

Esercizi di apprendimento

```
c:\qm4\1\esercizi\2es21.asm
1
2 ;contatore (P000M decimale da 1 digit
3
4 ;sul display a 7 segmenti collegato alla porta B verrà visualizzato il numero degli impulsi
5 ;applicati all'ingresso RC0, RC1 determina se il conteggio è ascendente (a "1")
6 ;o discendente
7
8          List    p=16870h           ;tipo di processore
9          Include "16870B.INC"       ;definizione dei registri interni
10
11 Contatore equ    0c7h             ;variabile del contatore
12
13 org     0c0h
14
15 ;inizio
16 clr    P000h                     ;Cancella i latch di uscita
17 bcf    STATUS,RP0                 ;Seleziona banco 1
18 cbr    TRISC                       ;Configura la porta B come uscita
19 movlw 0c7h                       ;Configura la porta C come ingresso
20 movwf TRISC
21 movlw 0'0000110'                 ;Prevalore di 101 per il TMRB
22 movwf SP10H,SP5
23 bcf    STATUS,RP0                 ;Seleziona banco 0
24 cbr    Contatore                  ;Azzerò il contatore
25
```

Questo è un ampliamento dell'esercizio dei contatori es20.asm, spiegato in precedenza. Si tratta di implementare un contatore ascendente e discendente, che però visualizzi il numero sul display a sette segmenti. Questo esercizio introdurrà due novità, in primo luogo l'utilizzo del display invece della barra dei diodi LED, e poi la necessità di implementare delle maschere, perché il numero da visualizzare dovrà essere compreso fra 0 e 9.

```
c:\qm4\1\esercizi\2es21.asm
26 ;Loop
27 call  Tabella                     ;Converte BCD a 7 segmenti
28 movwf PERSEL                       ;Visualizza il valore del contatore
29 cbrwf PERSEL                       ;Aggiorna il VBT
30 bcfwf PERSEL,0                     ;REF = 1?
31 goto  Volt_0                       ;No
32 call  Delay_20ms                   ;Elimina rimbaldi
33
34 ;Wait_1
35 cbrwf PERSEL                       ;Aggiorna il VBT
36 bcfwf PERSEL,0                     ;REF = 0 (impulso)?
37 goto  Volt_1                       ;No
38 call  Delay_20ms                   ;C'è un impulso, eliminare i rimbaldi
39
40 bcfwf PERSEL,1                     ;RC1 = 1
41 goto  Desc                           ;No, conteggio discendente
42
```

Il ciclo principale del programma è abbastanza simile a quello del contatore binario che abbiamo già realizzato. Abbiamo una variabile chiamata Contatore che contiene il valore del contatore e che deve essere visualizzata sul display a 7 segmenti. Esistono inoltre due cicli per attendere che il segnale RC0 passi a valore '1' e dopodiché torni a '0', completando il ciclo dell'impulso. Dopo aver inserito un impulso con RC0 si testa il pin RC1 per decidere se eseguire un conteggio ascendente o discendente.

```
c:\qm4\1\esercizi\2es21.asm
43 ;Up
44 incf  Contatore,f                 ;Incrementa contatore
45 movlw -10                          ;-10
46 subwf Contatore,W                 ;Contatore-M
47 btfsc STATUS,Z                     ;È maggiore di 0?
48 goto  Loop                          ;No
49 cbrwf Contatore                    ;SI, resetta il contatore
50 goto  Loop
51
52 ;Down
53 decf  Contatore,f                 ;Decrementa il contatore
54 movlw 0c7h                          ;0c7h
55 subwf Contatore,W                 ;Contatore-M
56 btfsc STATUS,Z                     ;È minore di 0?
57 goto  Loop                          ;No
58 movlw 0c0h                          ;0c0h
59 movwf Contatore                    ;SI, imposta a 0 il contatore
60 goto  Loop
61
```

Sia nel ciclo di conteggio ascendente che in quello discendente, l'operazione si realizza nella variabile Contatore. Dopo aver eseguito l'operazione si richiama una maschera, perché il numero sia sempre compreso fra 0 e 9, che rappresentano il valore minimo e massimo visualizzabili sul display. In questo modo, se la variabile contatore passa a valore 10 la si visualizza con il numero 0, e se dopo un decremento passa a valore 255, la si visualizza con il numero 9.



```

53 .....
54 Tabella: Questa routine converte il codice BCD presente sul 4 bit: meno significativi
55 del reg. R nel suo equivalente a 7 segmenti. Incomparabilmente il codice a 7 segmenti
56 viene scritto anche nel registro R
57 .....
58
59 Tabella:      movl   %E, %R      ;Spostamento sopra la tabella
60              movl   0'00111111' %R      ;Digit 0
61              movl   0'00000111' %R      ;Digit 1
62              movl   0'01111011' %R      ;Digit 2
63              movl   0'01001111' %R      ;Digit 3
64              movl   0'01000111' %R      ;Digit 4
65              movl   0'01101101' %R      ;Digit 5
66              movl   0'01111101' %R      ;Digit 6
67              movl   0'00000111' %R      ;Digit 7
68              movl   0'01111111' %R      ;Digit 8
69              movl   0'01000111' %R      ;Digit 9
70 .....
71
72 .....
73 .....
74 .....
75 .....

```

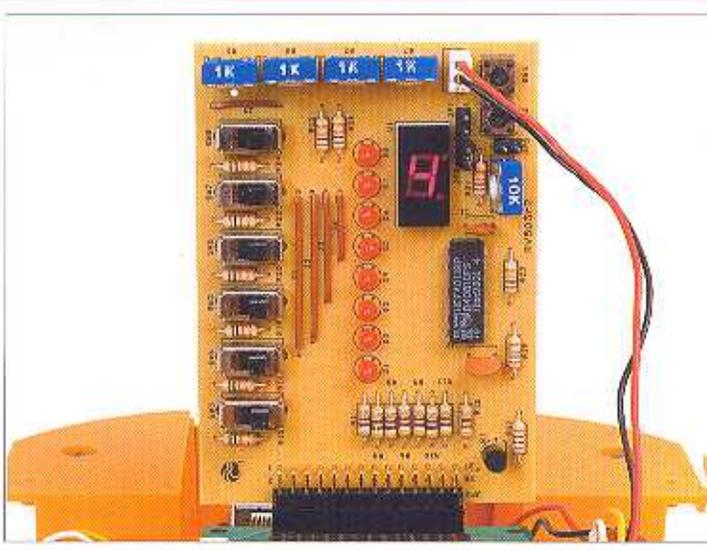
Questa è la funzione tabella che converte i numeri binari in codice numerico per il display a 7 segmenti. Nella variabile Contatore si memorizza il valore del contatore sotto forma di numero binario, che grazie alla maschera sarà sempre compreso fra 0 e 9. Prima di inviare il valore del contatore alla porta B nel ciclo Loop, è necessario richiamare la funzione Tabella per fare in modo che il valore binario sia convertito nel codice adatto al display a 7 segmenti.

```

75 .....
76 Delay 20 ms: Questa routine di ritardo ha come obiettivo eliminare "l'effetto rimbalzo",
77 caratteristica dei componenti elettromeccanici.
78 Realizza un ritardo di 20 ms. Se il PIC lavora ad una frequenza di 4 MHz,
79 il 1000 si aggiornerà ogni µs. Se si desidera temporizzare 20000 µs (20 ms) con
80 un processore di 128, il 1000 dovrà contare 150 eventi (150 * 128).
81 Il valore 150 equivale a 96 hex. e, dato che il 1000 è ascendente, lo stesso car-
82 icare con il suo complemento a 1 (53 hex.).
83 .....
84 Delay_20_ms:    movl   10000,1000      ;imposta il flag di overflow del 1000
85                movl   0x02          ;complemento hex. di 155
86                movl   1000          ;carica il 1000
87                cbrwl   0            ;aggiorna il 1000
88                stlsw  10000,1000     ;overload del 1000
89                goto   Delay_20_ms_1  ;non ancora
90                movl   10000,1000     ;carica il 1000
91                retlw  0              ;resettare il flag
92 .....
93 .....
94 .....
95 .....
96 .....

```

Questo programma utilizza degli interruttori per fornire gli impulsi di ingresso, e come già sappiamo, i dispositivi meccanici sono soggetti all'effetto rimbalzo. Questo effetto, ogni volta che un sensore cambia stato, genera un periodo di instabilità all'uscita del sensore stesso. Per contrastarlo, ogni volta che RCO cambia stato si richiama questa funzione; essa temporizza 20 ms, cosicché quando si ritorna al ciclo principale, l'uscita dell'interruttore si è già stabilizzata.



Questo esercizio si trova sul CDROM sotto il nome es21.asm. Dopo aver compilato l'esercizio lo memorizzeremo sulla Smartcard. Nell'immagine possiamo vedere l'esecuzione di questo esercizio. Per vedere i numeri sul display dobbiamo chiudere il jumper JP4 della scheda di ingressi e uscite, e aprire il resto dei jumper della scheda. Con l'interruttore SW3 inseriremo gli impulsi per il contatore; mediante l'interruttore SW4 selezioneremo se il conteggio dovrà essere ascendente o discendente.



