Componenti e connessioni

Capitolo 3

Componenti principali

- CPU (Unità Centrale di Elaborazione)
- Memoria
- Sistemi di I/O
- Connessioni tra loro

Architettura di Von Neumann

- Dati e instruzioni in memoria (lettura e scrittura)
- Memoria accessibile per indirizzo
- Esecuzione sequenziale delle istruzioni

Programma “cablato”

- Per eseguire un programma, possiamo costruire i componenti logici in modo che il risultato sia quello voluto
- Questo è un modo di costruire il programma “cablato”, cioè in forma hardware, che non può essere modificato

Programma cablato

- è un sistema non flessibile, che può eseguire solo le operazioni predeterminate
  - Accetta dati e produce risultati
- Con circuiti generici, accetta dati e segnali di controllo che dicono cosa eseguire, e produce risultati
- Per ogni nuovo programma, basta dare i giusti segnali di controllo

Cos'è un programma?

- Una sequenza di passi
- Ad ogni passo, una operazione logica o aritmetica
- Per ogni operazione, un diverso insieme di segnali di controllo

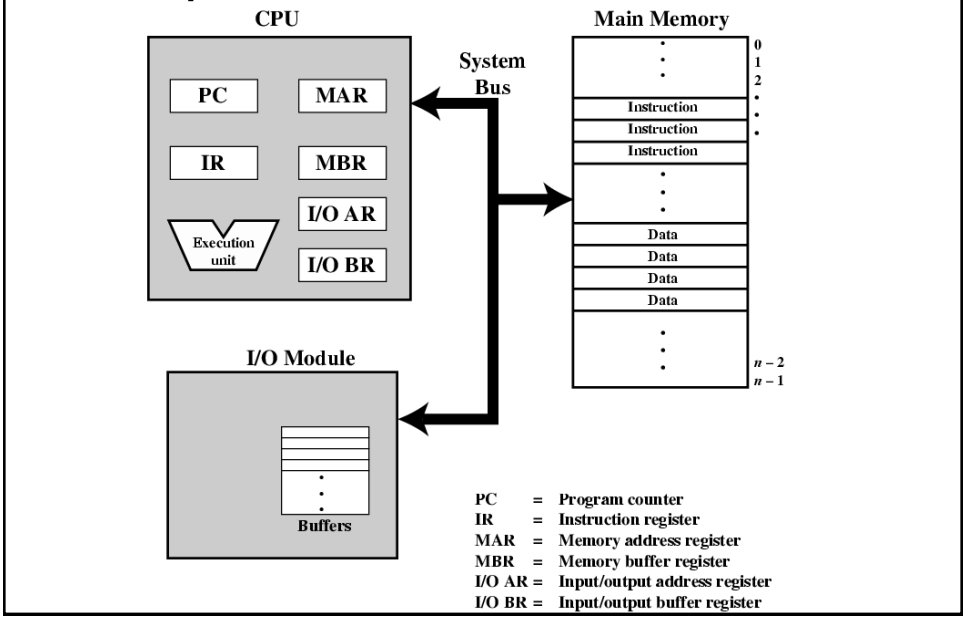
Programmazione software

- Hardware generico più una parte che preleva il codice di una istruzione e genera i segnali di controllo corrispondenti
- Programmazione software
- CPU = interprete delle istruzioni + generico modulo per operazioni aritmetico-logiche

Memoria principale

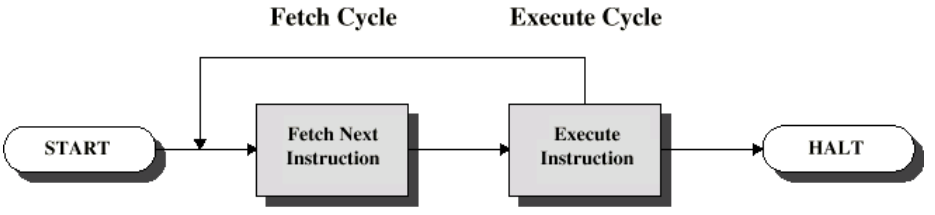
- Possibilità di salti oltre che esecuzione sequenziale
- Operazioni che richiedono accesso a più dati in memoria
- Immagazzinare temporaneamente sia istruzioni che dati

