ECE 376 - Homework #8

Timer 2 Interrupts. Due Monday, October 31st

Measuring Time to 1ms with Timer2 Interrupts

1) Write a routine for a count-down timer with a resolution of 1ms

- Time is measured to 1ms using Timer2 interrupts
- Each interrupt, pin RC0 is toggled (outputting a 500Hz square wave on RC0)
- Each interrupt (every 1ms), TIME is decremented to zero, stopping at zero
- TIME is displayed on the LCD display to 1ms: xx.xxx
- When you press RB0, the time is reset to 5.000 seconds
- When you press RB1, the time is reset to 10.000 seconds
- When you press RB2, the time is reset to 15.000 seconds
- When you press RB3, the time is reset to 20.000 seconds

Check the accuracy of your stopwatch

- Measure the frequency on RC0 when sent to a speaker using a cell phone app (Frequency Counter works)

Code:

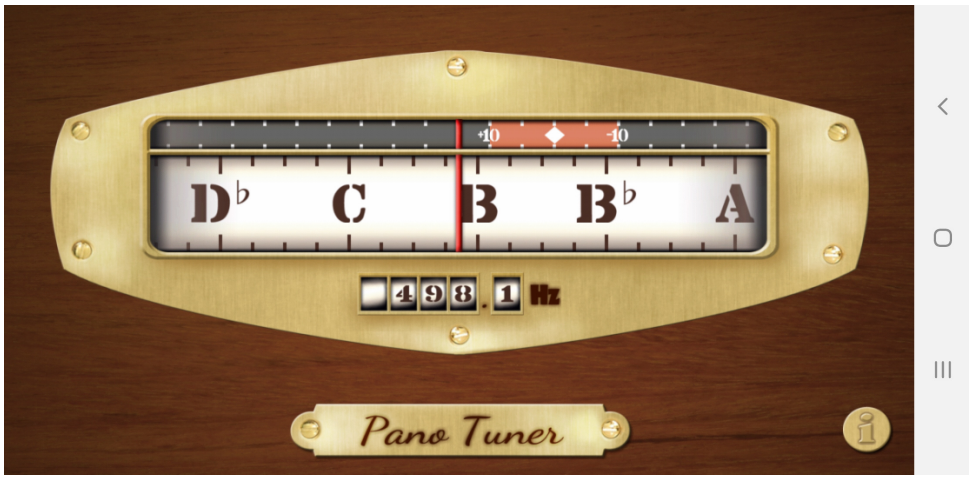
```
:
:
:
```

Compilation Results

```
Memory Summary:
    Program space        used   9FCh (  2556) of 10000h bytes   (  3.9%)
    Data space           used    33h (    51) of   F80h bytes   (  1.3%)
    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%)
```

Validation:

- Pressing RB0 initializes time to 5.000 seconds
- PRessing RB1 initialized time to 10.000 seconds
- (etc)
- Time decrements to 0.000 and stops
- Timer2 is called every 1.00ms (checked with Pano Tuner)



500Hz, which tells you that Timer2 is being called every 1.00ms

Generating Frequencies with Timer2 Interrupts

2) Write a routine which turns plays your PIC into a 1-string banjo using Timer2 interrupts

- Play note E4 (329.63Hz) on pin RC0 when button RB0 is pressed
- Check the accuracy of your music note using your cell phone (or whatever else you have on hand)

Calculations:

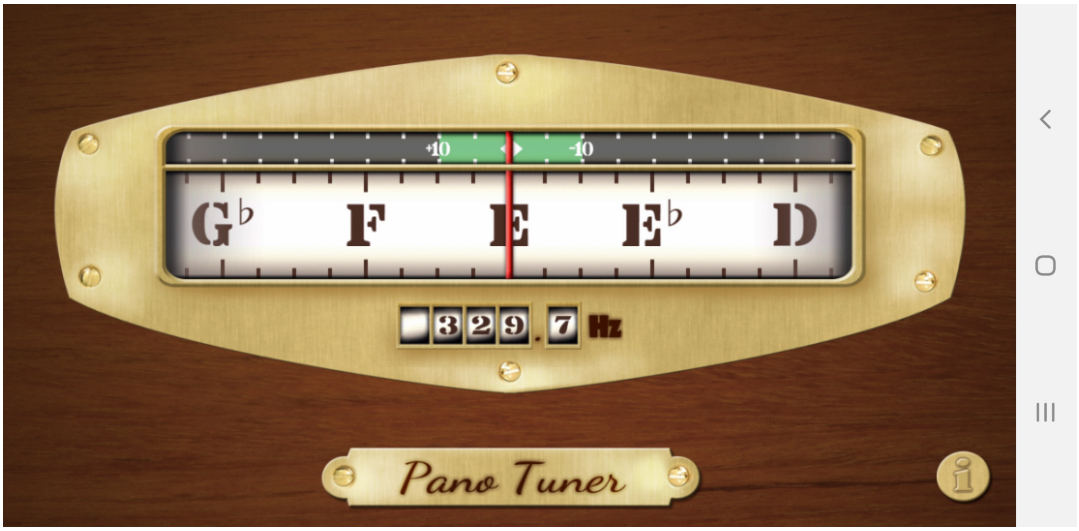
$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 15,168.5223$$

Find A, B, C so that N = A*B*C. One solution is

- A = 15
- B = 252.8 (round to 253)
- C = 4

T2CON is then 0x75

T2CON = 0x75							
7	6	5	4	3	2	1	0
0	1	1	1	0	1	0	1
	A = 15				T2E	C = 5	



Target = 329.63 Hz

```
Memory Summary:
    Program space        used   9BAh (  2490) of 10000h bytes   (  3.8%)
    Data space           used    33h (    51) of   F80h bytes   (  1.3%)
    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%)
```

Code:

```
// Global Variables

const unsigned char MSG0[21] = "Banjo               ";
const unsigned char MSG1[21] = "329.63 Hz           ";
```

Steppper Motor Roulette Wheel

3) Requirements: Explain what the inputs are / what the outputs are / and how they relate. Also explain how Timer2 interrupts will be used in your embedded system.

Input:

- RB0

Output:

- Stepper Motor (on PORTA)
- LCD Display (on PORTD)

Relationship:

- To start the game, press and release RB0.
- This generates a random number from 0..7
- The stepper motor then turns 3 rotations plus 25*N steps at a rate of 10ms/step (set by Timer2)
- The number (0..7) is also displayed on the LCD display as the stepper motor turns

Calculations:

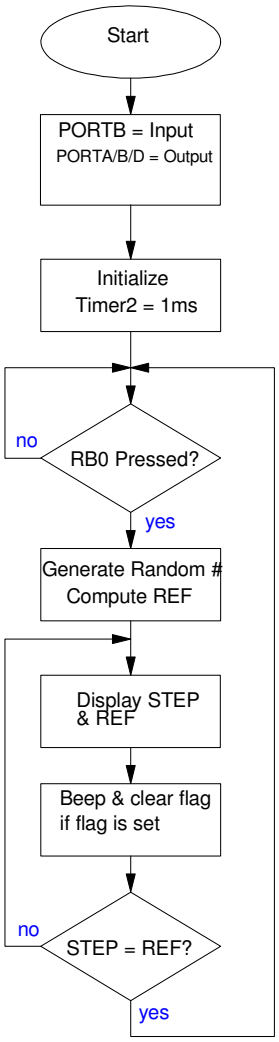
10ms/step is too large for Timer2 directly. So, a counter is added so that the stepper motor turns every 10th interrupt

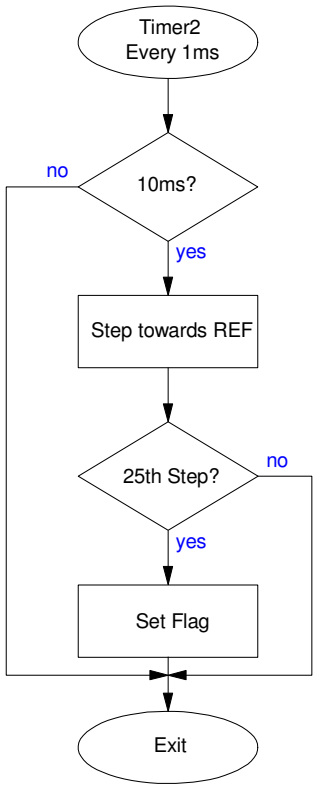
- Timer2: 1ms
  - A = 10
  - B = 250
  - C = 4
  - Toggle RD0 every interrupt (results in 500Hz square wave on RD0)
- 10th interrupt = 10ms
  - Step the motor every 10ms

