09. Linguaggio Macchina_Caratteristiche e Funzioni del Set di Istruzioni

Definizione e Elementi Fondamentali

Il **linguaggio macchina** è l'insieme delle istruzioni che la CPU può eseguire direttamente. Ogni istruzione macchina è composta da:

- **Codice operativo**: specifica l'operazione da eseguire
- **Riferimento all'operando sorgente**: indica l'input dell'operazione
- **Riferimento all'operando risultato**: indica dove memorizzare il risultato
- **Riferimento all'istruzione successiva**: determina il flusso di esecuzione

Localizzazione degli Operandi

Gli operandi possono trovarsi in quattro luoghi distinti:

1. **Memoria centrale o virtuale**: richiede la specifica di un indirizzo
2. **Registri della CPU**: identificati tramite un numero univoco
3. **Dato immediato**: contenuto direttamente nell'istruzione
4. **Dispositivi di I/O**: identificati tramite numero di modulo o indirizzo

Rappresentazione delle Istruzioni

Un'istruzione è una sequenza di bit suddivisa in campi. Per praticità si utilizza una rappresentazione simbolica: i codici operativi sono espressi tramite mnemonici (ADD, SUB, LOAD, STOR) e gli operandi tramite simboli (es. ADD A,B significa sommare il contenuto di A a quello di B).

Numero di Indirizzi nelle Istruzioni

Le istruzioni possono contenere da zero a tre indirizzi per operandi/risultati:

- **Tre indirizzi**: formato OP A, B, C dove A ← B OP C. Istruzioni potenti ma lunghe.
- **Due indirizzi**: formato OP A, B dove A ← A OP B. Un operando funge anche da destinazione.
- **Un indirizzo**: usa implicitamente un registro accumulatore (AC ← AC OP A). Tipico dei primi calcolatori.
- **Zero indirizzi**: opera su una pila (stack) con modalità LIFO. Es.: push a, push b, add, pop c.

Il trade-off fondamentale: meno indirizzi significa istruzioni più corte e CPU meno complessa, ma programmi più lunghi con tempi di esecuzione maggiori. Questa distinzione si riflette nell'architettura RISC (istruzioni elementari) versus CISC (istruzioni complesse).

Progettazione del Set di Istruzioni

La progettazione richiede decisioni su cinque aspetti: repertorio delle operazioni disponibili, tipi di dato supportati, formato delle istruzioni (lunghezza, numero indirizzi, dimensione campi), numero di registri indirizzabili, modalità di indirizzamento degli operandi.

Tipi di Operandi

Indirizzi: interi senza segno che indicano locazioni di memoria.

Numeri: comprendono interi in virgola fissa, numeri in virgola mobile, e decimali impaccati (packed BCD) dove ogni cifra decimale occupa 4 bit. Il formato packed BCD è inefficiente (usa solo 10 delle 16 configurazioni possibili) ma evita conversioni nelle operazioni di I/O.

Caratteri: codificati tipicamente in ASCII a 7 bit (128 caratteri), esteso a 8 bit per il controllo di parità nella rilevazione di errori di trasmissione.

Dati logici: sequenze di n bit manipolabili individualmente.

Tipi di Dato x86

L'architettura x86 supporta dimensioni di 8, 16, 32, 64 e 128 bit. L'indirizzamento avviene per unità di 8 bit. I tipi specifici includono: interi con e senza segno, BCD impaccato e non impaccato, puntatori near e far, campi e stringhe di bit, stringhe di byte, formati floating point (half, single, double, extended precision), e tipi SIMD packed per elaborazione parallela.

Tipi di Operazioni

Trasferimento dati: specifica sorgente, destinazione e lunghezza del dato. Può usare codici operativi diversi per trasferimenti diversi (approccio IBM 370) o un unico codice con specifica nell'operando (approccio VAX).

Aritmetiche: somma, sottrazione, moltiplicazione, divisione su interi con segno e spesso su virgola mobile. Includono anche incremento, decremento, negazione e valore assoluto.

Logiche: operazioni bit a bit (AND, OR, NOT, XOR, EQUAL) eseguibili in parallelo su tutti i bit di un registro. L'AND può fungere da maschera per isolare porzioni di dato.

Shift e rotazione: lo shift logico inserisce zeri, lo shift aritmetico preserva il segno, la rotazione fa circolare i bit.

Conversione: trasformazione tra formati di dato.

I/O: comunicazione con dispositivi esterni.

Sistema: operazioni privilegiate per la gestione del sistema.

Trasferimento del controllo: modifica del flusso sequenziale di esecuzione.

Trasferimento del Controllo

Salto condizionato (branch): il salto avviene solo se una condizione è verificata (es. BRZ salta se il risultato è zero, BRE R1, R2, X salta se R1 = R2). Permette l'esecuzione ripetuta di istruzioni e la scelta basata su condizioni.

Salto incondizionato (skip): scavalca l'istruzione successiva senza operandi espliciti (es. ISZ incrementa e salta se zero).

Chiamate di procedura: meccanismo fondamentale per economia di codice e modularità. L'istruzione CALL salva l'indirizzo di ritorno e trasferisce il controllo alla procedura; RETURN ripristina l'esecuzione dal punto di chiamata. L'indirizzo di ritorno può essere salvato in un registro, all'inizio della procedura, o preferibilmente in una pila (stack) che gestisce naturalmente le chiamate annidate secondo l'ordine LIFO.

Passaggio Parametri tramite Stack

Lo stack è il meccanismo più flessibile per il passaggio di parametri alle procedure. Al momento della chiamata, il processore impila sia l'indirizzo di ritorno sia i parametri. L'insieme di questi dati costituisce lo stack frame della procedura.

Linguaggio Assembly

Il linguaggio assembly sostituisce gli indirizzi numerici con simbolici e i codici operativi binari con mnemonici. L'assemblatore è il programma che traduce dal linguaggio assembly al linguaggio macchina.

Ordinamento dei Byte in Memoria (Endianness)

Per dati multibyte esistono due convenzioni di memorizzazione: **Big endian** memorizza il byte più significativo all'indirizzo più basso (ordine di scrittura occidentale); **Little endian** memorizza il byte meno significativo all'indirizzo più basso (ordine delle operazioni aritmetiche). Es.: il valore 12345678hex all'indirizzo 184 viene memorizzato come 12-34-56-78 in big endian e come 78-56-34-12 in little endian.