Componenti



Ciclo della CPU per eseguire un programma

- Due passi:
  - Fetch (reperimento dell'istruzione)
  - Execute (esecuzione dell'istruzione)



Fetch e execute

- Registro PC (program counter): indirizzo della cella di M contenente la prossima istruzione
- Prelievo dalla M, poi incremento di PC
- Esempio:
  - parole di M con 16 bit
  - PC contiene 300
  - CPU preleva l'istruzione nella cella 300, poi 301, poi 302, ...
- L'istruzione prelevata viene messa in IR (Instruction Register), poi l'operazione corrispondente viene eseguita

Operazioni di 4 tipi

- Processore-memoria
  - Trasferimento dati tra la CPU e la M
- Processore-I/O
  - Trasferimento dati tra CPU e I/O
- Elaborazione dati
  - Operazione logica o aritmetica sui dati
- Controllo
  - Può alterare la sequenza delle istruzioni
  - Esempio: prelievo istruzione dalla cella 149, che dice che la prossima istruzione è nella cella 182.

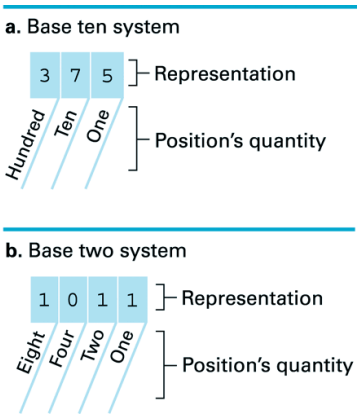
Esempio

- Ipotetica macchina con
  - Registri PC, IR, AC (accumulatore)
- Parole di M di 16 bit
- Dati e istruzioni di 16 bit
- Alcuni codici operativi (4 bit ➔ 16 diversi codici)
  - 0001: carica in AC una cella di M
  - 0010: scrive in M il contenuto di AC
  - 0101: somma una cella di M ad AC
- 2^{12} celle indirizzabili in una istruzione (4096=4K)

Kilo, Mega, Giga, Tera, ...

- Byte = 8 bit
- Kilo, dal greco khiloi (1000 = 10^3)
  - 2^{10} = 1024 = 1K (vicino a 1000)
- Mega, dal greco mega (grande)
  - 1.000.000 = 10^6
  - 2^{20} = 1.048.576
- Giga, dal latino gigas (gigante)
  - 1.000.000.000 = 10^9
  - 2^{30}
- Tera, dal greco tera (mostro)
  - 10^{12}
  - 2^{40}
- Peta, dal greco pente (5)
  - $1000^5 = 10^{15}$
  - 2^{50}

Base 10 e base 2



Rappresentazione decimale e binaria

- Base 10 ➔ cifre da 0 a 9
- Base 2 ➔ cifre 0 e 1
- Sequenza di cifre decimali

$$d_k\ d_{k-1} \ldots d_1\ d_0$$

➔ numero intero

$$d_k \times 10^k + d_{k-1} \times 10^{k-1} + \ldots d_1 \times 10 + d_0$$

- Esempio: 102 in base 10 è 1x100+0x10+2x1
- In generale: $\sum_{(k=n,n-1,\ldots,0)} d_k\ 10^k$

Valore di una rappresentazione binaria

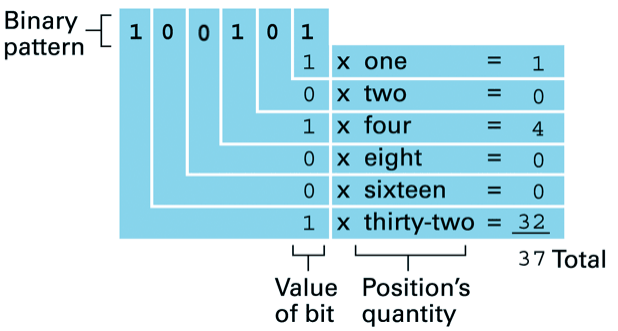
- Per un numero binario $d_k\ d_{k-1} \ldots d_1\ d_0$
- Stesso procedimento ma su base 2:

$$\sum_{(k=n,n-1,\ldots,0)} d_k\ 2^k$$

- Esempio:

$$0101101_2 = 1\cdot2^5 + 1\cdot2^3 + 1\cdot2^2 + 1\cdot2^0$$
$$= 32 + 8 + 4 + 1$$
$$= 45_{10}$$

