altre istruzioni ausiliarie per fare in modo che il programma funzioni. La figura successiva mostra il programma completo. In questo modo possiamo compilarlo. L'importanza di queste due istruzioni ausiliarie si può vedere anche dal fatto che, eseguendo l'operazione inversa, cioè togliendo le due istruzioni mostrate all'inizio, il programma sarebbe comunque valido anche se non fa nulla.

Istruzioni ausiliarie

Abbiamo già utilizzato in altre occasioni le istruzioni ausiliarie, anche se con parametri diversi. Con "DEVICE" specifichiamo il tipo di PIC a cui è diretto il programma, dato che i registri, le posizioni di memoria, ecc. variano da uno all'altro e il compilatore deve tenere conto di tutto questo per realizzare il programma eseguibile; poi bisogna includere le variabili e le routines di gestione del display LCD, operazione che si esegue tramite il comando "INCLUDE". In mancanza di queste, il compilatore non capisce le istruzioni come "cls" o "print" e produce un errore.

Non è sufficiente includere il pacchetto di lavoro con LCD, ma bisogna anche inizializzarlo. Questo lo realizza "INIT"; il primo parametro di "INIT" è la periferica utilizzata, nel nostro caso il display LCD, e il secondo varia in funzione del primo.

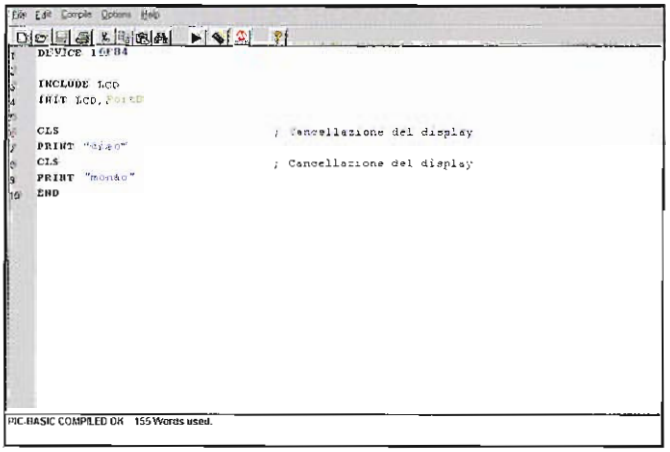
Per il display LCD i parametri possibili sono "portb" o "portc". Questo fa riferimento alle porte di ingresso/uscita del PIC dove deve essere collegato il display. In entrambi i casi le linee utilizzate sono fisse e uguali.

Le linee dei dati saranno quelle comprese fra RB4 e RB7 (RC4 e RC7), e quelle di controllo RB2 e RB3 (RC2 e RC3), sempre nell'ordine indicato nella figura. Questa è la configurazione che si realizza per lavorare con le istruzioni che fornisce il LetPicBasic. Nel caso di utilizzo di altri linguaggi di programmazione per il PIC bisognerà adattarsi alle istruzioni del linguaggio stesso al momento di collegare il display.

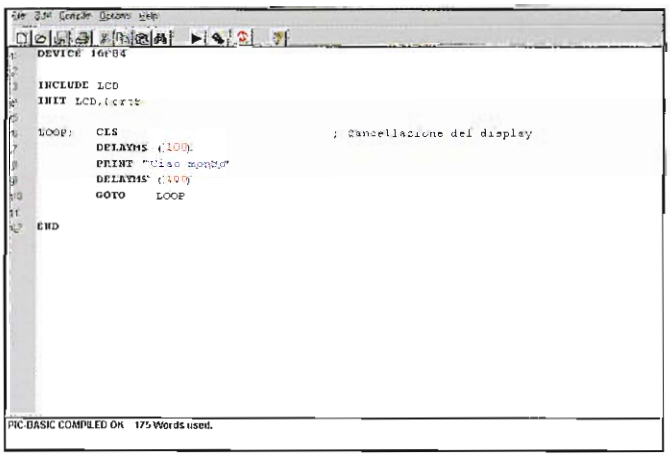
Inizio di scrittura

La prima istruzione utilizzata come parte del programma — cioè oltre ai comandi necessari — è la "cls". Questa istruzione di solito è la prima perché il suo compito è cancellare il display LCD e posizionare il cursore nella posizione denominata "home", in altre parole nella colonna 1 – fila 1. Il fatto che sia la prima non significa che non possa essere riutilizzata.

