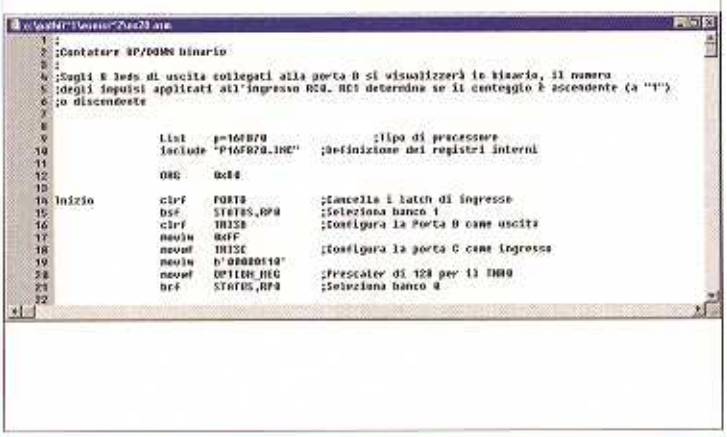
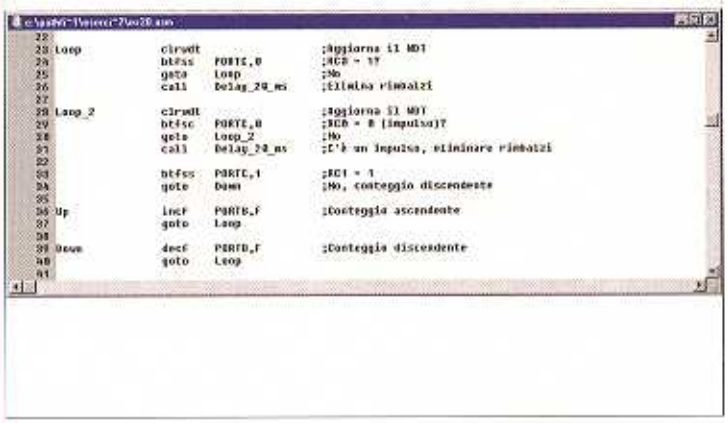


```
;Contatore UP/DOWN binario
;
;Sugli 8 leds di uscita collegati alla porta B si visualizzerà in binario, il numero
;degli impulsi applicati all'ingresso RC0. RC1 determina se il conteggio è ascendente (a "1")
;o discendente

            List    p=16F870            ;tipo di processore
            Include "P16F870.INC"       ;definizione dei registri interni

            ORG     0x00

Inizio      clrf    PORTB               ;Cancella il latch di ingresso
            bsf     STATUS,RP0          ;Seleziona banco 1
            clrf    TRISB               ;Configura la Porta B come uscita
            movlw   0xFF
            movwf   TRISC               ;Configura la porta C come ingresso
            movlw   b'00000010'
            movwf   OPTION_REG          ;Prescaler di 128 per il TMR0
            bcf     STATUS,RP0          ;Seleziona banco 0
```

Proponiamo un altro esercizio di apprendimento da realizzare con la scheda di ingressi e uscite. Si tratta di implementare un contatore ascendente o discendente. Quando il robot realizza traiettorie predefinite è necessario contare in avanti o indietro il numero di rotazioni compiute da ogni ruota, informazione che conosciamo tramite i sensori ottici. Mediante RC0 inseriremo gli impulsi che saranno contati e tramite RC1 determineremo se il conteggio è ascendente o discendente. Il valore del conteggio è riportato sui diodi LED della scheda.