Valore di una rappresentazione binaria

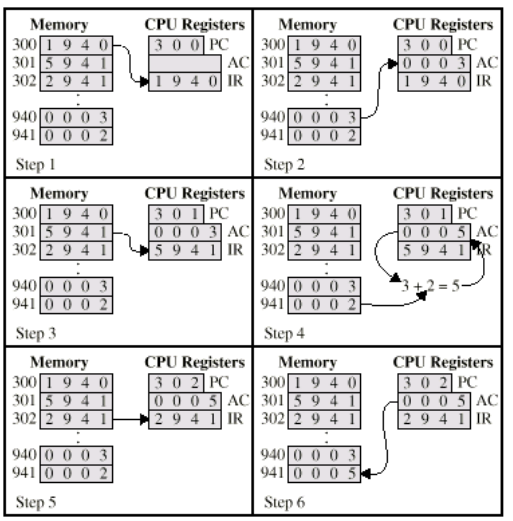


Rappresentazione binaria

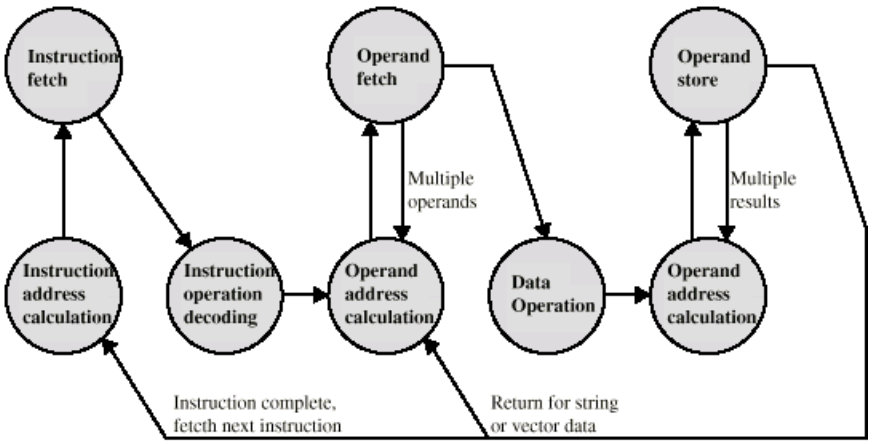
- Valore minimo di una sequenza di n cifre binarie: 000 … 0 (n volte) = 0_{10}
- Valore massimo: 1111…111 (n volte) = $2^{n-1} + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 = 2^n - 1$
- Esempio con n=3: $111 = 2^2 + 2 + 1 = 7 = 2^3 - 1$
- Da 0 a 8: 0, 1, 10, 11, 100, 101, 110, 111, 1000

Esempio

- Somma di cella 940 e 941 e memorizzazione del risultato nella cella 941
- Tre istruzioni
- All'inizio PC contiene 300
- Celle di M in esadecimale



