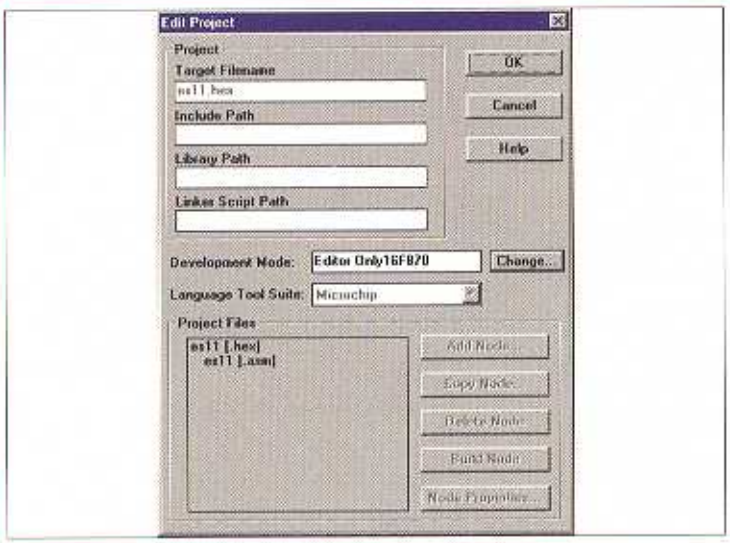
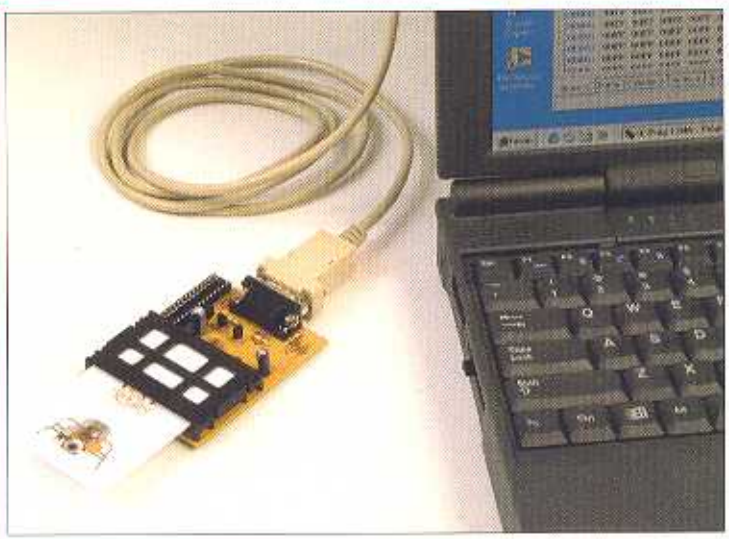


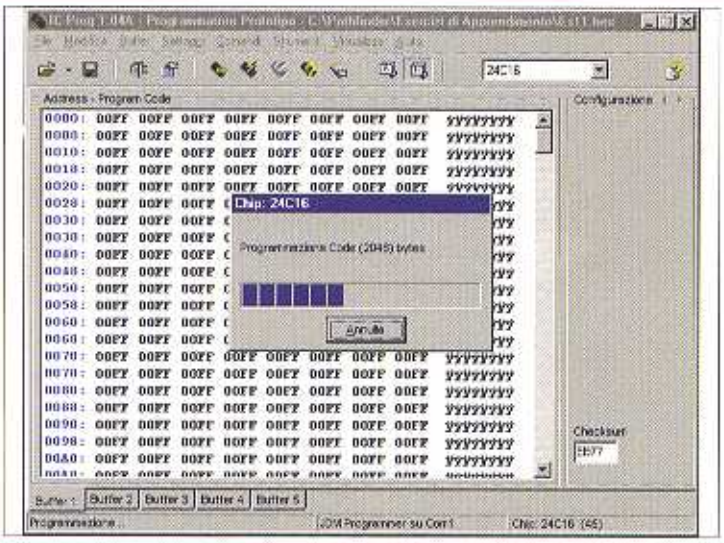
Realizzeremo un primo esercizio con la scheda di ingressi e uscite. Sarà un esercizio semplice, che consiste nel visualizzare sui diodi LED (da D1 a D8) della scheda il valore degli interruttori (da SW3 a SW8). Il codice sorgente di questo esercizio si trova nel CD che vi è stato fornito, con il nome di es11.asm. L'esercizio si realizzerà con il programma MPLAB. Dobbiamo creare un nuovo progetto in una directory del disco rigido, e copiare in questa stessa directory il file es11.asm. Selezioneremo il microcontroller modello PIC16F870.

```
c:\progra~1\mplab\es11.asm
;
;Leggere lo stato degli interruttori della scheda di ingresso(porta C)e visualizza
;il livello logico degli stessi sui leds RB0-RB7 collegati alla porta B

        List    p=16F870                ;Tipo di processore
        include "P16F870.INC"   ;Definizione dei registri interni

        ORG     0x00

Inizio  clrf    PORTB           ;Cancella il latch di uscita
        bsf     STATUS,RP0      ;Seleziona il banco 1
        clrf    TRISB           ;Porta B configurata come uscita
        movlw   0xFF
        movwf   TRISC           ;Porta C configurata come ingresso
        bcf     STATUS,RP0      ;Seleziona il banco 0

Loop    movf    PORTC,W         ;Leggi gli ingressi
        movwf   PORTB           ;Visualizzali sulle uscite
        goto    Loop            ;Ciclo senza fine

        end                     ;Fine del programma sorgente
```

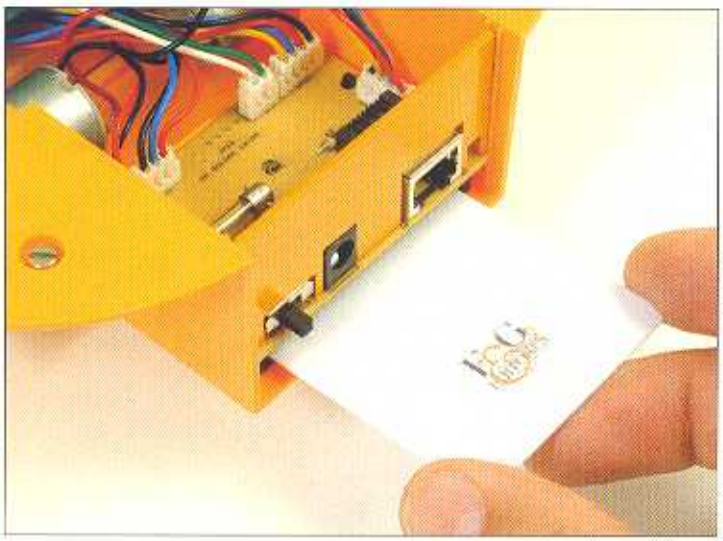
In questa immagine possiamo vedere il codice sorgente del programma. Per prima cosa viene configurata la porta C del microcontroller come ingresso, per poter leggere lo stato degli interruttori, e la porta B come uscita, per inviare dati ai diodi LED. Dopo aver realizzato le configurazioni sul banco 1 del microcontroller, si torna al banco 0 e si entra in un ciclo infinito, in cui si va continuamente a leggere il valore della porta C e si manda il risultato alla porta B. Come tutte le operazioni fra registri, abbiamo la necessità di utilizzare il registro di lavoro W per realizzare questo compito.



