ECE 376 - Homework #9

INT, Timer 0/1/2/3 Interrupts - Due Wednesday, November 13th

Timer0 Interrupts

1) Write a C routine using Timer0 interrupts to measure time to 100ns. Using this routine, determine how long a the following operations in C take:

a) Press RB0 ten times:

```
TRISB = 0xFF;

for(i=0; i<10; i++) {              // start
   while(!RB0);
   while(RB0);
   }                               // end
```

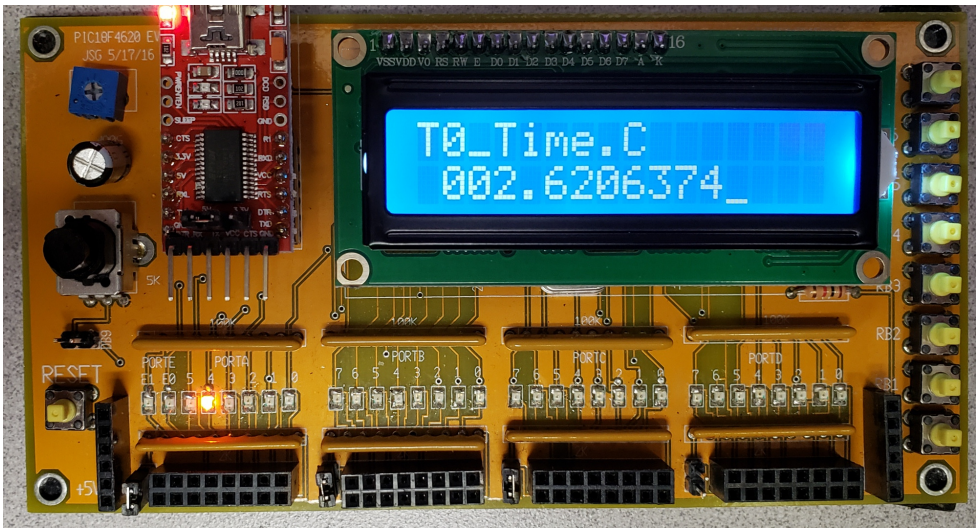
ans: 1.4642419 seconds

b) Input code 314157

```
TRISB = 0xFF

// start
while(!RB3);  while(!RB3);
while(!RB1);  while(!RB1);
while(!RB4);  while(!RB4);
while(!RB1);  while(!RB1);
while(!RB5);  while(!RB5);
while(!RB7);  while(!RB7);
// end
```

ans: 2.6206374 seconds

comment: With Timer0, you can measure time with absurd precision.



INT & Timer0 Interrupts

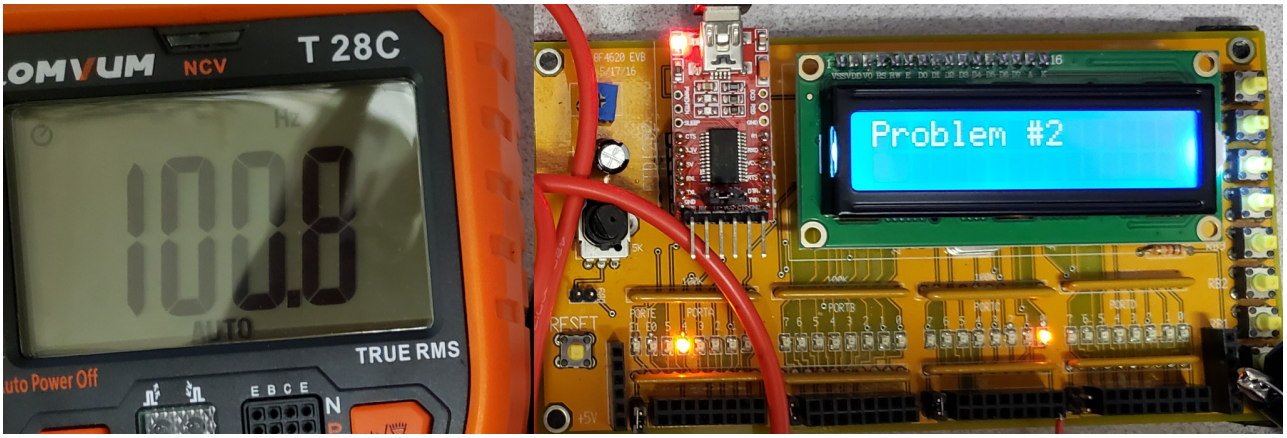
Write a routine which measures the frequency of a square wave using INT and Timer0 interrups

