzione contenuta nel programma da eseguire.

1. Lettura di un'istruzione dalla memoria.
2. Decodifica.
3. Calcolo degli indirizzi degli operandi in memoria.
4. Caricamento degli operandi dalla memoria.
5. Esecuzione dell'istruzione.
6. Analisi delle interruzioni, cioè prima di passare all'istruzione successiva controlla e gestisce eventuali eventi prioritari rispetto al processo che sta eseguendo.
7. Calcolo della posizione dell'istruzione successiva e ripetizione della procedura da passo 1.

In alcuni sistemi il processore viene scomposto in più unità, ognuna dedicata a una parte delle funzioni complessive del processore stesso. Per esempio in alcuni sistemi il processore viene diviso in due unità sequenziali:

- la prima si occupa delle lettura delle istruzioni, cioè del primo passo della precedente procedura;
- la seconda gestisce i successivi 6 passi della procedura.

I microprocessori di questo tipo si chiamano processori con **prefetch**.

Vi sono inoltre processori che ripartiscono in più fasi, come in una catena di montaggio, il compito da svolgere, assegnando l'esecuzione di ciascuna fase a sottounità a ciò preposte: in questo caso si hanno processori con **pipeline**. Il funzionamento è analogo a quello di un'azienda che produce automobili: essa divide il lavoro tra vari reparti (motori, carrozzeria, tappezzeria e così via) organizzandoli in una catena di montaggio. Il prodotto finito è ottenuto dall'assemblaggio dei pezzi aggiunti nei vari reparti.

> Nei sistemi con *prefetch* e *pipeline* si ottiene il parallelismo nell'elaborazione delle istruzioni di uno stesso programma, con notevoli vantaggi nella velocità di calcolo.

■ Unità di ingresso-uscita

Le unità di ingresso e di uscita sono classificate in tre categorie:

1. Unità di interazione: permettono all'utente di interagire con il sistema di calcolo. Appartengono a questa categoria tastiere, mouse, scanner, stampanti, video, microfoni, cuffie, webcam.
2. Unità di memorizzazione: memorizzano permanentemente le informazioni in esse contenute. Vengono chiamate anche memorie di lungo termine per differenziarle dalla memoria centrale (memorie di breve termine). Appartengono a questa categoria dischi magnetici, CD-ROM, DVD, CD-R, CD-RW, nastri magnetici.
3. Unità di comunicazione: permettono di interconnettere sistemi di calcolo in modo da realizzare una rete. Per esempio: i modem, le schede ISDN, le schede di rete LAN.

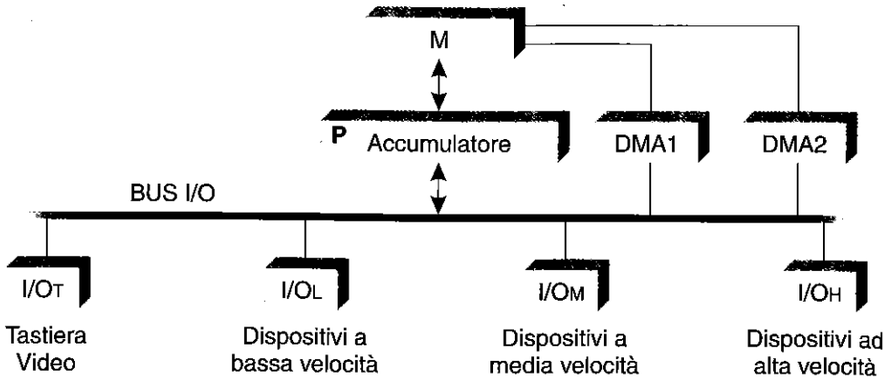
Le unità di ingresso-uscita (**unità di I/O**, *Input/Output*) hanno il compito di gestire il colloquio tra la parte centrale, costituita da memoria e processore, e l'ambiente esterno. Queste unità possono comprendere veri e propri processori specializzati per le funzioni di ingresso-uscita (**canali di I/O**) e per la gestione delle operazioni più comuni:

- lettura/scrittura di caratteri, stringhe, record
- posizionamento
- partenza
- terminazione
- interfacciamento con l'unità centrale.

La comunicazione tra la memoria centrale e le unità di gestione delle periferiche viene realizzata in uno dei seguenti modi:

1. a **controllo di programma:** in questa modalità si può trasferire un dato per volta sotto il controllo del microprocessore;
2. con **accesso diretto alla memoria**, senza l'intervento del processore, tramite opportune unità di controllo **DMAC** (*Direct Memory Access Controller*). In questo caso l'accesso alla memoria tramite il DMAC è prioritario rispetto a quello realizzato con il controllo di programma.
3. con il **bus mastering** in cui i controllori delle periferiche stesse gestiscono la comunicazione.

La figura seguente illustra le caratteristiche di un calcolatore convenzionale con particolare riguardo all'organizzazione delle periferiche.



Le informazioni tra unità centrale (M, P) e periferiche passano attraverso una via comune detta **bus di I/O**. Le unità di I/O comprendono l'interfacciamento fisico verso la periferia e il bus di I/O: esse possono eventualmente possedere la capacità di commutare tra diversi dispositivi dello stesso tipo, come accade nell'interfaccia IDE (di cui si parlerà più avanti) che consente di collegare due dischi fissi.

Le possibilità di gestione delle operazioni di I/O, a controllo di programma oppure con DMA (*Direct Memory Access*), sono importanti nella definizione dell'architettura di un sistema di calcolo.

La prima soluzione a controllo di programma, cioè la soluzione software, comporta l'intervento del processore nelle operazioni di I/O per l'esecuzione dei programmi di controllo. Praticamente si ha il massimo della flessibilità, ma una bassa efficienza a causa del tempo impiegato per eseguire le istruzioni.

La realizzazione hardware è più veloce in quanto la funzione di gestione dell'I/O è assegnata al controllore del dispositivo. Il programma è sostituito dall'algoritmo di controllo dell'unità specializzata. Il DMAC e il microprocessore concorrono per l'accesso alla memoria: pertanto l'accesso deve essere sincronizzato da un arbitro della memoria che assegni la priorità al DMAC.

Per implementare le funzioni di un sistema di elaborazione, oltre alle soluzioni software e hardware, esiste una terza via che si chiama *microprogrammazione* o **firmware**. In questa soluzione il funzionamento dell'unità è demandato a un programma memorizzato in una memoria veloce (memoria di controllo), le cui microistruzioni operano a livello del ciclo elementare di clock della macchina. In questo modo si possono ottenere velocità di poco inferiori alla soluzione hardware e una flessibilità vicina a quella software.