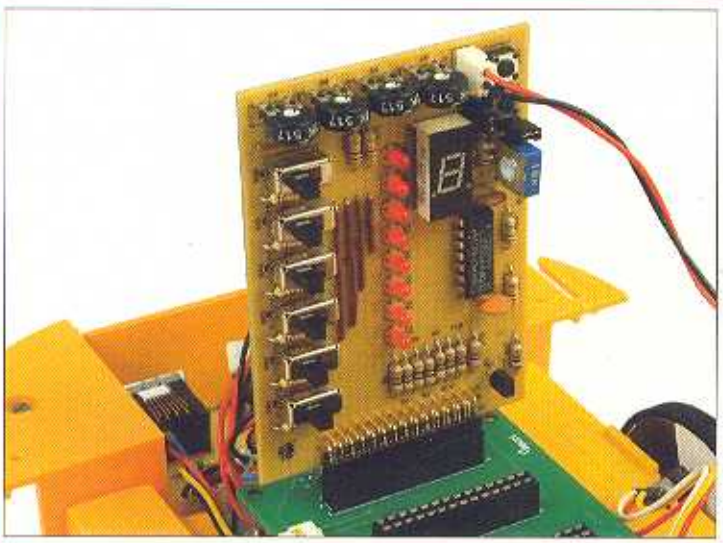
Dopo aver scritto il programma, il passo successivo consiste nel compilare il file. Per fare questo andiamo nel menù "Project" e scegliamo "Build all". Dopo aver compilato, otterremo il file con estensione esadecimale (.hex) che dobbiamo scrivere sulla Smartcard. A questo scopo inseriremo la scheda sullo scrittore e lo collegheremo al PC. Eseguiremo il programma di scrittura ICPROG.



Dopo aver aperto il programma ICPROG, il primo passo consiste nel selezionare il tipo di dispositivo che vogliamo programmare. In questo caso sarà la memoria tipo 24C16. Ora cerchiamo il file esadecimale da scrivere (es11.hex). Dopo aver selezionato il file sceglieremo l'opzione "Programma tutto", e il file verrà scritto sulla Smartcard. Per fare in modo che l'esercizio possa funzionare, Il microcontroller deve precedentemente essere stato scritto con il programma "uploader.hex", e montato nella scheda di controllo di Pathfinder (zoccolo U3).

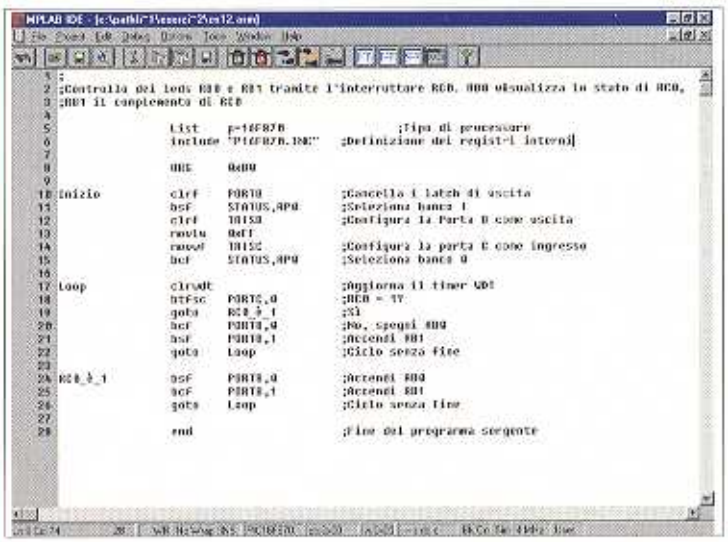


Dopo aver scritto il programma sulla Smartcard dobbiamo inserire la scheda in Pathfinder. A questo scopo utilizzeremo la scheda di alimentazione, inserendo la scheda Smartcard con lo stesso orientamento mostrato nell'immagine. Verificheremo che la scheda di alimentazione abbia tutti i connettori collegati alla scheda di controllo. Anche la scheda di interfaccia deve essere collegata alla scheda di controllo. Dopo aver inserito la Smartcard forniremo la tensione di alimentazione al robot, utilizzando delle batterie o un alimentatore esterno a corrente continua tramite J1.



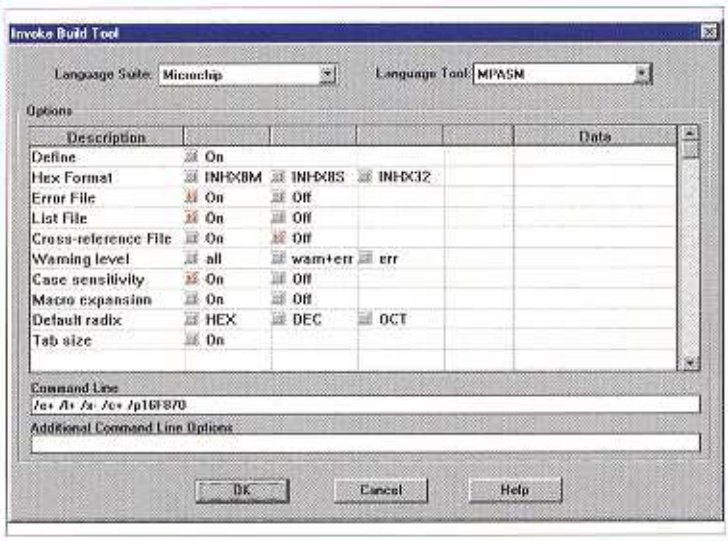
Per fare in modo che l'esercizio funzioni correttamente, la scheda di ingressi e uscite deve essere inserita sul connettore JP13 della scheda di interfaccia. Dobbiamo chiudere il jumper JP1 della scheda di ingressi e uscite ed estrarre il jumper JP4. Vi raccomandiamo di togliere i sensori ottici collegati alla scheda di controllo. Dopo aver alimentato il robot, il microcontroller impiegherà qualche secondo per leggere la scheda Smartcard, dopodiché ne inizierà l'esecuzione. Mediante i sei interruttori modificheremo lo stato dei sei diodi LED collegati alla scheda di ingressi e uscite. Se colleghiamo l'altoparlante e chiudiamo il jumper JP5, tramite l'interruttore SW8 faremo suonare l'altoparlante.