Ciclo di esecuzione



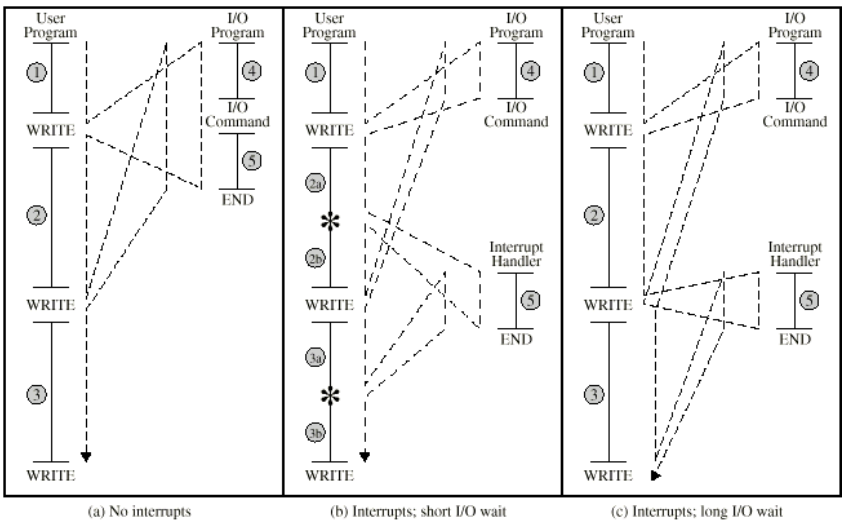
Interruzioni

- Meccanismo tramite il quale altri moduli (esempio I/O) possono interrompere la normale sequenza di esecuzione
- Tipiche interruzioni
  - Program
    - Esempio: overflow, division by zero
  - Timer
    - Generata da un timer interno alla CPU
  - I/O
    - Per segnalare la fine di un'operazione di I/O
  - Guasto hardware
    - Esempio: mancanza di alimentazione

Perché interrompere?

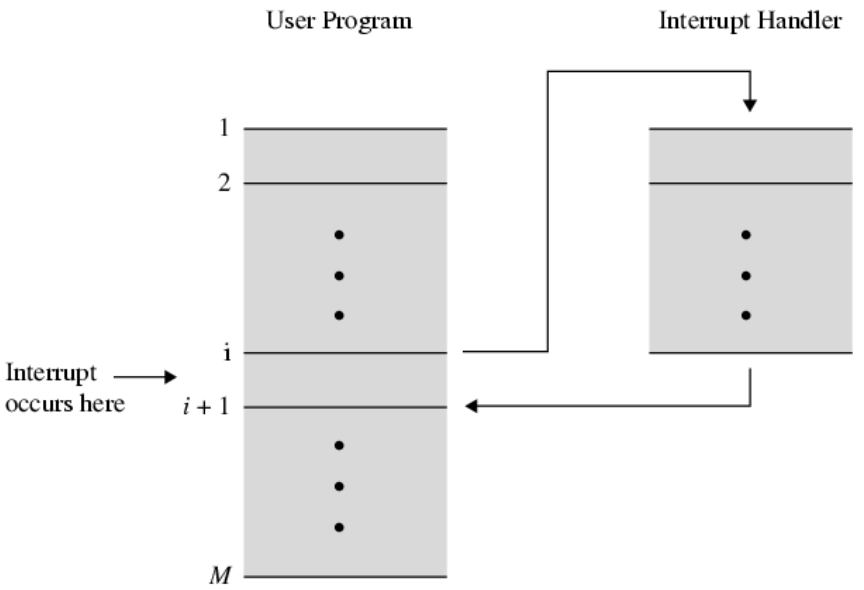
- Per migliorare l'efficienza della elaborazione
- Esempio:
  - Molti dispositivi esterni sono più lenti del processore
  - Per evitare che la CPU attenda la fine di un'operazione di I/O

Esempio



(a) No interrupts (b) Interrupts; short I/O wait (c) Interrupts; long I/O wait

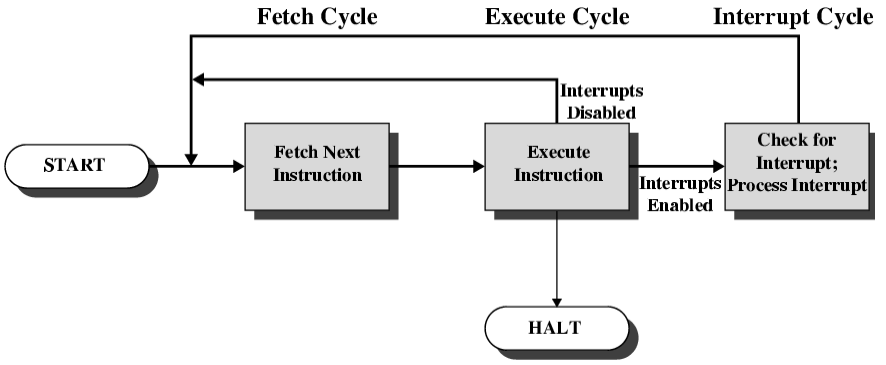
Trasferimento del controllo per una interruzione