Per esempio, il programma di controllo dei personal computer si chiama BIOS (*Basic Input Output System*), di cui si parlerà in dettaglio nei paragrafi successivi. È realizzato con soluzione firmware, risiede nella ROM-BIOS (una E^2PROM di tipo *Flash*) e usa i dati della CMOS (*Complementary Metal-Oxide Semiconductor*), memoria che contiene i parametri della macchina, per conoscere la configurazione del sistema. Inoltre può essere aggiornato quando viene rilasciata una nuova versione che aggiunge nuove funzioni non supportate dalla versione attualmente installata.

Per esempio, se la scheda madre del computer (**motherboard**) non è in grado di gestire hard disk da 40 GB, basta operare un aggiornamento del BIOS. Conoscendo il codice del BIOS e il codice della scheda madre, si può trovare nel sito Internet del produttore il *firmware* di aggiornamento.

ATTIVITÀ DI VERIFICA

Rispondere ai quesiti di base n. 3, 4 pag. 278

3. Le memorie elettroniche

Nella progettazione di un sistema di memorizzazione occorre trovare una situazione di equilibrio e un compromesso tra esigenze tra loro contrastanti:

- La memoria deve avere capacità illimitata
- La memoria deve essere veloce e quindi con un basso tempo di accesso
- La memoria deve consentire operazioni di lettura e scrittura
- La memoria non deve essere volatile
- La memoria deve avere bassi consumi energetici
- La memoria deve avere costi convenienti.

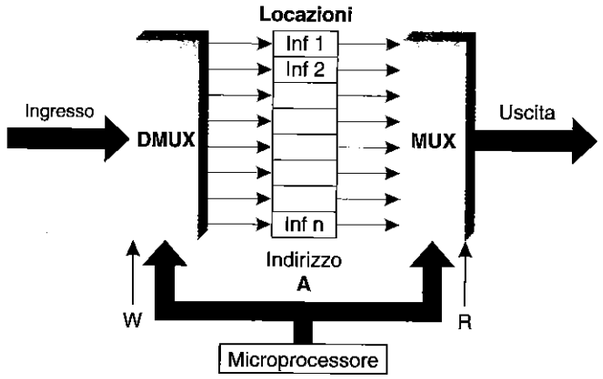
Le memorie elettroniche utilizzate nei sistemi di elaborazione possono essere classificate in questo modo:

- Ad accesso casuale (RAM, *Random Access Memory*) o ad accesso costante
- A sola lettura (ROM, *Read Only Memory*)
- Associative (AM, *Associative Memory*)
- Cache memory.

■ Memoria ad accesso casuale

In una memoria ad accesso casuale, o **memoria RAM** (*Random Access Memory*), una qualsiasi locazione (**cella**) è individuata da un numero (**indirizzo** o *address*) e il suo contenuto può essere letto o modificato in un intervallo di tempo costante detto **tempo di accesso**(t_a)

Lo schema generale di funzionamento è illustrato dalla seguente figura.



Quando viene richiesta un'operazione di lettura con il segnale R, l'indirizzo comanda il *Multiplexer* per passare sull'uscita il dato contenuto nella cella avente quell'indirizzo. Nell'operazione di scrittura il segnale W abilita la scrittura del dato, presente in ingresso, nella cella indicata dall'indirizzo, tramite il *Demultiplexer*.

La RAM è il dispositivo utilizzato dal processore per memorizzare temporaneamente (memoria a breve termine) i programmi, i dati raccolti in ingresso e i dati parziali o definitivi ottenuti durante l'elaborazione del programma.

Dal punto di vista fisico è un componente a semiconduttore realizzato in forma integrata.

La RAM può essere di tipo statico o dinamico. Le RAM *statiche* sono flip flop mentre quelle *dinamiche* sono microcondensatori (C-MOS), nei quali 1 corrisponde al condensatore carico e 0 corrisponde al condensatore scarico.

Nella pratica si usano sempre RAM dinamiche (**DRAM**, *Dynamic RAM*) per l'elevata capacità di integrazione e per i costi più bassi rispetto a quelle statiche: quest'ultime sono più veloci, ma hanno bisogno di dissipare potenza e quindi non possono essere integrate su larghissima scala.

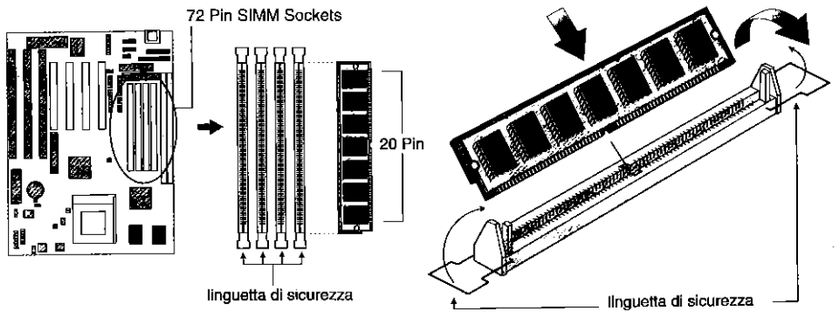
La RAM dinamica, avendo un comportamento elettrico tipico dei condensatori, è soggetta alla scarica, cioè tende a perdere l'informazione in essa contenuta, se non si utilizza una circuiteria per fare l'aggiornamento (**refresh**) della memoria.

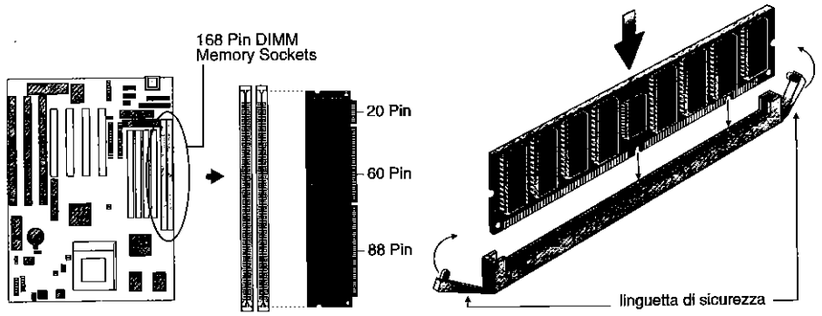
Le memorie utilizzate nei sistemi attuali sono le **DDR2 SDRAM** (*Double Data Rate Synchronous Dynamic RAM*), memorie dinamiche sincronizzate con il bus di sistema con frequenze reali di 400 MHz o 533 MHz, che trasferiscono però dati, sia sul fronte di salita e sia sul fronte di discesa di un impulso di clock: si comportano quindi come se avessero frequenze doppie, rispettivamente 800 MHz e 1066 MHz.