Esercizi di apprendimento



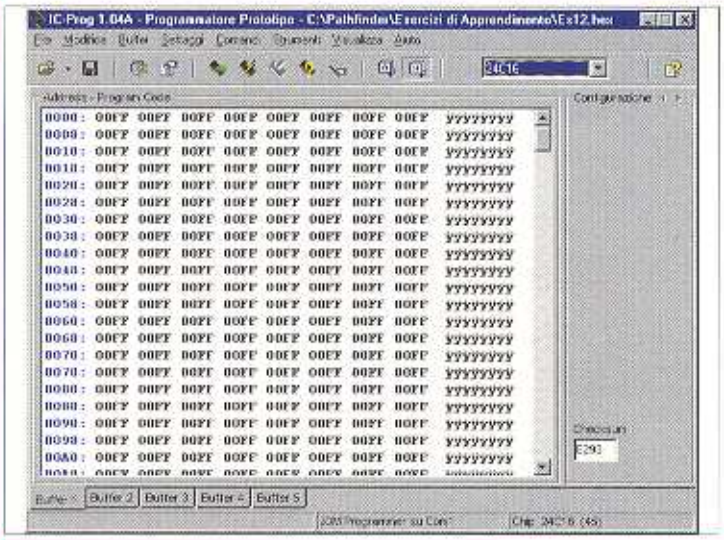
```
;Controlla dei leds RB0 e RB1 tramite l'interruttore RC0. RB0 visualizza lo stato di RC0,
;RB1 il complemento di RC0

                List    P=16F870                ;Tipo di processore
                include "P16F870.INC"   ;Definizione dei registri interni

                ORG     0x00

Inizio          clrf    PORTB           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona banco 1
                clrf    TRISB           ;Configura la Porta B come uscita
                movlw   0xff
                movwf   TRISC           ;Configura la porta C come ingresso
                bcf     STATUS,RP0      ;Seleziona banco 0

Loop            clrwdt                  ;Aggiorna il timer WDT
                btfsc   PORTC,0         ;RC0 = 1?
                goto    RC0_è_1         ;Sì
                bcf     PORTB,0         ;No, spegni RB0
                bsf     PORTB,1         ;Accendi RB1
                goto    Loop            ;Ciclo senza fine

RC0_è_1         bsf     PORTB,0         ;Accendi RB0
                bcf     PORTB,1         ;Accendi RB1
                goto    Loop            ;Ciclo senza fine

                end                     ;Fine del programma sorgente
```

Spiegheremo ora il secondo esercizio di apprendimento che potremo provare con la scheda di ingressi e uscite. Si tratta dell'esercizio es12.asm, incluso nel CDROM, nella directory Esercizi di apprendimento. Come il primo esercizio che abbiamo provato, si tratta di una prova di base sulle porte di ingresso e uscita. In questo caso abbiamo a disposizione un interruttore (SW3) che servirà per cambiare lo stato del pin RC0 del microcontroller. Se l'interruttore è a livello alto, il diodo D1 si accenderà e il diodo D2 rimarrà spento; se l'interruttore è a livello basso i diodi cambieranno il loro stato. I diodi D1 e D2 saranno gestiti con i pin RB0 e RB1 del PIC.

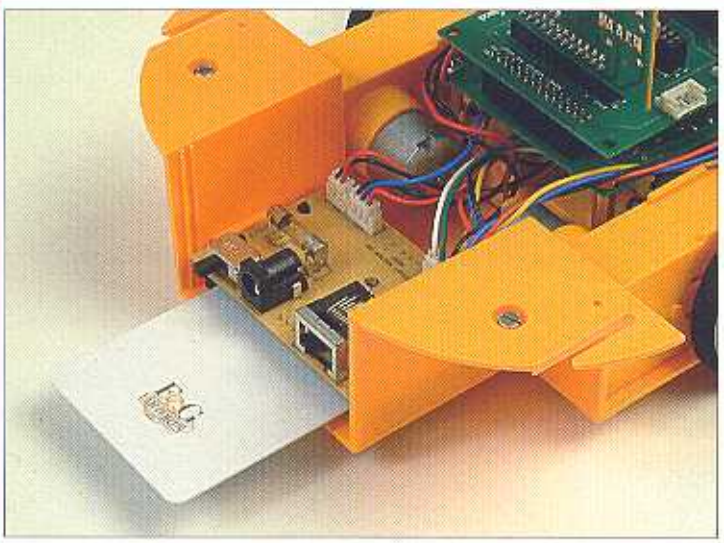


Per compilare l'esercizio dobbiamo utilizzare il programma MPLAB. Possiamo creare un nuovo progetto e inserirvi il programma, quindi compilarlo. Se non vogliamo creare un progetto, possiamo aprire il file (che avremo copiato sul nostro Hard Disk) ed eseguire l'opzione "Build node", presente nel menù "Project". Apparirà una videata di configurazione che dobbiamo impostare come quella che vediamo nell'immagine. Clicchiamo il pulsante OK e il programma si compilerà, generando il file es12.hex che scriveremo sul microcontroller.

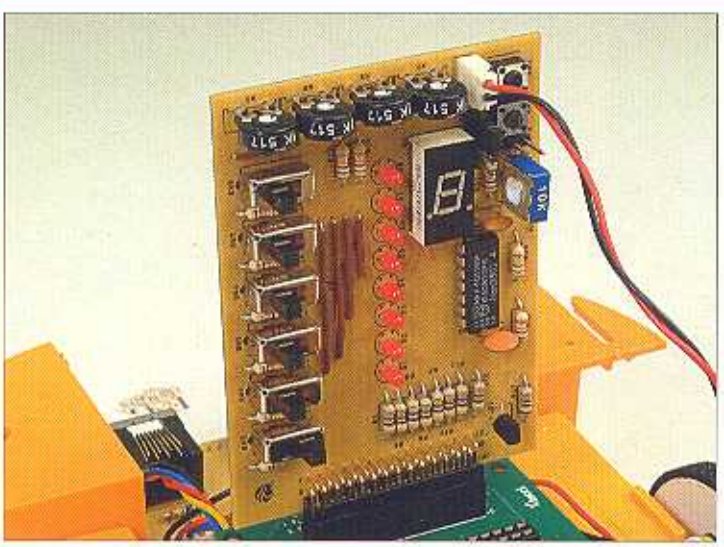
Tutti gli esercizi che si scrivono su Pathfinder saranno eseguiti tramite la Smartcard; la inseriremo nella scheda di scrittura con il verso mostrato nell'immagine. Lo scrittore si deve collegare a una porta seriale del PC, o tramite un cavo seriale, o direttamente al connettore DB9 presente sul PC. Dopo aver collegato la scheda di scrittura eseguiremo il programma ICPROG.



Per programmare la Smartcard dobbiamo selezionare il dispositivo 24C16.
Dopo aver selezionato il dispositivo, dobbiamo aprire il file es12.hex e cliccare sul pulsante di programmazione per trasferirlo alla Smartcard. Gli utenti di Windows NT, 2000 o XP devono copiare il file Icprog.sys nella stessa directory del file del programma Icprog.exe, e dovranno attivare la casella "Abilita il driver NT/2000" presente nella cartella "Misc" di "Opzioni", a cui si accede tramite il menù "Settaggi".

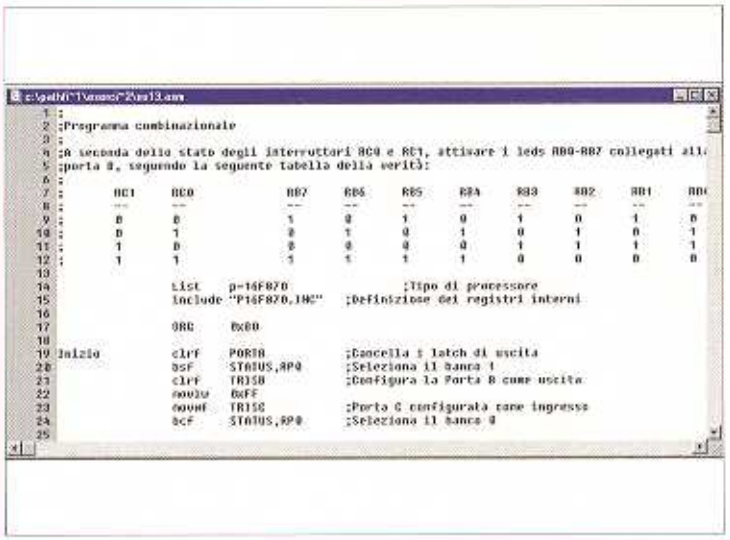


Dopo aver programmato la Smartcard, la inseriremo nel robot tramite la scheda di alimentazione, con l'orientamento adeguato. Sul robot dovrà essere montato il microcontroller programmato con il file uploader.hex. Per provare questo esercizio inseriremo la scheda di ingressi e uscite sul connettore JP13 della scheda di interfaccia. Alimenteremo Pathfinder con un alimentatore esterno a corrente continua, o con delle batterie.