2) Generate a 100Hz (ish) square wave on RC0

```
while(1) {
   RC0 = !RC0;
   Wait_ms(5);
   }
```

Check the frequency using a speaker and Pano Tuner app (or similar way to measure frequenc).

f = 100.8Hz

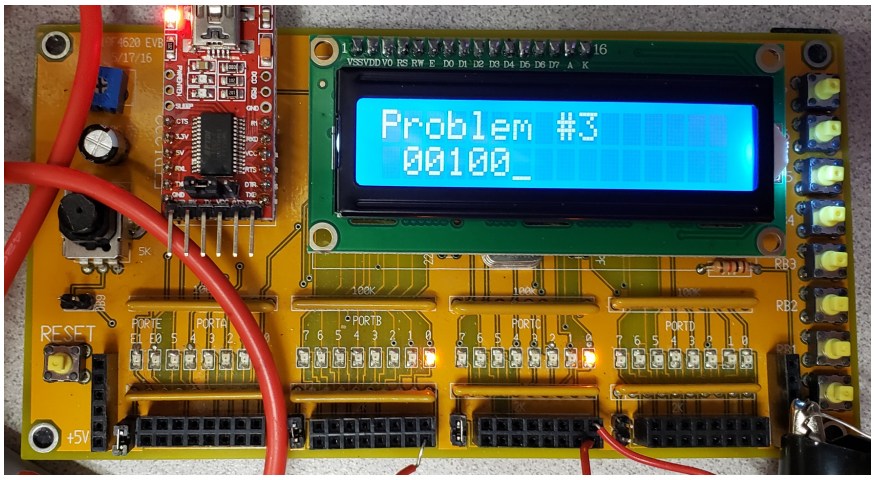


3) Use an INT0 interrupt and Timer0 to measure the frequency of this square wave (take 1)

- Measure each rising edge using an INT0 interrupt
- Measure time using a Timer0 interrupt

Count and display on the LCD how many edges you detect in 1.000 second

Result: Display shows 100 or 101 (varies)



note: This is a slight problem since the main routine is trying to do two things at once:

- Generate a square wave, and
- Display the frequency of the square wave

To get around this, the main routine:

- Generates a 100Hz square wave for two seconds (400 cycles)
- Once done, it then displayes the measured frequency in cycles per second

Interrupt:

```
void interrupt IntServe(void)
{
   if(INT0IF){
      N += 1;
      INT0IF = 0;
       }
   if (TMR0IF) {
      TMR0 = -39062;
      Hz = N;
      N = 0;
      TMR0IF = 0;
      }
   }
```

Main Routine

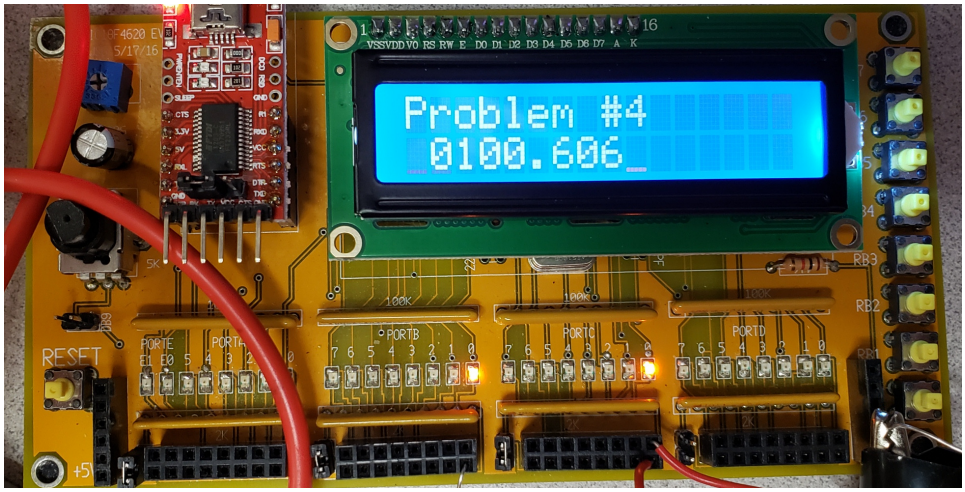
```
// set up Timer0 for PS = 256
   T0CS = 0;
   T0CON = 0x87;
   TMR0ON = 1;
   TMR0IE = 1;
   TMR0IP = 1;
   PEIE = 1;
// Turn on INT0 interrupt
   INT0IE = 1;
   TRISB0 = 1;
   INTEDG0 = 1;
// turn on all interrupts
   GIE = 1;

   while(1) {
       for(i=0; i<400; i++){
           RC0 = !RC0;
           Wait_ms(5);
           }
       LCD_Move(1,0);  LCD_Out(Hz, 5, 0);
   }
}
```