4) C-Code and flow chart.

```
< insert code >
Memory Summary:
    Program space        used  1256h (  4694) of 10000h bytes   (  7.2%)
    Data space           used    46h (    70) of   F80h bytes   (  1.8%)
    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%)
```

Note: With interrupts, you pretty much need to use separate flow charts for each interrupt





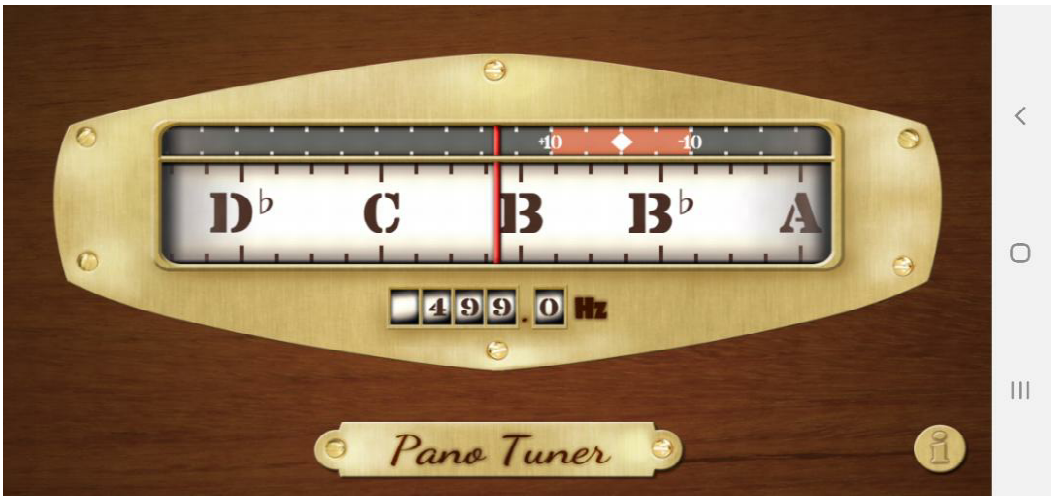
Validation of code:

- Pressing RB0 results in a random number (0..7)
- The stepper motor makes 2 rotations then stops at the winning number
- When passing by each number, the stepper motor beeps
- Pressing RB0 again results in a new winning number (0..7)

5) Data. Your raw data (at least two data points)

Timer2 Interrupt

- 499.0Hz
- It's being called every 1ms



Winning Numbers

- 1, 6, 3, 0, 1, 5, 0, 3, 7, 5, 6, 0, 5

6) Statistical Analysis: Analyze your data to determine

- The 90% confidence interval, or
- Who in your group can jump the highest (with what probability level), or
- Something else (your pick - just use some statistics to anlayze your data)

With only 14 numbers, there isn't enough data to do a chi-squared test with 8 bins, so use two bins

bin	p	np	N	chi-squared
even	0.5	7	8	1/7
odd	0.5	7	6	1/7
		Total		2/7

From StatTrek, with 1 degree of freedom, this corresponds to a probability of 0.41

I am 41% certain this is not a fair die

Using a different grouping:

bin	p	np	N	chi-squared
0..3	0.5	7	7	0/7
4..7	0.5	7	7	0/7
		Total		0/7

I am 0% certain this is not a fair die

Using yet another grouping:

bin	p	np	N	chi-squared
0 or 5	2/8	3.5	6	1.786
other	6/8	10.5	11	0.595
		Total		2.381

From a chi-squared table with 1 degree of freedom, this corresponds to a probability of 0.88

I am 88% certain this is not a fair die

7) Demo (in person during Zoom office hours or in a video)