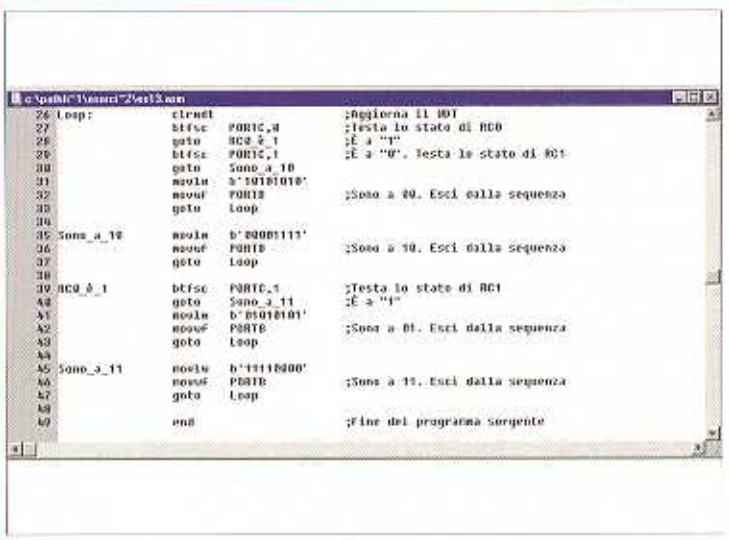
Per provare questo esercizio, il jumper JP1 della scheda di ingressi e uscite deve essere chiuso, e dovremo estrarre il jumper JP4. Dopo l'accensione, il robot impiegherà qualche secondo a leggere il contenuto della Smartcard e poi inizierà a eseguire il programma.
Trascorso il tempo di lettura verificheremo come sono gestiti i diodi LED D1 e D2, modificando lo stato del commutatore SW3.

Esercizi di apprendimento



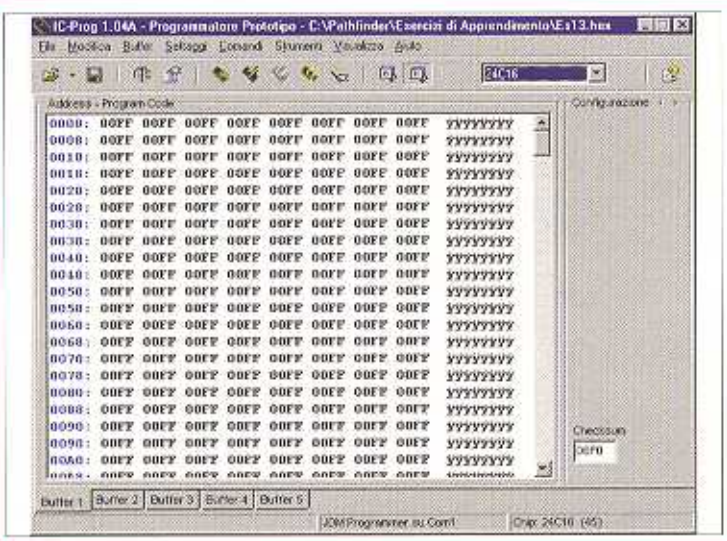
```
;
;Programma combinazionale
;
;A seconda dello stato degli interruttori RC0 e RC1, attivare i leds RB0-RB7 collegati alla
;porta B, seguendo la seguente tabella della verità:
;
;      RC1     RC0          RB7     RB6     RB5     RB4     RB3     RB2     RB1     RB0
;      --      --           --      --      --      --      --      --      --      --
;       0       0           1       0       1       0       1       0       1       0
;       0       1           0       1       0       1       0       1       0       1
;       1       0           0       0       0       0       1       1       1       1
;       1       1           1       1       1       1       0       0       0       0

                List    p=16F870                ;Tipo di processore
                include "P16F870.INC"   ;Definizione dei registri interni

                ORG     0x00

Inizio          clrf    PORTB           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona il banco 1
                clrf    TRISB           ;Configura la Porta B come uscita
                movlw   0xFF
                movwf   TRISC           ;Porta C configurata come ingresso
                bcf     STATUS,RP0      ;Seleziona il banco 0
```

Realizziamo ora un altro esercizio di apprendimento. Si tratta di un esercizio combinazionale; abbiamo due interruttori (SW3 e SW4) che gestiscono lo stato dei piedini RC0 e RC1 del microcontroller. Possiamo avere quattro stadi differenti in funzione del valore 1 o 0 di questi interruttori. Per ogni combinazione di ingresso genereremo una combinazione di uscita differente che consisterà in un dato a 8 bit da inviare alla porta B del microcontroller, in modo che si possa visualizzare sugli otto diodi LED.

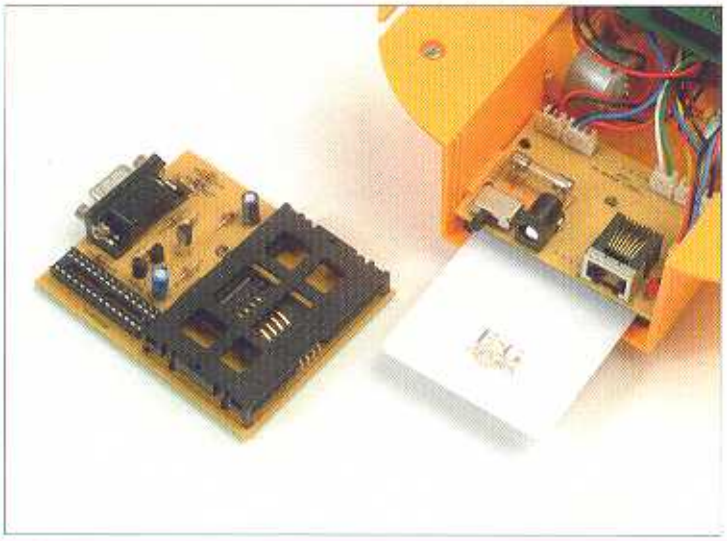


```
Loop:           clrwdt                  ;Aggiorna il WDT
                btfsc   PORTC,0         ;Testa lo stato di RC0
                goto    RC0_A_1         ;È a "1"
                btfsc   PORTC,1         ;È a "0". Testa lo stato di RC1
                goto    Sono_a_10
                movlw   b'10101010'
                movwf   PORTB           ;Sono a 00. Esci dalla sequenza
                goto    Loop

Sono_a_10       movlw   b'00001111'
                movwf   PORTB           ;Sono a 10. Esci dalla sequenza
                goto    Loop

RC0_A_1         btfsc   PORTC,1         ;Testa lo stato di RC1
                goto    Sono_a_11       ;È a "1"
                movlw   b'01010101'
                movwf   PORTB           ;Sono a 01. Esci dalla sequenza
                goto    Loop

Sono_a_11       movlw   b'11110000'
                movwf   PORTB           ;Sono a 11. Esci dalla sequenza
                goto    Loop

                end                     ;Fine del programma sorgente
```