4) Use an INT0 interrupt and Timer0 to measure the period of this square wave

- Measure time to 100ns using Timer0 interrupts
- Record the time of the rising edge using INT0 interrupts

From these, display on the LCD the period and the frequency of the square wave

- Frequency = 100.606Hz
- Period = 99,397 clocks



Interrupt

```
void interrupt IntServe(void)
{
   if(INT0IF){
      T2 = T1;
      T1 = TIME + TMR0;
      dT = T1 - T2;
      INT0IF = 0;
       }
   if (TMR0IF) {
      TIME += 0x10000;
      TMR0IF = 0;
      }
   }
```

Code

```
// set up Timer0 for PS = 1
   T0CS = 0;
   T0CON = 0x88;
   TMR0ON = 1;
   TMR0IE = 1;
   TMR0IP = 1;
   PEIE = 1;
// Turn on INT0 interrupt
   INT0IE = 1;
   TRISB0 = 1;
   INTEDG0 = 1;
// turn on all interrupts
   GIE = 1;

   while(1) {
       for(i=0; i<6; i++){
           RC0 = !RC0;
           Wait_ms(5);
           }
       Hz = 10000000.0 / dT;
       LCD_Move(1,0);
LCD_Out(dTHz*1000, 7, 3);
   }
```

Timer 0/1/2/3 Interrupts

5) Write a C routine using Timer0 / Timer1 / Tirme2 / Timer3 interrupts to play 4 notes at the same time when you press button RB0.. RB3 at the same time (each note plays if its input button is pressed)

Input Pin	RB0	RB1	RB2	RB3
Output Pin	RC0	RC1	RC2	RC3
Note	A2	B2	C3	D3
Frequency (Hz)	110 Hz	123.471 Hz	130.813 Hz	146.832 Hz
Interrupt	Timer0	Timer1	Timer2	Timer3
N	45,454.54	40,495.33	38,222.5	34,052.52
Actual Hz	110.1	123.6	131.0	146.9

Code:

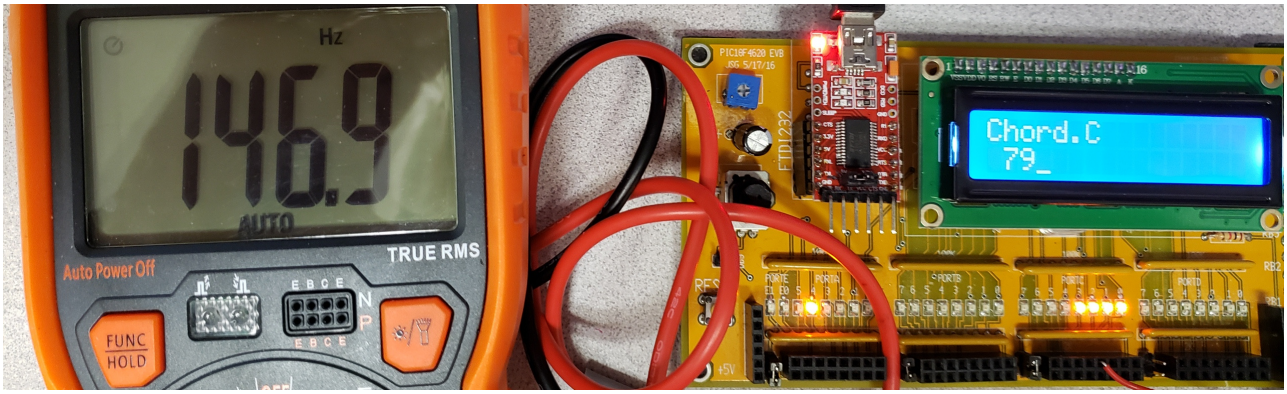
```
void interrupt IntServe(void)
{
   if (TMR0IF) {
      TMR0 = -45454 + 25;
      RC0 = !RC0;
      TMR0IF = 0;
      }
   if (TMR1IF) {
      TMR1 = -40495 + 32;
      RC1 = !RC1;
      TMR1IF = 0;
      }
   if (TMR2IF) {
      RC2 = !RC2;
      TMR2IF = 0;
      }
   if (TMR3IF) {
      TMR3 = -34052 + 41;
      RC3 = !RC3;
      TMR3IF = 0;
      }
   }
```

