ECE 376 - Homework #4

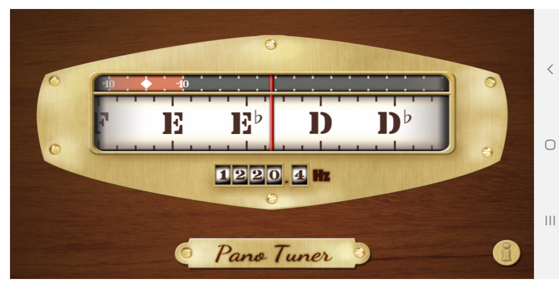
C Programming and LCD Displays
Please submit as a hard copy or submit on BlackBoard

1) Determine how many clocks the following C code takes to execute
   - Compile and download the code (modify working code and replace the main loop)
   - Measure the frequency you see on RC0 (toggles every loop).
     - Use an osiclloscope - or -
     - Connect a speaker to RC0 with a 200 Ohm resistor and measure the frequency with a cell phone app like Piano Tuner
     - RC1 is 1/2 the frequency of RC0, RC2 is 1/4th, RC3 = 1/8th, etc
   - The number of clocks it takes to execute each loop is

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right)$$

1a) Counting mod 8

```
unsigned char i
while(1) {
      i = (i + 1) % 8;
      if(i == 0) PORTC += 1;
      }
```



f(RC5) = 1220.4Hz

f(RC0) = 32 x f(RC5) = 39,052.8Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 128.032 \text{ clocks / toggle}$$

$$N/8 = 16.004 \qquad \text{clocks / loop}$$

It takes about 16 clocks to count mod 8

1b) Counting mod 7

```
unsigned char i
while(1) {
      i = (i + 1)% 7;
      if(i == 0) PORTC += 1;
      }
```

f(RC0) = 1023.2Hz

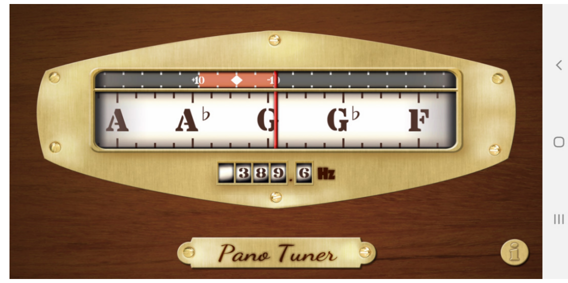
$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 4866.63 \text{ clocks / toggle}$$

N / 7 = 698.09 clocks / loop

It takes about 698 clocks to count mod 7

1c) Long Integer Division

```
unsigned long int A, B, C;
unsigned char i;
A = 0x12345678;
B = 0x1234;
while(1) {
    i = (i + 1)% 8;
    if (i == 0) PORTC += 1;
    C = A / B;
    }
```



f(RC0) = 389.6Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 12,833$$

$$N/8 = 1604$$

$$N/8 - 16 = 1588$$

It takes about 1588 clocks to do a long integer division

1d) Floating Point Cosine (need to add #include <math.h>)

```
float A, B, C;
A = 3.14159265379;
while(1) {
    i = (i + 1)% 8;
    if(i == 0) PORTC += 1;
    C = cos(A);
    }
```

f(RC0) = 120.6Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 41,459.37$$

$$\frac{N}{8} - 16 = 5166.42$$

It takes about 5166 clocks to do a floating point cosine function

Beep

2) Write a C program which plays 200Hz for 100ms on a speaker

Test Code:

```
void Beep(void) {
   unsigned int i, j;
   for(i=0; i<800; i++) {
      RC0 = !RC0;
      for(j=0; j<1000; j++);
   }
}
```

The measured frequency was 311.6Hz. To make the frequency 200Hz, change the counter for j:

$$N = \left(\frac{311.6Hz}{200Hz}\right)1000 = 1558$$

Now the frequency is 200.2Hz. For 100ms, count to 40

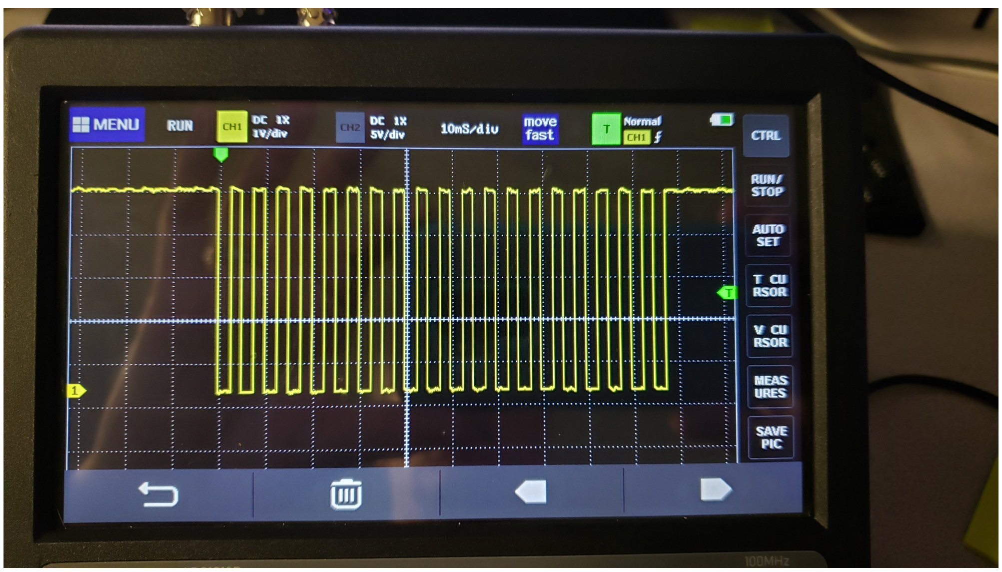
Final Code

```
void Beep(void) {
   unsigned int i, j;
   for(i=0; i<40; i++) {
      RC0 = !RC0;
      for(j=0; j<1588; j++);
   }
}
```

3) Verify the frequency and duration of your note

Frequency = 200.2Hz (from PanoTuner)

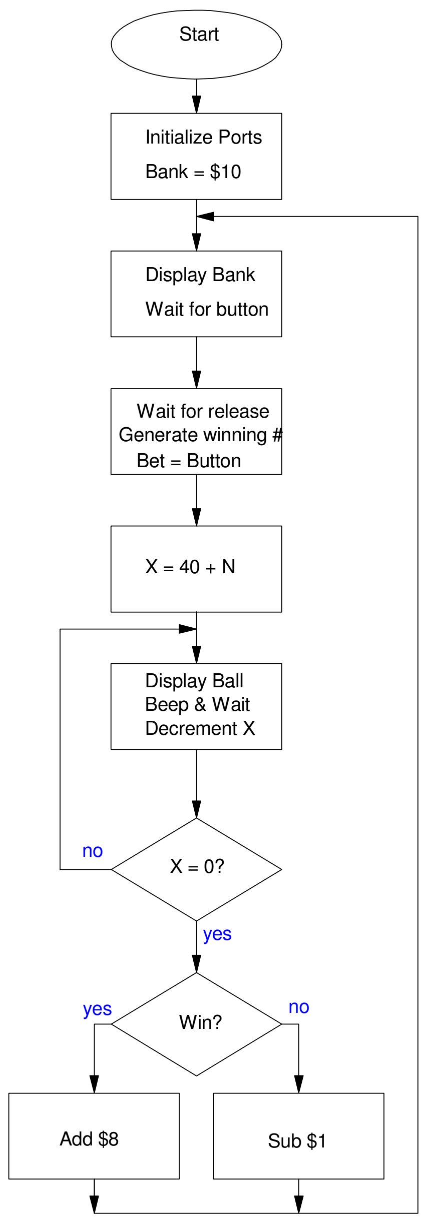
Period = 98ms (from an oscilloscope)



$65 Roulette Wheel

4) Give a flow chart for a program which turns your PIC into a Roulette wheel:

- On reset, you start with $10 in your bank (which is displayed on the LCD).
- The game starts by pressing a button (RB0 .. RB7). The number you're betting on is the button you press (0..7).
- When you press and release a button, a random number, N, is generated in the range of 0..7.
- The PIC will then count (mod 8) on the LCD display 40+N times, with one count every 200ms
- Each time you count, a speaker should beep for 100ms at 200Hz (problem #2)
- If the final count matches your bet, you win $8. If not, you lose $1.
- The game then repeats.
- The LCD displays your bank, the number you're betting on, and the current number on the roulette wheel