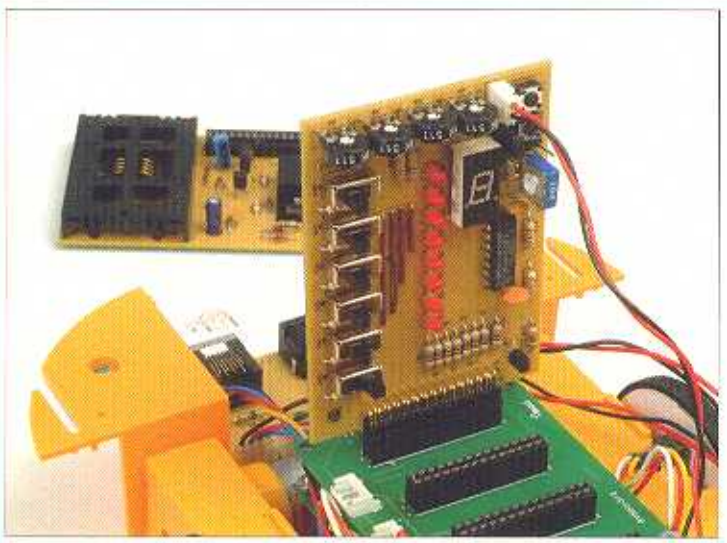
Questo esercizio si trova nel CDROM sotto il nome di es13.asm. Copieremo il file su una directory nel nostro hard disk per procedere alla sua compilazione. Il programma configura la porta C come ingresso per gli interruttori e la porta B come uscita per i diodi LED. Successivamente si entra nei cicli che testano mediante istruzioni di salto btsfc il valore dei bit RC0 e RC1, per le quattro combinazioni possibili. Per ogni stato viene inviato il valore corrispondente degli otto diodi della porta B.



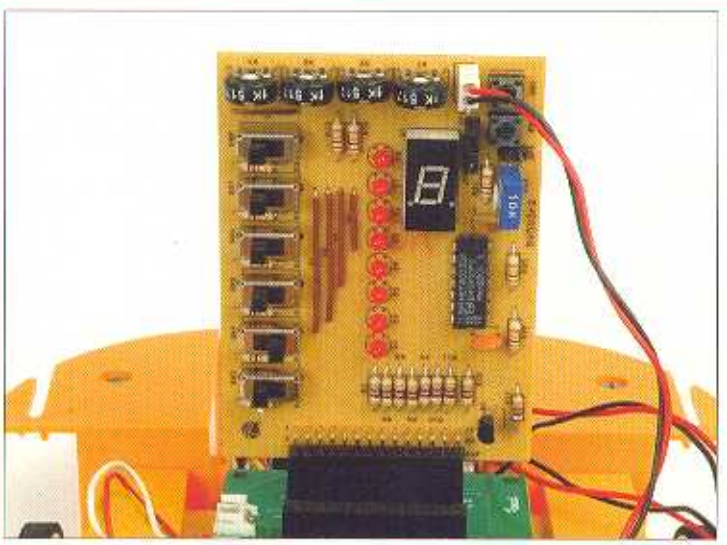
Dopo aver compilato il programma otterremo il file es13.hex, che dovrà essere caricato sulla Smartcard. Metteremo la Smartcard sulla scheda di scrittura e quest'ultima la collegheremo al PC mediante la porta seriale. Tramite il programma ICPROG selezioneremo il dispositivo 24C16, apriremo il file es13.hex e procederemo alla programmazione. Vi consigliamo di eseguire una verifica dopo aver programmato la Smartcard mediante l'opzione "Verifica" del menù "Comando".



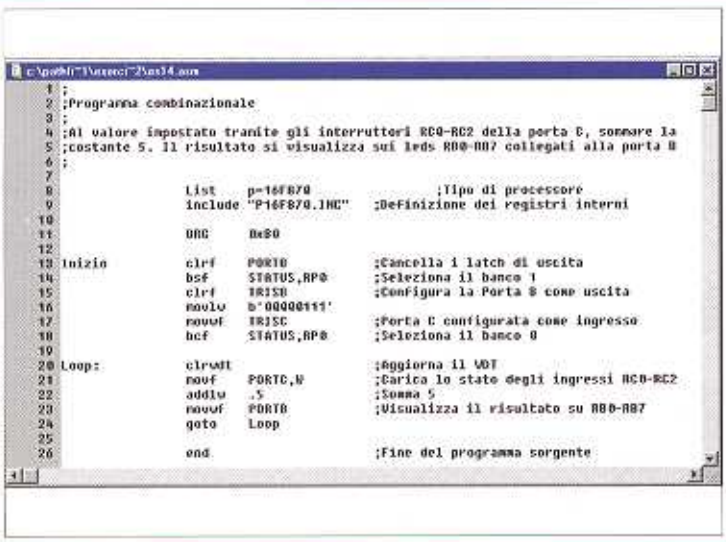
La procedura di programmazione di Pathfinder è la stessa che stiamo utilizzando per verificare gli esercizi di ingresso e uscita. Prima si inserisce la Smartcard sulla scheda di scrittura, per trasferire il programma sulla memoria, quindi dopo averla programmata la monteremo sul robot mediante la scheda di alimentazione. È importante che la Smartcard venga sempre inserita con l'orientamento adeguato su entrambe le schede.



Per verificare questo esercizio la scheda di ingresso e uscita deve essere inserita sul connettore della scheda di interfaccia JP13. Dopo aver alimentato il robot, il microcontroller impiegherà alcuni secondi a leggere il contenuto della Smartcard quindi inizierà l'esecuzione dell'esercizio. Per fare in modo che l'esercizio funzioni, il jumper JP1 deve essere chiuso, mentre JP4 dovrà essere aperto, per attivare i diodi LED e disattivare il display a sette segmenti.

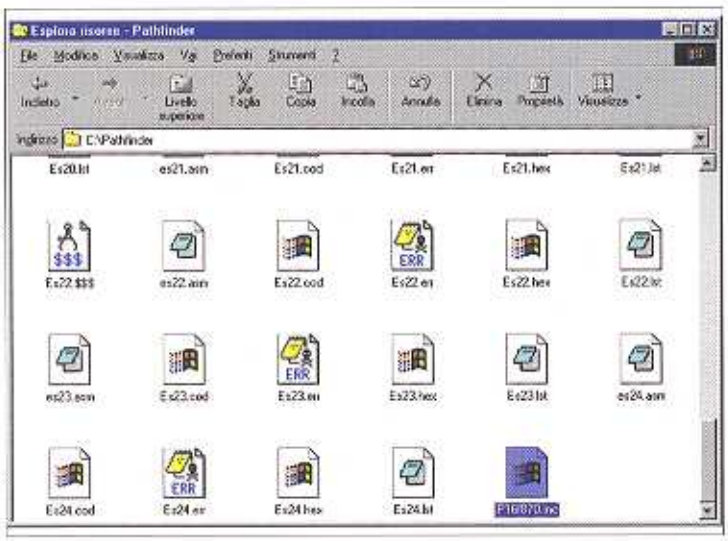


Nell'immagine possiamo vedere l'esercizio durante il funzionamento. Modificando i valori dei commutatori SW3 e SW4 inseriremo una delle quattro combinazioni possibili. Per ogni combinazione inserita osserveremo come cambia lo stato dei diodi LED che corrisponderà alle combinazioni di 1 e 0 selezionate nel programma.

