

Assembler per PIC

tuttavia tener presente che, diversamente da quanto succede in un'operazione aritmetica, il "riporto" non passa da un bit all'altro, perché non bisogna dimenticare che anche se si lavora con valori da 8 bit l'operazione è realizzata bit a bit.

Le istruzioni di somma esclusiva funzionano come quelle di somma e moltiplicazione logica, anche se seguono una particolare tabella di funzionamento.

Operazioni di negazione

Fra le operazioni logiche tipiche, non poteva mancare quella che cambia gli 1 con 0 e viceversa. È l'operazione di negazione, o complemento (a 1).

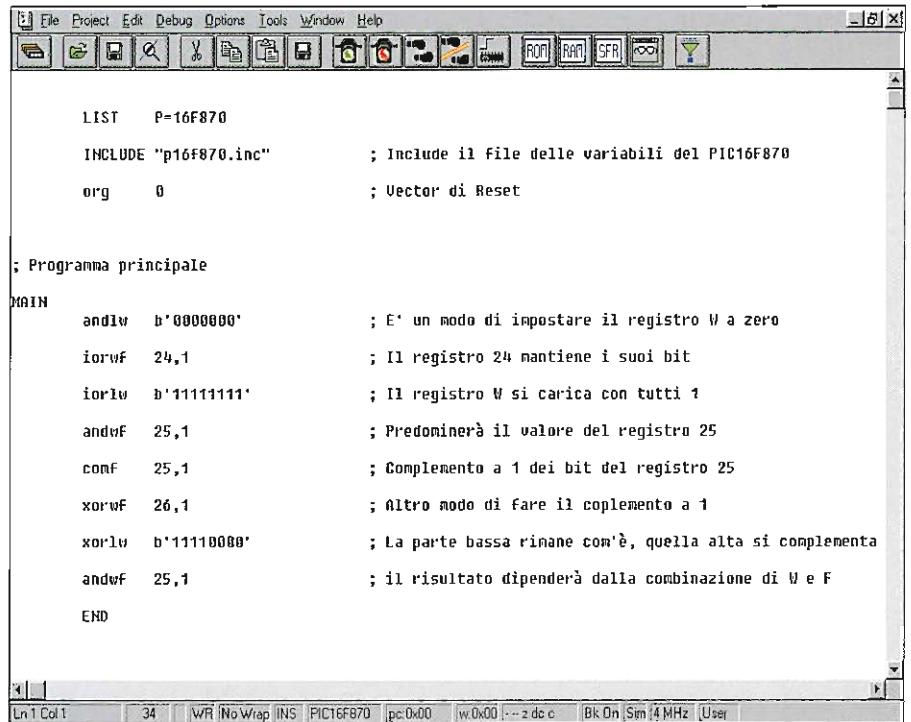
L'unica variante prevede l'intervento di un registro della memoria RAM, in questo caso il risultato dell'operazione può essere il registro medesimo oppure il registro di lavoro W.

L'unico bit del registro STATUS coinvolto in tutte queste istruzioni è Z, che verrà impostato a 1 quando il risultato dell'intero byte sarà 0, cioè otto 0.

Prova delle espressioni in MPLAB

Proveremo ora un programma che riunisce tutte le istruzioni logiche.