Esercizi di apprendimento

```

1
2 :Generazione di un numero casuale
3
4
5 :Ogni volta che si applica un impulso a RC0, si genera un numero binario da 8 bits casuale
6 :che sarà visualizzato sugli 8 LED collegati alla porta B, per il tempo di 3 secondi.
7
8
9 :Tra le diverse tecniche, quella utilizzata per ottenere il numero, consiste nel catturare
10 :il valore del TMR0 in un certo momento.
11
12
13 List p=16F87D ;Tipo di processore
14 Include "P16F87D.INC" ;Definizione dei registri interni
15
16 Numero equ 0x20 ;Numero casuale
17 Delay_3sec equ 0x21 ;Costante di intervalli
18
19 org 0x00
20
21 Inizio cirdt PORTB ;Cancella i latch di uscita
22 bcf STATUS,RP0 ;Seleziona banco 1
23 cirdt TRISA ;Configura la porta A come uscita
24 movlw 0x07 ;
25 movwf TRISC ;Configura la porta C come ingresso
26 movlw b"00001111" ;
27 movwf OPTION_REG ;Prescaler al 256 per il TMR0
28 bcf STATUS,RP0 ;Seleziona banco 0
  
```

Il presente esercizio è un generatore di numeri casuali. Il programma è il preludio a un altro esercizio che consisterà in un dado elettronico. Per poter sviluppare questo programma dobbiamo gestire interruttori e led come ingressi e uscite e anche il temporizzatore interno del microcontroller Timer0. Ogni volta che premeremo l'interruttore SW3 della scheda di ingressi e uscite si visualizzerà un numero binario casuale sulla barra dei diodi led, che rimarrà visibile per tre secondi.

```

26
27 Loop cirdt PORTB,0 ;Aggiorna il LED
28 btfsc RC0 ;E' stato premuto?
29 goto Loop ;No, ancora
30 movlw TMR0_H ;Tra il...
31 movwf Numero ;Cattura il valore del TMR0 (8 bits casuale)
32 call Delay_20_ms ;Elimina i rimbalzi
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
  
```

Il Timer0 è un contatore interno del microcontroller che si utilizza per realizzare temporizzazioni. Questo contatore è sempre attivo, per questo motivo ogni volta che l'interruttore RC0 passa a "1" si acquisisce il valore del Timer0 che sarà un numero casuale fra 0 e 255, essendo il Timer0 un contatore a 8 bit. Quando vogliamo realizzare funzioni casuali, un buon metodo è acquisire il valore del temporizzatore del microcontroller, dato che per ogni acquisizione si potrà avere un valore qualsiasi.

```

47
48
49 :Delay_20_ms: Questa routine di ritardo ha come obiettivo eliminare l'effetto rimbalzo,
50 :caratteristico dei componenti elettromeccanici. Realizza un ritardo di 20 ms.
51 :Se il PIC lavora a una frequenza di 4 MHz, il TMR0 si aggiorna ogni 1µs. Se vogliamo tempori-
52 :zare 20000 µs (20 ms) con un prescaler di 256, il INCR dovrà contare 20 eventi.
53 :((78 * 256). Il valore 78 equivale a 0x4e hex, e dato che il TMR0 è incrementato in decore
54 :l'aggiungere con il suo complemento a 1 (0x11 hex.).
55
56 Delay_20_ms: bcf INCON,TRIF ;Resetta il flag di overflow del TMR0
57 movlw 0x11 ;Complemento hex. di 78
58 movwf TMR0 ;Carica il TMR0
59 cirdt TRISA ;Aggiorna il LED
60 goto Delay_20_ms ;No, ancora
61
62
63
  
```

Il numero casuale da mostrare sulla barra dei diodi led si produce quando si introduce un impulso completo mediante l'interruttore SW3 della scheda, che invia segnali al pin RC0 del microcontroller. Bisogna eliminare l'effetto rimbalzo dei sensori meccanici, e per questo motivo quando RC0 cambia il suo valore a 1, dopo aver acquisito il valore casuale del Timer0, si chiama una routine di temporizzazione di 20 ms che elimina l'effetto rimbalzo dell'interruttore.

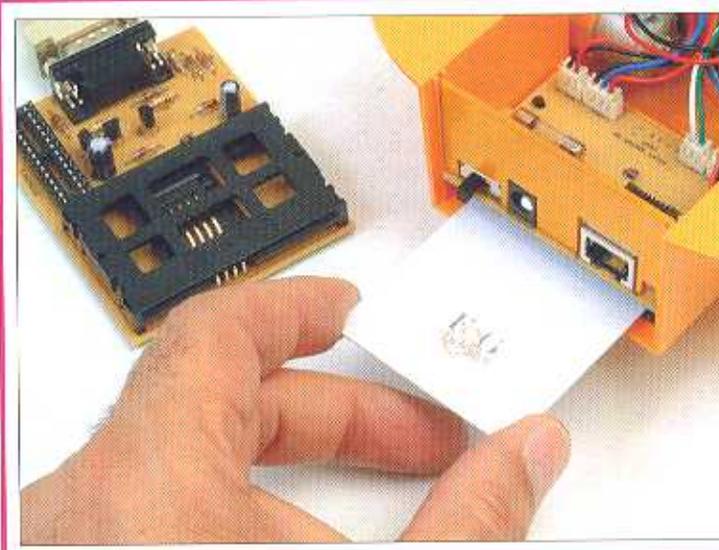


```

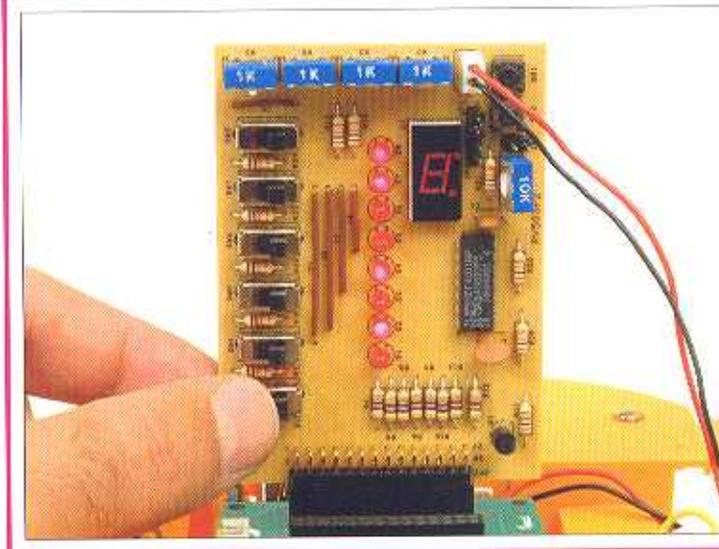
56 .....
57 .....
58 .....
59 .....
60 .....
61 .....
62 .....
63 .....
64 .....
65 .....
66 .....
67 .....
68 .....
69 .....
70 .....
71 .....
72 .....
73 .....
74 .....
75 Delay_var:   bcf     INDIR_1HIF    ;Resetta il flag di overflow
76             movlw  00h        ;Complemento hex. di 195
77             movwf  INDIR_1HIF ;Scrive il 195
78             clrwd  INDIR_1HIF ;aggiorna il 100
79             movlw  INDIR_1HIF ;Scrive il 100
80             goto  Intervallo   ;Inizia il 100ms
81             decfsz Delay_cont,f ;Decrementa il contatore degli intervalli
82             goto  Delay_var    ;Ripete l'intervallo di 50 ms
83 .....
84 .....
85 .....
86 .....
87 .....

```

Al ritorno dalla routine di temporizzazione per annullare i rimbalzi, è sufficiente attendere che l'interruttore RC0 torni a valore "0" per mostrare il dato acquisito sulla variabile "Numero" tramite i diodi, cosa che eseguiamo passando il contenuto della variabile "Numero" alla porta B; dobbiamo visualizzare il numero casuale per 3 secondi e poi spegnere i led. Per fare questo eseguiamo una chiamata a una routine di temporizzazione che attende 3 secondi. Nella routine si realizza una temporizzazione base di 50 ms con il Timer0, e la si ripete 60 volte. 60 è il contenuto caricato nella variabile Delay_cont. Il risultato sarà una temporizzazione di tre secondi dopo la quale spegneremo i led e torneremo ad attendere un altro cambio di stato sull'interruttore SW3 per ripetere il ciclo.



Questo esercizio si trova sul CDROM, sotto il nome di es22.asm. Dopo aver compilato l'esercizio e ottenuto il file es22.hex, lo scriveremo sulla Smartcard. Dobbiamo collegare la scheda di scrittura alla porta seriale del PC ed eseguire il programma ICPROG. Selezioneremo il dispositivo modello 24C16 e apriremo il file es22.hex. Ora selezioneremo il comando programma tutto. Dopo aver programmato la Smartcard la inseriremo nella scheda di alimentazione di Pathfinder con l'orientamento uguale a quello mostrato nell'immagine.



Quando realizzeremo gli esercizi di apprendimento con la scheda di ingressi e uscite, nessun altro sensore deve essere collegato al robot tramite la scheda di controllo o la scheda di interfaccia. Tutti i connettori dei sensori rimarranno senza collegamento. Il microcontroller impiegherà alcuni minuti a leggere il contenuto della Smartcard e ad eseguire l'esercizio. Mediante l'interruttore SW3 inseriremo gli impulsi in grado di generare numeri casuali che visualizzeremo per tre secondi sulla barra dei diodi led. Il jumper JP1 della scheda di ingressi e uscite rimarrà chiuso per attivare i led.

