Laboratory Worksheet #05
Timer Overflow Interrupts Exercise

The following is an exercise on Timer Overflow interrupts and will serve as a good starting point for Lab 1-2. The hardware used for this activity should already be ready on your board from Lab 1-1.

Exercise 1:

1) On what page of the LITEC manual does the discussion of Timer Functions start? 34

Utilizing the timer on the microcontroller requires initializing and interacting with a number of SFRs. Based on the following descriptions, identify which SFR is indicated.

2) The high byte of Timer0 0x8C

3) The SFR which contains a bit to turn the Timer0 on/off 0x88 (Bit 4)

4) The SFR that sets Timer0 to 16 bit mode 0x89 (Bit 0, Bit 1) 0

5) What is the frequency of the system clock (to 6 decimal places)? 22.1184×10^6 Hz^1

Exercise 2:

The program configures Timer0 to use SYSCLK/12 as its source and in a 13-bit mode.

1) Complete the initialization code that follows. Note: this example code is only part of the full initialization routine. The indicated initial settings of the SFRs are here for example.

```
//current state of TMOD is xxxx 1111 =>   xxxx 0000
//current state of CKCON is xxxx xxxx
TMOD  &= F0
CKCON &= F7          xxxx 0xxx
```

2) How much time (in seconds) is required for a timer overflow interrupt (assume the Timer is initialized to 0)? 4.444×10^{-3} s

3) How many overflows will occur in 1 second? 225

4) Assume a variable counts keeps track of the number of overflows. After 2.5 seconds, what is the value of counts? (assume counts is zero a 0 seconds) 562

Exercise 3:

The sample code, *Worksheet_05.c*, is available on the LMS website under the "Laboratories" section, under Lab 1, part 2. You need to complete the initialization routines for the Port I/O, the Timer, and Interrupts. Once you have done that, compile, link, download and run this program. This program counts the number of timer overflows occurring while the slide switch is in the Off position.

Exercise 4:

In the *Worksheet_05.c* code, the variable *counts* keeps track of the number of timer overflows and this value is printed on the terminal.

1) Modify the printf statement to also print out the corresponding time period in seconds.
2) Using a handy watch or online clock, turn the switch to the 'count' position for 10 seconds and compare the accuracy of the counter to the time you measure.

