ECE 376 - Homework #5

Keypads in C, Stepper Motors, NeoPixels in C

NeoPixel Flashlight

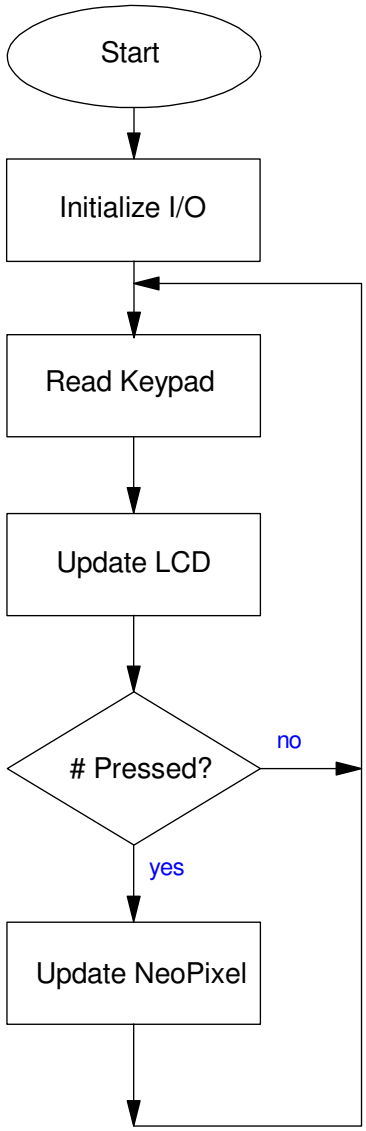
1) Requirements: Specify the inputs / outputs / how they relate.

- Input a number from 0..255 using the keypad
- Press RB0
- The NeoPixel then lights up with a white light at that brightness level (0..255)

2) C code, flow chart, and resulting number of lines of assembler

Code: Main Loop

```
RED = 0;
GREEN = 0;
BLUE = 0;

while(1) {
   TEMP = ReadKey();

   if (TEMP < 10) X = (X*10) + TEMP;

   if (TEMP == 10) {
      RED = X;
      GREEN = X;
      BLUE = X;
      }

   if (TEMP == 11) {
      X = X / 10;
      }

   LCD_Move(1,5);  LCD_Out(X, 5, 0);


   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);
   NeoPixel_Display(RED, GREEN, BLUE);

   Wait_ms(100);
   }

}
```



Compiler Results

```
Memory Summary:
    Program space        used   F14h (  3860) of 10000h bytes   (  5.9%)
    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%)
```

3) Validation: Collect data in lab to verify you met the requirements.

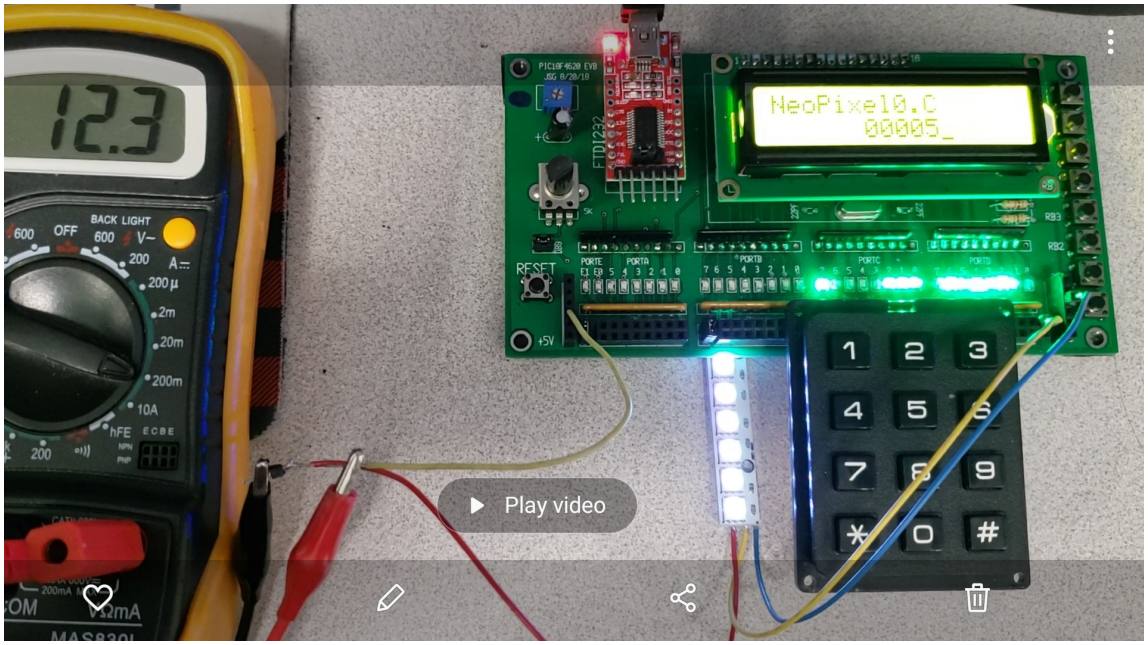
Requirement: Input a number from 000 to 255 using the keypad

