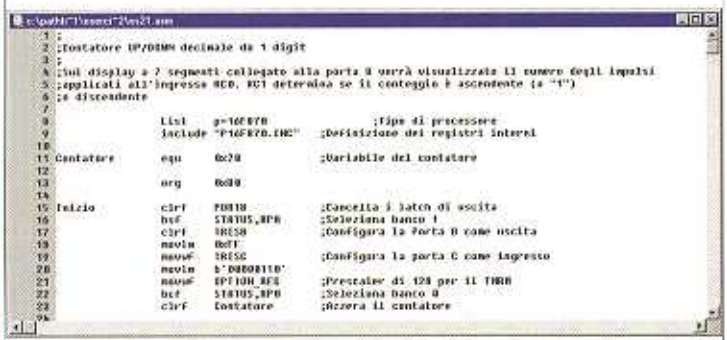
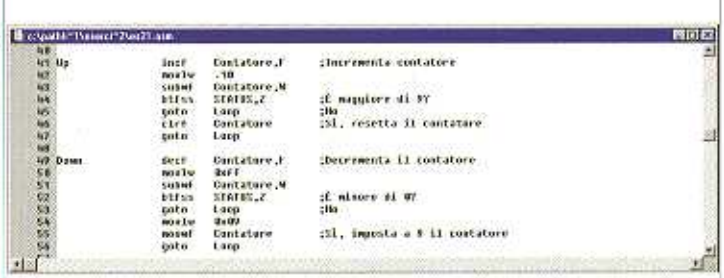


```
;
;Contatore UP/DOWN decimale da 1 digit
;
;sul display a 7 segmenti collegato alla porta B verrà visualizzato il numero degli impulsi
;applicati all'ingresso RC0. RC1 determina se il conteggio è ascendente (a "1")
;o discendente

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

Contatore       equ     0x20            ;Variabile del contatore

                org     0x00

Inizio          clrf    PORTB           ;Cancella il 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
                movlw   b'00000110'
                movwf   OPTION_REG      ;Prescaler di 128 per il TMR0
                bcf     STATUS,RP0      ;Seleziona banco 0
                clrf    Contatore       ;Azzera il contatore
```

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.

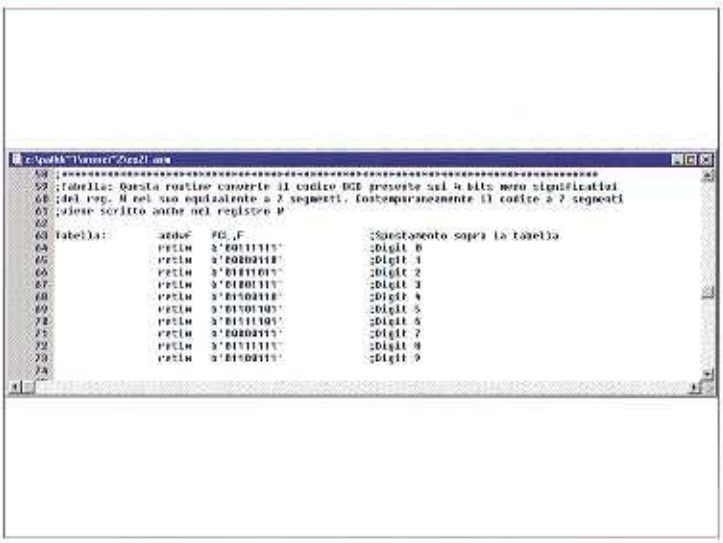
```
Loop            movf    Contatore,W
                call    Tabella         ;Converte BCD a 7 segmenti
                movwf   PORTB           ;Visualizza il valore del contatore
Wait_0          clrwdt                  ;Aggiorna il WDT
                btfsc   PORTC,0         ;RC0 = 1?
                goto    Wait_0          ;No
                call    Delay_20_ms     ;Elimina rimbalzi

Wait_1          clrwdt                  ;Aggiorna il WDT
                btfss   PORTC,0         ;RC0 = 0 (impulso)?
                goto    Wait_1          ;No
                call    Delay_20_ms     ;C'è un impulso, eliminare i rimbalzi

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

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.



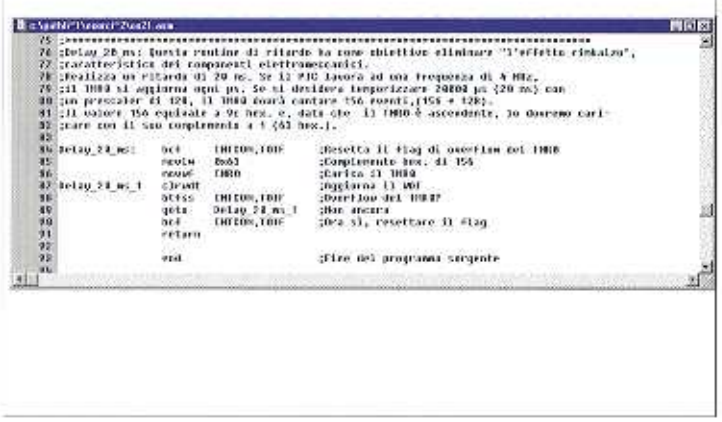
```
Up              incf    Contatore,F     ;Incrementa contatore
                movlw   .10
                subwf   Contatore,W
                btfss   STATUS,Z        ;È maggiore di 9?
                goto    Loop            ;No
                clrf    Contatore       ;Sì, resetta il contatore
                goto    Loop

