ECE 376 - Homework #4

C Programming, LCD Displays, Keypads - Due Monday, February 17th

1) Determine how many clocks the following C code takes to execute

1a) Counting mod 64

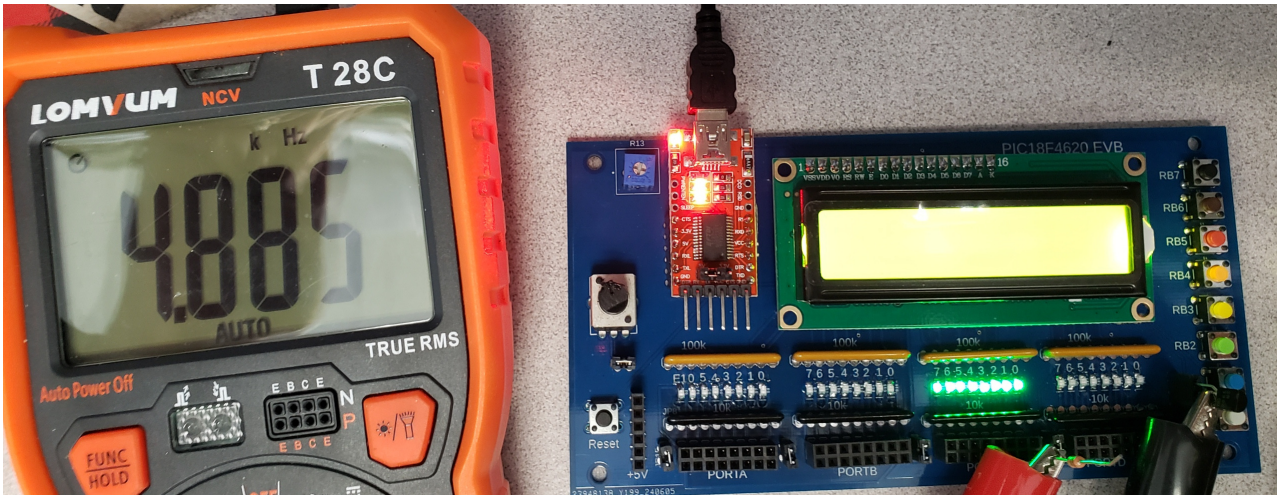
```
unsigned char i
while(1) {
      i = (i + 1) % 64;
      if(i == 0) PORTC += 1;
      }
```

f = 4885Hz+

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 1023.54 \text{ clocks / toggle}$$

$$\frac{N}{64} = 15.992 \text{ clocks / loop}$$

It takes 16 clocks to complete one loop (count mod 64)



1b) Counting mod 63

```
unsigned char i
while(1) {
      i = (i + 1)% 63;
      if(i == 0) PORTC += 1;
      }
```

f = 140.4Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 35{,}612.53 \text{ clocks / toggle}$$

$$\frac{N}{63} = 565.27 \text{ clocks / loop}$$

It takes 565 clocks to complete one loop (count mod 63)

1c) Integer Division

```
unsigned int A, B, C;
A = 12345;
B = 273;
while(1) {
    PORTC += 1;
    C = A / B;
    }
```

f = 10.10kHz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 495.05 \text{ clocks / loop}$$

It takes 495 clocks to complete one loop (integer divide)

1d) Floating Point Division

```
float A, B, C;
A = 123.456;
B = 7.2143;
while(1) {
    PORTC += 1;
    C = A / B;
    }
```

f = 2660Hz

$$N = \left(\frac{10{,}000{,}000}{2 \cdot Hz}\right) = 1879.7 \text{ clocks / loop}$$

It takes 1879 clocks to do a floating point divide

LCD Display & LED Flashlight!

Turn your PIC board into an LED flashlight

2) Write the C code for an LED flashlight. The requirements are:

- On power up, the LEDs are off
- Each button sets the color of the flashlight
- If a button isn't pressed for 10 seconds, the lights turn off
- The status of the LED flashlight is displayed on the LCD display (Off / Red / Green / Blue / White)

C-Code:

```
// Global Variables

// Subroutine Declarations
#include <pic18.h>
#include "lcd_portd.c"

// Subroutines

// Main Routine

void main(void)
{
    unsigned int TIME;

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

    while(1) {
        if(RB0) PORTC = 0;
        if(RB1) {
               PORTC = 1;
               TIME = 100;
               }
         if(RB2) {
               PORTC = 2;
               TIME = 100;
         }
         if(RB3) {
               PORTC = 4;
               TIME = 100;
         }
         if(RB4) {
               PORTC = 7;
               TIME = 100;
         }
         if(TIME == 0) PORTC = 0;
         if(TIME) TIME -= 1;
         PORTD = TIME;
         Wait_ms(100);
         }
   }
```