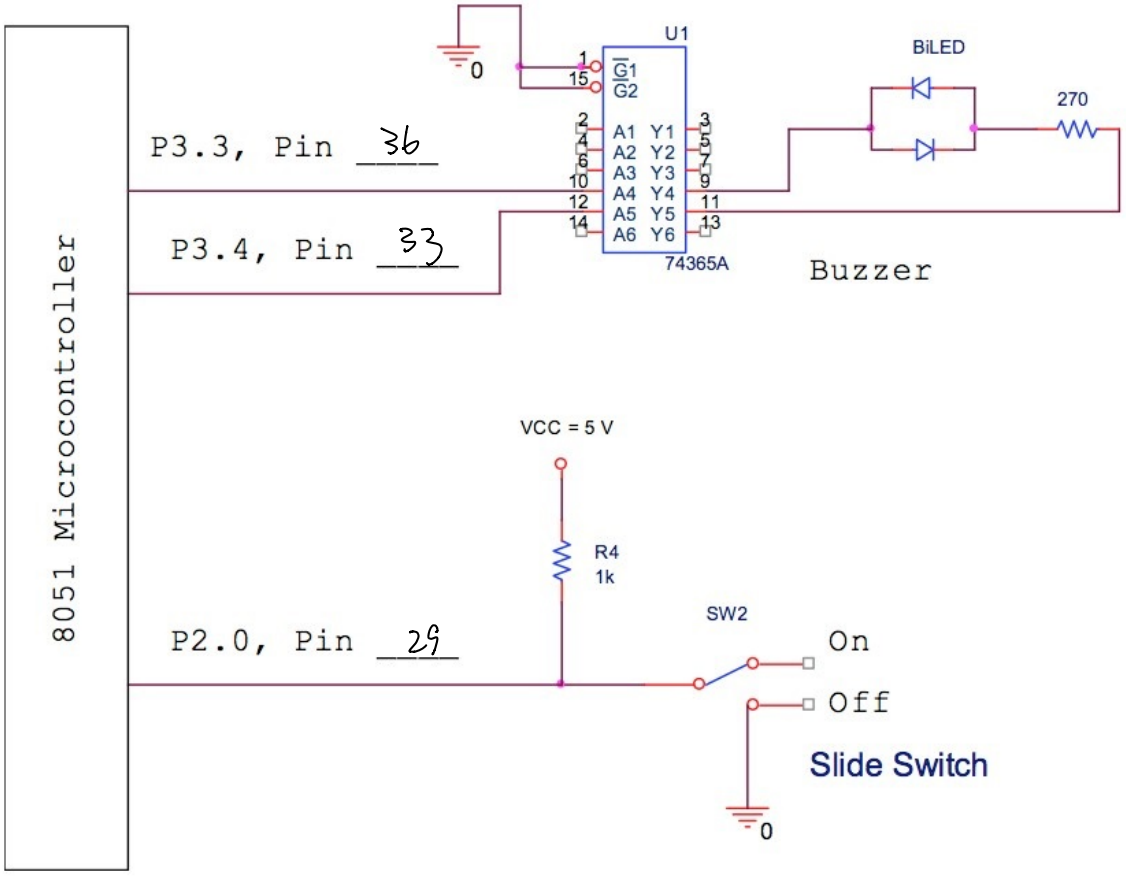

Figure 1: Worksheet 3 Schematic

When complete, include Worksheet 5 with your Laboratory 1-2 Pre-lab submission.

```
/* This program demonstrates the use of T0 interrupt. The code will count the
number of T0 timer overflows that occur while a slide switch is in Off position
Some editing is requited prior to running the code. Fill in the indicated blanks.
*/

#include <c8051_SDCC.h> // include files. You need to include stdio and c8051_SCDD.
#include <stdio.h>      // Add lines as needed

//-----------------------------------------------------------
// Function PROTOTYPES
//-----------------------------------------------------------
void T0_ISR(void) __interrupt 1; // Function Prototype for Interrupt Service Routine
void Port_Init(void);           // Initialize ports for input and output
void Timer_Init(void);           // Initialize Timer 0
void Interrupt_Init(void);
void Counter_off(void);
void Counter_on(void); //

//-----------------------------------------------------------
// Global variables
//-----------------------------------------------------------

// one end of bicolor LED0 is associated with Port 3 Pin 3
__sbit __at 0xB3 Biled1;
// other end of bicolor LED0 is associated with Port 3 Pin 4
__sbit __at 0xB4 Biled2;
__sbit __at 0xA0 SW; // Slide Switch associated with Port 2 Pin 0

int Counts = 0;

//***************
void main(void)
{
   Sys_Init();   // System Initialization Always do this first.
   putchar(' '); // line added to allow printf statements
   Port_Init();  // Initialize port 2 and 3
   Timer_Init(); // Initialize Timer 0
   Interrupt_Init();

   printf("Start\r\n");
   while (1)
   {
      Counter_off();
      Counter_on();
   }
}

void Port_Init(void)
{

   // Port 3
   P3MDOUT |= 0x18; // set output pins P3.3 and P3.4 in push-pull mode
```

```
    // Port 2
    P2MDOUT &= 0xFE; // set input pin P2.0 in open drain mode
    P2 |= ~0xFE;     // set input pin P2.0 to high impedance state
}

void Interrupt_Init(void)
{
    IE |= 0x02; //enable Timer0 interrupts by setting the appropriate bit in the SFR
    EA = 1;     //enable all interrupts using an existing sbit label
}

void Timer_Init(void)
{
    CKCON &= F7; // Make T1 intact and T0 use SYSCLK/12
    TMOD &= F0;  // Clear the 4 least significant bits
    TMOD &= F0;  // Leave T1 intact and set T0 mode as specified in Exercise 2
    TR0 = 0;     // Stop Timer0
    // 2 ways to clear 16-bit T0 counter: use a single command for all 16 bits
    TMR0 = 0; // Clear both bytes of T0
    // or use 2 commands for low and high bytes separately
    TL0 = 0; // Clear low byte of register T0
    TH0 = 0; // Clear high byte of register T0
}

void T0_ISR(void) __interrupt 1 //Interrupt service routine
{
    TF0 = 0;  // clear interrupt request (not required - cleared automatically by hardware)
    Counts++; // increment overflow counter
}

void Counter_off(void) // turn the BILED off and stop the counter
{
    TR0 = 0;    // turn off the counter
    Counts = 0; // reset counts to 0
    Biled1 = 0;
    Biled2 = 0;
    TL0 = 0x00;
    TH0 = 0x00; // initialize the Timer to a 0 start value
    while (SW)
        ; // while the switch is off, wait
}

void Counter_on(void) // turn the BILED on and count how long it the switch is on
{
    Biled1 = 1;
    Biled2 = 0;
    TR0 = 1; // start the counter
    while (!SW)
        ; // while the switch is on, wait
    printf("Number of Overflows = %d\n", Counts);
    float num_of_sec = 0;
    num_of_sec = Counts / 225; // 225 overflows per second.
    printf("Total: %f seconds", num_of_sec);
}
```