5) Write the C code for a roulette wheel

Code:

```
// Global Variables

const unsigned char MSG0[20] = "Bank:            ";
const unsigned char MSG1[20] = "Bet:             ";

// Subroutine Declarations
#include <pic18.h>

// Subroutines
#include        "lcd_portd.c"

void Beep(void) {
   unsigned int i, j;
   for(i=0; i<20; i++) {
      RC0 = !RC0;
      for(j=0; j<1558; j++);
      }
   }

// Main Routine

void main(void)
{
   unsigned int BANK;
   unsigned char N, X, BET;
   unsigned char i, j;

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

   LCD_Init();
   LCD_Move(0,0);  for(i=0; i<16; i++) LCD_Write(MSG0[i]);
   LCD_Move(1,0);  for(i=0; i<16; i++) LCD_Write(MSG1[i]);


   BANK = 10;
   while(1) {
      LCD_Move(0,8);  LCD_Out(BANK, 3, 0);

      while(PORTB == 0);
      while(PORTB) {
         if(RB0) BET = 0;
         if(RB1) BET = 1;
         if(RB2) BET = 2;
         if(RB3) BET = 3;
         if(RB4) BET = 4;
         if(RB5) BET = 5;
         if(RB6) BET = 6;
         if(RB7) BET = 7;
         N = (N + 1)%8;
         }
      LCD_Move(1,8);  LCD_Out(BET, 3, 0);


// Easter Egg:  4 always wins
      if(BET == 4) X = 4;
```

```
     for(X=0; X<40+N; X++) {
        LCD_Move(1,12); LCD_Out(X%8, 3, 0);
        Beep();
        Wait_ms(100);
        }

     X = X % 8;

     LCD_Move(1,12); LCD_Out(X, 3, 0);
     if(X == BET) BANK += 8;
     else BANK -= 1;

     }
  }
```

6) Verify your program

On reset, you start with $10 in your bank

- Check - bank starts at $10

Numbers generated are random, in the range of 0..7

- Check: winning numbers were always in the range of 0..7

The LCD displays information correctly

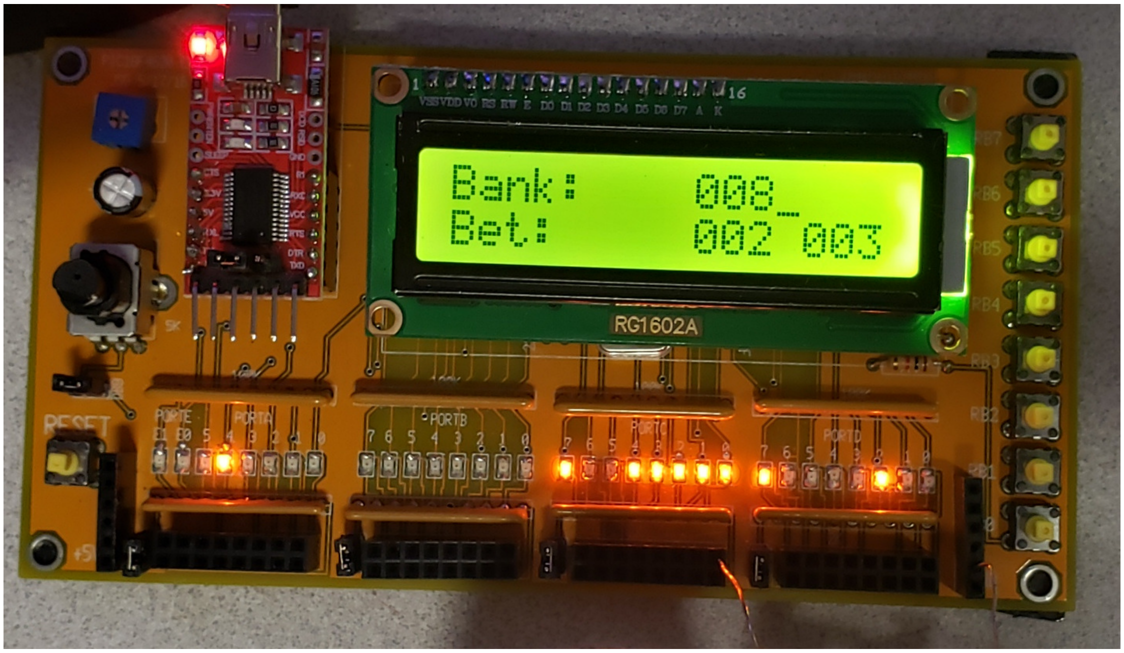
- Bank balance is correct
- Number betting on is correct
- Current number (X) is displayed

When you win, you gain $8. When you lose, you lose $1.

- Check: most of the time the two numbers don't match and I lose $1
- Once in a while, they do match and I win $8.

7) (20pt) Demonstration (in person or on a video) (1563 lines of assembly)

```
Memory Summary:
    Program space        used   C36h (  3126) of 10000h bytes   (  4.8%)
    Data space           used    2Ch (    44) of   F80h bytes   (  1.1%)
    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%)
```


Bank: 008_
Bet: 002 003
RG1602A
RESET
PORTE
PORTA
PORTB
PORTC
PORTD