Esercizi di apprendimento

```
1 ;
2 ;Generazione di numeri casuali. Il dado elettronico
3 ;
4 ;Si tratta di generare un numero casuale fra 1 e 6. Quando R00 è a "1", sopra il
5 ;display a 7 segmenti collegato alla porta B, si visualizzano in modo sequenziale
6 ;il numeri da 1 a 6, con intervalli di 0,25". Dopo che R00 passa a livello "0", si visualizzerà
7 ;il numero casuale ottenuto per un tempo di 3". Dopo il display si spegne e la
8 ;sequenza si ripete.
9 ;
10 List g="PR070" ;Tipo di processore
11 Include "PR070.INC" ;Definizione dei registri interni
12 ;
13 Numero equ R00 ;Numero casuale
14 Delay_Cnt equ R01 ;Contatore di intervalli
15 Temporale equ R02 ;Variabile temporale
16 ;
17 org R000
18 ;
19 ;
20 Inizio cldi PORTD ;Cancella i latch di uscita
21 bcf STATUS,RP0 ;Seleziona banca 0
22 cldi TRISB ;Configurazione della porta B come uscita
23 movlw 0xFF
24 movwf TRISB ;Configurazione della porta C come ingresso
25 movlw b'00000111'
26 movwf STATUS,RC0 ;Prescaler da 256 per il TIMER0
27 bcf STATUS,RP0 ;Seleziona banca 0
28 ;
```

Questo esercizio combina quasi tutte le conoscenze sulla programmazione che abbiamo acquisito sino a questo momento.

Si tratta di sviluppare un dado elettronico che genererà numeri casuali fra 1 e 6, visualizzati tramite il display a 7 segmenti della scheda di ingressi e uscite. Avremo bisogno di leggere l'interruttore SW3 per attivare il dado, eliminare l'effetto rimbalzo, gestire una routine di generazione di numeri casuali, fare in modo che il numero sia compreso fra 1 e 6 e realizzare una temporizzazione per fare spegnere il display dopo 3 secondi.

```
20 Loop cldi ;Aggiorna il V01
21 btfsc PORTC,0 ;Si è attivato R00?
22 goto Loop ;Non ancora
23 movf TRISB,W ;Intra al
24 movwf Numero ;Cattura il valore del TRISB (R0 casuale)
25 call Delay_25_ms ;Elimina i rimbalzi
26 ;
27 ;Il numero casuale è, mediante sottrazioni consecutive, diviso per 6. In questo modo
28 ;l'ultimo resto sarà fra 0 e 5 che sarà incrementato di una unità in modo che il
29 ;risultato sia un numero fra 1 e 6
30 ;
31 Dividi movlw 0*6
32 subwf Numero,W ;Sottra 6 al numero casuale
33 movwf Numero ;Numerizzato
34 movlw 0*5
35 btfsc STATUS,C ;Giorna se è minore di 5
36 goto Dividi ;No
37 lscf Numero,W ;Il numero è fra 1 e 6
38 ;
```

Nella routine di inizio realizzeremo le configurazioni dei registri per il programma. La porta C come ingresso, la porta B come uscita e il prescaler a 256 per il Timer0. Nel ciclo "Loop" si attende che l'interruttore RC0 passi a 1. In questo momento si acquisisce il valore del Timer0, portandolo sulla variabile "Numero", detto valore sarà un numero casuale fra 0 e 255. Il dato da visualizzare sul display a 7 segmenti deve essere compreso fra 1 e 6, per questo motivo nella routine "Dividi" si opera con la variabile "Numero" sino a che il suo valore casuale sia compreso in questo intervallo.

```
40 ;Questa sequenza di istruzioni ha per scopo visualizzare sul display i numeri
41 ;da 1 a 6 ad intervalli di 0,25" per dare la sensazione del movimento del dado.
42 ;Questo momento il mantiene sino a che R00 è a "1". Quando passa a "0" si
43 ;visualizza il numero casuale catturato in precedenza tramite il TRISB
44 ;
45 Dado: movlw 0*0
46 movwf Temporale ;Inizializza il contatore del dado
47 ;
48 R00_1 cldi ;Aggiorna il V01
49 btfsc PORTC,0 ;Guarda se R00 è a 1
50 goto Escita ;No, visualizza il N° casuale
51 movf Temporale,W ;Numero da visualizzare
52 call Tabella ;Conversione in BCD
53 movwf PORTB ;Visualizza sul display
54 movlw 0*1
55 movwf Delay_Cnt ;Variabile di temporizzazione
56 call Delay_Var ;Temporizza 0,25"
57 decfz Temporale,F ;Numero successivo
58 goto R00_1
59 goto Dado ;
60 call Delay_25_ms ;Elimina i rimbalzi
61 ;
62 Escita: movf Numero,W ;Recupera il N° casuale
63 call Tabella ;Lo converte in 7 segmenti
64 movwf PORTB ;Uscita sul Display
65 movlw 0*00
66 movwf Delay_Cnt ;Inizializza la variabile di temporizzazione
67 call Delay_Var ;Temporizza 3"
68 cldi PORTD ;Resetta l'uscita
69 goto Loop
```

Sino a quando l'interruttore RC0 non passa a valore 0, non si deve visualizzare il numero casuale sul display a 7 segmenti. Nell'attesa che ciò accada, la routine dado opera sul display visualizzando numeri casuali ad alta velocità; all'interno di questo ciclo si testa RC0, in modo che quando l'interruttore SW3 passa a valore 0, termini il ciclo e si visualizzi il numero casuale generato, scritto nella variabile "Numero", tramite il display a 7 segmenti. Dopo aver visualizzato il numero casuale si temporizzano tre secondi e si spegne il display.



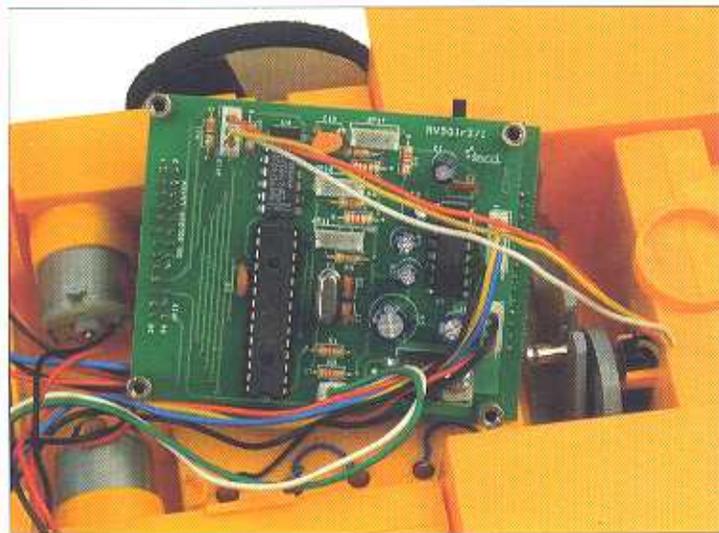
```

70 .....
71 ;Tabella: Questa routine converte il codice binario presente sui 4 bit meno significativi
72 del reg. R nel suo equivalente a 7 segmenti. Il codice a 7 segmenti viene anche scritto
73 sul reg. R
74 .....
75 ;
76 ;Tabella:
77 movl   PCL_F      ;spostamento sopra la tabella
78 movl   0x00111111 ;digit 0
79 movl   0x00001110 ;digit 1
80 movl   0x01111111 ;digit 2
81 movl   0x00001111 ;digit 3
82 movl   0x01110110 ;digit 4
83 movl   0x01111101 ;digit 5
84 .....
85 ;
86 ;Delay 20 ms: questa routine di ritardo ha come obiettivo eliminare l'effetto rimbalzo",
87 ;caratteristica dei componenti elettromeccanici. Realizza un ritardo di 20 ms.
88 ;Se il PIC lavora ad una frequenza di 4 MHz, il 1000 si aggiorna ogni µs. Se vogliamo
89 ;temporizzare 20000 µs (20 ms) con un prescaler di 254, il 1000 dovrà contare 74 eventi
90 ;(20 * 254). Il valore 20 equivale a 100 hex, e dato che il 1000 è ascendente lo dividerò
91 ;caricare con il suo complemento a 1 (001 hex.).
92 .....
93 ;
94 Delay_20_ms: bcf     INTCON,TOIF      ;resetta il flag di overflow del 1000
95              movlw  0x01             ;complemento hex. di 20
96              movwf  1000             ;carica il 1000
97              cwf     INTCON,TOIF      ;aggiorna il 1000
98              btfsc  INTCON,TOIF      ;overflow del 1000?
99              goto   Delay_20_ms_1    ;Non ancora
100             return
    
```

