Architettura degli elaboratori

Matteo

February 2, 2026

1 Introduction

Un calcolatore è un sistema composto da processsori, memorie, e dispositivi di input/output. un **Bus** è un insieme di connessioni elettriche parallele che trasportano dati da un componente all'altro.

Nell'architettura di Von Neumann viene introdotta l'idea di usare la memoria non solo per i dati ma anche per il programma.

Collega la cpu e la memoria con un bus indirizzi, indicando la posizione dei dati. Mentre su un altro bus dati memoria e cpu si scambiano i dati veri e propri.

CPU composta date

- Unita di controllo: legge e interpreta le istruzioni.
- Unita aritmetico logica (ALU): esegue operazioni aritmetiche e logiche.
- registri: memorie veloci interne alla cpu.

ci sono diversi registri specializzati:

- Program counter (PC): indica la prossima istruzione da eseguire.
- Memory address register (MAR): contiene l'indirizzo di memoria da cui leggere o scrivere dati.
- Memory data register (MDR): registro che accede ai bus dei dati.
- Instruction register (IR): contiene l'istruzione che si sta per eseguire.
- Program status word (PSW): contiene informazioni sullo stato della CPU.

in pratica il ciclo di esecuzione di un'istruzione è:

1. il contenuto di PC viene copiato in MAR e inviato alla memoria attraverso il bus indirizzi.
2. la memoria risponde inviando il dato all'MDR attraverso il bus dati.
3. il contenuto di MDR viene copiato in IR e decodificato.
4. l'istruzione passa alla ALU per essere eseguita.
5. se servono altri dati vengono letti dalla memoria nello stesso modo.

6. se serve il risultato e copiato in memoria attraverso MDR

7. il PC viene aggiornato per puntare alla prossima istruzione.

Questo ciclo e chiamato **fetch-decode-execute cycle**.

data path: insieme di registri e ALU che eseguono operazioni sui dati.

il percorso dei dati da memoria ad ALU, la loro esecuzione e il ritorno in memoria viene chiamato **ciclo di data path**, ed e governato da un clock.

durata ciclo di data path o ciclo di clock= 1/F, dove F e la frequenza di lavoro della CPU(cicli al secondo), misurata in heartz.

la velocita di esecuzione delle istruzioni ISA(instruction set architecture), è misurata con la durata di un ciclo di clock per i cicli necessari.

fissato il ciclo di clock, la velocita puo essere aumentata con il parallelismo.

negli anni 70 si sviluppo la differenza tra CISC e RISC. CISC: complex instruction set computer, set di istruzioni complesso, con istruzioni che fanno operazioni complesse in un singolo ciclo. RISC: reduced instruction set computer, set di istruzioni ridotto, con istruzioni semplici che richiedono piu cicli per operazioni complesse.

Risc usa la microprogrammazione, in modo da far fare diverse operazioni a pochi componenti.

piu cicli FDE possono essere eseguiti insieme con il **pipelining**

si sviluppano anche architetture che permettono il parallelismo. , ad esempio cpu con piu ALU e control units.

oppure molte cpu possono lavorare in modo coordinato sulla stessa istruzione su dati diversi, creando un **array computer**

se invece eseguono istruzioni diverse su dati, con una memoria condivisa, diversi si parla di **multiprocessor systems**.

la forma piu complessa sono i **multicomputer systems**, con piu cpu, ognuna con memoria propria, collegate in rete.

1.1 memoria

vari tipi di memoria

- memoria volatile: perde il contenuto quando viene spenta(es:ram).
- memoria non volatile: mantiene il contenuto quando spenta(es:rom, hard disk).
- memoria online: sempre accessibile
- memoria offline: il supporto deve essere montato

la memoria si organizza in celle, ogni cella è una sequenza di bit con il proprio indirizzo.

cella di 8 bit=byte.

molti calcolatori lavorano su blocchi da 32 o 64 bit, questi blocchi si chiamano word.

le word possono essere memorizzate in celle standard, occupando piu celle consecutive. si puo fare in 2 modi:

- big-endian: byte piu significativo all'indirizzo piu basso.
- little-endian: byte meno significativo all'indirizzo piu basso.

la **cache** è una memoria poco capiente ma veloce, che viene usata per contenere le word piu usate.

la cpu prima cerca i dati nella cache, se non li trova li va a prendere in memoria principale.

principio di località: dati usati recentemente gli uni dagli altri sono spesso in locazoni vicine.

tipi di supporti:

- hard disk: memoria non volatile, lenta, grande capacita, usa piatti magnetici rotanti.(puo essere resa piu veloce con la tecnica RAID, che usa piu dischi in parallelo).
- SSD: memoria non volatile, veloce, grande capacita, usa memoria flash, piu veloce ma meno capiente.
- CD e DVD: memoria non volatile, usano supporti ottici, capacita limitata.