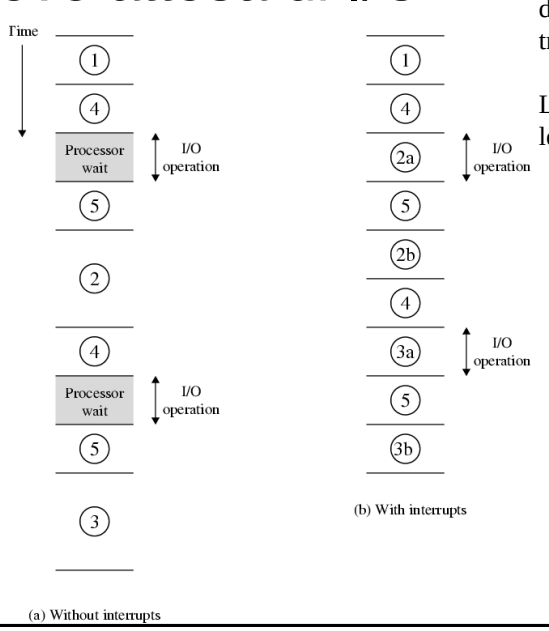
Ciclo di interruzione

- Aggiunto al ciclo di esecuzione
- La CPU controlla se ci sono interruzioni pendenti
- Se no, prende la prossima istruzione
- Se si:
  - Sospende l'esecuzione del programma corrente
  - Salva il contesto (es.: indirizzo prossima istruzione)
  - Imposta il PC all'indirizzo di inizio del programma di gestione dell'interruzione
  - Esegue il programma di gestione dell'interruzione
  - Rimette il contesto al suo posto e continua il programma interrotto

Ciclo di esecuzione con interruzioni



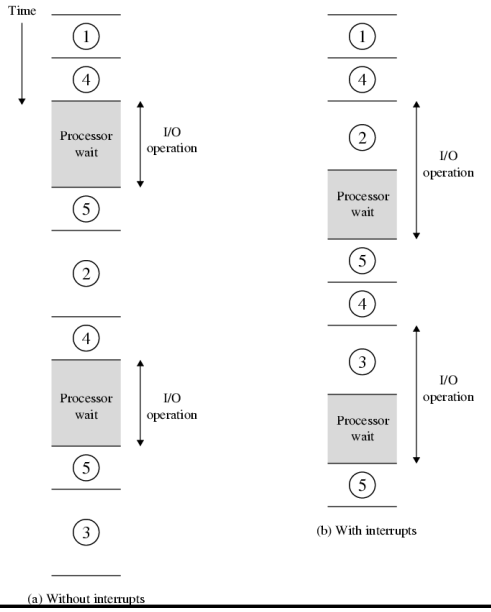
Breve attesa di I/O



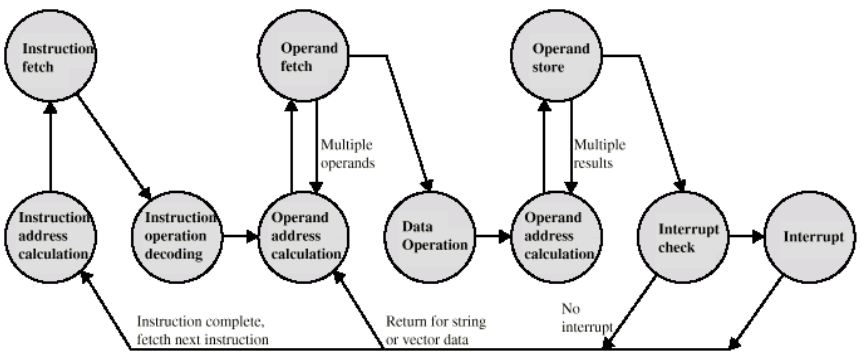
Tempo per operazione di I/O minore del tempo tra due istruzioni WRITE