EVB Pin		Port Bit	Bit Addresses & Labels
1	2	1.	
		2.	
3	4	3.	
		4.	
5	6	5.	
		6.	
7	8	7.	
		8.	
9	10	9.	
		10.	
11	12	11.	
		12.	
13	14	13.	
		14.	
15	16	15.	
		16.	
17	18	17.	
		18.	
19	20	19.	
		20.	
21	22	21.	
		22.	
23	24	23.	
		24.	
25	26	25.	
		26.	
27	28	27.	
		28.	
29	30	29. 0XA0	SS
		30.	
31	32	31. 0XB6	LED0
		32. 0XB7	BUZZER
33	34	33. 0XB4	B1LED2
		34. 0XB5	LED1
35	36	35.	
		36. 0XB3	B1LED1
37	38	37. 0XB1	PB2
		38. 0XB0	PB1
39	40	39.	
		40.	

41 ⟷ 60

Software Initializations

A) Port I/0

```
P3MDOUT |= 0xF8;
P3MDOUT &= 0xFC;
P3 |= ~0xFC;

P2MDOUT &= 0xFE;
P2 |= ~0xFE;
```

B) Timers

```
CKCON |= 0x08;
TMOD &= 0xF0;
TMOD |= 0x02;
TR0 = 0;
TMR0 = 0;
```

C) Interrupts

```
IE |= 0x02;
EA = 1;
```

D) A/D

E) PCA

F) XBAR

G) I2C

Compile directives
```
#include <c8051_SDCC.h>
#include <stdio.h>
#include <stdlib.h>
```

Function prototypes
```
void Port_Init(void);
void Timer_Init(void);
void Interrupt_Init(void);
void Timer0_ISR(void) __interrupt 1;
void random(void);
unsigned char IsPB1Pushed(void);
unsigned char IsPB2Pushed(void);
unsigned char IsSSOn(void);
void LEDController(void);
void ResultController(void);
void TurnBILEDGreen(void);
void TurnBILEDRed(void);
void TurnBILEDOff(void);
void TurnLED0On(void);
void TurnLED0Off(void);
void TurnLED1On(void);
void TurnLED1Off(void);
```

Declare global variables
```
sbit LED0, LED1, BILED1, BILED2, BUZZER, SS, PB1, PB2, TR0.
unsigned int Counts.
unsigned char Wins, CurrentRandom, PreviousRandom, Tries.
```

Main function
```
declare local variables
    (NONE)
Initialization functions
    Sys_Init();
    Port_Init();
    Interrupt_Init();
    Timer_Init();
    putchar(' ');
    Turn off all the outputs
    Turn off Timer0
    Begin infinite loop
        Wait until slide switch is on
        loop for 10 times
            Execute LEDController control the LED based on the CurrentRandom
            Turn Timer0 on
            Clear the counts on Timer0
            Wait for 1 second
            Execute ResultController to determine the user inputs
            Reset the Counts on Timer0.
        End the loop
        print the result
        Reset the number of wins to 0
        Turn off all the outputs
```

    End infinite loop
End main function

Functions
  void Port_Init(void)
    Set Port 3 input pins to open drain mode.
    Set Port 3 output pins to push-pull mode.
    Set Port 3 input pins to high impedance state.
    Set Port 2 input pins to open drain mode.
    Set Port 2 input pins to high impedance state.
  void Interrupt_Init(void)
    Enable Timer0 Interrupt request (by masking).
    Enable global interrupts (by sbit).
  void Timer_Init(void)
    Set Timer0 as stated in the manual, Use 16bit and SYSCLK.
    Stop the Timer0 for now.
    Clear high and low byte of T0
  void Timer0_ISR(void) __interrupt 1
    Increment the Counts
  void random(void)
    Begin infinite loop
      Generate a random number.
      Exit while the new random number is not the same as the old one.
      Set the PreviousRandom the same as the CurrentRandom.
    End infinite loop
  void LEDController(void)
    Use the random function to generate a random number.
    Turn on LED(s) based on the result.
  void ResultController(void)
    Determine the inputs from the user and light up the BILED accordingly.
  unsigned char IsPB1Pushed(void)
    Return 1 if PB1 is pushed.
    Return 0 if PB1 is not pushed.
  unsigned char IsPB2Pushed(void)
    Return 1 if PB2 is pushed.
    Return 0 if PB2 is not pushed.
  unsigned char IsSSOn(void)
    Return 1 if slide switch is on.
    Return 0 if slide switch is off.
  void TurnBILEDGreen(void): Turn the BILED to green.
  void TurnBILEDRed(void): Turn the BILED to red.
  void TurnBILEDOff(void): Turn off the BILED.
  void TurnLED0On(void): Turn on LED0.
  void TurnLED0Off(void): Turn off LED0.
  void TurnLED1On(void): Turn on LED1.
  void TurnLED1Off(void): Turn off LED1.