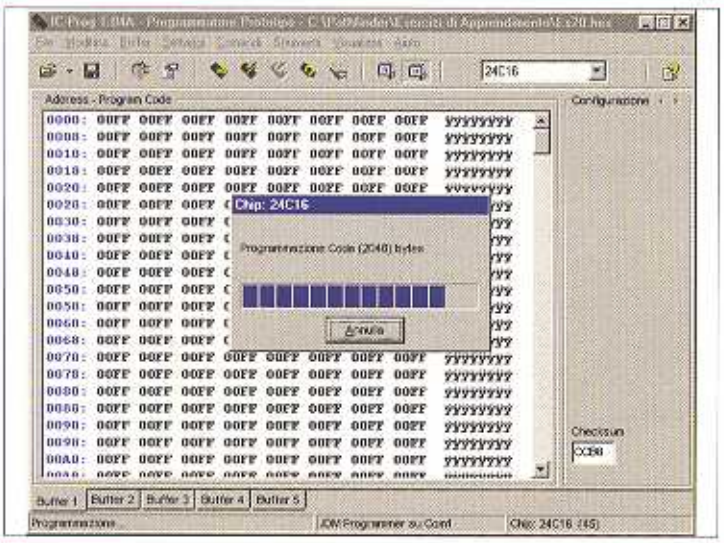


```
Loop        clrwdt                  ;Aggiorna il WDT
            btfss   PORTC,0         ;RC0 = 1?
            goto    Loop            ;No
            call    Delay_20_ms     ;Elimina rimbalzi

Loop_2      clrwdt                  ;Aggiorna il WDT
            btfsc   PORTC,0         ;RC0 = 0 (impulso)?
            goto    Loop_2          ;No
            call    Delay_20_ms     ;C'è un impulso, eliminare rimbalzi

            btfss   PORTC,1         ;RC1 = 1
            goto    Down            ;No, conteggio discendente

Up          incf    PORTB,F         ;Conteggia ascendente
            goto    Loop

Down        decf    PORTB,F         ;Conteggia discendente
            goto    Loop
```

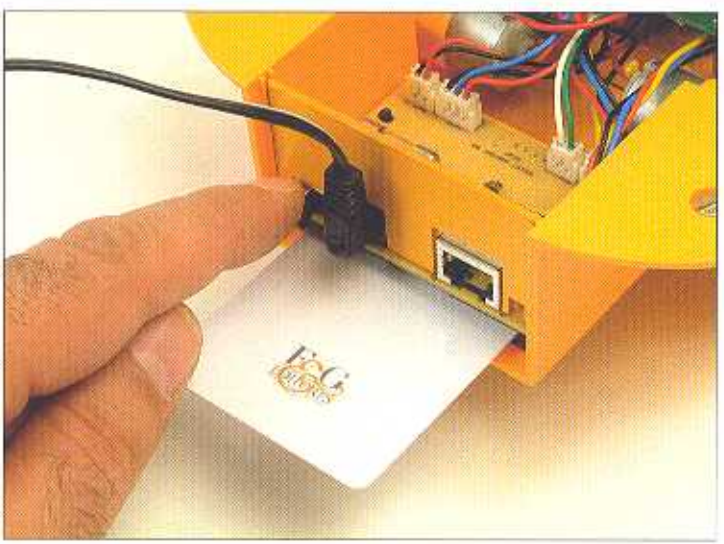
L'esercizio configura la porta B come uscita per i diodi, e la porta C come ingresso per gli interruttori. Inoltre viene assegnato il prescaler 128 al Timer 0, dato che utilizzeremo questo temporizzatore per la routine antirimbalzo. Il ciclo del programma inizia testando lo stato dell'interruttore RC0. Prima si attende che l'interruttore passi a valore 1, dopo di che si rimane in un altro ciclo sino a che ritorna a valore 0. Quando questo succede si dovrà generare un impulso; in quel momento dovremo solo leggere lo stato di RC1 per aumentare o diminuire il valore dei diodi LED della porta B.

```
;*******************************************************************************
;Delay_20_ms: Questa routine di ritardo ha come obiettivo eliminare "l'effetto rimbalzo",
;caratteristica dei componenti elettromeccanici.
;Realizza un ritardo di 20 ms. Se il PIC lavora ad una frequenza di 4 MHz,
;il TMR0 si aggiorna ogni µs. Se si desidera temporizzare 20000 µs (20 ms) con
;un prescaler di 128, il TMR0 dovrà contare 156 eventi,(156 * 128).
;Il valore 156 equivale a 9C hex e, date che  il TMR0 è ascendente, la dovremo cari-
;care con il suo complemento 1 (63 hex.).

Delay_20_ms:    bcf     INTCON,T0IF     ;Resetta il flag di overflow del TMR0
                movlw   0x63            ;Complemento hex. di 156
                movwf   TMR0            ;Carica il TMR0
Delay_20_ms_1   clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Delay_20_ms_1   ;Non ancora
                bcf     INTCON,T0IF     ;Ora sì, resettare il flag
                return

                end                     ;Fine del programma sorgente
```

