Linguaggio Macchina

Linguaggio Macchina

- Linguaggio di basso livello composto da numeri o binari comprensibili dalla macchina
- insieme delle istruzioni che la CPU può eseguire
- Difficile da comprendere
- Tutti i programmi o linguaggi di programmazione producono o eseguono programmi in linguaggio macchina prima di essere eseguiti
- I computer comprendo solo dati in binario

Elementi del codice macchina

- Codice Operativo
  - Specifica l'operazione da eseguire
- Riferimento all'operando sorgente
  - Specifica l'operando, ossia l'input dell'operazione
- Riferimento all'operando risultato
  - Dove memorizzare il risultato
- Riferimento all'istruzione successiva

Gli operandi su cui vengo eseguite le istruzioni sono contenuti in vari posti:

- Memoria centrale o virtuale
  - È necessario fornire l'indirizzo della locazione di memoria
- Registri della CPU
  - identificati tramite dei numeri
- Dato immediato
- Dispositivi di I/O
  - È necessario fornire il numero del modulo o l'indirizzo di memoria per I/O

Le istruzioni sono rappresentate come sequenze di 16 bit, suddivise in campi, utilizzando una rappresentazione simbolica (es: ADD, SUB) con dimensione di 4 bit. Anche gli operandi hanno rappresentazioni simboliche, ogni operando è di 6 bit.

Tipi di Istruzioni

- Elaborazione dati
  - Istruzioni aritmetiche e logiche

- Immagazzinamento dei dati in memorie
- Trasferimento dei dati I/O
- Controllo del flusso del programma
    - Salti anche condizionati
      Per ogni istruzione sono necessari 1 o 2 indirizzi per ogni operando, 1 per il risultato, 1 per l'istruzione successiva. Il massimo è quindi 4 indirizzi per istruzione, ma avviene solo in casi rari.
- **1 indirizzo**
    - il secondo è implicito
    - il secondo operando e il risultato si trovano in un registro (accumulatore)
    - l'istruzione contiene 1 campo per l'indirizzo e 1 per il codice operativo
- **0 indirizzi**
    - utilizzato nelle pile (stack)
    - tutti gli indirizzi sono impliciti
    - le operazioni si eseguono implicitamente in base alla cima dello stack
    - es: c = a + b è realizzato con: push a push b add pop c

Meno indirizzi ci sono più le istruzioni sono semplici e quindi la CPU è meno complessa. Però avendo maggior numero di istruzioni l'esecuzione sarà più lunga.

1. **RISC** Reduced Instruction Set Computer, quindi istruzioni più complesse
2. **CISC** Complex Instruction Set Computer,quindi più istruzioni ma semplici

Progettare un insieme di istruzioni

- Repertorio
    - quante e quali istruzioni eseguire
- Tipi di dato
    - su che dati operare
- Formato
    - lunghezza, numero di indirizzi, dimensione campi, ...
- Registri
    - numero dei registri della CPU indirizzabili
- Indirizzamento
    - modo per accedere agli operandi

Tipi di operandi

- Indirizzi (interi senza segno)

- Numeri
  - interi
  - virgola mobile
  - per operazioni di I/O si utilizzano i **decimali impaccati**
    - es: 246 = 0010 0100 0110 più lunga della notazione binaria, ma evita la conversione
- Caratteri
  - codice standard ASCII
  - si aggiunge un bit di controllo errori
  - 7 bit sono per i caratteri alfabetici
- Dati logici
  - formati da n bit, invece che un singolo dato
  - servono per manipolare i dati

Tipi di dati x86

8 (byte), 16 (parola), 32 (doppia parola), 64 (quadword), 128 (quadword doppia)
L'indirizzamento avviene a unità di 8 bit con memoria accessibile al byte. Nella memoria non sono allineati gli indirizzi quindi possono esserci strutture dati di dimensioni diverse nella memoria. Gli indirizzi sono allineati solo per il trasferimento in bus.

Tipi di operazioni

- Trasferimento dati
  - Sorgente: dove è il dato da trasferire
  - Destinazione: dove va messo
  - Lunghezza del dato da trasferire
  - Codici operativi diversi per trasferimenti diversi o stesso codice
- Aritmetiche
  - somma, sottrazione, moltiplicazione, divisione
  - interi con segno
  - incremento, decremento, negazione e valore assoluto
- Logiche
  - operazioni sui bit
  - and, or, not, xor, equal
  - possono essere eseguite in parallelo su tutti i bit di un registro
- Shift
  - spostamento logico/aritmetico verso destra o sinistra

- Rotazione
  - una volta che si shifta alla fine si riparte dall'inizio o dalla fine
- Trasferimento del controllo
  - tramite salto condizionato (branch)
  - tramite salto incondizionato (skip)

Procedura

Pezzo di programma a cui viene attribuito un nome, in modo da poterlo eseguire da qualunque punto del programma, basta chiamare il nome.
In questo modo si risparmia codice, dovendo scrivere solo una volta la funzione. Inoltre posso delegare la scrittura della procedura ad un'altro programmatore.
Per le procedure si usano due istruzioni di salto: la chiamata e il ritorno, che possono essere anche annidate. Per le procedure non annidate, l'indirizzo di ritorno può essere memorizzato in un registro o all'inizio della procedura chiamata. Per le procedure anche annidate si memorizza nella cima della pila.

Conclusione

Assembly è un linguaggio di basso livello destinato a comunicare direttamente con l'hardware del computer. A differenza del linguaggio macchina che usa binari ed esadecimali, assembly cerca di essere comprensibile dall'uomo: i codici operativi sono rappresentati da simboli, gli indirizzi simbolici rappresentano gli operandi e le istruzioni.
Pe traddure dall'assembly al linguaggio macchina si usa un assembler. I byte sono memorizzati in:

- Big endian: i dati vengono archiviati in big end, dal byte più significativo a quello meno

Indirizzo	Valore
184	12
185	34
186	56
187	78

- Little endian: memorizzazione dal byte meno significativo per finire con quello più significativo

Indirizzo	Valore
184	78
185	56
186	34
187	12