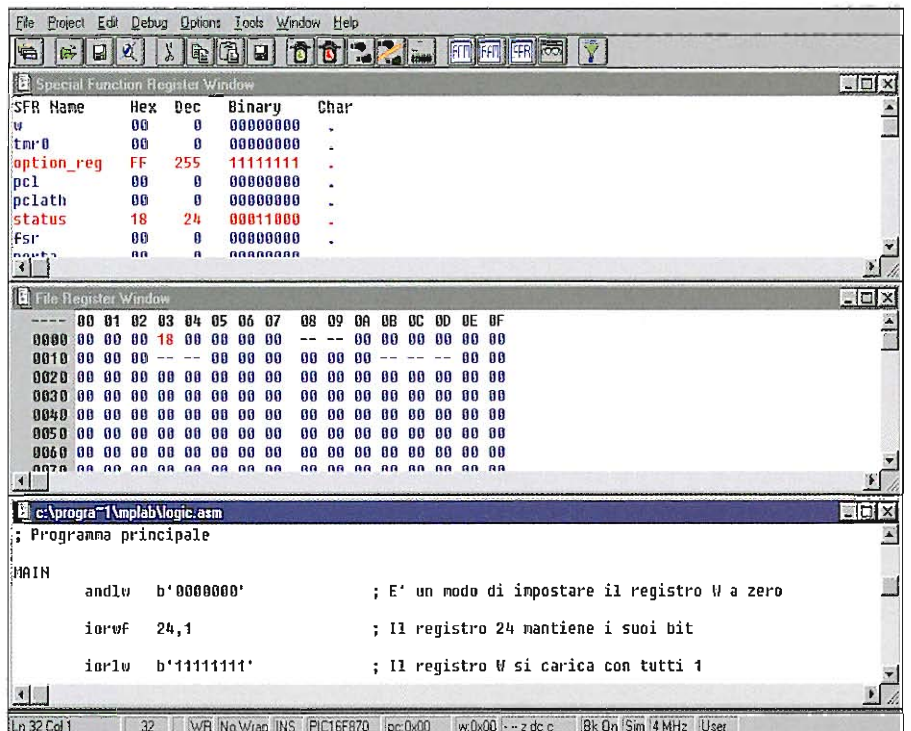
Copiate il programma, assegnatelo a un progetto e compilatelo. Ricordate che il modo di funzionamento dovrà essere per



```
LIST    P=16F870
INCLUDE "p16f870.inc"      ; Include il file delle variabili del PIC16F870
org     0                  ; Vector di Reset

; Programma principale
MAIN
    andlw b'0000000'      ; E' un modo di impostare il registro W a zero
    iorwf 24,1            ; Il registro 24 mantiene i suoi bit
    iorlw b'11111111'    ; Il registro W si carica con tutti 1
    andwf 25,1            ; Predominerà il valore del registro 25
    conf  25,1            ; Complemento a 1 dei bit del registro 25
    xorwf 26,1            ; Altro modo di fare il complemento a 1
    xorlw b'11110000'    ; La parte bassa rimane com'è, quella alta si complementa
    andwf 25,1            ; il risultato dipenderà dalla combinazione di W e F
END
```

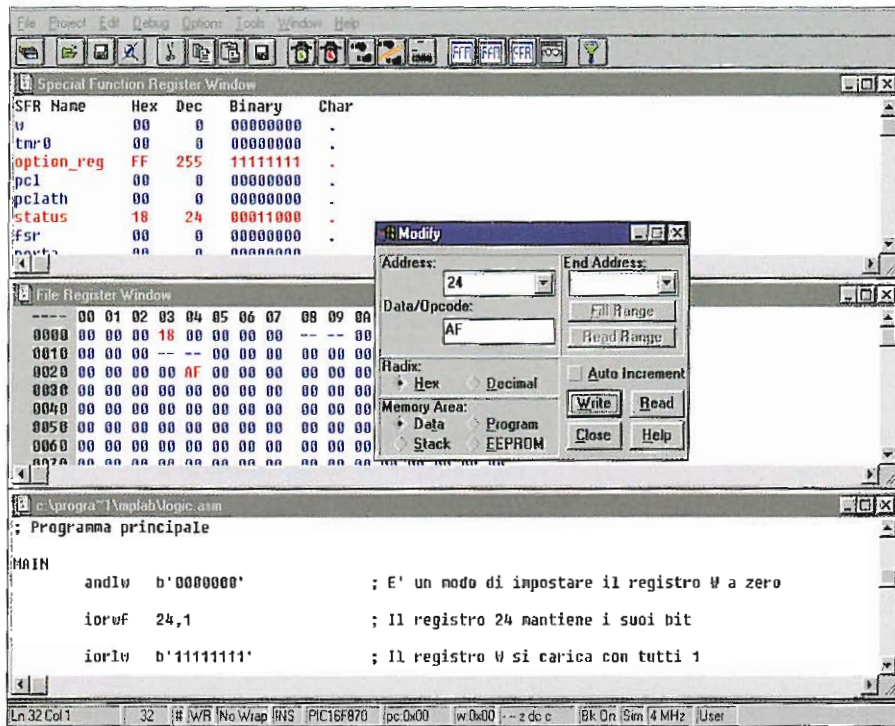
Programma per provare le istruzioni viste.