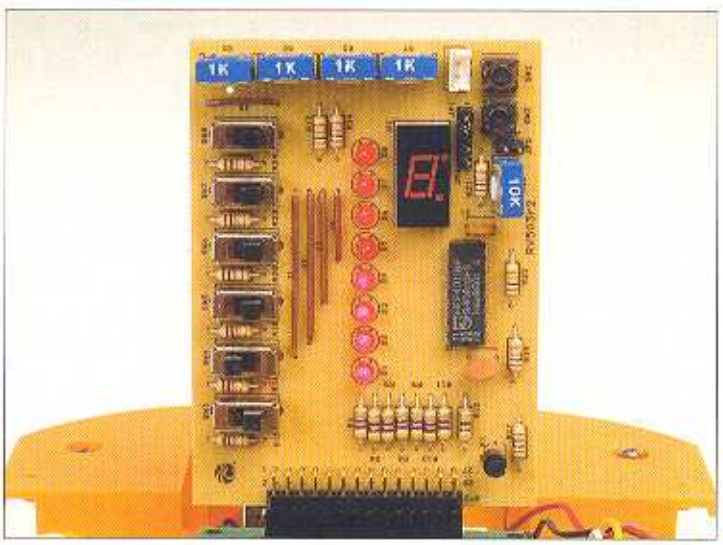
Gli interruttori sono sensori meccanici e quindi generano l'effetto rimbalzo. Per eliminare questo problema, ogni volta che l'interruttore generatore di impulsi (RC0) cambia stato, viene chiamata la subroutine di temporizzazione da 20 ms. Dopo questo tempo il segnale all'uscita dell'interruttore si sarà stabilizzato e si potrà leggere in modo affidabile. Anche i sensori meccanici tipo finecorsa del robot patiscono l'effetto del rimbalzo e il loro trattamento dovrà essere simile a quello degli interruttori.



Questo esercizio si trova sul CDROM sotto il nome es20.asm. Dopo aver compilato il file procederemo alla sua programmazione mediante il software ICPROG sulla scheda di scrittura. Nel software ICPROG sceglieremo la memoria modello 24C16 e verrà aperto il file es20.hex. Dopodiché inizieremo la sua scrittura sulla Smartcard che dovrà essere inserita sulla scheda di scrittura con l'orientamento corretto.



Quando la scheda sarà stata programmata, la inseriremo sulla scheda di alimentazione del robot. Per realizzare questi esercizi di apprendimento vi consigliamo di alimentare il robot con un alimentatore a corrente continua tramite connettore J1 della scheda di alimentazione. La scheda di ingressi e uscite deve essere inserita sulla scheda di interfaccia, e il microcontroller dovrà essere presente sulla scheda di controllo con il programma uploader caricato.



Per verificare l'esercizio, modificheremo lo stato degli interruttori SW3 (RC0) e SW4 (RC1). Ogni volta che realizzeremo un ciclo 0-1-0, mediante SW3, inseriremo un impulso, e il contatore aumenterà o diminuirà il suo valore binario in funzione dello stato dell'interruttore SW4, che invia segnali al pin RC1 del microcontroller.