Result: Program = 384 bytes (192 lines of assembler)

- 74 lines of assembler back in homework #3
- C code is 5.18x larger

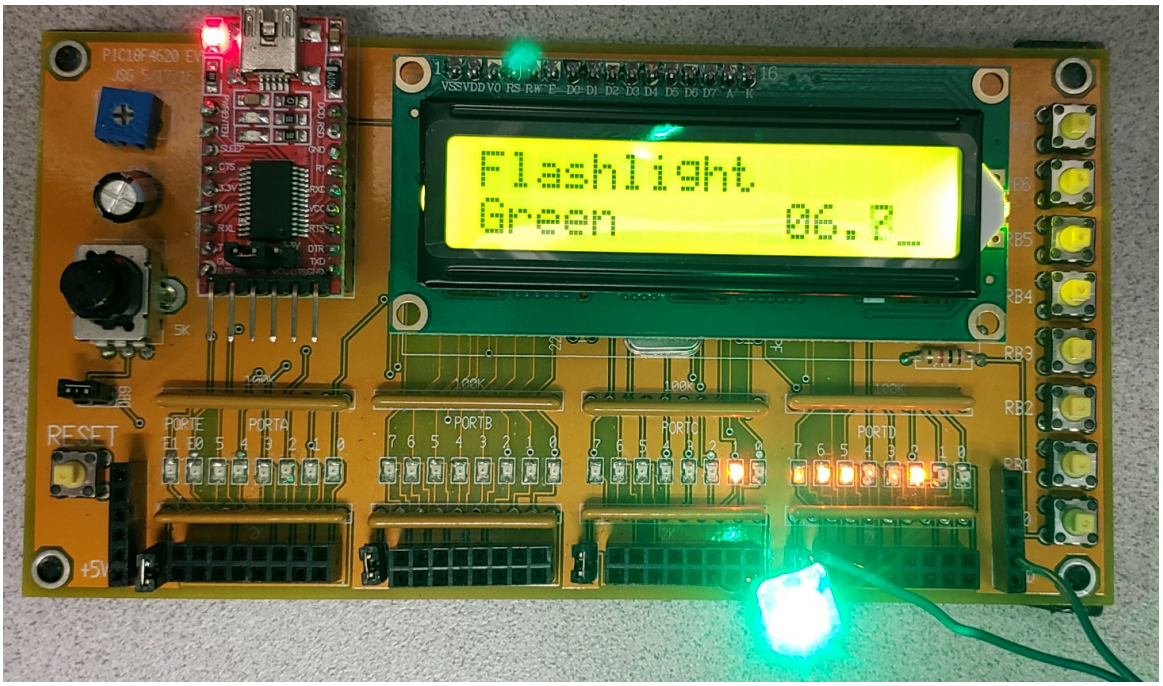
```
Memory Summary:
    Program space        used   180h (   384) of 10000h bytes    (  0.6%)
    Data space           used     9h (     9) of   F80h bytes    (  0.2%)
    EEPROM space         used     0h (     0) of   400h bytes    (  0.0%)
    ID Location space    used     0h (     0) of     8h nibbles  (  0.0%)
    Configuration bits   used     0h (     0) of     7h words    (  0.0%)
```

If you add LCD routines (photo below), the code size becomes 3198 bytes

```
Memory Summary:
    Program space        used   C7Eh (  3198) of 10000h bytes    (  4.9%)
    Data space           used    26h (    38) of   F80h bytes    (  1.0%)
    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) Collect data to verify your LED flashlight meets the requirements

- RB1 turns on RC0 (red)
- RB2 turns on RC1 (green)
- RB3 turns on RC2 (blue)
- RB4 turns on RC1/2/3 (white)
- RB0 turns off lights
- 10 seconds after pressing a button, the lights turn off



```
// LED Flashlight Code (with LCD functions)
// Global Variables

const unsigned char msg[20] = "Flashlight       ";
const unsigned char msg0[20] = "Off              ";
const unsigned char msg1[20] = "Red              ";
const unsigned char msg2[20] = "Green            ";
const unsigned char msg3[20] = "Blue             ";
const unsigned char msg4[20] = "White            ";

// Subroutine Declarations
#include <pic18.h>
#include "lcd_portd.c"

// Subroutines

// Main Routine

void main(void)
{
    unsigned int TIME;
   unsigned char i;