2 porte logiche e circuiti combinatori

Porte logiche sono basate sull'algebra di Boole, un'espressione booleana si costruisce con: le costanti di boole(0,1), gli opeartori booleani, e variabili(x,y, ecc..).

mintermine: un prodotto (AND) di tutte le variabili della funzione, ognuna presente una sola volta, negata o non negata, che vale 1 per una sola combinazione di valori delle variabili.

forma canonica: l'OR di tutti mintermini veri

porta nand è una porta logica utile per costruire circuiti combinatori, falsa solo quando a=1 e b=1. è universale e facile da costruire con transistor

array logici programmmabili: collegando dei fusibili è possibile creare porte logiche personalizzate.

2.1 mappe di karnaugh

mappe di karnaugh: modo di rappresentare funzioni booleane. permettono di costruire un circuito combinatorio minimale.

tabella bidimensionale in cui ogni mintermine ha una cella, e la cella ha un valore, 1 se vale, se 0 se no.

puoi creare raggruppamenti che comprendono gruppi di certi letterali =1.

all'interno di una mappa di karnaugh una copertura si dice minimale quando:

- i suoi raggruppamenti non è contenuto in raggruppamenti piu grandi.
- i suoi raggruppamenti contengono almeno una cella che non appare in altri raggruppamenti della copertura

2.2 bus multi bit

circuiti combinatori collegati a connessioni da piu bit sono detti **bus**

sono praticamente un raggruppamento di segnali e ogni coppia lavora separatamente come in una porta normale, per poi far uscire i seganli uno dopo l'altro.

3 codice binario

codifiche posizionali principali:

- binario
- ottale
- esadecimale

per convertire da bianrio a ottale prendi 3 cifre alla volta e calcola il numero, per esadecimale prendine 4.

i numeri possono anche essere codificati **in eccesso** cioe dato k= bits e bias= 2^{k-1}. somma numero e bias, e codifica regolarmente il risultato.

es: -127 + bias(128) = 1. 1= 00000001

se poi vuoi decodificare, decodifica il binario nel numero e sottrai il bias.

nelle somme tra binari, il riporto finale nel complemento a 1 viene sommato all'inizio, nel complemento a 2 viene scartato.

puo verificarsi **overflow**: se i due addendi hanno segni diversi non accade mai, se hanno lo stesso segno e il risultato ha segno opposto, si verifica overflow.

3.1 codifica a virgola mobile

composto da sue parti: **mantissa**(o frazione): numero, ed **esponente**: potenza

le operazioni su numeri floating point(virgola mobile) possono dare 2 tipi di errore

- overflow: numero in valore assoluto troppo grande
- underflow: numero in valore assoluto troppo piccolo

normalizzare un numero significa scrivere un numero in una forma standard dove la mantissa ha sempre la stessa struttura.

in binario vuoi fare in modo che il numero piu significativo della mantissa(il primo) sia 1.

puoi farlo spostando il primo 1 a sinistra di x posti e poi sottraendo x all'esponente e ricodificandolo.

3.2 standard di codifica

IEEE 754 è lo standard piu usato per i floating point. definisce diversi formati tra cui Binary32 che usa 1 bit di segno, 8 di esponente e 23 di mantissa.

era impossibile codificare caratteri extraeuropei in ASCII quindi fu creato UNICODE, che usava 16 bit. UNICODE pero usa troppi bit per i caratteri piu semplici ed e vicino oramai all'esaurimento, quindi fu creato UTF-8. UTF-8 puo dinamicamente occupare da 1 a 4 byte, I bit iniziali indicano il formato specifico e la quantità di bit utilizzati.

3.3 bit di controllo

i codici binari o in lettura o in scrittura possono essere soggette a errori, per evitare questo esistono i bit di controllo.

esistono sia bit di rilevazione che di correzione.

distanza di Hamming: è il numero di bit di differenza tra due parole. la distanza di hamming di un codice è invece la minima dist. di Hamming tra tutte le parole del codice.

per rilevare d bit errati servono, la distanza di Hamming deve essere uguale a d+1, per correggerli deve essere d+2.

uno dei codici di controllo piu semplici e il **bit di parita**: un bit che fa in modo che il numero di 1 sia pari, la dist. di hamming minima risulta essere 2.

i bit di parita devono sempre rispettare l'equazione m(numero bit)+ r(num bit di contollo)+1 minore o uguale a 2^r e vanno collocati nelle posizioni equivalenti alle potenze di 2 (1,2,4,8,16 ....) e ognuno controlla un sottogruppo di bit.