Le interruzioni eliminano le attese

Lunga attesa di I/O



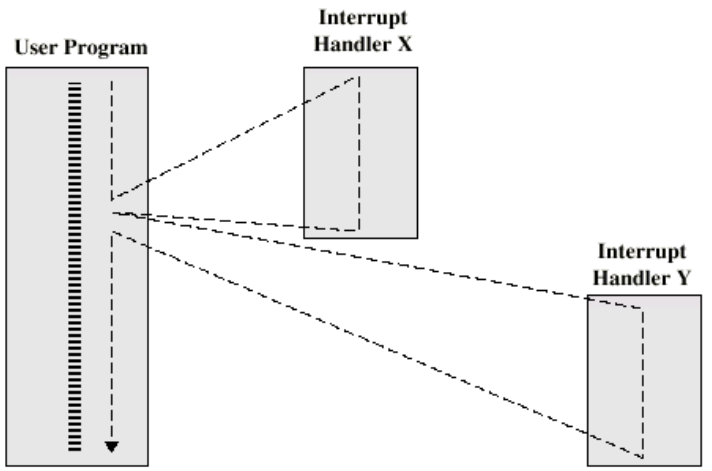
Ciclo con interruzioni



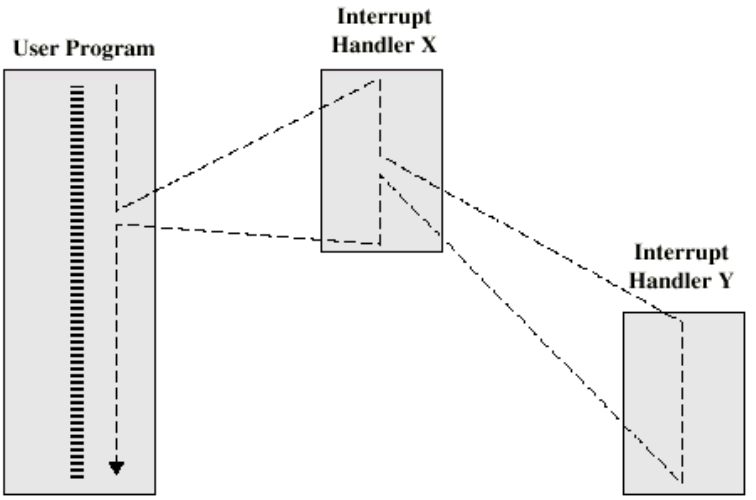
Interruzioni multiple

- Disabilitare le interruzioni
  - La CPU ignorerà altre interruzioni mentre gestisce la prima
  - Le interruzioni rimangono pendenti e sono controllate solo dopo che la prima è stata gestita completamente
  - Interruzioni gestite nella sequenza in cui sono richieste
- Definire delle priorità
  - Interruzioni con bassa priorità possono essere interrotte da interruzioni con priorità più alta
  - Quando l'interruzione con priorità più alta è stata gestita, la CPU ritorna all'interruzione precedente