L'ente di standardizzazione dei circuiti integrati, JEDEC, ha approvato le specifiche dei chip e dei banchi (schede) di memoria. I chip sono identificati con la sigla **DDR2**-xxx dove xxx indica la frequenza di trasferimento che è doppia rispetto a quella reale. Per esempio, DDR2-800 vuol dire che lavora con una frequenza reale di 400 MHz, però trasferisce come se avesse una frequenza di 800 MHz.

Le schede di memoria sono identificati con la sigla **PC2**-xxxx dove xxxx indica la banda, espressa in MBps, di trasmissione del banco. Per esempio, PC2-8500 significa che il banco ha una banda di 8,5 GBps.

In precedenza queste memorie venivano realizzate su schede dette **SIMM** (*Serial In-line Memory Module*), più recentemente si usano schede **DIMM** (*Dual In-line Memory Module*).



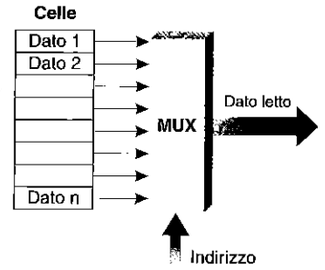


■ Memoria a sola lettura

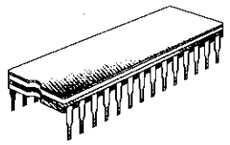
La memoria a sola lettura o **memoria ROM** (*Read Only Memory*) è una memoria in cui l'informazione è registrata permanentemente nel momento della sua costruzione: l'informazione può essere solo letta e non può mai essere modificata.

Le memorie ROM sono usate come parti di memoria di lavoro contenenti informazioni particolarmente protette, come il BIOS (*Basic Input Output System*) del sistema di elaborazione.

Lo schema di funzionamento è mostrato nella figura seguente: come si può notare, la logica di indirizzamento è analoga a quella della RAM, ma con possibilità di sola lettura.

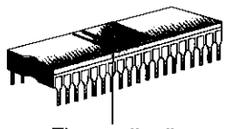


■ PROM



Nelle ROM la programmazione avviene durante il loro processo di costruzione attraverso procedure particolari e costose. Una versione più flessibile, che consente all'utente di programmare autonomamente i contenuti binari, è rappresentato dalle **PROM** (*Programmable ROM*). Le PROM sono quindi memorie ROM programmabili una sola volta.

■ EPROM



Le **EPROM** (*Erasable PROM*) rappresentano una prima soluzione al problema della riprogrammabilità che limita l'uso delle ROM e delle PROM.

In questi dispositivi è prevista la possibilità di rimuovere la programmazione mediante l'esposizione del chip alla radiazione ultravioletta: a tale scopo il chip presenta una piccola finestra di quarzo trasparente alle radiazioni ultraviolette. Il processo di cancellazione prevede l'esposizione del chip alle radiazioni per un periodo di 15-20 minuti.

Gli inconvenienti connessi alla modalità di cancellazione per queste memorie sono:

- Non consente una cancellazione selettiva, tutta la memoria deve essere cancellata.
- Il procedimento di riprogrammazione può essere ripetuto un numero limitato di volte (alcune decine di volte).
- La riprogrammazione avviene *off-line*, cioè occorre togliere il chip dalla scheda madre e inserirlo nel sistema programmatore di EPROM, a radiazione ultravioletta.
- La necessità di proteggere la finestrella da un'esposizione prolungata alla luce solare, il cui contenuto di radiazione ultravioletta potrebbe alterare il contenuto delle celle di memoria. Il chip di memoria viene protetto con una striscia di carta sopra la finestrella.

■ EEPROM

Le memorie **EEPROM** o **E²PROM** (*Electrically Erasable PROM*) sono memorie di tipo programmabile e cancellabile come le EPROM, ma offrono alcuni vantaggi rispetto ad esse:

- I tempi di cancellazione sono nell'ordine delle decine di millisecondi contro i 20-30 minuti richiesti dalle EPROM.
- La cancellazione richiede tensioni molto più basse, rispetto a quelle necessarie per le EPROM, e possono essere fornite dallo stesso alimentatore della scheda madre: quindi il processo di cancellazione può essere realizzato *online*, cioè senza togliere il chip dagli zoccoli della scheda madre. Per questo motivo queste memorie sono spesso chiamate anche *RAM non volatili*. Nei sistemi di elaborazione moderni sono indicate come memorie **Flash ROM programmabili**. Questi tipi di chip consentono all'utente di aggiornare il BIOS del proprio computer con le nuove versioni rilasciate dal produttore.

- La cancellazione può riguardare anche una singola cella: in questo caso le memorie sono indicate con il nome **EAPROM** (*Electrically Alterable PROM*).

■ Sintesi di una rete combinatoria utilizzando una ROM

Con l'introduzione delle tecnologia a larga scala di integrazione, i tradizionali metodi di sintesi delle reti logiche, nelle quali più funzioni logiche sono implementate con le porte logiche AND, OR, NOT, sono sostituiti dall'uso di componenti in larga scala di integrazione come le ROM.

Supponiamo di avere due funzioni combinatorie, Z0 e Z1, aventi come ingressi X0, X1, X2.

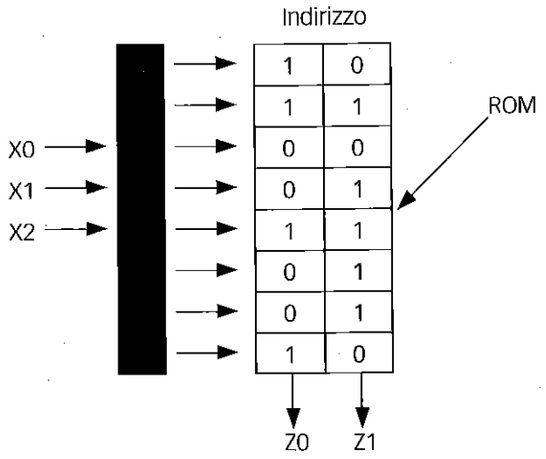
La tabella di verità è la seguente:

Indirizzo	X0	X1	X2	Z0	Z1
0	0	0	0	1	0
1	0	0	1	1	1
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	1	1
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	0

Per sintetizzare una rete logica avente *n* variabili d'ingresso e *m* variabili di uscita utilizziamo una ROM avente tante celle quante sono le configurazioni degli ingressi (2^n): ogni cella ha una lunghezza pari al numero delle uscite.