Nel caso in cui volessimo cancellare il display completamente per scrivere un altro messaggio utilizzeremo questa istruzione. Nel secondo programma il messaggio è scritto in due parti, con una cancellazione nel mezzo per riutilizzare la stessa zona del display. Nel caso non sia utilizzata l'istruzione "cls" fra le due "print", la seconda parola



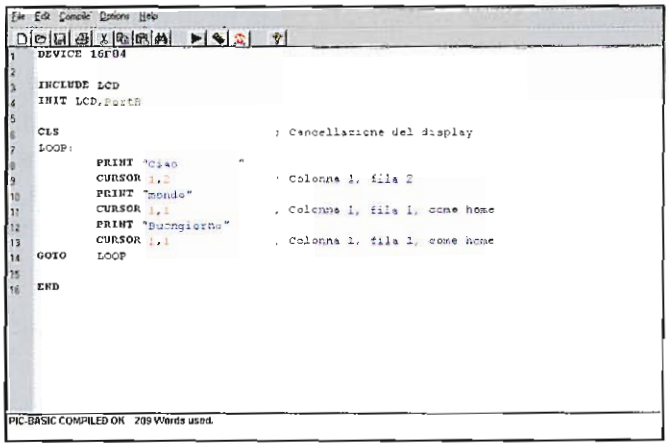
Scrittura di due messaggi consecutivi sul display LCD.



Modifica della visualizzazione utilizzando l'istruzione "cls".

sarà scritta di seguito alla prima, cioè dove è rimasto il cursore dopo la prima "print".

Facciamo ora un'altra prova. Che cosa pensate possa succedere se inseriamo in un ciclo infinito le istruzioni "cls" e "print" messe una dopo l'altra, intervallate solo da un'istruzione di attesa? Si tratta della stessa cosa che abbiamo fatto in un altro caso per far lampeggiare il LED: accenderlo, attendere un determinato tempo, spegnerlo, attendere un determinato tempo, accenderlo, ecc. In questo caso, invece di accendere un LED, si tratta di scrivere un messaggio, però il procedimento per farlo lampeggiare è il medesimo.



Esempio di posizionamento del cursore.

Comando di scrittura

Nei programmi precedenti abbiamo scritto delle catene di caratteri con l'istruzione "print", è altresì possibile scrivere altri tipi di dati, come il contenuto di variabili definite dall'utente, oppure numeri.

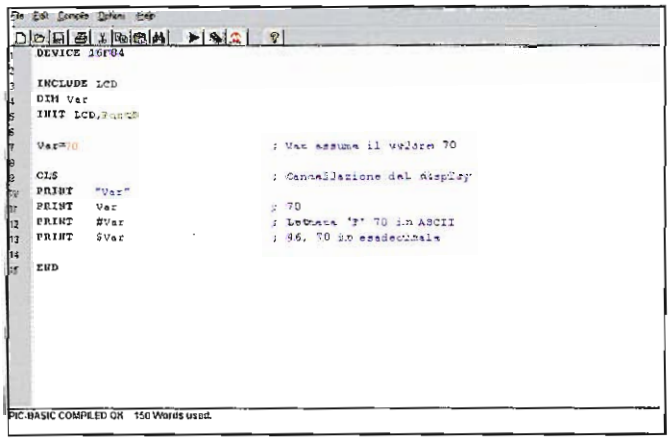
Sia le variabili che i numeri possono essere visualizzati come caratteri, anteponendo il simbolo # o come numeri esadecimali se il simbolo è $. Così nella prima istruzione "print" della figura si mostra il nome della variabile "A", nella seconda il suo contenuto, nella terza il carattere ASCII corrispondente al contenuto della variabile, e nella quarta il contenuto della variabile in codice esadecimale.

Posizione dei messaggi

I messaggi scritti sino a questo momento sono stati collocati nella prima fila e colonna del display LCD, oppure lì dove è rimasto il cursore dopo l'ultima scrittura. Questo però non ha molto senso, normalmente si sceglie la posizione in cui vogliamo far apparire i messaggi. Questo si ottiene grazie all'istruzione "cursor". Con essa collochiamo il cursore nella fila e nella colonna che desideriamo fra quelle esistenti (dipenderà dal modello di display LCD con cui si lavora).

Il primo parametro di questa istruzione è la colonna, il secondo la fila; dopo di che possiamo utilizzare uno di questi tre comandi: LEFT, RIGHT o HOME. Il primo muove di un carattere il cursore a sinistra, il secondo a destra, e il terzo lo porta alla colonna 1 – fila 1. Il movimento del cursore non implica la cancellazione di alcun carattere, in nessun caso.

Nel LetPicBasicLite, per via della versione gratuita, gli ultimi tre comandi presentati per l'istruzione "cursor" non sono abilitati. È comunque possibile ottenere lo stesso risultato muovendo il cursore con una semplice istruzione di colonna-fila. Nell'esempio della figura si riscrive in continuazione la prima fila con due messaggi differenti, ma senza prima cancellare il precedente.



Differenti esempi con l'istruzione "print".