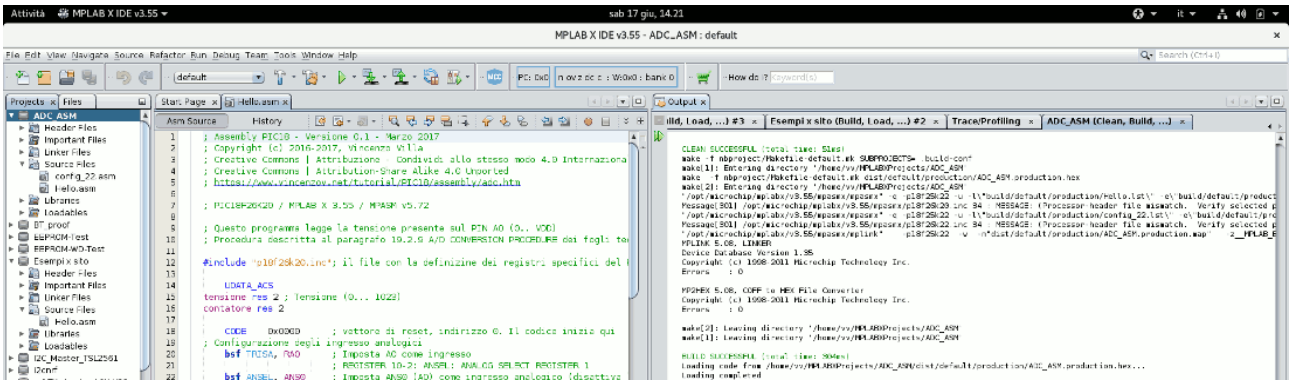
L'assemblatore



In questa pagina e nelle successive verrà mostrato come rendere (un poco) più semplice la lettura e la scrittura del codice in assembly. Per fare ciò utilizzeremo istruzioni e parole chiave dell'assemblatore MPASM, il programma che effettivamente useremo per scrivere il codice.

Un po' di nomenclatura

Il termine *linguaggio macchina* (*Opcode*) si riferisce alle istruzione binarie che un microprocessore/microcontrollore può eseguire. Queste istruzioni sono descritte nei fogli tecnici dell'hardware effettivamente utilizzato.

Il termine *assembly* si riferisce al codice mnemonico (*human readable*) che corrisponde direttamente al codice macchina. Queste istruzioni sono descritte nei fogli tecnici dell'hardware effettivamente utilizzato.

Il termine *assembler* (assemblatore) si riferisce al programma, descritto in questa pagina, che traduce l'assembly in istruzioni macchina. Per estensione il termine si riferisce anche all'insieme delle *direttive* che permettono di controllare e guidare il processo di *traduzione*. Una *direttiva* non corrisponde ad alcuna istruzione macchina. Le direttive sono descritte nel documento *MPASM™ Assembler User's Guide* fornito con MPLABX o disponibile sul sito Microchip.

Alcune direttive le abbiamo già viste:

- `CODE` specifica l'indirizzo di memoria in cui iniziare a salvare le istruzioni macchina
- `END` specifica la fine del testo da assemblare

L'insieme di codici mnemonici e direttive che costituiscono il programma scritto dal programmatore viene spesso indicato come *codice sorgente*.

Le direttive dell'assemblatore

Il manuale dell'assemblatore MPASM è installato insieme a MPLAB e ad esso si rimanda per una discussione approfondita delle tante direttive. Qui è riportata una breve descrizione di quelle di uso più frequente

Le label

Una *label* (etichetta) è usata per identificare con un nome significativo una particolare cella di memoria contenente dati oppure istruzioni. L'uso di label evita di dover utilizzare indirizzi numerici, difficili da calcolare, impossibili da ricordare e frequente causa di errori o difficoltà di interpretazione.

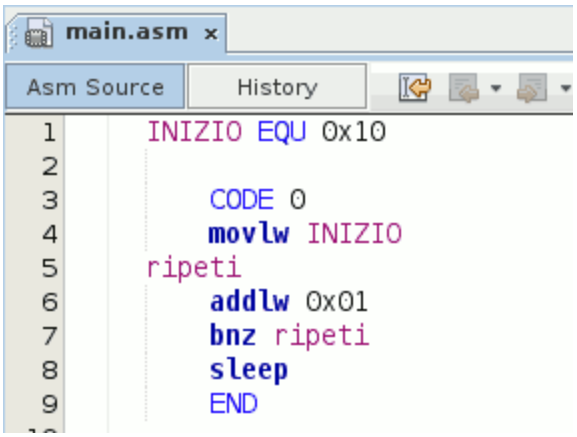
Una label è una stringa alfanumerica scelta dal programmatore e scritta a partire dalla prima colonna del codice sorgente.