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

    LCD_Init();
   Wait_ms(100);
   LCD_Move(0,0);  for(i=0; i<16; i++) LCD_Write(msg[i]);


    while(1) {
        if(RB0) {
            PORTC = 0;
                 LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg0[i]);
                 }
        if(RB1) {
                 PORTC = 1;
                 TIME = 100;
                 LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg1[i]);
                 }
          if(RB2) {
                 PORTC = 2;
                 TIME = 100;
                 LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg2[i]);
          }
          if(RB3) {
                 PORTC = 4;
                 TIME = 100;
                 LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg3[i]);
          }
          if(RB4) {
                 PORTC = 7;
                 TIME = 100;
                 LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg4[i]);
          }
          if(TIME == 0) PORTC = 0;
          if(TIME) {
                 TIME -= 1;
                 if(TIME == 0) {
                        LCD_Move(1,0);  for(i=0; i<10; i++) LCD_Write(msg0[i]);
                        }
                 }
          LCD_Move(1,10);  LCD_Out(TIME, 3, 1);
          Wait_ms(100);
          }
   }
```

Keypads:

Turn your PIC board into a clock

- The LCD displays the time as hours : minutes : seconds
- When you type in a number on the keypad and press a button, one of these numbers is updated:
  - RB0: Update seconds (0-59)
  - RB1: Update minutes (0-59)
  - RB2: Update hours (0-23)

4) Write a C program to read the keypad and update the LCD display. Include

- Your C code
- The compiled size of your C code

```
// Global Variables

unsigned char msg[20] = "hr : min : sec   ";

// Subroutine Declarations
#include <pic18.h>
#include "lcd_portd.c"

// Subroutines

void Display(unsigned int hr, unsigned int min, unsigned int sec) {
    LCD_Move(1,0);
     LCD_Out(hr, 2, 0);
    LCD_Write(':');
    LCD_Out(min, 2, 0);
    LCD_Write(':');
    LCD_Out(sec, 3, 1);
}

// Keypad Routines

char GetKey(void)
{
   int i;
   unsigned char RESULT;
   TRISC = 0xF8;
   RESULT = 0xFF;
   PORTC = 4;
   for (i=0; i<100; i++);
   if (RC6) RESULT = 1;
   if (RC5) RESULT = 4;
   if (RC4) RESULT = 7;
   if (RC3) RESULT = 10;
   PORTC = 2;
   for (i=0; i<100; i++);
   if (RC6) RESULT = 2;
   if (RC5) RESULT = 5;
   if (RC4) RESULT = 8;
   if (RC3) RESULT = 0;
   PORTC = 1;
   for (i=0; i<100; i++);
   if (RC6) RESULT = 3;
   if (RC5) RESULT = 6;
   if (RC4) RESULT = 9;
   if (RC3) RESULT = 11;
   if (RB0) RESULT = 12;
   if (RB1) RESULT = 13;
   if (RB2) RESULT = 14;
   if (RB3) RESULT = 15;
   if (RB4) RESULT = 16;
   PORTC = 0;
   return(RESULT);
}

char ReadKey(void)
{
   char X, Y;
   do {
      X = GetKey();
      }   while(X > 20);
   do {
      Y= GetKey();
      }   while(Y < 20);
   Wait_ms(100);  // debounce
   return(X);
   }
```

```
// Main Routine

void main(void)
{
    unsigned int hr, min, sec;
   unsigned int i, TEMP, X;


    TRISA = TRISC = TRISD = TRISE = 0;
    TRISB = 0xFF;
    PORTA = PORTB = PORTC = PORTD = PORTE = 0;
    ADCON1 = 0x0F;

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

   X = 0;

   while(1) {
         TEMP = ReadKey();

         if (TEMP < 10) X = (X*10) + TEMP;
         if (TEMP == 11) X = X / 10;
         if(TEMP == 12) hr = X;
         if(TEMP == 13) min = X;
         if(TEMP == 14) sec = X;
         if(TEMP == 15) {
               while( (hr+min+sec>0) ){
                     if(sec == 0) {
                           if(min == 0) {
                                 hr -= 1;
                                 min = 59;
                                 }
                           else {
                                 min -= 1;
                                 sec = 599;
                                 }
                           }
                     else sec -= 1;
                     Display(hr, min, sec);
                     Wait_ms(100);
                     }
               }

         LCD_Move(0,10);  LCD_Out(X, 3, 0);
         Display(hr, min, sec);

      }

   }
```

5) Collect data to verify your keypad-controlled clock

Press 55 then RB0

- 55 shows up under hours

press RB1

- 55 shows up under minutes

Press RB2

- 5.5 shows up under seconds

Press ##

- clears out 55 one digit at a time

Press RB3 (showing off - not requried)

- Counts down to 00:00:00.0
- One count every 100ms

Demo

6) Demo either the flashlight or the keypad.

- Video or in-person