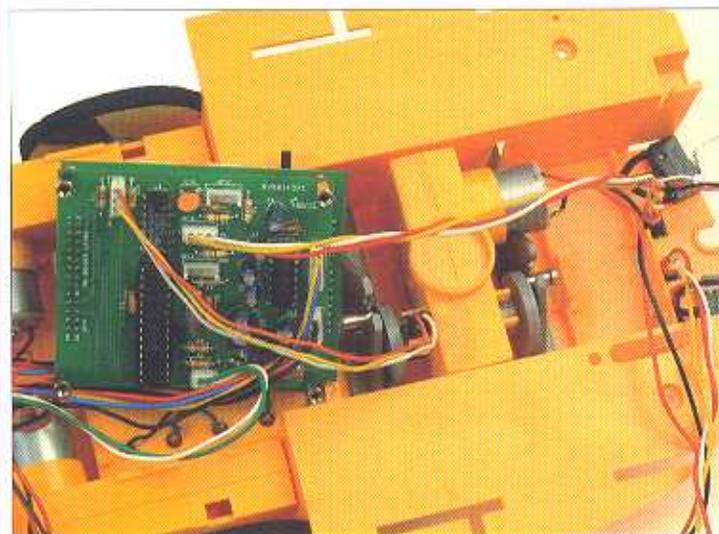
Negli esercizi precedenti abbiamo utilizzato le due routines mostrate nell'immagine. La prima di esse è la routine di conversione dei numeri binari a numeri in codice per il display a 7 segmenti. È necessario richiamare questa routine prima di inviare il numero casuale alla porta B. L'altra funzione è una temporizzazione di 20 ms utilizzata per eliminare l'effetto rimbalzo dell'interruttore.

```

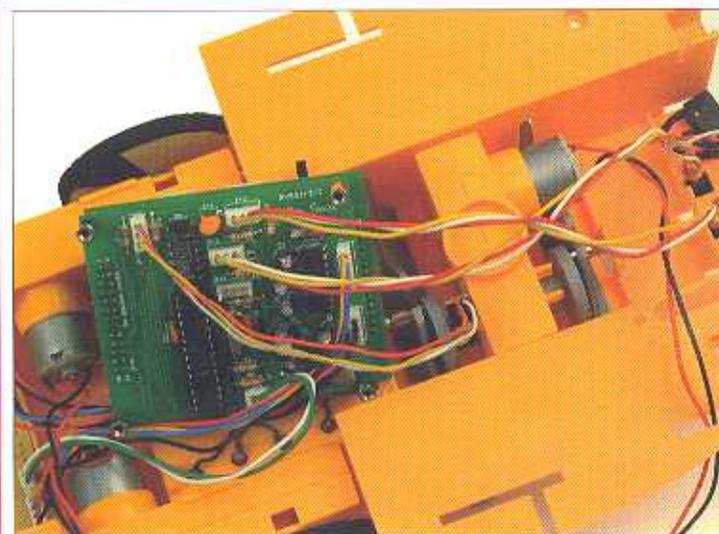
103 .....
104 ;
105 ;Delay_var: Questa routine di utilizzo generale realizza una temporizzazione variabile
106 ;fra 50 ms e 12,0". Si utilizza un prescaler di 256 e sul 1000 si carica 195.
107 ;La velocità di lavoro è di 4 MHz e quindi il 1000 si incrementa ogni µs. In
108 ;questo modo, il 1000 deve contare 745 eventi che, con un prescaler di 128 danno un
109 ;intervallo totale di 50000 µs = 50 ms (195 * 256). Il valore 195 lo dobbiamo esprimere
110 ;in hex. (c3) e, dato che il 1000 è ascendente dovremo caricare il suo complemento a 1 (3c hex.)
111 ;Questo intervallo di 50 ms si ripete tante volte quante indicate dalla variabile "Delay_cont".
112 ;Per questo il ritardo minimo è di 50 ms ("Delay_cont=1") e quello massimo di 12,0 sec.
113 ;("Delay_cont=255").
114 .....
115 ;
116 Delay_var: bcf     INTCON,TOIF      ;resetta il flag di overflow
117              movlw  0xc3            ;complemento hex. di 195
118              movwf  1000            ;carica il 1000
119              cwf     INTCON,TOIF      ;aggiorna il 1000
120              btfsc  INTCON,TOIF      ;overflow del 1000?
121              goto   Intervallo       ;Non ancora
122 Intervallo: decf   Delay_cont,F      ;decrementa il contatore degli intervalli
123              goto   Delay_var        ;ripete l'intervallo di 50 ms
124 .....
125 ;
126 ;
127 ;
128 ;
129 ;
130 ;
131 ;
132 ;
133 ;
134 ;
135 ;
136 ;
137 ;
138 ;
139 ;
140 ;
141 ;
142 ;
143 ;
144 ;
145 ;
146 ;
147 ;
148 ;
149 ;
150 ;
151 ;
152 ;
153 ;
154 ;
155 ;
156 ;
157 ;
158 ;
159 ;
160 ;
161 ;
162 ;
163 ;
164 ;
165 ;
166 ;
167 ;
168 ;
169 ;
170 ;
171 ;
172 ;
173 ;
174 ;
175 ;
176 ;
177 ;
178 ;
179 ;
180 ;
181 ;
182 ;
183 ;
184 ;
185 ;
186 ;
187 ;
188 ;
189 ;
190 ;
191 ;
192 ;
193 ;
194 ;
195 ;
196 ;
197 ;
198 ;
199 ;
200 ;
201 ;
202 ;
203 ;
204 ;
205 ;
206 ;
207 ;
208 ;
209 ;
210 ;
211 ;
212 ;
213 ;
214 ;
215 ;
216 ;
217 ;
218 ;
219 ;
220 ;
221 ;
222 ;
223 ;
224 ;
225 ;
226 ;
227 ;
228 ;
229 ;
230 ;
231 ;
232 ;
233 ;
234 ;
235 ;
236 ;
237 ;
238 ;
239 ;
240 ;
241 ;
242 ;
243 ;
244 ;
245 ;
246 ;
247 ;
248 ;
249 ;
250 ;
251 ;
252 ;
253 ;
254 ;
255 ;
256 ;
257 ;
258 ;
259 ;
260 ;
261 ;
262 ;
263 ;
264 ;
265 ;
266 ;
267 ;
268 ;
269 ;
270 ;
271 ;
272 ;
273 ;
274 ;
275 ;
276 ;
277 ;
278 ;
279 ;
280 ;
281 ;
282 ;
283 ;
284 ;
285 ;
286 ;
287 ;
288 ;
289 ;
290 ;
291 ;
292 ;
293 ;
294 ;
295 ;
296 ;
297 ;
298 ;
299 ;
300 ;
301 ;
302 ;
303 ;
304 ;
305 ;
306 ;
307 ;
308 ;
309 ;
310 ;
311 ;
312 ;
313 ;
314 ;
315 ;
316 ;
317 ;
318 ;
319 ;
320 ;
321 ;
322 ;
323 ;
324 ;
325 ;
326 ;
327 ;
328 ;
329 ;
330 ;
331 ;
332 ;
333 ;
334 ;
335 ;
336 ;
337 ;
338 ;
339 ;
340 ;
341 ;
342 ;
343 ;
344 ;
345 ;
346 ;
347 ;
348 ;
349 ;
350 ;
351 ;
352 ;
353 ;
354 ;
355 ;
356 ;
357 ;
358 ;
359 ;
360 ;
361 ;
362 ;
363 ;
364 ;
365 ;
366 ;
367 ;
368 ;
369 ;
370 ;
371 ;
372 ;
373 ;
374 ;
375 ;
376 ;
377 ;
378 ;
379 ;
380 ;
381 ;
382 ;
383 ;
384 ;
385 ;
386 ;
387 ;
388 ;
389 ;
390 ;
391 ;
392 ;
393 ;
394 ;
395 ;
396 ;
397 ;
398 ;
399 ;
400 ;
401 ;
402 ;
403 ;
404 ;
405 ;
406 ;
407 ;
408 ;
409 ;
410 ;
411 ;
412 ;
413 ;
414 ;
415 ;
416 ;
417 ;
418 ;
419 ;
420 ;
421 ;
422 ;
423 ;
424 ;
425 ;
426 ;
427 ;
428 ;
429 ;
430 ;
431 ;
432 ;
433 ;
434 ;
435 ;
436 ;
437 ;
438 ;
439 ;
440 ;
441 ;
442 ;
443 ;
444 ;
445 ;
446 ;
447 ;
448 ;
449 ;
450 ;
451 ;
452 ;
453 ;
454 ;
455 ;
456 ;
457 ;
458 ;
459 ;
460 ;
461 ;
462 ;
463 ;
464 ;
465 ;
466 ;
467 ;
468 ;
469 ;
470 ;
471 ;
472 ;
473 ;
474 ;
475 ;
476 ;
477 ;
478 ;
479 ;
480 ;
481 ;
482 ;
483 ;
484 ;
485 ;
486 ;
487 ;
488 ;
489 ;
490 ;
491 ;
492 ;
493 ;
494 ;
495 ;
496 ;
497 ;
498 ;
499 ;
500 ;
501 ;
502 ;
503 ;
504 ;
505 ;
506 ;
507 ;
508 ;
509 ;
510 ;
511 ;
512 ;
513 ;
514 ;
515 ;
516 ;
517 ;
518 ;
519 ;
520 ;
521 ;
522 ;
523 ;
524 ;
525 ;
526 ;
527 ;
528 ;
529 ;
530 ;
531 ;
532 ;
533 ;
534 ;
535 ;
536 ;
537 ;
538 ;
539 ;
540 ;
541 ;
542 ;
543 ;
544 ;
545 ;
546 ;
547 ;
548 ;
549 ;
550 ;
551 ;
552 ;
553 ;
554 ;
555 ;
556 ;
557 ;
558 ;
559 ;
560 ;
561 ;
562 ;
563 ;
564 ;
565 ;
566 ;
567 ;
568 ;
569 ;
570 ;
571 ;
572 ;
573 ;
574 ;
575 ;
576 ;
577 ;
578 ;
579 ;
580 ;
581 ;
582 ;
583 ;
584 ;
585 ;
586 ;
587 ;
588 ;
589 ;
590 ;
591 ;
592 ;
593 ;
594 ;
595 ;
596 ;
597 ;
598 ;
599 ;
600 ;
601 ;
602 ;
603 ;
604 ;
605 ;
606 ;
607 ;
608 ;
609 ;
610 ;
611 ;
612 ;
613 ;
614 ;
615 ;
616 ;
617 ;
618 ;
619 ;
620 ;
621 ;
622 ;
623 ;
624 ;
625 ;
626 ;
627 ;
628 ;
629 ;
630 ;
631 ;
632 ;
633 ;
634 ;
635 ;
636 ;
637 ;
638 ;
639 ;
640 ;
641 ;
642 ;
643 ;
644 ;
645 ;
646 ;
647 ;
648 ;
649 ;
650 ;
651 ;
652 ;
653 ;
654 ;
655 ;
656 ;
657 ;
658 ;
659 ;
660 ;
661 ;
662 ;
663 ;
664 ;
665 ;
666 ;
667 ;
668 ;
669 ;
670 ;
671 ;
672 ;
673 ;
674 ;
675 ;
676 ;
677 ;
678 ;
679 ;
680 ;
681 ;
682 ;
683 ;
684 ;
685 ;
686 ;
687 ;
688 ;
689 ;
690 ;
691 ;
692 ;
693 ;
694 ;
695 ;
696 ;
697 ;
698 ;
699 ;
700 ;
701 ;
702 ;
703 ;
704 ;
705 ;
706 ;
707 ;
708 ;
709 ;
710 ;
711 ;
712 ;
713 ;
714 ;
715 ;
716 ;
717 ;
718 ;
719 ;
720 ;
721 ;
722 ;
723 ;
724 ;
725 ;
726 ;
727 ;
728 ;
729 ;
730 ;
731 ;
732 ;
733 ;
734 ;
735 ;
736 ;
737 ;
738 ;
739 ;
740 ;
741 ;
742 ;
743 ;
744 ;
745 ;
746 ;
747 ;
748 ;
749 ;
750 ;
751 ;
752 ;
753 ;
754 ;
755 ;
756 ;
757 ;
758 ;
759 ;
760 ;
761 ;
762 ;
763 ;
764 ;
765 ;
766 ;
767 ;
768 ;
769 ;
770 ;
771 ;
772 ;
773 ;
774 ;
775 ;
776 ;
777 ;
778 ;
779 ;
780 ;
781 ;
782 ;
783 ;
784 ;
785 ;
786 ;
787 ;
788 ;
789 ;
790 ;
791 ;
792 ;
793 ;
794 ;
795 ;
796 ;
797 ;
798 ;
799 ;
800 ;
801 ;
802 ;
803 ;
804 ;
805 ;
806 ;
807 ;
808 ;
809 ;
810 ;
811 ;
812 ;
813 ;
814 ;
815 ;
816 ;
817 ;
818 ;
819 ;
820 ;
821 ;
822 ;
823 ;
824 ;
825 ;
826 ;
827 ;
828 ;
829 ;
830 ;
831 ;
832 ;
833 ;
834 ;
835 ;
836 ;
837 ;
838 ;
839 ;
840 ;
841 ;
842 ;
843 ;
844 ;
845 ;
846 ;
847 ;
848 ;
849 ;
850 ;
851 ;
852 ;
853 ;
854 ;
855 ;
856 ;
857 ;
858 ;
859 ;
860 ;
861 ;
862 ;
863 ;
864 ;
865 ;
866 ;
867 ;
868 ;
869 ;
870 ;
871 ;
872 ;
873 ;
874 ;
875 ;
876 ;
877 ;
878 ;
879 ;
880 ;
881 ;
882 ;
883 ;
884 ;
885 ;
886 ;
887 ;
888 ;
889 ;
890 ;
891 ;
892 ;
893 ;
894 ;
895 ;
896 ;
897 ;
898 ;
899 ;
900 ;
901 ;
902 ;
903 ;
904 ;
905 ;
906 ;
907 ;
908 ;
909 ;
910 ;
911 ;
912 ;
913 ;
914 ;
915 ;
916 ;
917 ;
918 ;
919 ;
920 ;
921 ;
922 ;
923 ;
924 ;
925 ;
926 ;
927 ;
928 ;
929 ;
930 ;
931 ;
932 ;
933 ;
934 ;
935 ;
936 ;
937 ;
938 ;
939 ;
940 ;
941 ;
942 ;
943 ;
944 ;
945 ;
946 ;
947 ;
948 ;
949 ;
950 ;
951 ;
952 ;
953 ;
954 ;
955 ;
956 ;
957 ;
958 ;
959 ;
960 ;
961 ;
962 ;
963 ;
964 ;
965 ;
966 ;
967 ;
968 ;
969 ;
970 ;
971 ;
972 ;
973 ;
974 ;
975 ;
976 ;
977 ;
978 ;
979 ;
980 ;
981 ;
982 ;
983 ;
984 ;
985 ;
986 ;
987 ;
988 ;
989 ;
990 ;
991 ;
992 ;
993 ;
994 ;
995 ;
996 ;
997 ;
998 ;
999 ;
1000 ;
1001 ;
1002 ;
1003 ;
1004 ;
1005 ;
1006 ;
1007 ;
1008 ;
1009 ;
1010 ;
1011 ;
1012 ;
1013 ;
1014 ;
1015 ;
1016 ;
1017 ;
1018 ;
1019 ;
1020 ;
1021 ;
1022 ;
1023 ;
1024 ;
1025 ;
1026 ;
1027 ;
1028 ;
1029 ;
1030 ;
1031 ;
1032 ;
1033 ;
1034 ;
1035 ;
1036 ;
1037 ;
1038 ;
1039 ;
1040 ;
1041 ;
1042 ;
1043 ;
1044 ;
1045 ;
1046 ;
1047 ;
1048 ;
1049 ;
1050 ;
1051 ;
1052 ;
1053 ;
1054 ;
1055 ;
1056 ;
1057 ;
1058 ;
1059 ;
1060 ;
1061 ;
1062 ;
1063 ;
1064 ;
1065 ;
1066 ;
1067 ;
1068 ;
1069 ;
1070 ;
1071 ;
1072 ;
1073 ;
1074 ;
1075 ;
1076 ;
1077 ;
1078 ;
1079 ;
1080 ;
1081 ;
1082 ;
1083 ;
1084 ;
1085 ;
1086 ;
1087 ;
1088 ;
1089 ;
1090 ;
1091 ;
1092 ;
1093 ;
1094 ;
1095 ;
1096 ;
1097 ;
1098 ;
1099 ;
1100 ;
1101 ;
1102 ;
1103 ;
1104 ;
1105 ;
1106 ;
1107 ;
1108 ;
1109 ;
1110 ;
1111 ;
1112 ;
1113 ;
1114 ;
1115 ;
1116 ;
1117 ;
1118 ;
1119 ;
1120 ;
1121 ;
1122 ;
1123 ;
1124 ;
1125 ;
1126 ;
1127 ;
1128 ;
1129 ;
1130 ;
1131 ;
1132 ;
1133 ;
1134 ;
1135 ;
1136 ;
1137 ;
1138 ;
1139 ;
1140 ;
1141 ;
1142 ;
1143 ;
1144 ;
1145 ;
1146 ;
1147 ;
1148 ;
1149 ;
1150 ;
1151 ;
1152 ;
1153 ;
1154 ;
1155 ;
1156 ;
1157 ;
1158 ;
1159 ;
1160 ;
1161 ;
1162 ;
1163 ;
1164 ;
1165 ;
1166 ;
1167 ;
1168 ;
1169 ;
1170 ;
1171 ;
1172 ;
1173 ;
1174 ;
1175 ;
1176 ;
1177 ;
1178 ;
1179 ;
1180 ;
1181 ;
1182 ;
1183 ;
1184 ;
1185 ;
1186 ;
1187 ;
1188 ;
1189 ;
1190 ;
1191 ;
1192 ;
1193 ;
1194 ;
1195 ;
1196 ;
1197 ;
1198 ;
1199 ;
1200 ;
1201 ;
1202 ;
1203 ;
1204 ;
1205 ;
1206 ;
1207 ;
1208 ;
1209 ;
1210 ;
1211 ;
1212 ;
1213 ;
1214 ;
1215 ;
1216 ;
1217 ;
1218 ;
1219 ;
1220 ;
1221 ;
1222 ;
1223 ;
1224 ;
1225 ;
1226 ;
1227 ;
1228 ;
1229 ;
1230 ;
1231 ;
1232 ;
1233 ;
1234 ;
1235 ;
1236 ;
1237 ;
1238 ;
1239 ;
1240 ;
1241 ;
1242 ;
1243 ;
1244 ;
1245 ;
1246 ;
1247 ;
1248 ;
1249 ;
1250 ;
1251 ;
1252 ;
1253 ;
1254 ;
1255 ;
1256 ;
1257 ;
1258 ;
1259 ;
1260 ;
1261 ;
1262 ;
1263 ;
1264 ;
1265 ;
1266 ;
1267 ;
1268 ;
1269 ;
1270 ;
1271 ;
1272 ;
1273 ;
1274 ;
1275 ;
1276 ;
1277 ;
1278 ;
1279 ;
1280 ;
1281 ;
1282 ;
1283 ;
1284 ;
1285 ;
1286 ;
1287 ;
1288 ;
1289 ;
1290 ;
1291 ;
1292 ;
1293 ;
1294 ;
1295 ;
1296 ;
1297 ;
1298 ;
1299 ;
1300 ;
1301 ;
1302 ;
1303 ;
1304 ;
1305 ;
1306 ;
1307 ;
1308 ;
1309 ;
1310 ;
1311 ;
1312 ;
1313 ;
1314 ;
1315 ;
1316 ;
1317 ;
1318 ;
1319 ;
1320 ;
1321 ;
1322 ;
1323 ;
1324 ;
1325 ;
1326 ;
1327 ;
1328 ;
1329 ;
1330 ;
1331 ;
1332 ;
1333 ;
1334 ;
1335 ;
1336 ;
1337 ;
1338 ;
1339 ;
1340 ;
1341 ;
1342 ;
1343 ;
1344 ;
1345 ;
1346 ;
1347 ;
1348 ;
1349 ;
1350 ;
1351 ;
1352 ;
1353 ;
1354 ;
1355 ;
1356 ;
1357 ;
1358 ;
1359 ;
1360 ;
1361 ;
1362 ;
1363 ;
1364 ;
1365 ;
1366 ;
1367 ;
1368 ;
1369 ;
1370 ;
1371 ;
1372 ;
1373 ;
1374 ;
1375 ;
1376 ;
1377 ;
1378 ;
1379 ;
1380 ;
1381 ;
1382 ;
1383 ;
1384 ;
1385 ;
1386 ;
1387 ;
1388 ;
1389 ;
1390 ;
1391 ;
1392 ;
1393 ;
1394 ;
1395 ;
1396 ;
1397 ;
1398 ;
1399 ;
1400 ;
1401 ;
1402 ;
1403 ;
1404 ;
1405 ;
1406 ;
1407 ;
1408 ;
1409 ;
1410 ;
1411 ;
1412 ;
1413 ;
1414 ;
1415 ;
1416 ;
1417 ;
1418 ;
1419 ;
1420 ;
1421 ;
1422 ;
1423 ;
1424 ;
1425 ;
1426 ;
1427 ;
1428 ;
1429 ;
1430 ;
1431 ;
1432 ;
1433 ;
1434 ;
1435 ;
1436 ;
1437 ;
1438 ;
1439 ;
1440 ;
1441 ;
1442 ;
1443 ;
1444 ;
1445 ;
1446 ;
1447 ;
1448 ;
1449 ;
1450 ;
1451 ;
1452 ;
1453 ;
1454 ;
1455 ;
1456 ;
1457 ;
1458 ;
1459 ;
1460 ;
1461 ;
1462 ;
1463 ;
1464 ;
1465 ;
1466 ;
1467 ;
1468 ;
1469 ;
1470 ;
1471 ;
1472 ;
1473 ;
1474 ;
1475 ;
1476 ;
1477 ;
1478 ;
1479 ;
1480 ;
1481 ;
1482 ;
1483 ;
1484 ;
1485 ;
1486 ;
1487 ;
1488 ;
1489 ;
1490 ;
1491 ;
1492 ;
1493 ;
1494 ;
1495 ;
1496 ;
1497 ;
1498 ;
1499 ;
1500 ;
1501 ;
1502 ;
1503 ;
1504 ;
1505 ;
1506 ;
1507 ;
1508 ;
1509 ;
1510 ;
1511 ;
1512 ;
1513 ;
1514 ;
1515 ;
1516 ;
1517 ;
1518 ;
1519 ;
1520 ;
1521 ;
1522 ;
1523 ;
1524 ;
1525 ;
1526 ;
1527 ;
1528 ;
1529 ;
1530 ;
1531 ;
1532 ;
1533 ;
1534 ;
1535 ;
1536 ;
1537 ;
1538 ;
1539 ;
1540 ;
1541 ;
1542 ;
1543 ;
1544 ;
1545 ;
1546 ;
1547 ;
1548 ;
1549 ;
1550 ;
1551 ;
1552 ;
1553 ;
1554 ;
1555 ;
1556 ;
1557 ;
1558 ;
1559 ;
1560 ;
1561 ;
1562 ;
1563 ;
1564 ;
1565 ;
1566 ;
1567 ;
1568 ;
1569 ;
1570 ;
1571 ;
1572 ;
1573 ;
1574 ;
1575 ;
1576 ;
1577 ;
1578 ;
1579 ;
1580 ;
1581 ;
1582 ;
1583 ;
1584 ;
1585 ;
1586 ;
1587 ;
1588 ;
1589 ;
1590 ;
1591 ;
1592 ;
1593 ;
1594 ;
1595 ;
1596 ;
1597 ;
1598 ;
1599 ;
1600 ;
1601 ;
1602 ;
1603 ;
1604 ;
1605 ;
1606 ;
1607 ;
1608 ;
1609 ;
1610 ;
1611 ;
1612 ;
1613 ;
1614 ;
1615 ;
1616 ;
1617 ;
1618 ;
1619 ;
1620 ;
1621 ;
1622 ;
1623 ;
1624 ;
1625 ;
1626 ;
1627 ;
1628 ;
1629 ;
1630 ;
1631 ;
1632 ;
1633 ;
1634 ;
1635 ;
1636 ;
1637 ;
1638 ;
1639 ;
1640 ;
1641 ;
1642 ;
1643 ;
1644 ;
1645 ;
1646 ;
1647 ;
1648 ;
1649 ;
1650 ;
1651 ;
1652 ;
1653 ;
1654 ;
1655 ;
1656 ;
1657 ;
1658 ;
1659 ;
1660 ;
1661 ;
1662 ;
1663 ;
1664 ;
1665 ;
1666 ;
1667 ;
1668 ;
1669 ;
1670 ;
1671 ;
1672 ;
1673 ;
1674 ;
1675 ;
1676 ;
1677 ;
1678 ;
1679 ;
1680 ;
1681 ;
1682 ;
1683 ;
1684 ;
1685 ;
1686 ;
1687 ;
1688 ;
1689 ;
1690 ;
1691 ;
1692 ;
1693 ;
1694 ;
1695 ;
1696 ;
1697 ;
1698 ;
1699 ;
1700 ;
1701 ;
1702 ;
1703 ;
1704 ;
1705 ;
1706 ;
1707 ;
1708 ;
1709 ;
1710 ;
1711 ;
1712 ;
1713 ;
1714 ;
1715 ;
1716 ;
1717 ;
1718 ;
1719 ;
1720 ;
1721 ;
1722 ;
1723 ;
1724 ;
1725 ;
1726 ;
1727 ;
1728 ;
1729 ;
1730 ;
1731 ;
1732 ;
1733 ;
1734 ;
1735 ;
1736 ;
1737 ;
1738 ;
1739 ;
1740 ;
1741 ;
1742 ;
1743 ;
1744 ;
1745 ;
1746 ;
1747 ;
1748 ;
1749 ;
1750 ;
1751 ;
1752 ;
1753 ;
1754 ;
1755 ;
1756 ;
1757 ;
1758 ;
1759 ;
1760 ;
1761 ;
1762 ;
1763 ;
1764 ;
1765 ;
1766 ;
1767 ;
1768 ;
1769 ;
1770 ;
1771 ;
1772 ;
1773 ;
1774 ;
1775 ;
1776 ;
1777 ;
1778 ;
1779 ;
1780 ;
1781 ;
1782 ;
1783 ;
1784 ;
1785 ;
1786 ;
1787 ;
1788 ;
1789 ;
1790 ;
1791 ;
1792 ;
1793 ;
1794 ;
1795 ;
1796 ;
1797 ;
1798 ;
1799 ;
1800 ;
1801 ;
1802 ;
1803 ;
1804 ;
1805 ;
1806 ;
1807 ;
1808 ;
1809 ;
1810 ;
1811 ;
1812 ;
1813 ;
1814 ;
1815 ;
1816 ;
1817 ;
1818 ;
1819 ;
1820 ;
1821 ;
1822 ;
1823 ;
1824 ;
1825 ;
1826 ;
1827 ;
1828 ;
1829 ;
1830 ;
1831 ;
1832 ;
1833 ;
1834 ;
1835 ;
1836 ;
1837 ;
1838 ;
1839 ;
1840 ;
1841 ;
1842 ;
1843 ;
1844 ;
1845 ;
1846 ;
1847 ;
1848 ;
1849 ;
1850 ;
1851 ;
1852 ;
1853 ;
1854 ;
1855 ;
1856 ;
1857 ;
1858 ;
1859 ;
1860 ;
1861 ;
1862 ;
1863 ;
1864 ;
1865 ;
1866 ;
1867 ;
1868 ;
1869 ;
1870 ;
1871 ;
1872 ;
1873 ;
1874 ;
1875 ;
1876 ;
1877 ;
1878 ;
1879 ;
1880 ;
1881 ;
1882 ;
1883 ;
1884 ;
1885 ;
1886 ;
1887 ;
1888 ;
1889 ;
1890 ;
1891 ;
1892 ;
1893 ;
1894 ;
1895 ;
1896 ;
1897 ;
1898 ;
1899 ;
1900 ;
1901 ;
1902 ;
1903 ;
1904 ;
1905 ;
1906 ;
1907 ;
1908 ;
1909 ;
1910 ;
1911 ;
1912 ;
1913 ;
1914 ;
1915 ;
1916 ;
1917 ;
1918 ;
1919 ;
1920 ;
1921 ;
1922 ;
1923 ;
1924 ;
1925 ;
1926 ;
1927 ;
1928 ;
1929 ;
1930 ;
1931 ;
1932 ;
1933 ;
1934 ;
1935 ;
1936 ;
1937 ;
1938 ;
1939 ;
1940 ;
1941 ;
1942 ;
1943 ;
1944 ;
1945 ;
1946 ;
1947 ;
1948 ;
1949 ;
1950 ;
1951 ;
1952 ;
1953 ;
1954 ;
1955 ;
1956 ;
1957 ;
1958 ;
1959 ;
1960 ;
1961 ;
1962 ;
1963 ;
1964 ;
1965 ;
1966 ;
1967 ;
1968 ;
1969 ;
1970 ;
1971 ;
1972 ;
1973 ;
1974 ;
1975 ;
1976 ;
1977 ;
1978 ;
1979 ;
1980 ;
1981 ;
1982 ;
1983 ;
1984 ;
1985 ;
1986 ;
1987 ;
1988 ;
1989 ;
1990 ;
1991 ;
1992 ;
1993 ;
1994 ;
1995 ;
1996 ;
1997 ;
1998 ;
1999 ;
2000 ;
2001 ;
2002 ;
2003 ;
2004 ;
2005 ;
2006 ;
2007 ;
2008 ;
2009 ;
2010 ;
2011 ;
2012 ;
2013 ;
2014 ;
2015 ;
2016 ;
2017 ;
2018 ;
2019 ;
2020 ;
2021 ;
2022 ;
2023 ;
2024 ;
2025 ;
2026 ;
2027 ;
2028 ;
2029 ;
2030 ;
2031 ;
2032 ;
2033 ;
2034 ;
2035 ;
2036 ;
2037 ;
2038 ;
2039 ;
2040 ;
2041 ;
2042 ;
2043 ;
2044 ;
2045 ;
2046 ;
2047 ;
2048 ;
2049 ;
2050 ;
2051 ;
2052 ;
2053 ;
2054 ;
2055 ;
2056 ;
2057 ;
2058 ;
2059 ;
2060 ;
2061 ;
2062 ;
2063 ;
2064 ;
2065 ;
2066 ;
2067 ;
2068 ;
2069 ;
2070 ;
2071 ;
2072 ;
2073 ;
2074 ;
2075 ;
2076 ;
2077 ;
2078 ;
2079 ;
2080 ;
2081 ;
2082 ;
2083 ;
2084 ;
2085 ;
2086 ;
2087 ;
2088 ;
2089 ;
2090 ;
2091 ;
2092 ;
2093 ;
2094 ;
2095 ;
2096 ;
2097 ;
2098 ;
2099 ;
2100 ;
2101 ;
2102 ;
2103 ;
2104 ;
2105 ;
2106 ;
2107 ;
2108 ;
2109 ;
2110 ;
2111 ;
21
```