Nell'esempio precedente serve una ROM di 8 celle, aventi ciascuna una lunghezza di 2 bit.

La configurazione di ingresso serve per selezionare la cella, l'uscita è data dal contenuto letto.

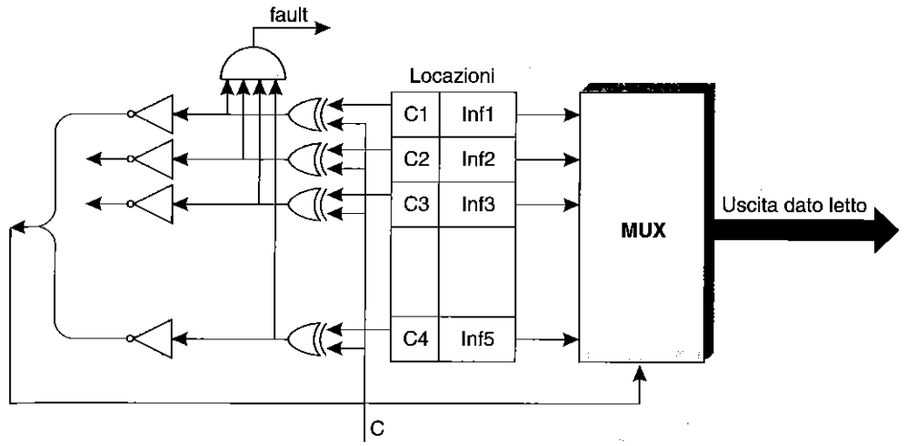


■ Memorie associative

Una **memoria associativa** è essenzialmente una memoria ad accesso casuale che ha la capacità di accedere a una locazione, per le operazioni di lettura e scrittura, non solo in base a un indirizzo, ma anche in base al contenuto della locazione stessa.

Fornendo in ingresso un valore, questo viene confrontato simultaneamente con il contenuto di tutte le celle: il contenuto della cella, che ha dato esito positivo nel confronto, viene letto con un solo tempo di accesso. Se nessuna cella soddisfa il confronto, viene attivato un segnale di esito negativo (*fault*).

Nella figura seguente è mostrato il modello di funzionamento di una memoria associativa a sola lettura relativo a un accesso veloce a una tabella.



Le celle hanno due campi: un campo chiave e un campo informativo. Per esempio, nelle informazioni anagrafiche, la chiave può essere il codice fiscale e il campo informativo è costituito da tutti i dati sulla persona.

La chiave C di ricerca viene confrontata simultaneamente con i campi chiave di tutti le celle tramite l'operazione di XOR. L'operatore **XOR** produce il risultato 0 se i due ingressi sono uguali, produce il risultato 1 se sono diversi.

Se la chiave cercata non esiste in tabella il segnale di fault va alto, perché tutte le uscite degli XOR sono alte: di conseguenza l'uscita della porta AND va alta. La configurazione di uscita che va al commutatore è costituita da tutti 0 e nessuna cella viene selezionata: tutti 0 con nessuna selezione.

Se esiste una cella che ha la chiave uguale a quella cercata, l'uscita di uno degli XOR ha valore 0, quindi il segnale di fault va a 0, mentre la configurazione degli ingressi del commutatore ha un bit 1 in corrispondenza della cella selezionata: in uscita si ottiene il campo informativo corrispondente.

Una cella di memoria associativa, o indirizzata dal contenuto, è quindi un dispositivo in cui alla funzione di memorizzazione sono aggiunte capacità logiche.

Un dispositivo di questo genere è molto importante nelle applicazioni che richiedono la ricerca di un dato all'interno di un vasto insieme di elementi, perché un solo accesso è sufficiente per completare la ricerca. La sua utilità è diventata ancora più importante da quando esistono memorie associative in grado di fare non solo il confronto per uguaglianza, ma anche per maggiore o per minore o in base a maschere (come accade con l'uso dei caratteri *jolly* nella ricerca dei file su disco).

Le memorie associative hanno come svantaggi l'alto costo e la difficoltà a essere integrate su larghissima scala a causa delle capacità logiche di cui sono dotate, che corrispondono a una maggiore potenza da dissipare.

Per questi motivi le applicazioni sono limitate a impieghi particolari, come il riconoscimento ottico di caratteri (OCR, *Optical Character Reader*), l'elaborazione dei segnali video, l'ausilio nell'indirizzamento di grandi capacità di memoria.

■ Memoria cache

La **memoria cache** (letteralmente *memoria ripostiglio*) è una memoria più veloce della RAM, ma con una capacità di memorizzazione molto più bassa. La *cache memory* si trova logicamente situata tra la memoria centrale e il microprocessore.

Nella cache memory viene mantenuta una copia di una parte della memoria centrale, cercando di conservare le informazioni statisticamente più richieste, in modo da sostituire parte degli accessi in memoria centrale con accessi alla cache memory che è più veloce. La gestione della memoria è trasparente all'utente, cioè l'utente opera con i suoi programmi come se la memoria cache non ci fosse: può solo osservare un aumento di velocità nella memoria centrale.

Il funzionamento logico della memoria cache è il seguente:

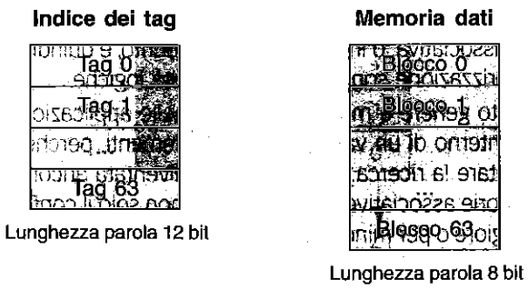
1. Nella lettura di un dato in memoria centrale, se il dato si trova nella memoria cache, la lettura avviene dalla cache, altrimenti la memoria cache copia il blocco di memoria centrale contenente il dato richiesto e lo fornisce all'unità di elaborazione.
2. Nella scrittura di un dato in memoria centrale, il dato viene scritto in memoria centrale e copiato nella memoria cache. La scrittura agisce sempre sia sulla memoria centrale, sia sulla memoria cache: questo meccanismo viene detto di tipo *write through*.