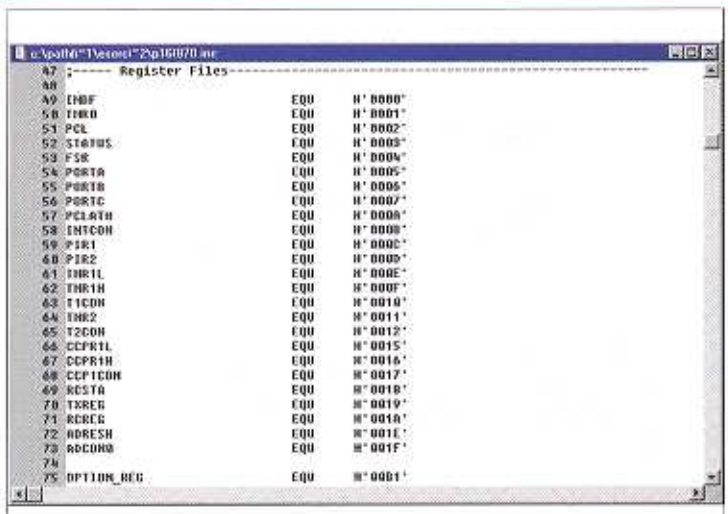


```
;
;Programma combinazionale
;
;Al valore impostato tramite gli interruttori RC0-RC2 della porta C, sommare la
;costante 5. Il risultato si visualizza sui leds RB0-RB7 collegati alla porta B
;

                List    p=16F870                ;Tipo di processore
                include "P16F870.INC"   ;Definizione dei registri interni

                ORG     0x80

Inizio          clrf    PORTB           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona il banco 1
                clrf    TRISB           ;Configura la Porta B come uscita
                movlw   b'00000111'
                movwf   TRISC           ;Porta C configurata come ingresso
                bcf     STATUS,RP0      ;Seleziona il banco 0

Loop:           clrwdt                  ;Aggiorna il WDT
                movf    PORTC,W         ;Carica lo stato degli ingressi RC0-RC2
                addlw   .5              ;Somma 5
                movwf   PORTB           ;Visualizza il risultato su RB0-RB7
                goto    Loop

                end                     ;Fine del programma sorgente
```

Nell'immagine possiamo vedere il codice di un altro esercizio per la gestione degli ingressi e le uscite (file es14.asm). Viene catturato il dato presente sui primi tre interruttori della porta C (SW3 - SW4 - SW5), si somma il numero 5 a questo dato, e si visualizza il risultato in binario sui diodi LED collegati alla porta B del microcontroller. Il programma configura i primi tre pin della porta C come ingressi, la porta B come uscita, dopodiché entra in ciclo infinito in cui legge continuamente il contenuto della porta C, esegue l'operazione sul dato e invia il risultato alla porta B per la visualizzazione sui diodi LED.

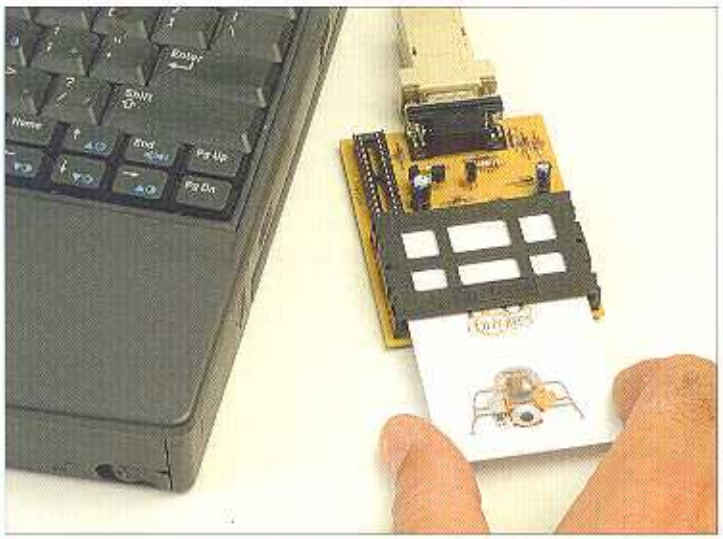


Tutti gli esercizi di Pathfinder devono essere compilati partendo dall'indirizzo 0xB0 della memoria di programma. Questa funzione è realizzata dalla direttiva ORG 0xB0. L'impostazione di questo indirizzo di partenza è dovuto al fatto che i primi indirizzi della memoria di programma sono già occupati dal programma scritto in precedenza sul microcontroller, cioè il programma uploader, che ha il compito di leggere il contenuto della memoria EEPROM e autoprogrammarlo nella memoria del microcontroller. Il file P16F870.inc, incluso nel programma, contiene tutte le definizioni dei registri e i bit di controllo del microcontroller. Possiamo aprirlo con un qualsiasi programma di editor per vederne il contenuto.

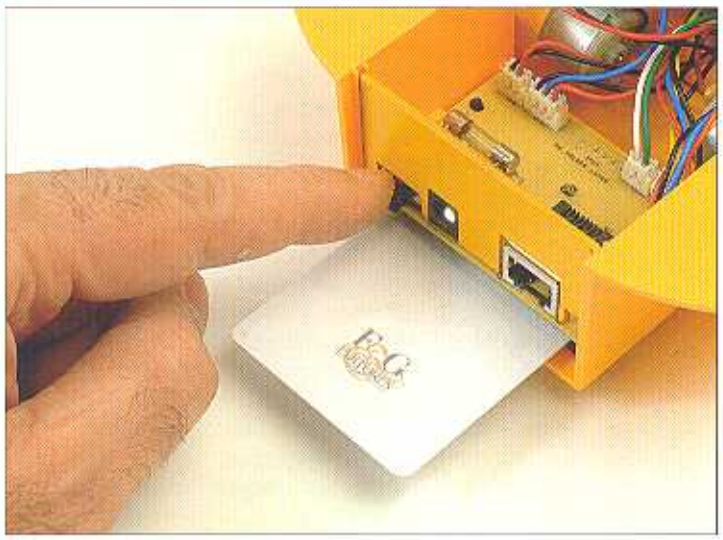


```
;----- Register Files------------------------------------------------------

INDF                    EQU     H'0000'
TMR0                    EQU     H'0001'
PCL                     EQU     H'0002'
STATUS                  EQU     H'0003'
FSR                     EQU     H'0004'
PORTA                   EQU     H'0005'
PORTB                   EQU     H'0006'
PORTC                   EQU     H'0007'
PCLATH                  EQU     H'000A'
INTCON                  EQU     H'000B'
PIR1                    EQU     H'000C'
PIR2                    EQU     H'000D'
TMR1L                   EQU     H'000E'
TMR1H                   EQU     H'000F'
T1CON                   EQU     H'0010'
TMR2                    EQU     H'0011'
T2CON                   EQU     H'0012'
CCPR1L                  EQU     H'0015'
CCPR1H                  EQU     H'0016'
CCP1CON                 EQU     H'0017'
RCSTA                   EQU     H'0018'
TXREG                   EQU     H'0019'
RCREG                   EQU     H'001A'
ADRESH                  EQU     H'001E'
ADCON0                  EQU     H'001F'

OPTION_REG              EQU     H'0081'
```