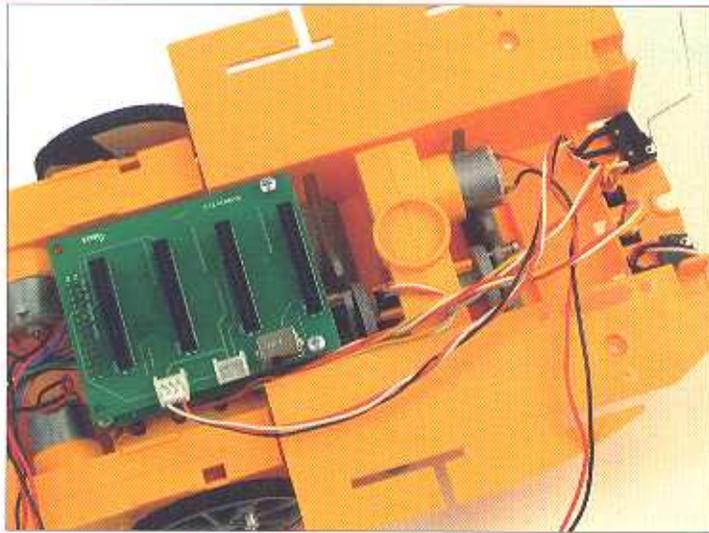
Inizieremo realizzando alcuni programmi di apprendimento con i motori e sensori di Pathfinder. Per questo avremo bisogno di utilizzare la scheda di potenza che abbiamo già montato. Però prima di realizzare i programmi il primo passo consisterà nel collegare adeguatamente i sensori alle diverse schede del robot. Il primo sensore da collegare sarà il sensore ottico CNY70, che abbiamo montato per il controllo del motore centrale e che si collegherà sul connettore JP18 della scheda di controllo.