Esempio

Supponiamo di avere una memoria centrale da 64 KB (servono 16 bit per indirizzarla) e una memoria cache di 1 KB (servono 10 bit per indirizzarla). La memoria centrale e la memoria cache sono suddivise in blocchi da 16 byte. Con questo accorgimento, la memoria cache risulta costituita da 64 blocchi, mentre la memoria centrale contiene 4096 blocchi (indirizzabili con 12 bit, $2^{12} = 4096$).

L'indirizzo di una cella a 16 bit è suddiviso in 2 parti: una di 12 bit, detta **tag** o *numero di blocco*, e l'altra di 4 bit, detta **offset** o *distanza*. Per esempio, l'indirizzo di cella 0BA9 è considerato come indirizzo della cella 9 (*offset*) del blocco 0BA (*tag*).

La cache memory è formata da due parti: la prima, detta **indice dei tag**, è composta da 64 celle di 12 bit, la seconda, detta **memoria dati**, è composta da blocchi aventi ciascuno 16 locazioni da 1 byte. In questo esempio, essendo costituita da 64 blocchi, la memoria cache ha la capacità di 1 KB.



Quando si deve leggere un dato, per esempio all'indirizzo FAA5, la parte tag FAA è confrontata con il contenuto dell'indice dei tag e, supponendo che il tag esista nella cella *Tag1*, il dato è individuato nella cella 5 (*offset*) del blocco 1.

Se il tag non si trova nella tabella degli indici dei tag, il blocco di memoria centrale FAA viene copiato nella zona *memoria dati* della cache e il numero del blocco viene inserito nella tabella degli indici dei tag.

Per inserire il nuovo tag e il nuovo blocco nella cache memory, occorre eliminare un tag e il blocco corrispondente. La scelta del tag e del blocco da eliminare può essere dettata da un algoritmo di casualità o da politiche che si possono applicare mediante l'esame di alcune informazioni conservate insieme all'indice dei tag. Si può per esempio togliere il blocco che da più tempo è in cache, oppure il blocco che ha accessi meno frequenti. L'*indice dei tag* è una memoria associativa, mentre la *memoria dati* è una memoria convenzionale con tecnologia più veloce rispetto a quella della memoria centrale.

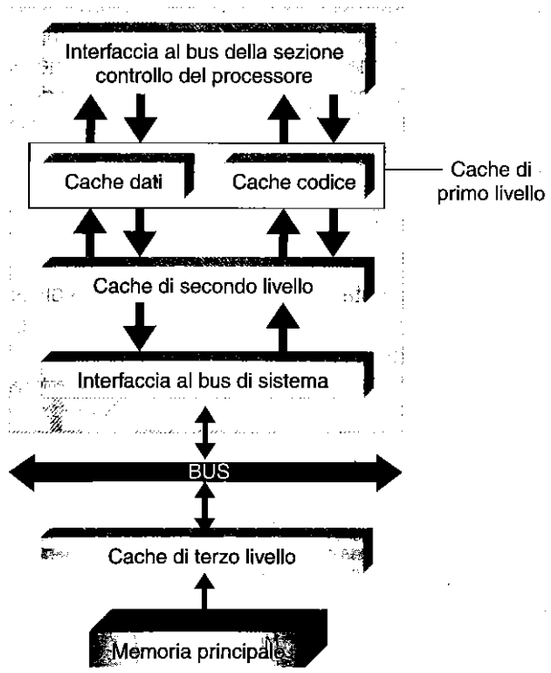
Nei personal computer si possono avere tre **livelli di cache**: due si trovano all'interno del microprocessore, dette *cache di livello 1* (L1) e *cache di livello 2* (L2), e una sulla scheda madre, detta *cache di livello 3* (L3).

Quando si dice che un processore ha un clock di *f* MHz (per esempio 600, 700, 900 MHz oppure 1.5, 1.7 GHz), questo non significa che l'intero sistema di elaborazione funziona ad un ritmo di *f* MHz, ma che solo il microprocessore lavora a questa frequenza; all'esterno invece il lavoro segue la velocità del bus (BIU, *Bus Interface Unit*): valori tipici dei computer attuali sono 133, 266, 400, 533, 1066 MHz, cioè velocità molto più basse rispetto a quella del microprocessore.

Poiché aumentare la velocità del bus non è un'operazione né semplice né economica, per incrementare le prestazioni del sistema conviene far lavorare il microprocessore, in modo da avere una maggiore efficienza, adottando un'architettura che prevede l'uso di cache memory.

Il microprocessore al suo interno lavora alla frequenza *f* con le cache memory e si collega poi al bus con la frequenza del bus.

Riepilogando, si può dire che la memoria cache è piccola, per poter mantenere i costi moderati, ma velocissima. Nei sistemi di elaborazione è usata per contenere dati e codice ai quali il microprocessore accede frequentemente.



Con una **struttura a più livelli** il meccanismo di accesso si modifica nel seguente modo: quando il microprocessore ha bisogno di leggere un'informazione, interroga la cache di primo livello, se la risposta è positiva lo preleva e lo elabora. Se il dato non è presente, lo cerca nella cache di secondo livello: se l'informazione è contenuta nella cache, il dato è trasferito al microprocessore e il blocco contenente il dato viene ricopiato nella cache di primo livello. Se invece il processore non trova l'informazione neppure nella cache di secondo livello, accede alla cache di terzo livello: se il dato si trova nella cache di terzo livello, il dato viene trasferito al microprocessore ed il blocco relativo è copiato nelle cache di livello inferiore. Se il dato non si trova neppure nella cache di terzo livello, esso viene prelevato dalla memoria principale, copiando il blocco contenente il dato in tutte le cache. Statisticamente meno del 2% dei dati non si trova in una delle cache.

■ Shadow RAM

Il **BIOS** (*Basic Input Output System*) è un insieme di routine a livello *firmware* che operano da interfaccia tra l'hardware e il sistema operativo di un computer, per permettere il trasferimento di informazioni tra le diverse unità del sistema.

