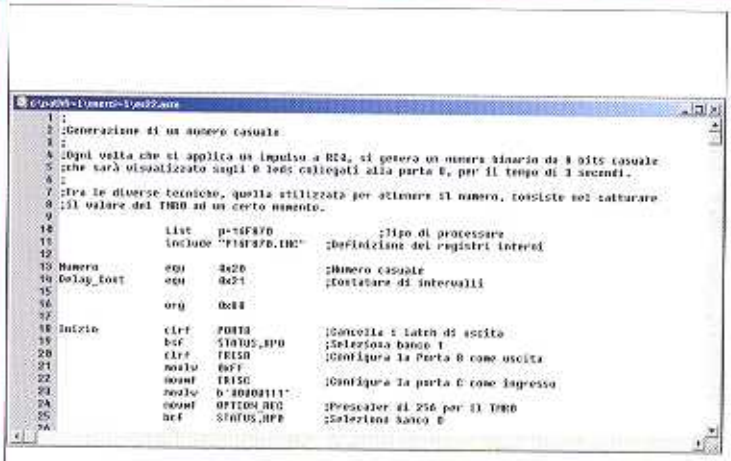
Esercizi di apprendimento

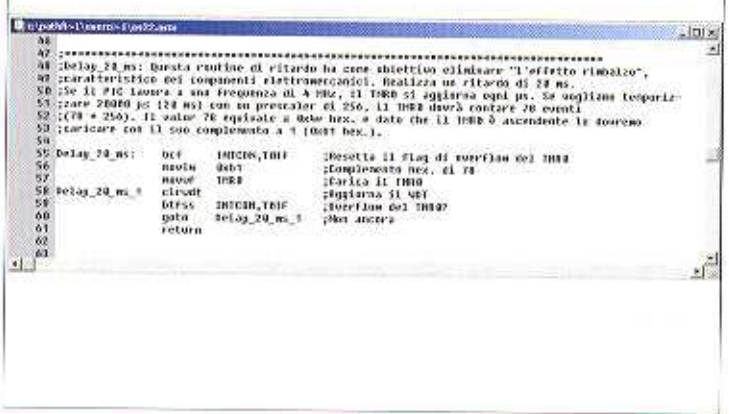


```
;
;Generazione di un numero casuale
;
;Ogni volta che si applica un impulso a RC3, si genera un numero binario da 8 bits casuale
;che sarà visualizzato sugli 8 leds collegati alla porta B, per il tempo di 3 secondi.
;
;Tra le diverse tecniche, quella utilizzata per ottenere il numero, consiste nel catturare
;il valore del TMR0 ad un certo momento.

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

Numero          equ     0x20            ;Numero casuale
Delay_Cost      equ     0x21            ;Costatore di intervalli

                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
                movlw   b'00000111'
                movwf   OPTION_REG      ;Prescaler di 256 per il TMR0
                bcf     STATUS,RP0      ;Selezione 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.

```
Loop            clrwdt                  ;Aggiorna il WDT
                btfsc   PORTC,0         ;E' attivo RC0?
                goto    Loop            ;Non ancora
                movf    TMR0,W          ;Sì.
                movwf   Numero          ;Catturo il valore del TMR0 (NP casuale)
                call    Delay_20_MS     ;Elimina rimbalzi

RC0_1           clrwdt                  ;Aggiorna il WDT
                btfsc   PORTC,0         ;RC0 è stato disattivato?
                goto    RC0_1           ;Non ancora
                movf    Numero,W        ;E' stato generato un impulso su RC0, si legge
                movwf   PORTB           ;il valore catturato dal TMR0 (NP casuale)
                                        ;e lo si visualizza sui leds della PORTB 0

                movlw   d'66'
                movwf   Delay_Cost      ;Inizializza la variabile di temporizzazione
                call    Delay_sec       ;Temporizza 3 secondi
                clrf    PORTB           ;Resetta le uscite
                goto    Loop
```

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.

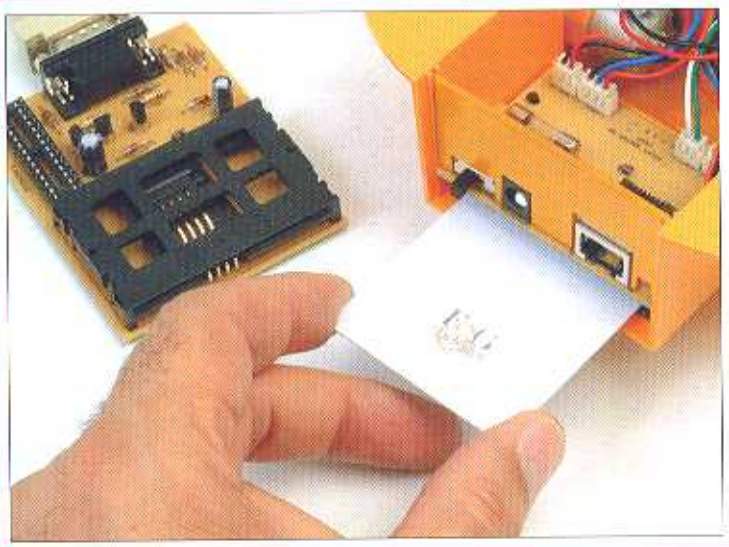