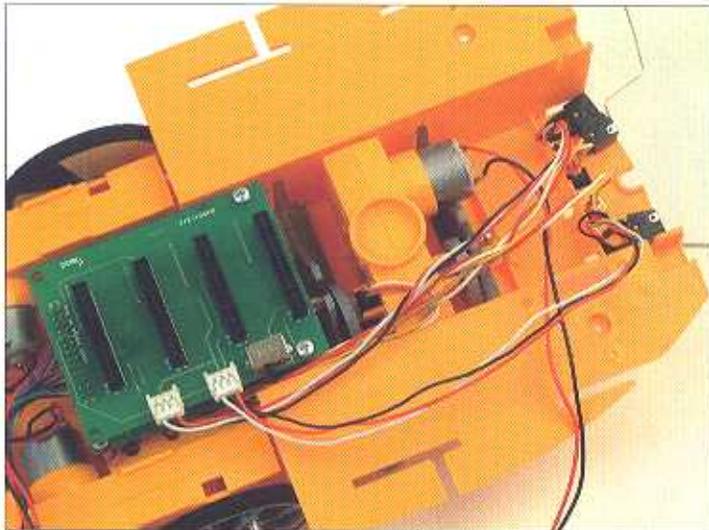
Collegheremo ora gli altri due sensori ottici che abbiamo montato nella parte anteriore del telaio del robot. Il riferimento per distinguere se il sensore è sinistro o destro consiste nel guardare il robot dalla sua parte posteriore. Il sensore situato nel foro sinistro del telaio si collegherà al connettore JP12 della scheda di controllo.