Malgrado la sua forte influenza sulle prestazioni, il BIOS è invisibile per gli utenti di computer; la sua conoscenza, invece, è di fondamentale importanza per il sistemista. Il BIOS fisicamente risiede in una memoria E^2PROM. Poiché questi dispositivi sono meno veloci della RAM, per migliorare le prestazioni globali del sistema, una parte del BIOS (per esempio il *video BIOS*, cioè le routine che gestiscono l'interfaccia grafica) viene copiata nella memoria RAM: la zona della RAM che contiene le routine del BIOS si chiama **Shadow RAM** (letteralmente *ombra*). A volte si copia in RAM l'intero BIOS, oppure il contenuto delle ROM che si trovano in qualche scheda di interfaccia. Complessivamente si ottiene un sistema più veloce e una capacità di RAM inferiore. L'opzione viene attivata attraverso la routine di **setup del BIOS**: per entrare nella routine di *setup* del BIOS alla fase di bootstrap del sistema, prima di avviare il sistema operativo, si deve premere un tasto predefinito (di solito è il tasto DEL, CANC nelle tastiere italiane).

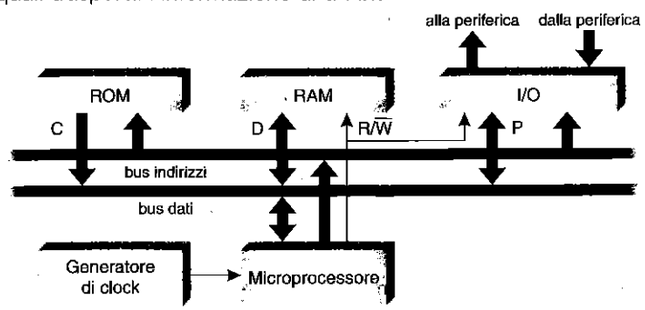
ATTIVITÀ DI VERIFICA

Rispondere ai quesiti di Recall n. 6, 7 pag. 286

Risolvere i problemi di base dal n. 1 al n. 4 pag. 286

4. Bus

La figura seguente rappresenta uno schema di un sistema di calcolo: si può notare come il flusso delle informazioni costituito da dati, codici e indirizzi, viaggi su linee a più fili, ognuno dei quali trasporta l'informazione di un bit.



Queste linee o vie di comunicazione tra il microprocessore e i dispositivi di supporto sono indicate con il termine **bus** (parola che deriva dal latino *omnibus*, che significa *per tutti*).

Si chiama **bus dati** l'insieme delle linee che trasmettono dati, indicati nella figura con D, P, C: D identifica dati da e per la RAM, P identifica dati da e per le unità periferiche, mentre C identifica codici provenienti dalla ROM.

Si chiama **bus indirizzi** o *address bus* l'insieme delle linee che trasmettono gli indirizzi.

A seconda del microprocessore, il bus dati può avere 8, 16, 32, 64, 80 linee (bit). Il numero di bit del *bus dati* rappresenta il **parallelismo** del microprocessore, cioè la massima dimensione degli operandi che è in grado di elaborare con una singola istruzione.

Il *bus indirizzi*, anch'esso dipendente dal microprocessore, rappresenta lo spazio fisico di indirizzamento, inteso come spazio degli indirizzi disponibili. Il *bus indirizzi* di solito ha 16, 20, 24, 32 linee (bit): di conseguenza, lo spazio di indirizzamento del microprocessore è di 64 KB, 1 MB, 16 MB e 4 GB rispettivamente, essendo 2^{10} = 1K; 2^{20} = 1M; 2^{30} = 1G.

Lo spazio di indirizzamento deve essere ripartito tra la memoria RAM, la memoria di tipo ROM e le interfacce di ingresso e uscita.

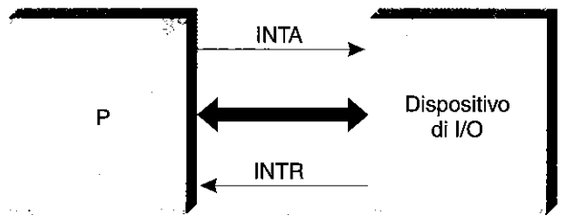
Nella figura precedente si può notare una **linea di controllo**, indicata con R/$\overline{W}$, mediante la quale il microprocessore controlla le due operazioni di lettura (R=1) e di scrittura (W=0): più precisamente, se la linea è alta si ha un'operazione di lettura, mentre se è bassa si ha un'operazione di scrittura, dal o nel dispositivo indirizzato, rispettivamente memoria RAM o interfaccia di Input/Output.

Il soggetto che gestisce il processo di elaborazione è il microprocessore, quindi quando si parla di lettura il dato entra nel microprocessore, mentre quando si parla di scrittura il dato esce dal microprocessore.

In un sistema più completo ci sono altre linee di controllo generate sia dal microprocessore che dai dispositivi stessi. L'insieme delle linee di controllo forma il **bus di controllo**.

Vi sono due linee molto importanti: una che consente alle unità di Input/Output di richiedere al microprocessore l'interruzione dell'attività in corso perché si deve gestire un trasferimento di dati, l'altra, gestita dal microprocessore, serve ad avvisare il dispositivo che il microprocessore è pronto a gestire il trasferimento dati.

La prima linea si chiama **INTR** (*Interrupt Request*, richiesta di interruzione dell'attività del microprocessore), la seconda **INTA** (*Interrupt Acknowledge*, riconoscimento dell'interruzione).



In un sistema ci sono più unità periferiche indipendenti che possono interagire con il microprocessore per stabilire una comunicazione. Questo impone che ogni periferica possa essere identificata in modo diverso per essere univocamente riconosciuta dal microprocessore.

Il microprocessore al termine di ogni istruzione del programma in corso, prima di iniziare la successiva, controlla se qualche periferica abbia richiesto un trasferimento dati.

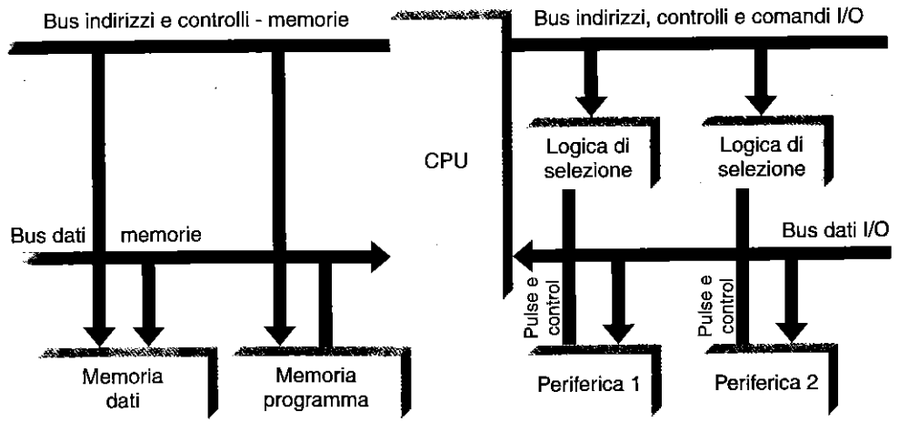
Il protocollo di trasmissione tra una periferica di Input/Output e il microprocessore è descritto dalla seguente sequenza di passi:

1. La periferica quando deve trasmettere/ricevere un dato al/dal microprocessore attiva la linea INTR.
2. Il microprocessore attende di terminare l'istruzione in corso.
3. Il microprocessore attiva la linea INTA.
4. La periferica si identifica trasmettendo il proprio codice di tipo.
5. La periferica trasmette/riceve il dato.
6. Il microprocessore continua il processo interrotto dall'istruzione successiva all'ultima eseguita.