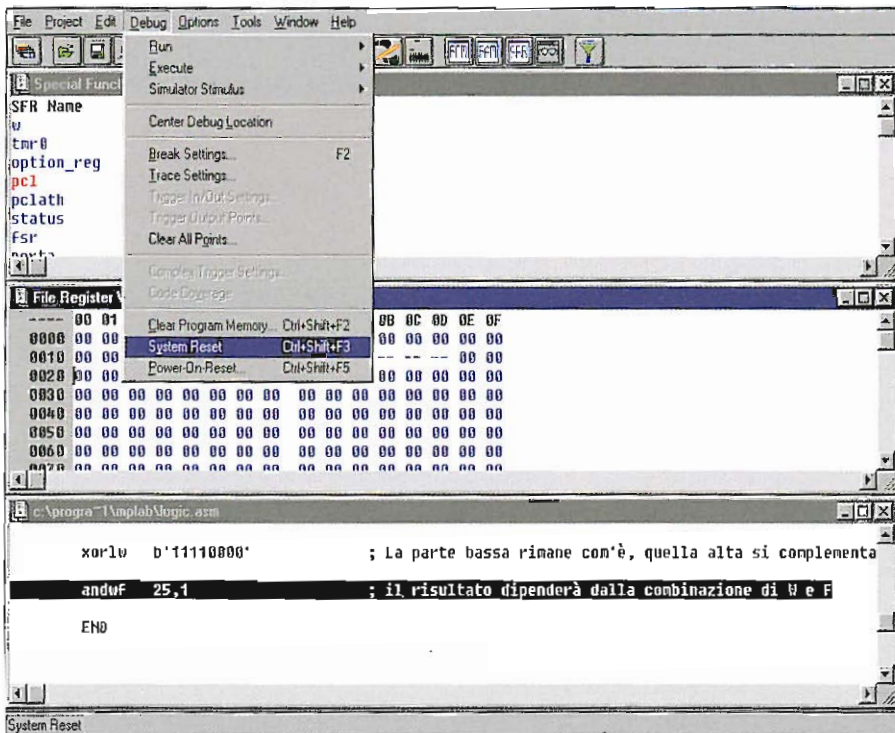
The screenshot shows three windows in MPLAB IDE:

- Special Function Register Window:** A table showing the state of various registers. The **option_reg** register is highlighted in red, showing its value as 0xFF (255) in decimal and 11111111 in binary.
- File Register Window:** A memory dump showing the contents of memory locations from 0000 to 0070. The value at address 0018 is 18, which corresponds to the STATUS register.
- Assembly Code Window:** Shows the same assembly code as the first screenshot, but only the first few lines are visible.

Finestre da aprire per la simulazione.



Modifica del valore di un registro.



Opzione di reset del sistema.

il PIC16F870 in modo simulazione. A partire da questo momento sarà sempre così, a meno che non vengano date specifiche indicazioni diverse.

Aperte le finestre necessarie per questa simulazione. Prima di iniziare la simulazione, modifichiamo i valori dei registri che sono implicati nella simulazione stessa, per fare in modo che si noti bene la variazione. Questa modifica si realizza tramite Window → Modify. Si può impostare l'indirizzo del registro da modificare direttamente, o sceglierne uno fra quelli presentati nella lista del menù a tendina.

In questo caso modificheremo la memoria dei dati RAM, però si potrebbe modificare in egual modo quella EEPROM dei dati, la memoria di programma e anche lo STACK. Dopo aver impostato il valore con

il comando "Write" rimarrà modificato l'indirizzo scelto, e apparirà con caratteri di colore rosso. Modificate gli indirizzi 24, 25 e 26; i valori da inserire sono indifferenti. Premete ora il tasto F7, per realizzare una simulazione passo a passo.

Prima di ogni attivazione di F7, guardate il valore del registro di lavoro W, dato che è coinvolto in tutte le operazioni.

Potete anche verificare che dei tre bit che abbiamo già visto del registro STATUS, solo il valore del flag Z verrà modificato da alcune istruzioni, quelle che determineranno il risultato zero. Se si desidera ricominciare la simulazione bisognerà prima resettare il sistema.