Qualche esempio di uso delle label:

- Nell'uso del *File Register* identifica una *variabile*, cioè una cella specifica senza indicare l'indirizzo a cui su trova.
- Nei salti, condizionati o non, identifica l'istruzione a cui saltare. Per esempio un salto condizionato è decisamente più leggibile se scritto usando una label

```
   CODE 0
   movlw 0x55
ripeti
   addlw 0x01
   bnz ripeti
   sleep
   END
```

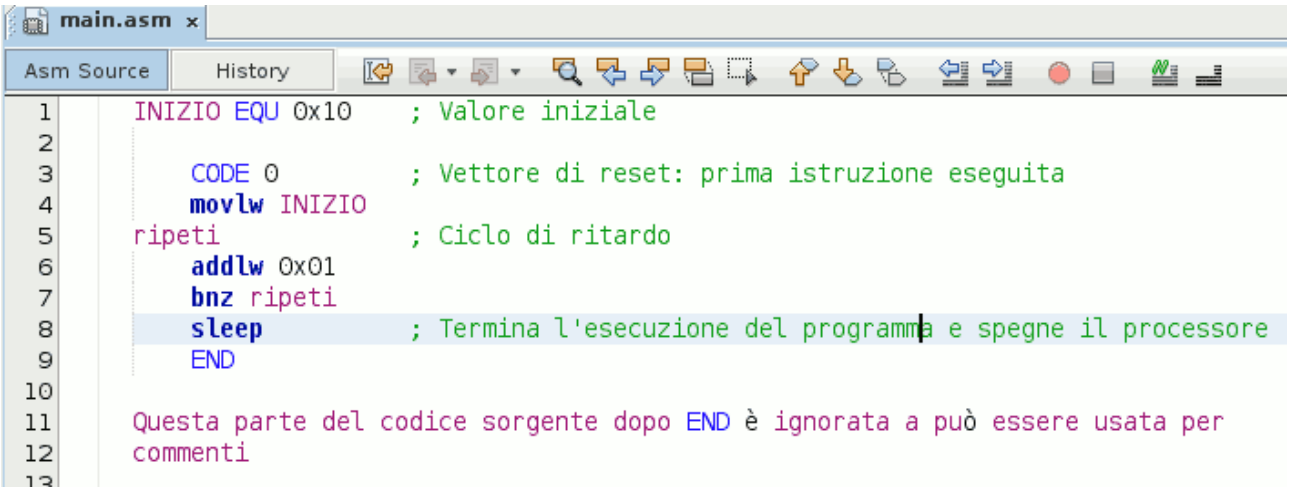
- Spesso è scomodo e fonte di errori specificare numeri all'interno del codice. La direttiva `EQU` permette di assegnare ad una label un valore costante. Per esempio l'istruzione `movlw INIZIO` mostrata ha esattamente lo stesso effetto di `movlw 0x10`, ma potrebbe rendere il codice più semplice da capire e modificare.



Commenti

Tutto ciò che segue un `;` è ignorato dall'assemblatore e serve esclusivamente agli umani per comprendere il codice. Se un codice è privo commenti non può essere ritenuto un codice "adeguato" in quanto rende complicato, se non impossibile, il suo uso, la sua comprensione, l'individuazione degli errori...

Un caso particolare è costituito da tutto quanto segue la direttiva `END`, semplicemente ignorato dall'assemblatore.



Base numerica

Ovviamente il codice all'interno del microcontrollore è esclusivamente in binario.

Al solo uso degli umani, durante lo sviluppo o l'analisi del codice, potrebbe essere più comodo vedere o scrivere le variabili interne in decimale oppure esadecimale attraverso le direttive `RADIX DEC` oppure `RADIX HEX` (predefinita) oppure `RADIX BIN`.

In genere è meglio specificare esplicitamente la base ogni volta, soprattutto se diversa da 10. Per fare ciò è per esempio possibile scrivere:

- Per la base 2: `B'10000000'`
- Per la base 10: `D'128'`
- Sempre per la base 10: `.128`
- Per la base 16; `H'80'`
- Sempre per la base 16: `0x80`

Tutte queste cinque forme sono esattamente equivalenti.

Tutte le lettere utilizzate nella rappresentazione di numeri possono indifferentemente essere maiuscole o minuscole, ma è bene scegliere un proprio stile.

Nomi dei registri

A volte è utile includere alcuni parti di codice all'interno di un file. Uno dei modi è quello di scrivere questi blocchi all'interno di un file di testo (in genere con estensione `.inc`) da includere nel file sorgente con la direttiva `#include`.

Per esempio i nomi dei registri non sono conosciuti dall'assemblatore, ma disponibili in un file separato, che dipende dal modello specifico di PIC in uso. Per includere tale file:

```
#include "p18f26k22.inc"
```