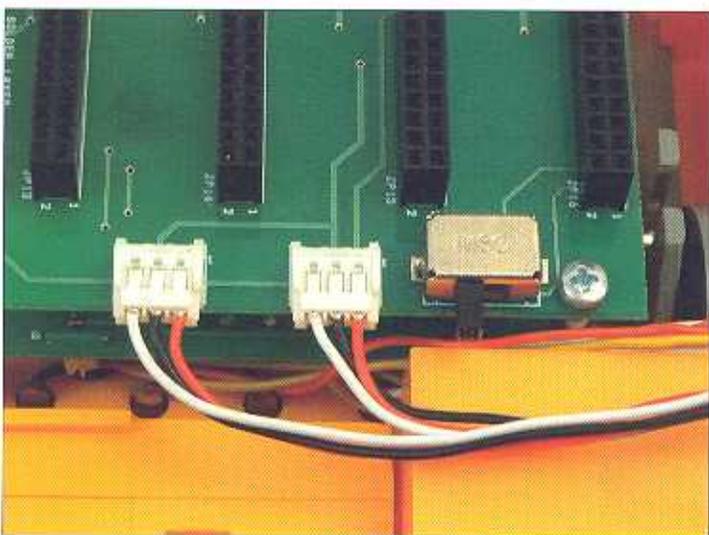
Per ultimo collegheremo il sensore ottico CNY70, che è l'altro sensore della parte anteriore del telaio ed è montato sulla parte destra. Questo sensore verrà collegato sul connettore JP7 della scheda di controllo. I connettori hanno solo un verso di inserzione corretto, e dispongono di alcuni impedimenti meccanici mediante i quali è permessa l'inserzione solo nell'orientamento adeguato.



Ora collegheremo i sensori meccanici tipo finecorsa. Per distinguere il sensore sinistro dal destro utilizzeremo lo stesso riferimento dei sensori ottici, cioè osserveremo il robot dalla sua parte posteriore. Il sensore finecorsa sinistro verrà collegato al connettore JP7 posizionato sulla scheda di interfaccia.



Infine l'ultimo sensore tipo finecorsa è quello situato sulla parte destra del robot. Si collegherà al connettore JP8 della scheda di interfaccia. Questi connettori così come i sensori ottici CNY70, accettano solo il collegamento con l'orientamento corretto. Il finecorsa sinistro comunicherà con il pin RA1 del microcontroller, mentre il finecorsa destro invierà il suo stato al pin RA2.



A lato dei connettori per i finecorsa sulla scheda di interfaccia troviamo il commutatore a due vie SW3. Questo commutatore serve a fare in modo che il robot funzioni con i sensori meccanici tipo finecorsa o con i sensori a ultrasuoni. Nei primi esercizi utilizzeremo i finecorsa. Per fare in modo che i finecorsa siano attivi la posizione del commutatore deve coincidere con quella mostrata nell'immagine.

