ECE 376 - Homework #5

C Programming. Due Monday, October 2nd

1) Write a C subroutine, Wait(X), which waits X milliseconds then returns. For example

2) Check the accuracy of your wait routine.

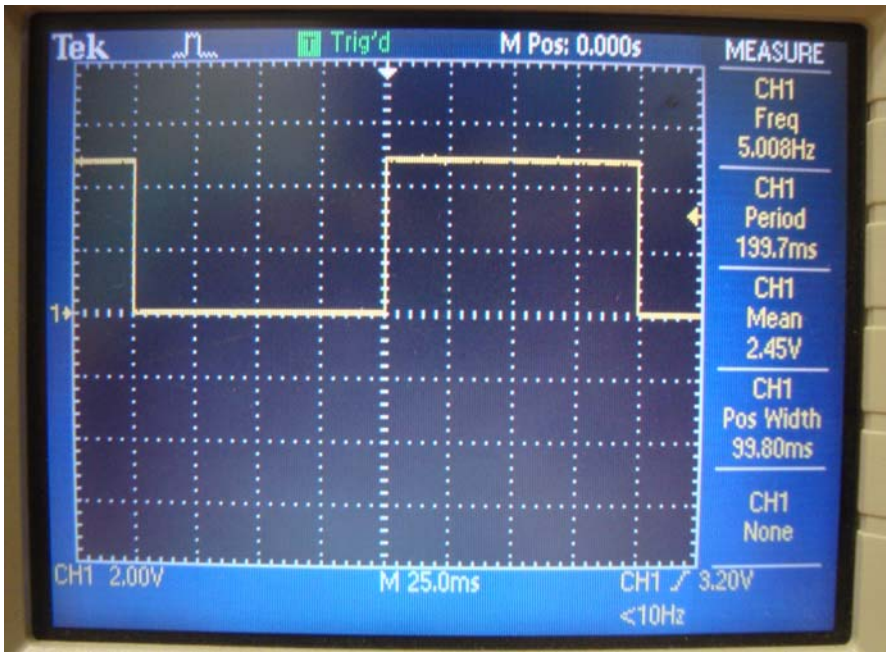
```
#include <pic18.h>

void Wait(unsigned int X)
{
   unsigned int i, j;
   for (i=0; i<X; i++)
      for (j=0; j<622; j++);
}

void main(void)
{

   TRISA = 0;
   TRISB = 0;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   while(1) {
      PORTB = PORTB + 1;
      Wait(100);
      }
   }
```



Verification of the Wait() routine. The positive width should be 100ms (actual = 99.8ms)

Problem 3-6: Write a program in C which includes some timing and outputs. Some suggestions are

- 8-key piano in C. Play notes C2 to C3 when you press buttons RB0 to RB7. Each note should be accurate to 1%

3) Requirements: Specify

Inputs:

- Buttons RB0 . . RB7

Outputs:

- 8 Ohm speaker connected to pin RC0 with a 200 Ohm resistor

Relationship: Play a note based upon the button pressed:

Button	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
Note	C3	B2	A2	G2	F2	E2	D2	C2
Hz	130.81	123.47	110	98	87.31	82.41	73.42	65.41

Tolerance: +/- 1%

4) Analysis: The following code was used to test the frequency played on pin RC0

```
void main(void)
{
   unsigned int i;

   TRISA = 0;
   TRISB = 0xFF;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   while(1) {
      if (PORTB) RC0 = !RC0;
      if (RB0)  for(i=0; i<1000; i++);
      }
```

This produces a frequency of 312.077Hz on RC0 when button RB0 was pressed.

To generate the other frequencies, the number 1000 is scaled according to the frequency

$$N = \left(\frac{312.088Hz}{\text{desired Hz}}\right)1000$$

Button	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
Note	C3	B2	A2	G2	F2	E2	D2	C2
Hz	130.81	123.47	110	98	87.31	82.41	73.42	65.41
N	2,385.73	2,527.55	2,837.06	3,184.46	3,574.36	3,786.88	4,250.57	4,771.09

5) C Code and number of lines of assembler the resulting code took up.

```
// ---  Piano.C -------------------

// Subroutine Declarations
#include <pic18.h>

// Main Routine

void main(void)
{
   unsigned int i;

   TRISA = 0;
   TRISB = 0xFF;
   TRISC = 0;
   TRISD = 0;
   TRISE = 0;
   ADCON1 = 0x0F;

   while(1) {
      if (PORTB) RC0 = !RC0;
      if (RB0)  for(i=0; i<4771; i++);
      if (RB1)  for(i=0; i<4250; i++);
      if (RB2)  for(i=0; i<3786; i++);
      if (RB3)  for(i=0; i<3574; i++);
      if (RB4)  for(i=0; i<3184; i++);
      if (RB5)  for(i=0; i<2837; i++);
      if (RB6)  for(i=0; i<2527; i++);
      if (RB7)  for(i=0; i<2385; i++);
      }
   }
```

note: Copy the message when you compile your code - it should look like the following. The number of lines of assembler are half of the program space (each instruction takes 2 bytes)

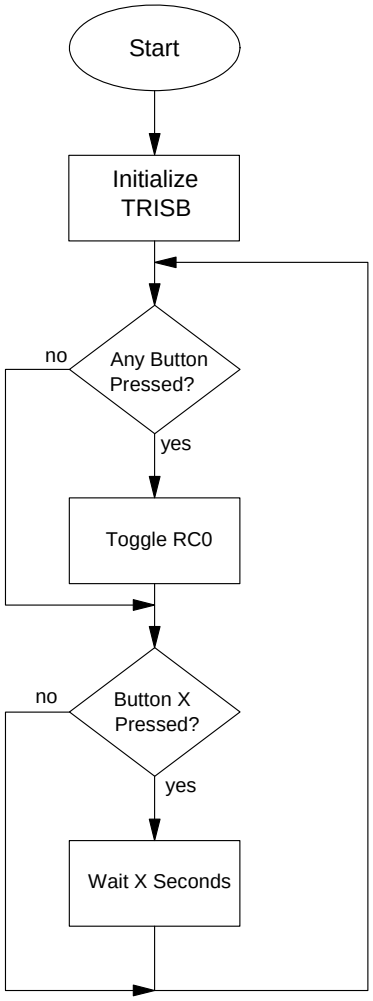
```
Memory Summary:
    Program space        used   294h (   660) of 10000h bytes   (  1.0%)
    Data space           used     3h (     3) of   F80h bytes   (  0.1%)
    EEPROM space         used     0h (     0) of   400h bytes   (  0.0%)
    ID Location space    used     0h (     0) of     8h nibbles (  0.0%)
    Configuration bits   used     0h (     0) of     7h words   (  0.0%)
```

The resulting C Code compiled into 330 lines of assembler.

In contrast, the assembler code took 208 lines of code

The C code is 58% larger than the corresponding code written in assembler

6) Flow chart



7) Validation: Include data from the lab to show you did / did not meet the requirements

Button	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
Note	C3	B2	A2	G2	F2	E2	D2	C2
Hz (ideal)	130.81	123.47	110	98	87.31	82.41	73.42	65.41
Hz (actual)	130.87	123.53	110.05	98.07	87.38	82.49	73.49	65.48
Error (%)	0.05	0.05	0.05	0.07	0.08	0.1	0.1	0.11

All frequencies are within 1% of the desired frequency

8) Demo. Either with a video or in person.