Le informazioni che la CPU scambia con le memorie e i dispositivi di Input/Output sono i dati, indirizzi e segnali di controllo. Ognuna di queste entità è distinta dalle altre o perché trasmesse su linee elettriche dedicate o perché trasmesse su un'unica via e gestite da segnali di controllo che ne specificano il contenuto. Si hanno quindi **bus specializzati** per inviare un unico tipo di informazioni (sono i *bus dati*, *bus indirizzi* e *bus di controllo*) e **bus generalizzati** che in tempi differenti trasportano un tipo diverso di informazione.

■ Bus dedicati all'I/O

Nei sistemi che adottano questo schema si hanno bus dati, bus indirizzi e bus di controllo dedicati alle memorie e bus dati, bus indirizzi e bus di controllo dedicati esclusivamente all'I/O.



Questa architettura ha il vantaggio di avere una logica di selezione esterna più semplice rispetto alla precedente, oltre a poter sovrapporre operazioni di Input/Output con le memorie. Gli spazi di indirizzamento sono distinti per la memoria e le periferiche, però i bus di questo tipo sono molto complicati da realizzare.

Nello schema della figura le periferiche sono considerate come locazioni di memoria. Quando alla ripartizione dello spazio degli indirizzi partecipano anche le periferiche, si parla di memoria **mapped I/O**. In questo caso non sono necessarie particolari istruzioni per l'Input/Output, perché si possono estendere a tutte le periferiche le istruzioni utilizzate per la memoria, comprese quelle di elaborazione. Per il microprocessore non vi è alcuna differenza tra le periferiche e le memorie.

■ Bus con distinti indirizzi di memoria e di I/O

Alcuni microprocessori, come *Intel*, utilizzano una soluzione intermedia alle precedenti perché, pur utilizzando fisicamente gli stessi bus sia per le memorie sia per le periferiche, attraverso una linea di controllo, di solito chiamata **Mem/IO**, sono in grado di individuare, nello spazio degli indirizzi, l'indirizzo che si trova sul bus.

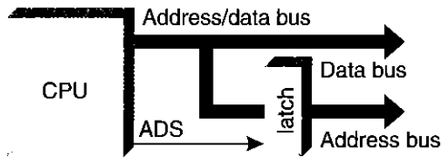
I microprocessori che gestiscono i dispositivi di Input/Output in questo modo utilizzano **I/O isolato**, ossia hanno uno spazio indirizzi per le periferiche e uno per le memorie. In questi sistemi il microprocessore viene dotato di due istruzioni adatte allo scopo, **IN** e **OUT**, meno flessibili di quelle utilizzate dai processori che utilizzano il *memory mapped* I/O perché non hanno proprietà di elaborazione.

■ Bus contenenti indirizzi e dati in tempi differiti

In alcuni microprocessori, al fine di ridurre i piedini del microprocessore, viene utilizzata una parte del bus indirizzi come bus dati. Un opportuno segnale di controllo emesso dal microprocessore consente di memorizzare l'indirizzo su un registro esterno, chiamato **latch** (blocco temporaneo), prima che venga trasmesso il dato.

Il segnale emesso dal microprocessore è un segnale di **strobe**, cioè un segnale impulsivo di memorizzazione, che in questo caso si chiama **ADS** (*address*).

In sostanza, poiché le linee sono le stesse, sia per gli indirizzi che per i dati, e gli indirizzi e il bus dati devono essere disponibili contemporaneamente, si risolve il problema memorizzando l'indirizzo nel registro *latch*, per lasciare libero il bus e avere disponibili, nello stesso momento, sia l'indirizzo sia il bus dati per la ricezione o per la trasmissione del dato stesso.



ATTIVITÀ DI VERIFICA

Rispondere al quesito di base n. 8 pag. 279
Risolvere i problemi di base dal n. 5 al n. 7 pag. 286, 287

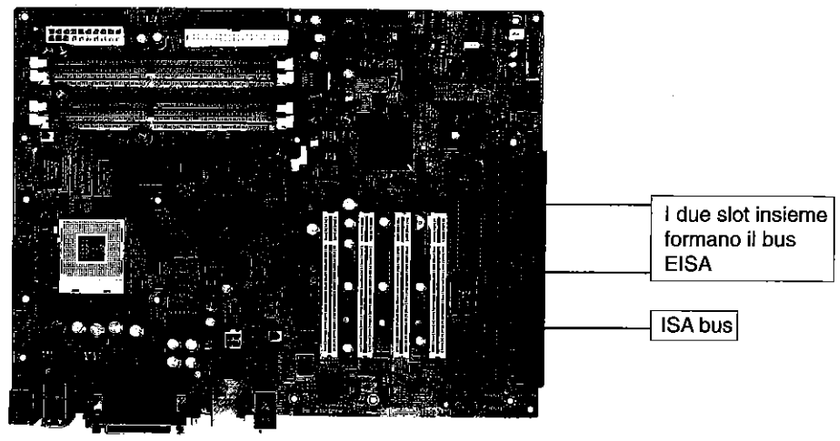
Bus standard e bus ISA **APPROFONDIMENTO**

Gli utenti di un computer hanno spesso la necessità di aggiungere componenti al sistema, per cui i sistemi di elaborazione dispongono di uno o più **slot di espansione** del sistema, che fanno parte del bus e sono fisicamente i connettori nei quali vengono inserite le schede di interfaccia.

Quando tali schede vengono inserite, si collegano elettricamente ai bus e diventano a tutti gli effetti parti del sistema. La velocità con cui le informazioni sono trasferite dipende dalla frequenza di funzionamento del bus, che è il vero sistema di collegamento tra gli elementi del sistema. Se un bus lavora con frequenza di 10 MHz e trasferisce dati a 8 bit, la velocità di trasferimento dei dati nel sistema è V = 8*10 = 80Mbps.

