Danilo B (ildani)

LAMPEGGIO LED TRAMITE TIMER INTERRUPT - PARTE 3

12 January 2013

Introduzione

Questa è la terza parte dell'articolo Lampeggio led tramite Timer Interrupt.

Sviluppo del Codice

Decido di collegare il led da far lampeggiare al pin RA0.

Anzichè ripetere noiose pagine di codice, cerco di riassumere i principali passaggi evolutivi che ha avuto il mio codice:

1. **Versione 1**
    1. **imposto le porte A** con questo codice: TRISA=0b00000001; che è **sbagliato**, perchè così il pin RA0 viene configurato come input;
    2. **copio e incollo il codice ottenuto con il PicTimer**; lo rendo compatibile col compilatore XC8 cambiando OPTION=0b00000001; in OPTION_REG=0b00000001;
    3. **precarico il Timer0** usando il valore con correzione suggerito da PicTimer;
    4. nella routine di interrupt scrivo il codice per **controllare la sorgente dell'interrupt** e **azzerare il timer overflow** (o meglio il Timer0 Interrupt Flag T0IF).
1. **Versione 2**
    1. **sposto la routine di interrupt fuori dal programma principale**, cioè dove deve stare correttamente;
    2. nella routine di interrupt, dentro al codice che controlla se si è accesa la "spia" T0IF, **incremento la variabile Tempo di una unità**. La variabile chiamata Tempo sarà quella che userò per conteggiare il numero di T0IF, ossia il tempo che passa. Il codice è semplicemente **Tempo++;**
    3. sopra al programma principale devo ovviamente **dichiarare ed inizializzare questa varibile**, il codice è **unsigned int Tempo;** (provate a capire come mai il Tipo della variabile è un int);

4. sopra al programma principale definisco il TempoLampeggio per poterlo richiamare facilmente nel programma (utilissimo anche per modificare il tempo di lampeggio senza modificarne tutte le occorrenze nel codice) e definisco anche il pin utilizzato per collegare il led. I codici sono **#define TempoLampeggio 500** e **#define Led RA0**, quest'ultima istruzione è errata.
5. sotto l'impostazione errata della Porta A, **imposto come uscite la Porta B e C** col codice **TRISB = 0; TRISC = 0;** per assicurarmi di non lasciarle come ingressi che portano disturbi.

2. **Versione 3**
    1. nel programma principale, sotto l'impostazione delle porte, accendo il led col codice **Led = 1;**
    2. nella routine di interrupt, dopo l'incremento della variabile Tempo, **controllo se ha superato il TempoLampeggio** con **if (Tempo >= TempoLampeggio) {**; se questo accade inverto lo stato del pin della porta dove ho collegato il led tramite **Led = !Led;** ed azzero la variabile Tempo con **Tempo = 0;**.
3. **Versione 4**
    1. mi rendo conto che nella ISR, dopo aver verificato la sorgente T0IF, è necessario **reimpostare il precaricamento del Timer0** con **TMR0 = 8;** perchè se si è accesa la spia T0IF significa che è avvenuto il Timer0 overflow e quindi il Timer0 è tornato a 0 ed è pronto ad effettuare un nuovo conteggio, ma per farlo nei tempi corretti deve essere di nuovo precaricato dello stesso valore presente nel programma principale (potrebbe essere utile definire un TempoPreload);
    2. alla fine del programma principale devo inserire un "loop infinito" che impegna il PIC sino al Timer0 Interrupt, il codice è **while(1) { }**;
    3. **sistemo l'impostazione di tutti i pin della Porta A** come uscite cambiando il codice in **TRISA = 0;**
    4. **sistemo la definizione del pin utilizzato per collegare il led** cambiando il codice in **#define Led PORTAbits.RA0**.

Il codice che ottengo alla fine è questo:

```
#include <xc.h>

//DEFINE
#define TempoLampeggio 500
#define Led            PORTAbits.RA0


// CONFIG
```

```
#pragma config FOSC = XT        // Oscillator Selection bits (XT oscillator)
#pragma config WDTE = OFF       // Watchdog Timer Enable bit (WDT disabled)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT disabled)
#pragma config CP = OFF         // FLASH Program Memory Code Protection bits (Code prot
#pragma config BOREN = OFF      // Brown-out Reset Enable bit (BOR disabled)
#pragma config LVP = OFF        // Low Voltage In-Circuit Serial Programming Enable bit
#pragma config CPD = OFF        // Data EE Memory Code Protection (Code Protection off)
#pragma config WRT = ON         // FLASH Program Memory Write Enable (Unprotected progr

//VARIABILI
unsigned int Tempo = 0;

void main(void) {
    //IMPOSTAZIONE PORTE INPUT/OUTPUT
    //PORTA is a 6-bit wide, bi-directional port.
    TRISA = 0;
    TRISB = 0;
    TRISC = 0;

    //ACCENSIONE Led
    Led = 1;

    //IMPOSTARE PRESCALER
    // Generato da PicTimer 1.0.1.0
    // © Bernardo Giovanni - www.settorezero.com
    // Codice valido per Hitec-C

    // Fosc           :4MHz
    // Timer0 preload :6
    // Prescaler      :4
    // Interrupt time :1,0000mS

    // OPTION
    // bit 0 -> PS0    Prescaler Rate Select bit 0
    // bit 1 -> PS1    Prescaler Rate Select bit 1
    // bit 2 -> PS2    Prescaler Rate Select bit 2
    // bit 3 -> PSA    Prescaler assegnato a Timer0 (1=Watchdog Timer)
    // bit 4 -> T0SE   Timer0 Signal Edge: 0=low->high 1=high->low
    // bit 5 -> T0CS   Timer0 Clock Select: internal clock (1=T0CKI transition)
    // bit 6 -> INTEDG INTerrupt Edge (1=raise 0=fall)
    // bit 7 -> RBPU   PortB PullUp (0=off 1=on)

    OPTION_REG = 0b00000001;
```

```
    //IMPOSTARE INTERRUPT
    // INTCON
    // bit 0 -> RBIF   PortB Interrupt Flag
    // bit 1 -> INTF   RB0/INT Interrupt Flag
    // bit 2 -> T0IF   Timer0 Interrupt Flag
    // bit 3 -> RBIE   PortB Interrupt Enable (off)
    // bit 4 -> INTE   INT Interrupt Enable (off)
    // bit 5 -> TMR0IE Timer0 Interrupt Enable (on)
    // bit 6 -> PEIE   PEripheral Interrupt Enable (off)
    // bit 7 -> GIE    Global Interrupt Enable (on)

    INTCON = 0b10100000;

    //PRECARICARE IL TIMER
    // Preload Timer0
    // TMR0=8; // valore con correzione

    TMR0 = 8; // con correzione

    while(1) {

    }
}

//ROUTINE DI INTERRUPT

void interrupt isr(void) {
    //CONTROLLO SORGENTE INTERRUPT TIMER0 OVERFLOW
    if (T0IF) {
        //REIMPOSTAZIONE PRECARIMENTO DEL TIMER
        TMR0 = 8;
        //INCREMENTO VARIABILE E CONTROLLO VALORE
        Tempo++;
        if (Tempo >= TempoLampeggio) {
            Led = !Led;
            Tempo = 0;
        }
        //RESET TIMER OVERFLOW
        T0IF = 0;
    }
}
```

Sembra tutto corretto, lo carico sul PIC16F876 tramite MPLAB X IDE ed il PicKit3, ma ottengo solamente un led semiacceso che non lampeggia...