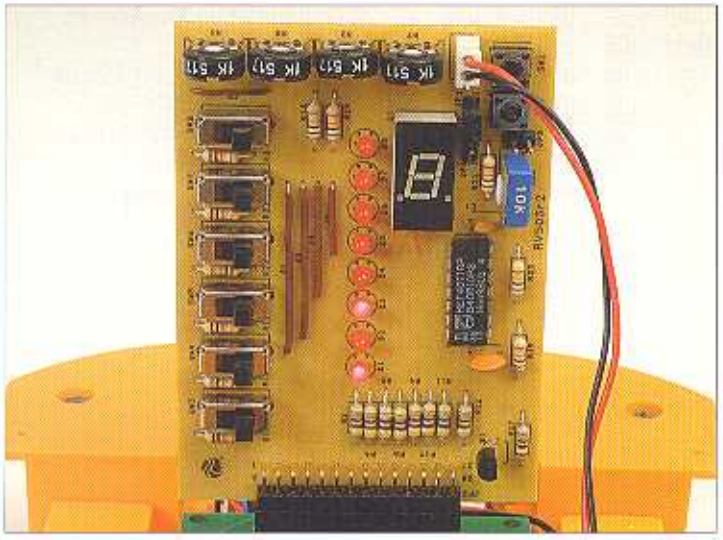
L'immagine mostra una parte del contenuto del file dove possiamo vedere la definizione dei registri di controllo del Banco 0 del microcontroller. Includendo il file P16F870.inc nel nostro programma, eviteremo di dover definire manualmente i registri di controllo, ma sarà sufficiente assegnare un nome ai registri di lavoro che potremo generare a partire dall'indirizzo 0x20 della memoria RAM, la prima posizione libera per i registri di lavoro.



Dopo aver compilato il file es14.asm e aver ottenuto il file esadecimale es14.hex, passeremo alla scrittura della Smartcard. La dovremo inserire nella scheda di scrittura con l'orientamento adeguato. Selezioneremo nel programma ICPROG la memoria modello 24C16, apriremo il file es14.hex e daremo il via alla programmazione della Smartcard (opzione Programma Tutto). Vi consigliamo di verificare la programmazione mediante l'opzione Verifica del menù Comandi.



Dopo aver scritto la Smartcard, è necessario inserirla nella scheda di alimentazione del robot, con l'orientamento adeguato. Per fare in modo che il programma venga eseguito dal microcontroller, quest'ultimo dovrà essere presente sulla scheda di controllo, ed essere stato precedentemente programmato con il file uploader.hex. Dopo aver alimentato il robot, il microcontroller impiegherà alcuni secondi a leggere il contenuto della Smartcard e a iniziare l'esecuzione del programma. La scheda di ingressi e uscite dovrà essere inserita sul connettore JP13 della scheda di interfaccia.



Per eseguire il programma è necessario chiudere il jumper JP1 e lasciare il resto dei jumper della scheda aperti, in questo modo abiliteremo solamente i diodi LED. A questo punto inseriremo i dati binari mediante i primi tre interruttori e vedremo sui diodi LED il dato inserito sommato a cinque, la quantità costante. Il programma rimarrà in esecuzione per un tempo indefinito.

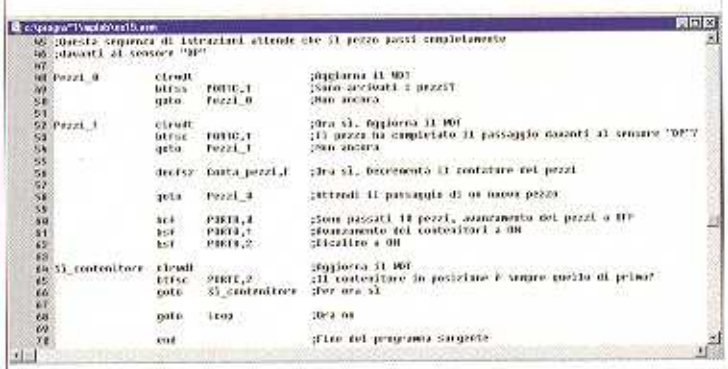
Esercizi di apprendimento

```
;
;Macchina di confezionamento
;
;Due relè "H1" (RB0)  e "H2" (RB1), comandano due motori che fanno muovere due nastri
;trasportatori. "H1" (RB0) trasporta pezzi e "H2" (RB1) imballaggi. Un sensore "DP" (RC1)
;rileva il passaggio dei pezzi e l'altro, "DE" (RC2), rileva il corretto posizionamento di
;un contenitore. Dopo che sono passati 10 pezzi, la confezione si considera piena, si
;attiva un segnale acustico, "H" (RB2), e il nastro che trasporta gli imballaggi parte,
;sino a posizionare un nuovo imballo vuoto. In questo momento si disattiva il segnale
;acustico "H" (RB2), e avanza nuovamente il nastro dei pezzi ripetendo così il ciclo.
;Un interruttore "I" (RC0) attiva o disattiva l'intero sistema.

                LIST    p=16F870                ;Tipo di processore
                include "P16F870.INC"   ;Definizione dei registri interni

Conta_pezzi     equ     0x20            ;Variabile N° di pezzi

                org     0x00

Inizio          clrf    PORTB           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona banco 1
                clrf    TRISB           ;Configura la Porta B come uscita
                movlw   0xFF
                movwf   TRISC           ;Configura la Porta A come ingresso
                bcf     STATUS,RP0      ;Seleziona banco 0
```

Gli esercizi che abbiamo realizzato sinora erano esempi combinatori. In questo tipo di esercizi si disponeva di un insieme di segnali di ingresso che venivano letti una volta, si operava su di essi e si ottenevano dei risultati che erano visualizzati tramite i diodi led. Questo tipo di funzionamento è comune in molti algoritmi per i robot.
Si legge lo stato dei sensori e in funzione della combinazione letta si eseguono diversi ordini sui motori. Ora realizzeremo un tipo diverso di esercizi, quelli chiamati sequenziali. Le azioni sono determinate dal modo in cui si attivano i sensori.

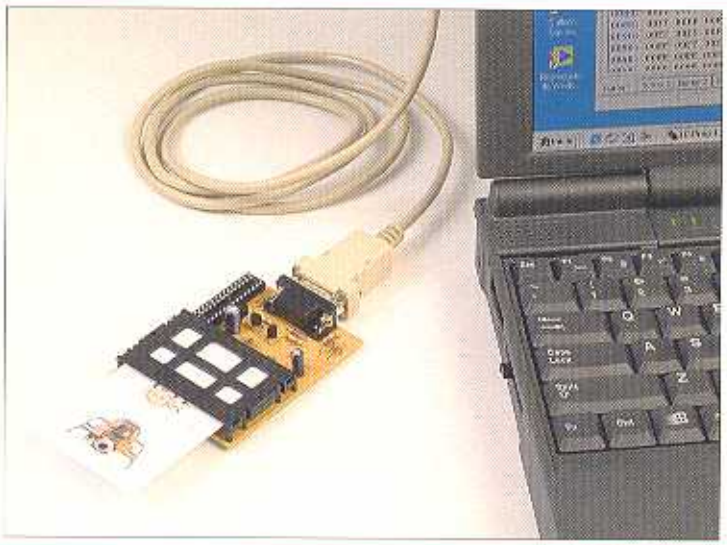
```
Loop_0          clrf    PORTB           ;Resetta le uscite
Loop:           clrwdt                  ;Aggiorna il WDT
                movlw   d'10'
                movwf   Conta_pezzi     ;Inizializza la variabile N° di pezzi
                btfss   PORTC,0         ;Testa lo stato dell'interruttore I
                goto    Loop_0          ;Il su OFF, sistema fermo

                bcf     PORTB,0         ;Avanzamento dei pezzi a OFF
                bsf     PORTB,1         ;Avanzamento dei contenitori a ON

Ha_contenitore  clrwdt                  ;Aggiorna il WDT
                btfss   PORTC,2         ;Contenitore in posizione?
                goto    Ha_contenitore  ;No.

                bsf     PORTB,0         ;Sì. Avanzamento dei pezzi a ON
                bcf     PORTB,1         ;Avanzamento dei contenitori a OFF
                bcf     PORTB,2         ;Cicalino a OFF
```

L'esercizio sequenziale che vogliamo provare si trova sul CDROM sotto il nome di es15.asm. Simula una macchina di confezionamento che trasporta pezzi su due nastri trasportatori. È necessario realizzare una serie di operazioni sui pezzi, non tutte assieme ma in modo sequenziale, quindi secondo come si attivano i sensori (simulati con gli interruttori della porta C), entreranno in funzione diversi attuatori (simulati con i diodi LED della porta B).