- Input 000 (works)
- Input 255 (works)
- Input 123 (works)

Requirement: Press #. The NeoPixel goes to that brightness (255 = 100%)

Input Number	NeoPixels	Current (mA)	% Full Scale theory	% Full Scale measured
0	off	7.1	0%	0.0%
5	dim	12.0	1.9%	1.9%
50		58.9	19.6%	20.48%
100		110.0	39.2%	40.69%
255	really bright	260	100%	100.0%

4) Demo. Video or in person.



Analog Inputs

5) Determine how long it takes to do an A/D conversion with a PIC processor

```
void main(void)
{
   TRISC = 0;
   ADCON1 = 0x0F;

// Turn on the A/D input
   TRISA = 0xFF;
   TRISE = 0x0F;
   ADCON2 = 0x95;
   ADCON1 = 0x07;
   ADCON0 = 0x01;

   while(1) {
      A2D = A2D_Read(0);
      PORTC = PORTC + 1;
      }
   }
```

RC4 = 963.0Hz

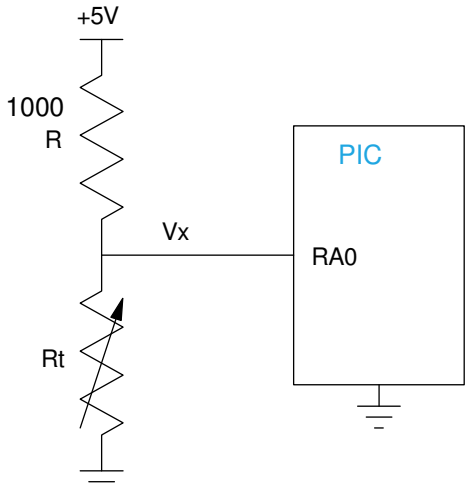
RC0 = 16 x RC4

RC0 = 15,408Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot 15408Hz}\right) = 324.5$$

It takes about 324 clocks (32.4us) to do a single A/D read

6) Assume the A/D reads 713 for the following circuit



What is the voltage Vx?

$$V_x = \left(\frac{713}{1023}\right)5V$$

$$V_x = 3.4848V$$

What is the resistance Rt?

$$V_x = \left(\frac{R_t}{R_t+1000}\right)5V$$

solving for Rt

$$R_t = \left(\frac{V_x}{5-V_x}\right)1000\Omega$$

$$R_t = 2300\Omega$$

What is the temperature?

$$R_t = 1000 \cdot \exp\left(\frac{3905}{T+273} - \frac{3905}{298}\right)\Omega$$

$$2300\Omega = 1000 \cdot \exp\left(\frac{3905}{T+273} - \frac{3905}{298}\right)\Omega$$

