

1.1.1 Assembler e linguaggi Macro

1.1.1.1 Assembler

Le istruzioni nei linguaggi assembler ricalcano le istruzioni macchina del calcolatore permettendo di indicare dei nomi simbolici per istruzioni e variabili di memoria, normalmente è prevista la possibilità di utilizzare macro istruzioni, che si espandono in una serie di istruzioni macchina, tipicamente per funzioni di accesso ad archivi, ai servizi del sistema operativo, o macro proprie dell'applicazione.

È probabilmente vero che, a parte i primi calcolatori, ogni calcolatore ha un proprio assembler, talvolta è un linguaggio assembler interpretato (Olivetti TC 800 Audit 5 Audit 6 DE 700), analogo ad un P-code.

1.1.1.2 MIXAL

Assembler 1973

E' il linguaggio assembler per una calcolatore che, nella fondamentale opera di Donald Knuth, "The Art of Computer Programming", è utilizzato per codificare gli algoritmi presentati.

La macchina ha 4000 voci di 5 bytes più segno, 9 registri e due indicatori, uno binario per l'overflow ed uno ternario per il risultato del confronto.

I registri sono due per le operazioni, A e X, sei numerati da 1 a 6, per l'indirizzamento della memoria e J il program counter.

Ogni istruzione occupa una voce ed ha la struttura riportata qui accanto dove AA è l'indirizzo, I è il registro d'indirizzamento

| | | | | | | | |
|---|---|---|---|---|---|-----|----------------|
| 0 | 1 | 2 | 3 | 4 | 5 | OP | ADDRESS, I (F) |
| ± | A | A | I | F | C | STX | b2(1:4) |

F indica quanto della voce indirizzata è oggetto dell'istruzione ed infine C è il codice dell'operazione.

Le istruzioni sono aritmetiche e di spostamento di dati fra memoria e registri e viceversa:

- LDR e LDRN per spostare da memoria a registro *R* dove *R* è il nome di uno dei registri (tranne J).
- STR per spostare in memoria il contenuto di uno dei 9 registri; Z è usato per azzerare.
- ENTR per caricare costanti nei registri

Oltre alle funzioni matematiche elementari su numeri interi, esistono anche operazioni su numeri floating point.

Sono previste istruzioni per leggere e scrivere voci di memoria su diversi dispositivi quali nastri e schede perforate.

Nell'esempio sottostante è realizzato un semplice algoritmo che trasforma il numero di partenza dividendolo per 2 se è pari o moltiplicandolo per 3 e sommando 1 se dispari. L'algoritmo termina quando il numero diventa 1.

```
* Dan's MIX Simulator and MIXAL Compiler
* Erik Schoenfelder algorithm
* (321.a60: oct '90)
bufout EQU 50
printer EQU 18
input EQU 5
*
clrbuf sub clear buffer
      STJ exclb Return address
      ENT1 bufout
      MOVE space(1) spaces
      DEC1
      MOVE bufout(23) rolling
exclb JMP *
start OUT bufout(printer)
      IN N(input)
```

```

        JMP      clrbuf      clear printer buffer
        LDA      N
        CHAR
        STX      bufout(1:4)
loop    ENT6      0          loop counter
        NOP
        LDX      N
        CMPX     one
        JE       OneFind
        INC6     1
        ENTA     0          0 in a register
        DIV      due       oddness test
        JXZ      Even
        LDA      N
        MUL      tre       N = 3*N
        INCX     1         N= N*3 + 1
        STX      N
        JMP      loop
Even    STA      N
        JMP      loop
OneFind NOP
        ENTA     0,6
        CHAR
        STX      b2(1:4)
        OUT      bufout(printer)
Stop   HLT
one    CON      1
due    CON      2
tre    CON      3
N      CON      0
space ALF
        ORIG    bufout
        ALF     ENTER
b2     ALF     NUMB
        ALF     ER
        ORIG    bufout+24
        END     start

```

1.1.1.3 Linguaggi Macro

I linguaggi Macro sono strumenti che estendono le capacità espressive dei linguaggi, tramite costrutti che durante la compilazione sono espansi in istruzioni proprie del linguaggio. Sono principalmente utilizzati per semplificare e rendere più veloce la scrittura di un programma, o per facilitare l'accesso a servizi di applicativi particolari come data base, programmi di grafica, ecc...

Alcuni linguaggi Macro sono strettamente legati a degli assembler di linguaggio macchina, altri sono di impiego più generale. Normalmente non hanno le funzionalità complete di un linguaggio, ma hanno funzioni per trattare le variabili dichiarate nel programma ospite, strutture di controllo e trattamento di variabili.

Sui sistemi IBM Serie 1, macchine indirizzate all'automazione industriale, il linguaggio di programmazione era formato da istruzioni macro.

L'esempio qui riportato è relativo al Macroassembler IBM 370.

```

&LAB   MACRO
        SOMMA  &ADD1, &ADD2      * &ADD1 = &ADD1 + &ADD2
        L      R1, &ADD1         * CARICO &ADD1 IN REG. 1
        A      R1, &ADD2         * SOMMO &ADD2 AL REG. 1
        ST     R1, &ADD1         * DA REG. 1 A &ADD1
        MEND

```

```
.....  
SOMMA TOT, IMPORTO * TOT = TOT + IMPORTO  
.....
```

Si notino i delimitatori MACRO e MEND che racchiudono il *prototipo* della *macro* SOMMA, ed i segnaposto, individuati da & , che sono sostituiti all'atto dell'espansione della macro dai valori reali.