Interruzioni multiple – sequenziali

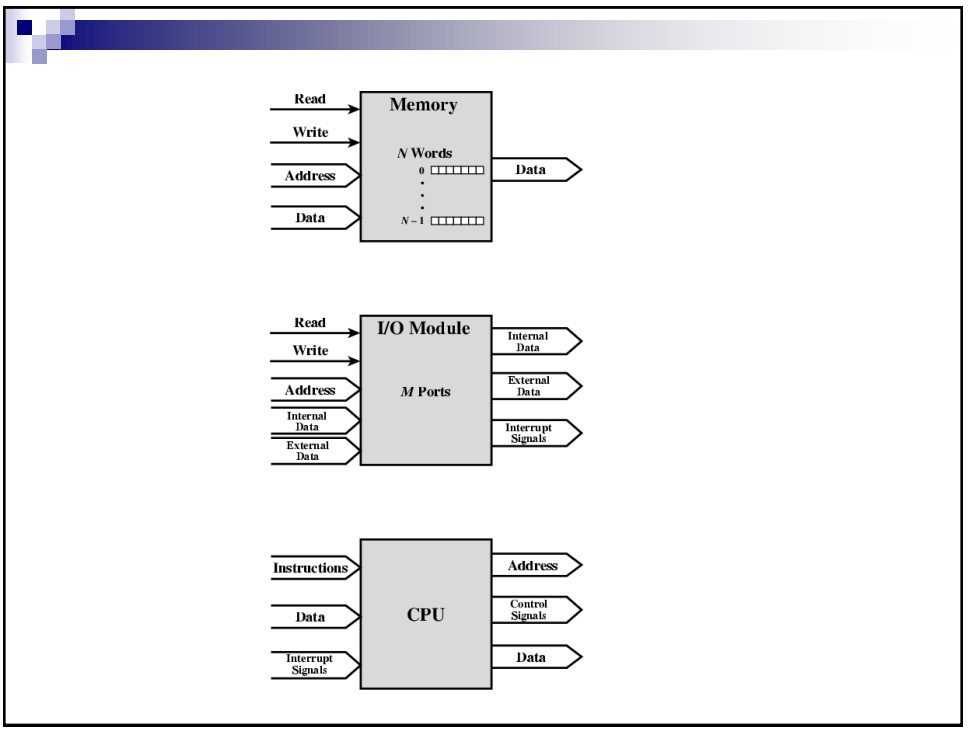


Interruzioni multiple – annidate



Connessioni

- Tutte le componenti di un calcolatore devono essere connesse
- Tipi diversi di connessione per diversi tipi di componente
  - Memoria
  - Input/Output
  - CPU



Connessioni per la memoria

- Riceve e spedisce dati (scrittura e lettura)
- Riceve indirizzi (di locazioni di M)
- Riceve segnali di controllo
  - Lettura
  - Scrittura

Connessioni dell' Input/Output (1)

- Modulo di I/O: simile ad una memoria dal punto di vista della CPU
- Output
  - Riceve dati dalla CPU
  - Manda dati alle periferiche
- Input
  - Riceve dati dalle periferiche
  - Manda dati alla CPU

Connessioni dell'Input/Output (2)

- Riceve segnali di controllo dalla CPU
- Manda segnali di controllo alle periferiche
- Riceve indirizzi dalla CPU (n.ro di porta per identificare una periferica)
- Manda segnali di interruzione

Connessioni per la CPU

- Legge istruzioni e dati
- Scrive dati (dopo l'elaborazione)
- Manda segnali di controllo alle altre unità
- Riceve segnali di interruzione

Connessioni

- Da M a CPU: la CPU legge un'istruzione o un dato dalla M
- Da CPU a M: la CPU scrive un dato in M
- Dall'I/O alla CPU: la CPU legge i dati di una periferica
- Dalla CPU all'I/O: la CPU invia dati ad una periferica
- Dall'I/O alla M o viceversa: accesso diretto alla M da parte di un dispositivo di I/O

Bus

- Collega due o più dispositivi
- Mezzo di trasmissione condiviso
- Un segnale trasmesso da uno dei dispositivi collegati ad un bus è disponibile a tutti gli altri
- Solo un dispositivo alla volta può trasmettere, altrimenti i segnali si sovrappongono
- Più linee di comunicazione, ogni linea trasmette uno 0 o un 1
- Insieme, più linee trasmettono in parallelo numeri binari
  - Esempio: dato da 8 bit tramesso in parallelo da un bus a 8 bit

Bus di sistema

- Connette CPU, I/O, M
- Da 50 a qualche centinaio di linee (ampiezza del bus)
- Tre gruppi di linee
  - Dati: su cui viaggiano i dati (bus dati)
  - Indirizzi
  - Controllo

Bus dati

- Trasporta i dati (o le istruzioni)
- L'ampiezza è importante per l'efficienza del sistema
  - Se poche linee, più accessi in M per prendere un dato

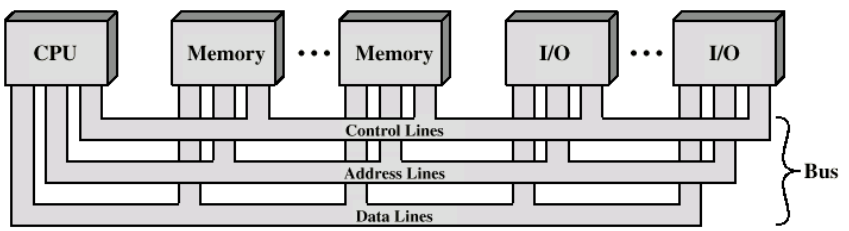
Bus indirizzi

- Indica la sorgente o la destinazione dei dati
  - Es.: la CPU vuole leggere un dato dalla M
- L'ampiezza determina la massima quantità di M indirizzabile