Solving

$$T = 7.1907^0 C$$

Combination Lock

7) Requirements & Flow Chart

Input: A/D input. button RB0

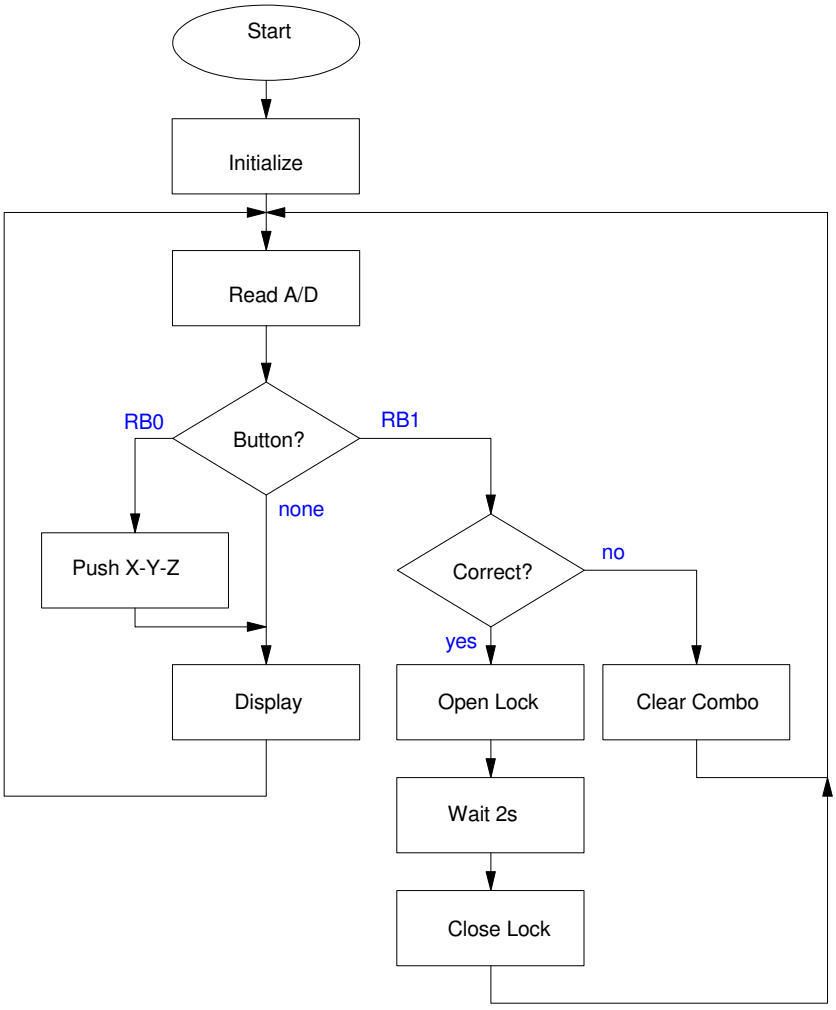
Output: Stepper Motor

Relationship:

- Turning the potentiometer sets the number from 0..20
- Press RB0 to push that number on the stack
- After 3 numbers are input, press RB1 to test that combination

If the combination is correct,

- The stepper motor rotates 180 degrees at 10ms/step,
- Pauses 2 seconds, then
- Rotates back to zero degrees



8) C code

```
Memory Summary:
    Program space        used  1540h (  5440) of 10000h bytes   (  8.3%)
    Data space           used    31h (    49) of   F80h bytes   (  1.2%)
    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%)


// Ohm_Meter.C
//
// This program reads the A/D on RA0
// Computes the voltage

// Global Variables

const unsigned char MSG0[21] = "Combination Lock   ";
const unsigned char TABLE[4] = {1, 2, 4, 8};


// Subroutines
#include <pic18.h>
#include        "lcd_portd.c"


unsigned int A2D_Read(unsigned char c)
{
   unsigned int result;
   unsigned char i;
   c = c & 0x0F;
   ADCON0 = (c << 2) + 0x01; // set Channel Select
   for (i=0; i<20; i++); // wait 2.4us (approx)
   GODONE = 1; // start the A/D conversion
   while(GODONE); // wait until complete (approx 8us)
   return(ADRES);
   }


// Main Routine

void main(void)
{
   int A2D;
   int X, Y, Z, T;
   int STEP;
   unsigned int i, j;

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

   LCD_Init();                  // initialize the LCD

   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   Wait_ms(500);

// Initialize the A/D port
   TRISA = 0xFF;
   TRISE = 0x0F;
   ADCON2 = 0x85;
   ADCON1 = 0x07;
   ADCON0 = 0x01;
```

```
i = 0;
X = 0;
Y = 0;
Z = 0;
T = 0;

while(1) {

   A2D = A2D_Read(0) / 51.15 + 1;

   if(RB0) {
      T = Z;
      Z = Y;
      Y = X;
      X = A2D;
      while(RB0);
   }

   if(RB1) {
      if( (X == 1)*(Y == 2)*(Z == 3) ) {
         for(i=0; i<100; i++) {
            STEP = STEP + 1;
            PORTC = TABLE[STEP % 4];
            Wait_ms(10);
            }
         Wait_ms(2000);
         for(i=0; i<100; i++) {
            STEP = STEP - 1;
            PORTC = TABLE[STEP % 4];
            Wait_ms(10);
            }
          }
      X = 0;
      Y = 0;
      Z = 0;
      }

   LCD_Move(1,0); LCD_Out(A2D, 2, 0);
   LCD_Move(1,4);  LCD_Out(X, 2, 0);
   LCD_Move(1,8);  LCD_Out(Y, 2, 0);
   LCD_Move(1,12); LCD_Out(Z, 2, 0);
   }
}
```

9) Validation (refer to the requirements)

Input a number from 1..20 using the analog input

- Check: as you turn the pot the number comes up as 1..20

Press RB0 to push that number onto the stack

- Check: the number gets pushed as A/D >> X >> Y >> Z >> T

Press RB1 to check the combination

- Correct (1/2/3) results in the motor turning 180 degrees (100 steps), pauses 2 seconds, then back to zero.
- Incorrect (other) results in the comination being cleared.

10) Demo