Initialization

```
// set up Timer0 for PS = 1
   T0CS = 0;
   T0CON = 0x88;
   TMR0ON = 1;
   TMR0IE = 1;
   TMR0IP = 1;
   PEIE = 1;
// set up Timer1 for PS = 1
   TMR1CS = 0;
   T1CON = 0x81;
   TMR1ON = 1;
   TMR1IE = 1;
   TMR1IP = 1;
   PEIE = 1;
// set up Timer2 for A = 12, B = 199, C = 16
   T2CON = 0x5F;
   PR2   = 198;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;
// set up Timer3 for PS = 1
```

```
    TMR3CS = 0;
    T3CON = 0x81;
    TMR3ON = 1;
    TMR3IE = 1;
    TMR3IP = 1;
    PEIE = 1;
// turn on all interrupts
    GIE = 1;
```

The main routine doesn't do anything



Three-channel PWM output for an RGB LED:

Write a C routine using Timer interrupts to drive an RGB LED with PWM outputs:

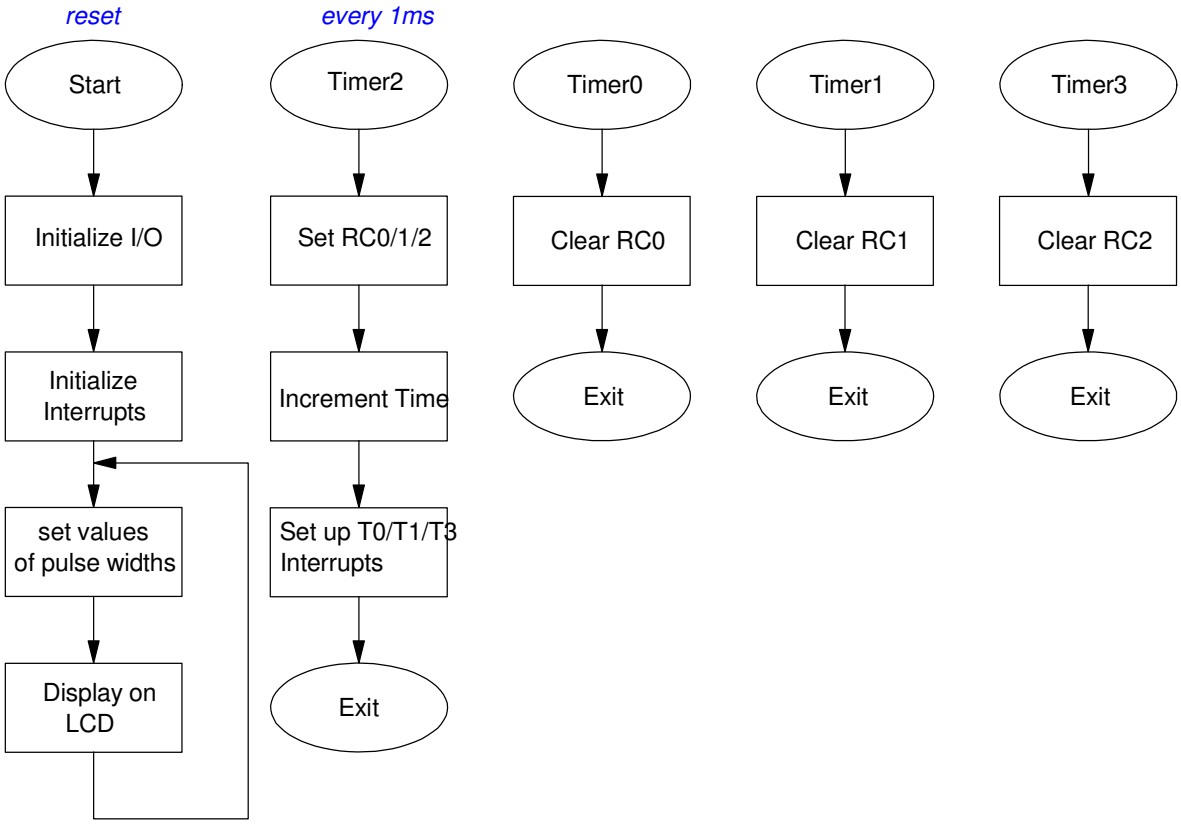
- Each LED is driven at 200Hz (5ms period)
- The brightness of each LED can be varied from 0% to 100% on