Bus di controllo

- Per controllare accesso e uso delle linee dati e indirizzi
  - M write: scrittura dei dati sul bus alla locazione di M
  - M read: mette sul bus i dati della locazione di M
  - Richiesta bus: un modulo vuole il controllo del bus
  - Bus grant: è stato concesso il controllo ad un modulo
  - Interrupt request: c'è una interruzione pendente
  - Clock: per sincronizzare le operazioni

Schema di interconnessione a bus



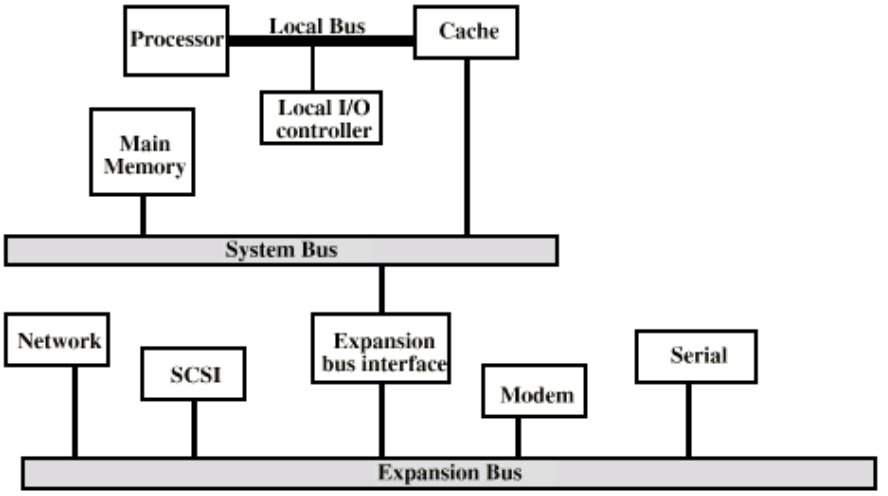
Uso del bus

- Se un modulo vuole inviare dati ad un altro, deve:
  - Ottenere l'uso del bus
  - Trasferire i dati sul bus
- Se un modulo vuole ricevere dati da un altro modulo, deve:
  - Ottenere l'uso del bus
  - Trasferire una richiesta all'altro modulo sulle linee di controllo
  - Attendere l'invio dei dati

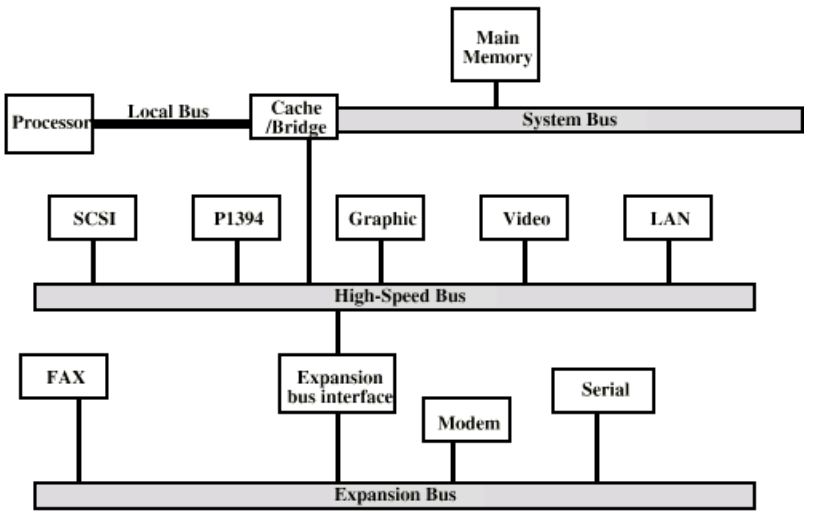
Bus singoli e multipli

- Se un solo bus, possibilità di ritardo e congestione
- Molti sistemi usano più bus per risolvere questi problemi

Bus multipli



Bus multipli



Temporizzazione

- Coordinazione degli eventi su un bus
- Sincrona
  - Eventi determinati da un clock
  - Una linea di clock su cui viene spedita una sequenza alternata di 0 e 1 di uguale durata
  - Una singola sequenza 1-0 è un ciclo di clock
  - Tutti i dispositivi connessi al bus possono leggere la linea di clock
  - Tutti gli eventi partono dall'inizio di un ciclo di clock