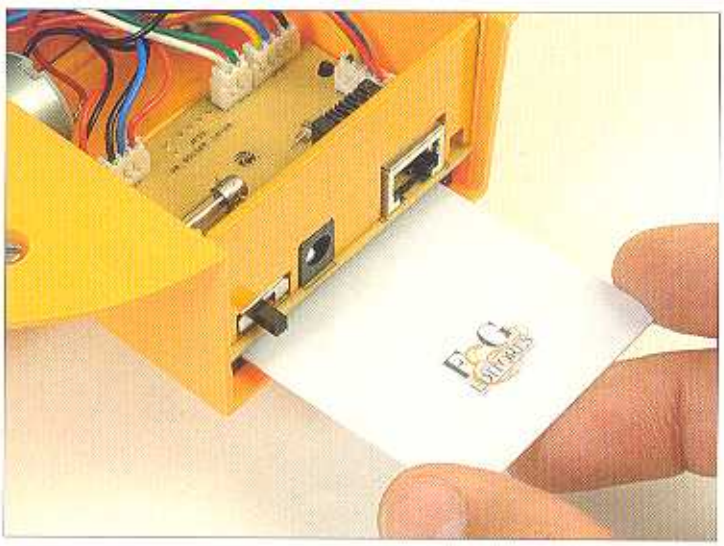


```
;Questa sequenza di istruzioni attende che il pezzo passi completamente
;davanti al sensore "DP"

Pezzi_0         clrwdt                  ;Aggiorna il WDT
                btfss   PORTC,1         ;Sono arrivati i pezzi?
                goto    Pezzi_0         ;Non ancora

Pezzi_1         clrwdt                  ;Ora sì. Aggiorna il WDT
                btfsc   PORTC,1         ;Il pezzo ha completato il passaggio davanti al sensore "DP"?
                goto    Pezzi_1         ;Non ancora

                decfsz  Conta_pezzi,f   ;Ora sì. Decremento il contatore dei pezzi

                goto    Pezzi_0         ;Attendi il passaggio di un nuovo pezzo

                bcf     PORTB,0         ;Sono passati 10 pezzi, avanzamento dei pezzi a OFF
                bsf     PORTB,1         ;Avanzamento dei contenitori a ON
                bsf     PORTB,2         ;Cicalino a ON

Sì_contenitore  clrwdt                  ;Aggiorna il WDT
                btfsc   PORTC,2         ;Il contenitore in posizione è sempre quello di prima?
                goto    Sì_contenitore  ;Per ora sì

                goto    Loop            ;Ora no

                end                     ;Fine del programma sorgente
```

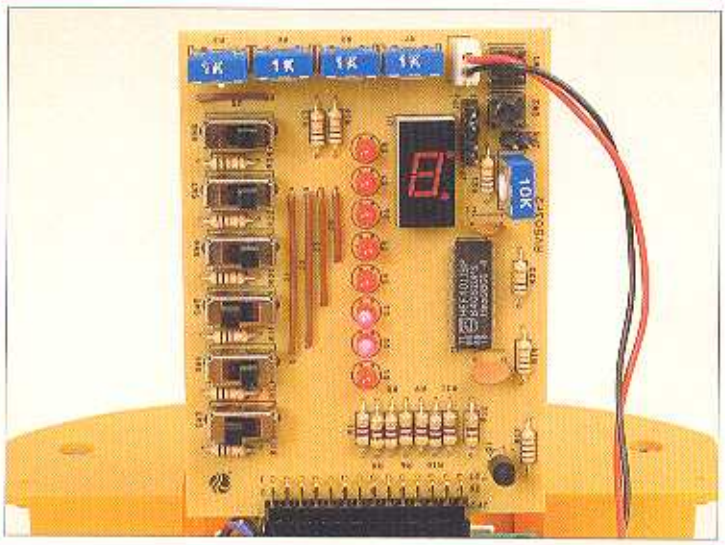
All'inizio del programma si realizzano le configurazioni: porta C come ingresso per leggere i sensori e porta B come uscita per inviare segnali agli attuatori.
In seguito si entra in una serie di cicli di attesa con le istruzioni di salto btfss e btfsc.
In questo modo, finché i sensori non hanno il valore appropriato al momento giusto, il programma rimane in attesa. Secondo come si compiono le condizioni di attivazione di ogni sensore vengono attivate le uscite corrispondenti, e si passa al ciclo successivo.



Per compilare il programma copieremo il file es15.asm sull'hard disk. Nella cartella in cui si copia dovrà essere inserito anche il file P16F870.INC, che contiene tutte le definizioni dei registri del microcontroller. Potremo generare un progetto e inserire in esso il file o semplicemente aprirlo e selezionare l'opzione Build Node del menù Project per compilare l'esercizio. Dopo aver ottenuto il file es15.hex, daremo seguito alla programmazione della Smartcard utilizzando il programma ICPROG e la scheda di scrittura con lo stesso procedimento che conosciamo già.



Per verificare questo esercizio, la scheda di ingresso e uscita dovrà essere collegata alla scheda di interfaccia tramite il connettore JP13. Il microcontroller dovrà essere inserito sulla scheda di controllo con il programma uploader.hex caricato. La Smartcard verrà inserita nella scheda di alimentazione con il programma es15.hex scritto. Dopo aver alimentato il robot, il microcontroller impiegherà alcuni secondi a leggere il contenuto della scheda Smartcard e a iniziare l'esecuzione del programma. Sulla scheda di ingresso e uscita dovrà essere chiuso il jumper JP1 e aperti i jumper JP4 e JP5.



Quando si inizia a provare l'esercizio, tutti gli interruttori dovranno essere posizionati in modo da inviare uno zero logico (la levetta dell'interruttore verso l'interno della scheda). Ora procederemo ad attivare gli interruttori nell'ordine adeguato, simulando l'attivazione dei sensori del nastro trasportatore. I diodi LED si attiveranno in modo sequenziale, simulando le diverse operazioni che realizzano gli attuatori sui pezzi presenti sul nastro trasportatore. A causa dell'effetto rimbalzo dei sensori meccanici, che vi abbiamo spiegato in passato, in alcuni casi anche con meno di dieci attivazioni dell'interruttore, il sistema conterà ugualmente dieci pezzi. In altri esercizi sarà applicata la soluzione software per correggere l'effetto rimbalzo.