Set up the interrupts as follows:

- Timer2 sets each LED's output pin every 5ms.
  - Timer2 also sets up the following Timer0/1/3 interrupts
- Timer0 clears the red LED's pin
- Timer1 clears the green LED's pin, and
- Timer3 clears the blue LED's pin

6) Give the flow chart for this program.

- It will need at least four flow charts: one for the main routine and one for each interrupt.



7) C-code. Write the corresponding C code

Interrupts

```
void interrupt IntServe(void)
{
   if (TMR0IF) {
      RC0 = 0;
      TMR0IF = 0;
      }
   if (TMR1IF) {
      RC1 = 0;
      TMR1IF = 0;
      }
   if (TMR2IF) {
      PORTC = 7;
      TMR0 = -N0;
      TMR1 = -N1;
      TMR3 = -N2;
      TMR2IF = 0;
      }
   if (TMR3IF) {
      RC2 = 0;
      TMR3IF = 0;
      }
   }
```

Initialization

```
// set up Timer0 for PS = 1
   T0CS = 0;
   T0CON = 0x88;
   TMR0ON = 1;
   TMR0IE = 1;
   TMR0IP = 1;
   PEIE = 1;
// set up Timer1 for PS = 1
   TMR1CS = 0;
   T1CON = 0x81;
   TMR1ON = 1;
   TMR1IE = 1;
   TMR1IP = 1;
   PEIE = 1;
// set up Timer2 2.5ms for A = 7, B = 223, C = 16
   T2CON = 0x37;
   PR2   = 222;
   TMR2ON = 1;
   TMR2IE = 1;
   TMR2IP = 1;
   PEIE = 1;
// set up Timer3 for PS = 1
   TMR3CS = 0;
   T3CON = 0x81;
   TMR3ON = 1;
   TMR3IE = 1;
   TMR3IP = 1;
   PEIE = 1;
// turn on all interrupts
   GIE = 1;
```

Main Loop

```
N0 = 0.9*25000;
N1 = 0.5*25000;
N2 = 0.1*25000;

while(1) {
   LCD_Move(0,8);  LCD_Out(N0, 5, 0);
   LCD_Move(1,0);  LCD_Out(N1, 5, 0);
   LCD_Move(1,8);  LCD_Out(N2, 5, 0);
   Wait_ms(1000);
   }
}
```

8) Validation and Testing. Verify you can vary the duty cycle of each LED independently

Red = 90%, Green = 50%, Blue = 10%

- Power = 5.101V
- RC0 = 4.530V 88.81%
- RC1 = 2.525V 49.50%
- RC2 = 0.523V 10.25%

Red = 30%, Green = 90%, Blue = 40%

- Power = 5.101V
- RC0 = 1.517V 29.74%
- RC1 = 4.532V 88.85%
- RC2 = 2.026V 39.72%