```
;*****************************************************************************
;Delay_20_ms: Questa routine di ritardo ha come obiettivo eliminare "l'effetto rimbalzo",
;caratteristico dei componenti elettromeccanici. Realizza un ritardo di 20 ms.
;Se il PIC lavora a una frequenza di 4 MHz, il TMR0 si aggiorna ogni µs. Se vogliamo temporiz-
;zare 20000 µs (20 ms) con un prescaler di 256, il TMR0 dovrà contare 78 eventi
;(78 * 256). Il valore 78 equivale a 0x4e hex. e dato che il TMR0 è ascendente lo dovremo
;caricare con il suo complemento a 1 (0x31 hex.).

Delay_20_ms:    bcf     INTCON,T0IF     ;Resetta il Flag di overflow del TMR0
                movlw   0x31            ;Complemento hex. di 78
                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
                return
```

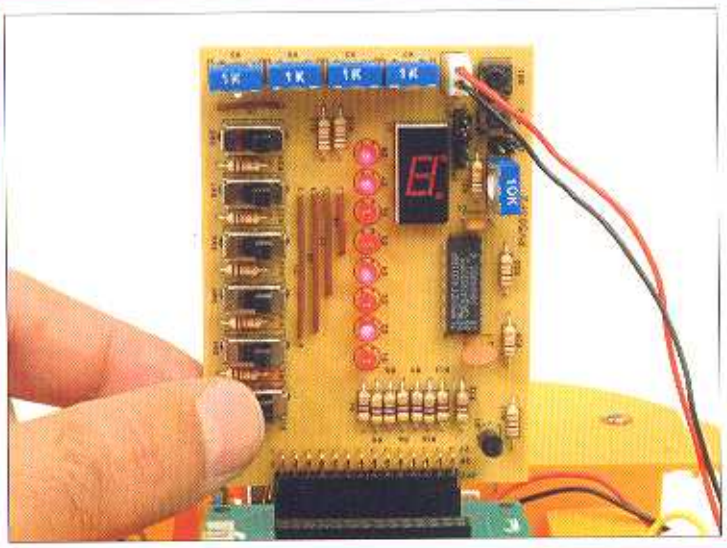
Il numero casuale da mostrare sulla barra dei diodi led si produce quando si introduce un impulso completo mediante l'interruttore SW3 della scheda, che invia segnali al pin RC0 del microcontroller. Bisogna eliminare l'effetto rimbalzo dei sensori meccanici, e per questo motivo quando RC0 cambia il suo 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.

```
;Delay_var: Questa routine di utilizzo generale realizza una temporizzazione variabile
;fra 50 ms e 12.8 sec. Si utilizza un prescaler di 256 e sul TMR0 si carica 195.
;La velocità di lavoro è di 4 Mhz e quindi il TMR0 si incrementa ogni µs. In
;questo modo, il TMR0 deve contare 195 eventi che, con un prescaler di 128 danno un
;intervallo totale di 50000 µs = 50 ms (195 * 256). Il valore 195 lo dobbiamo caricare
;in hex. (C3) e, dato che il TMR0 è ascendente dovremo caricare il suo complemento a 1 (3C hex.)
;Questo intervallo di 50 ms si ripete tante volte quante indicate dalla variabile "Delay_cont",
;per questo il ritardo minimo è di 50 ms ("Delay_cont=1) e quello massimo di 12.8 sec.
;(Delay_cont=255).

Delay_var:      bcf     INTCON,T0IF     ;Resetta il flag di overflow
                movlw   0x3C            ;Complemento hex. di 195
                movwf   TMR0            ;Carica il TMR0
Intervallo      clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Intervallo      ;Non ancora
                decfsz  Delay_cont,f    ;Decrementa il contatore degli intervalli
                goto    Delay_var       ;Ripete l'intervallo di 50 ms
                return

                end                     ;Fine del programma sorgente
```

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 eseguiremo 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 eseguiremo 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.