Struttura di programmi MAL
Elementi lessicali

- Ogni microistruzione è definita su una linea di programma
- Le linee vuote vengono ignorate
- Le linee con solo commenti vengono ignorate
- Il carattere di fine linea (RETURN | ENTER | INVIO) ***deve*** terminare una linea

Struttura di programmi MAL
Elementi lessicali

- MAL è una notazione ***case-sensitive*** cioè, lo scrivere in maiuscolo o minuscolo differenzia gli elementi di un programma: **esempio:**
  OR **è una parola riservata (indica la funzione dell'ALU corrispondente) ma si può usare** *or* **(minuscolo) come etichetta.**

Struttura di programmi MAL
Commenti

- I commenti iniziano con due slash “//” e continuano fino alla fine della linea
- Le linee di commento sono ignorate dal micro-assemblatore

Struttura di programmi MAL Direttive

- Si possono impartire delle speciali direttive (comandi) per il micro-assemblatore (sono comandi per il micro-assemblatore e NON determinano la generazione di parole di 36 bit da caricare nel control store). Iniziano con il carattere punto “.” e possono contenere caratteri alfanumerici.

Struttura di programmi MAL
Direttive

- ***.default***

  Dato che non tutte le 512 parole del control store contengono istruzioni si può specificare un'istruzione di default da inserire in ogni locazione non in uso in modo da controllare eventuali errori di esecuzione (vedremo l'esempio d'uso per il codice del micro-interprete). Obbligatoria ed è consigliabile consista di un goto.

  ESEMPIO:
  .default ***goto errorlabel***

Struttura di programmi MAL
Direttive

- ***.label***

Associa ad un'etichetta un indirizzo nel control store in modo da determinare a partire da quale indirizzo caricare le micro-istruzioni associate all'etichetta.

ESEMPIO:
.label ciclo ***0x3A***

Struttura di programmi MAL
Parole riservate

- Alcuni nomi sono riservati, ovvero non possono essere usati come etichette.
- Registri:
  MAR; MDR; PC; MBR; MBRU; SP; LV; CPP; TOS; OPC; H
- Pseudo-registri
  Z; N
- Linee di controllo della memoria
  rd; wr; fetch
- Statement del linguaggio (*nop* è una micro-istruzione che fa nulla e passa alla micro-istruzione successiva)
  if; else; goto; nop
- Operazioni dell'ALU
  AND; OR; NOT

Struttura di programmi MAL
Costanti

- Una sequenza di caratteri composta da cifre (caratteri da “0” a “9”) viene interpretata come una costante numerica in notazione decimale
- Per usare la notazione esadecimale si deve fa precedere la stringa (che ora può comprendere I caratteri da “a” ad “f” maiuscole o minuscole) dalla coppia di caratteri “0x”

  Esempio

  32 denota il valore decimale 32

  0x32 denota il valore decimale 50

Struttura di programmi MAL
Caratteri speciali

- Alcuni caratteri hanno un significato speciale in MAL:

(Servono per le istruzioni di ***if*** e ***goto(MBR)***

) " "

+ Denota l'operazione di somma dell'ALU

- Denota l'operazione di sottrazione dell'ALU

= Indica la copia in uno o più registri del contenuto del bus C

; Separa le varie componenti di una istruzione su una linea

< Denota l'operazione di shift a sinistra dell'ALU (<<8)

> Denota l'operazione di shift a destra dell'ALU (>>1)

Struttura di programmi MAL
Terminazione programmi

Non esiste una (pseudo)microistruzione che ci permetta di fermare l'esecuzione (ad esempio, halt o stop).

Il modo di terminare i programmi MAL sarà l'esecuzione di un ciclo inoperoso del tipo

…..

…..

halt goto halt

Problema numero 1

- Supponiamo che in memoria centrale siano memorizzate due sequenze di K elementi ognuna V1 e V2.
- Vogliamo scrivere in memoria centrale una sequenza di K elementi (V3) ognuno dei quali è la somma del corrispondenti elementi delle due sequenze, cioè il primo elemento di V3 è pari alla somma del primo elemento di V1 con il primo di V2, e così via.

Problema numero 1

- Supponiamo che:
  - SP contenga l'indirizzo di partenza di V1.
  - LV contenga l'indirizzo di partenza di V2.
  - CPP contenga l'indirizzo di partenza di V3.
  - TOS contenga K.

Problema numero 1 soluzione

- Scriviamo la direttiva .default

  .default goto errore

- Decidiamo a quale indirizzo nel control store caricare il nostro programma MAL con una direttiva .label

  .label inizio 0x0

Problema numero 1 soluzione

- Dobbiamo analizzare K elementi di V1 e K elementi di V2 con K memorizzato in TOS. Bisogna controllare il contenuto di TOS diminuirlo di uno ciclando per sommare gli elementi e memorizzarli in V3 fino a quando TOS (cioè K) non si azzera segnalando la fine degli elementi da sommare.

Problema numero 1 soluzione

```
.label inizio 0x0                // definiamo dove caricare il micro-programma
.label errore 0xFF               // definiamo dove caricare la micro-istruzione per l'errore
.default goto errore             // ovunque nel control store carica l'istruzione goto errore

inizio   Z = TOS; if (Z) goto fine; else goto somma  // controllo di TOS (K) se > esegui la somma
somma    MAR = SP; rd            // leggi l'elemento puntato da SP (V1)
         TOS = TOS – 1           // decrementa TOS
         OPC = MDR               // salva in OPC  l'elemento di V1
         MAR = LV; rd            // leggi l'elemento puntato da LV (V2)
         SP = SP + 1             // aumenta il puntatore SP (posizionati per leggere il prossimo in V1)
         H = MDR                 // salva in H l'elemento di V2
         MAR = CPP               // usa CPP per puntare alla parola dove scrivere la somma
         MDR = OPC + H; wr       // effettua la somma e scrivi il risultato (MDR contiene la somma)
         LV = LV + 1             // aumenta il puntatore LV (posizionati per leggere il prossimo in V2)
         CPP = CPP + 1; goto inizio   // aumenta il puntatore al prossimo elemento di V3 e cicla
fine     goto fine               // quando TOS (K) vale 0 si cicla qui indefinitamente
errore   goto errore             // se per caso si salta qui allora si cicla indefinitamente in errore
```