Il primo tipo di bus, utilizzato nei personal computer, fu il bus **ISA** (*Industry Standard Architecture*) in disuso ormai nelle schede madri che montano i processori da *Pentium III* in poi. Introdotto in origine con un bus dati a 8 bit e a 16 bit, fu poi potenziato per includere un collegamento a 32 bit, assumendo il nome **EISA** (*Extended ISA*). Comunemente, con la sigla ISA ci si riferisce agli stessi slot di espansione, a 8 o a 16 o a 32 bit.

Uno slot a 16 bit, in realtà, è costituito da due slot distinti adiacenti, montati in modo da poter accogliere una singola scheda a 16 bit. Anche una scheda a 8 bit può essere inserita in uno slot a 16 bit, ma occupa uno solo dei due connettori allineati. Una scheda a 16 bit, invece, non può essere usata con uno slot a 8 bit. La frequenza di lavoro per questo bus è 8,3 MHz.



ATTIVITÀ DI VERIFICA

Rispondere ai quesiti di approfondimento n. 44, 45 pag. 284

Bus locali e bus VESA — APPROFONDIMENTO

A partire dagli anni '90, da applicazioni basate su un interfaccia a carattere (come nel sistema operativo *DOS*), si è passati ad applicazioni basate su un'interfaccia grafica a causa della grande diffusione del sistema operativo *Windows*.

La quantità di informazioni che si devono trasmettere tra microprocessore, video, memoria e hard disk aumenta notevolmente, se si usa un'interfaccia grafica, in confronto ad applicazioni basate solo su caratteri. Uno schermo standard monocromatico completo (80 colonne per 25 righe) ha una dimensione di 4000 byte di informazioni (2000 per i caratteri e 2000 per gli attributi dello schermo). Un schermo standard completo in modalità grafica (640x480 pixel a 256 colori) richiede più di 300.000 byte, mentre per schermi ad altissima risoluzione (1600x1200) servono 5,8 milioni di byte di informazione.

La trasformazione del software da testo a grafica vuol dire anche un aumento considerevole dei programmi, con conseguente aumento della memoria necessaria sia ad eseguirli, sia a memorizzarli. Pertanto, dal punto di vista dell'I/O, c'era bisogno di una maggiore larghezza di banda per gestire l'incremento notevole di informazioni che si dovevano trasmettere. Il bus ISA aveva sempre la stessa frequenza, per cui era il collo di bottiglia del sistema che impediva l'incremento delle prestazioni del sistema. Queste considerazioni portarono alla creazione di un nuovo bus più veloce per incrementare solamente le prestazioni nel trasferimento di informazioni tra video e processore: il bus locale o **local bus**.

Il primo bus locale fu **VESA** (*Video Electronics Standard Association*), chiamato anche *VL-Bus* o VLB per brevità.

Il VLB è un bus con un parallelismo a 32 bit e con una frequenza di 33,3 MHz, per cui si ha una larghezza di banda pari 127,02 MBps (Megabyte al secondo).

La **larghezza di banda** si calcola con la seguente formula:
Larghezza di banda = Frequenza sul bus * Lunghezza della parola in bit
Per esempio, con una frequenza di 33,3 MHz e una parola di 32 bit, si ottiene
Larghezza di banda = $33,3*10^6*32/8 = 133,2*10^6$ byte
Per esprimere il risultato in Megabyte, si divide per 1.048.586 (1 MB = 2^{20} = 1.048.586).
Si ottiene quindi una velocità di trasferimento di $133,2*10^6/2^{20} = 127,02$ MBps.

L'uso della scheda video su bus VLB aumenta notevolmente le prestazioni del sistema. Uno slot VLB altro non è che uno slot ISA a 16 bit con altri due connettori aggiunti alla fine dello slot ISA. Poiché il bus VLB è un'estensione del bus ISA, una scheda ISA può essere usata in uno slot VLB.

Il bus VLB era molto diffuso nelle schede con processore Intel 80486 e fu poi abbandonato con l'introduzione dei microprocessori della famiglia *Pentium* e del suo nuovo bus PCI, di cui si parlerà più avanti.

Intel abbandonò il VLB per i seguenti motivi:

- VLB era fortemente dipendente dalle schede madri 486 e l'adattamento al *Pentium* causava una perdita di compatibilità.
- Dal punto di vista elettrico VLB era molto delicato: il numero di schede che potevano essere usate erano due e a volte anche una sola.
- Non supportava il *Plug and Play*.

ATTIVITÀ DI VERIFICA

Rispondere ai quesiti di approfondimento n. 46, 47 pag. 284
Risolvere i problemi per l'approfondimento n. 12, 13 pag. 287

5. Plug and Play

I personal computer hanno la caratteristica di essere sistemi aperti: questo significa che gli standard sono di pubblico dominio, per cui qualunque produttore di hardware può realizzare schede per i personal computer.

Per identificare univocamente una scheda all'interno di un personal computer, il microprocessore deve conoscere 3 parametri:

1. L'indirizzo dell'interfaccia o **porto** (*port*), cioè come raggiungere la scheda.
2. Il **numero di interrupt**, cioè quale linea deve utilizzare la periferica per chiedere al microprocessore di trasmettere o ricevere informazioni.
   Si può pensare a un centralino di un ufficio, nel quale ogni impiegato ha il suo numero interno: il numero interno rappresenta il numero di interrupt, mentre il porto rappresenta la posizione fisica dell'ufficio.
3. L'informazione sulla modalità di comunicazione, cioè se si devono leggere o scrivere i dati in memoria passando attraverso il microprocessore o direttamente con la memoria stessa (DMA): nel secondo caso occorre specificare anche il canale, dal momento che ci possono essere più canali collegati direttamente con la memoria.

Le schede utilizzate nei personal computer meno recenti contenevano i **jumper**, cioè piccoli ponticelli per impostare manualmente i parametri. Successivamente furono prodotte schede aventi una E^2PROM con relativo *firmware*, per consentire all'utente di impostare i parametri via software. In entrambi i casi l'impostazione dei parametri era a carico dell'utente, il quale doveva scegliere i tre parametri tra quelli non utilizzati dalle altre schede già installate. Bisognava conoscere bene la configurazione della macchina per evitare conflitti e quindi il blocco del sistema.