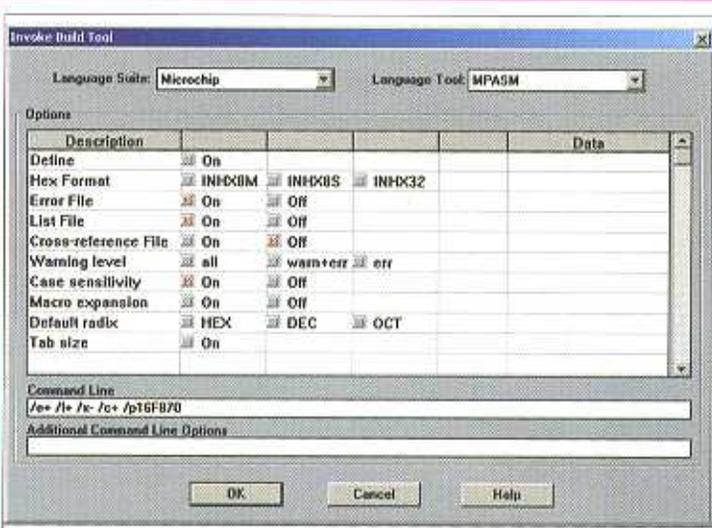
Esercizi con motori e sensori

```
1 :
2 :pot1.asm
3 ;Programma che esegue l'accensione o lo spegnimento dei motori posteriori
4 ;in funzione dello stato del finecorsa sinistro (connettore JP7, segnale su RA1)
5
6 LIST p=16F870 ;Tipo di processore
7 include "p16F870.HC" ;definizione dei registri interni
8
9 ORG 0x00
10
11 inizio
12 bsf STATUS,RP0 ;selezione del banco 1
13 movlw 0x07
14 movwf 80C000 ;Porta B come ingresso digitale
15 cldf TRISB ;la Porta B si configura come uscita
16 bcf STATUS,RP0 ;selezione del banco 0
17 cldf PORTB ;si spengono i motori
```

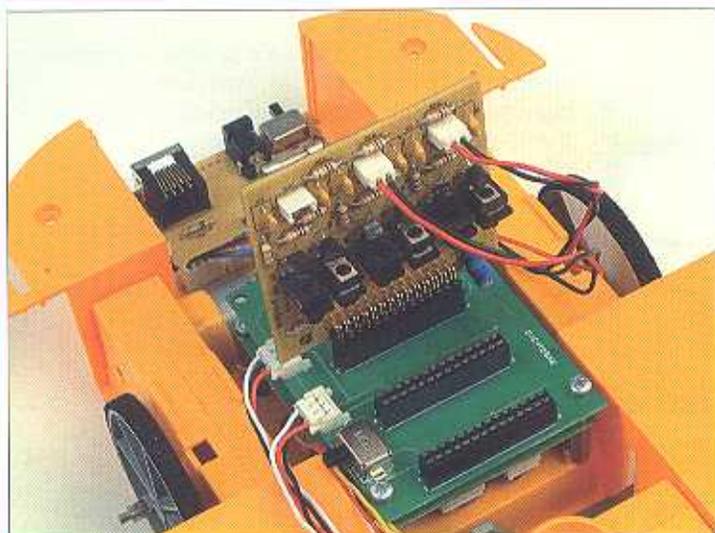
Il primo esercizio che realizzeremo consisterà nell'accensione e spegnimento dei motori posteriori del robot, mediante lo stato del finecorsa sinistro. Quando il finecorsa si attiva i motori cominceranno a funzionare e quando si trova disattivato si fermeranno. Il finecorsa sinistro invierà i segnali al piedino RA1 del microcontroller, i motori si controllano mediante la porta B del microcontroller. Per questo motivo la porta A si configura come ingresso e la porta B come uscita.

```
18
19 ;RB2 e RB3 sono per il motore della ruota sinistra
20 ;RB4: 0
21 ;RB2: 1 -> Motore avanza
22 ;RB4 e RB5 sono per il motore della ruota destra
23 ;RB4: 1
24 ;RB5: 0 -> Motore avanza
25
26 Loop btfss PORTA, 1 ;Si legge un '1' o uno '0' su RA1
27 goto OFF
28 goto ON
29
30 OFF cldf PORTB
31 goto Loop
32
33 ON movlw b'00011000'
34 movwf PORTB
35 goto Loop ;Ciclo infinito
36
37 END ;Fine del programma sorgente
```

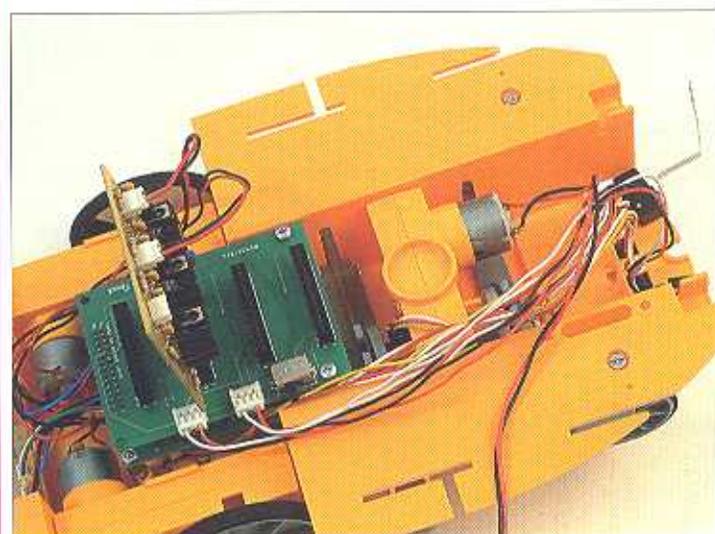
Nell'immagine del programma possiamo vedere la connessione dei motori posteriori alla porta B. Per accendere il motore sinistro e farlo muovere in avanti è necessario inviare un "1" logico tramite RB3 e uno "0" tramite RB2. Per accendere il motore destro nel verso dell'avanzamento bisogna inviare un "1" tramite RB4 ed uno "0" tramite RB5. Il ciclo principale del programma testa lo stato del finecorsa, quando il finecorsa si chiude invia un "1" logico al pin RA1 del microcontroller e quando rimane aperto uno "0". Nel momento in cui il finecorsa si chiude si salta alla routine ON per accendere i motori.



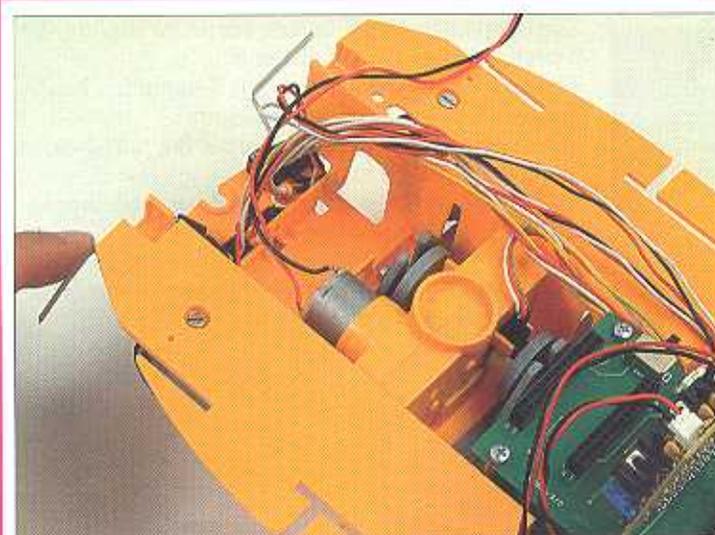
Dobbiamo scrivere il codice sorgente utilizzando il programma MPLAB. Dopo averlo scritto lo memorizzeremo con il nome pot1.asm, dobbiamo compilare il programma senza errori, al fine di ottenere il file pot1.hex; che sarà il file da scrivere sulla Smartcard utilizzando per questo il programma ICPROG. Il procedimento per la programmazione è lo stesso che abbiamo utilizzato per tutti gli esercizi di apprendimento con la scheda di ingressi e uscite.



Per verificare l'esercizio, la scheda di ingressi e uscite, deve essere inserita nel connettore JP13 della scheda di interfaccia. Collegheremo la scheda di potenza sul connettore JP14 della scheda di interfaccia con lo stesso orientamento mostrato dall'immagine. Collegheremo anche il motore destro del robot al connettore JP2 della scheda di potenza e il motore sinistro al connettore JP4 della stessa scheda. Per capire qual è il motore destro e qual è quello sinistro il riferimento è guardare il robot dalla sua parte posteriore, lato in cui si inserisce la Smartcard.



Il finecorsa sinistro deve essere inserito sul connettore JP7 della scheda di interfaccia, la posizione del commutatore SW2 della scheda di interfaccia deve essere la stessa di quella che appare nell'immagine, in modo che i segnali dei finecorsa siano attivi verso il microcontroller della scheda di controllo.



Dopo aver scritto il programma pot1.hex sulla Smartcard la inseriremo nella scheda di alimentazione del robot. Dobbiamo alimentare il robot mediante cinque pile collocate nel porta batterie o con un alimentatore esterno avente un'uscita di circa 6 V. Quando il microcontroller legge il contenuto della Smartcard inizierà l'esecuzione del programma, attivando il finecorsa sinistro i motori cominceranno a girare e rilasciando il finecorsa si fermeranno.