Down            decf    Contatore,F     ;Decrementa il contatore
                movlw   0xFF
                subwf   Contatore,W
                btfss   STATUS,Z        ;È minore di 0?
                goto    Loop            ;No
                movlw   0x09
                movwf   Contatore       ;Sì, imposta a 9 il contatore
                goto    Loop
```

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.



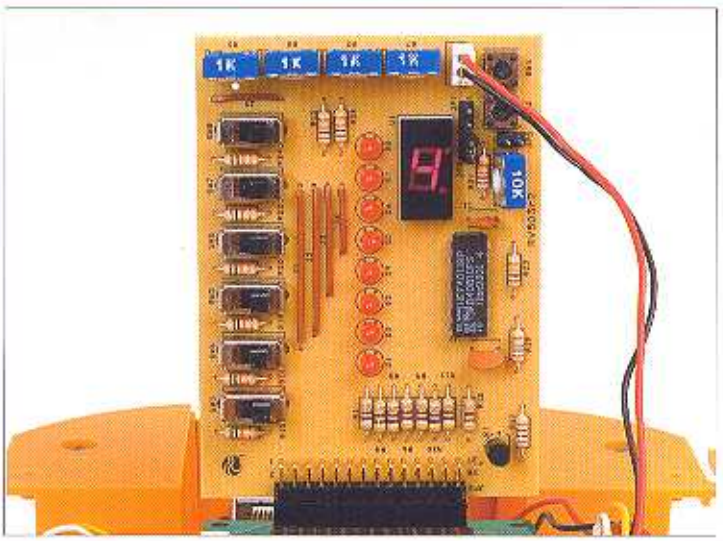
```
;*******************************************************************************
;Tabella: Questa routine converte il codice BCD presente sui 4 bits meno significativi
;del reg. W nel suo equivalente a 7 segmenti. Contemporaneamente il codice a 7 segmenti
;viene scritto anche nel registro W
;
Tabella:        andwf   PCL,F                   ;Spostamento sopra la tabella
                retlw   b'00111111'             ;Digit 0
                retlw   b'00000110'             ;Digit 1
                retlw   b'01011011'             ;Digit 2
                retlw   b'01001111'             ;Digit 3
                retlw   b'01100110'             ;Digit 4
                retlw   b'01101101'             ;Digit 5
                retlw   b'01111101'             ;Digit 6
                retlw   b'00000111'             ;Digit 7
                retlw   b'01111111'             ;Digit 8
                retlw   b'01100111'             ;Digit 9
```

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.



```
;*******************************************************************************
;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,(155 + 128).
;Il valore 156 espresso a 9c hex. e, dato che  il TMR0 è ascendente, lo dovremo cari-
;care con il suo complemento a 1 (63 hex.).

Delay_20_ms:    movlw   B'11010111'             ;Resetta il flag di overflow del TMR0
                movwf   TMR0                    ;Carica il TMR0
Delay_20_ms_1   clrwdt                          ;Aggiorna il WDT
                btfss   INTCON,T0IF             ;Overflow del TMR0?
                goto    Delay_20_ms_1           ;No: ancora
                bcf     INTCON,T0IF             ;Sì: sì, resettare il flag
                return

                end                             ;Fine del programma sorgente
```

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 RC0 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.