Basic and Finite State Machine LED and Switch I/O Programming
6th Laboratory Report for ECE 383 Microcomputers

Submitted by
Patrick Brooks
11650957

The University of Alabama
Tuscaloosa, AL 35487

3/19/21

Abstract:

The main objectives of this lab are to continue to instruct students in a PIC24-based hardware system using GPIO ports for switch input and LED output using basic and algorithmic state machine processing. They will get a glimpse into the PIC24 ecosystem and how certain pins are used. Students will carry these skills with them throughout their careers. Task one starts by having students build a schematic using PCB Artist. Task two instructs students to create a PCB based off of the schematic from task 1. Task three is where students will create a basic LED program. Task four requires students to create a program that follows what is laid out in the lab instructions. Task 5 instructs students to create a variable rotating LED using binary and gray code to create a specific color.

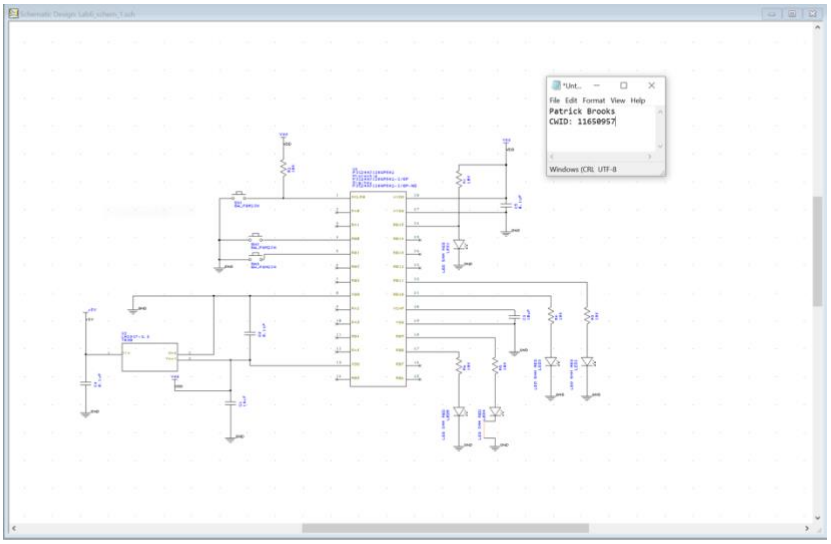
Introduction:

The objectives of the lab are to continue to instruct students in a PIC24-based hardware system using GPIO ports for switch input and LED output using basic and algorithmic state machine processing Task one starts by having students build a schematic using PCB Artist. In task two students create a PCB based off of the schematic from task 1. Task three is where students will create a basic LED program. Task four requires students to create a program that follows what is laid out in the lab instructions. Task 5 instructs students to create a variable rotating LED using binary and gray code to create a specific color.

Procedure/Results:

Task 1: Expanding PIC24 Reference Schematic

a. Use PCB Artist to add on to existing PIC24
    i. Configure the schematic as seen in the lab instructions



Patrick Brooks
CWID: 11650957

Figure 1. Deliverable 1

<u>Task 2:</u> Expanded PIC24 PCB Layout

a. Create a PCB by navigating through the PCB wizard

i. Follow the lab instructions to create the PCB

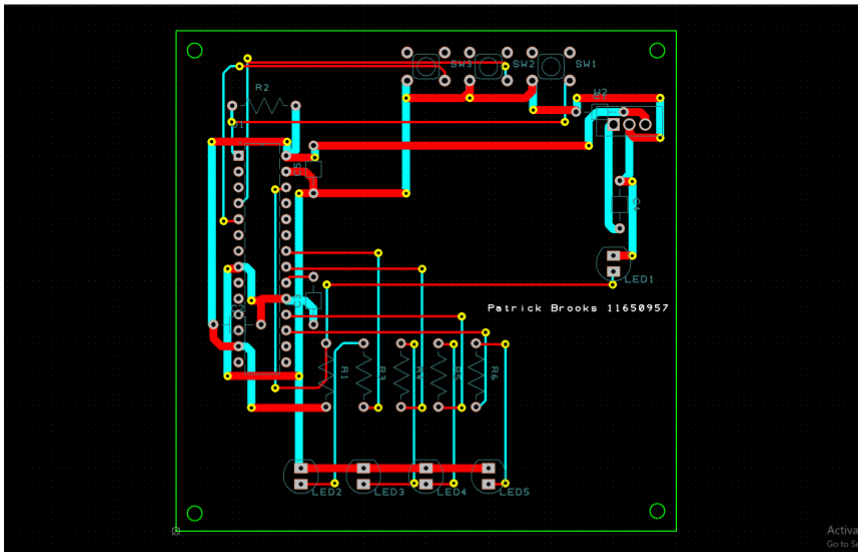
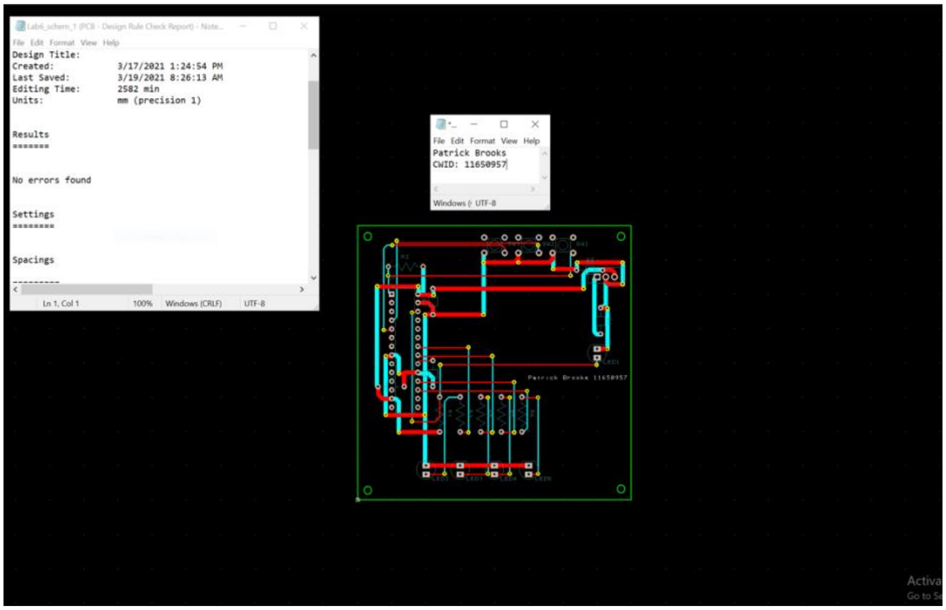


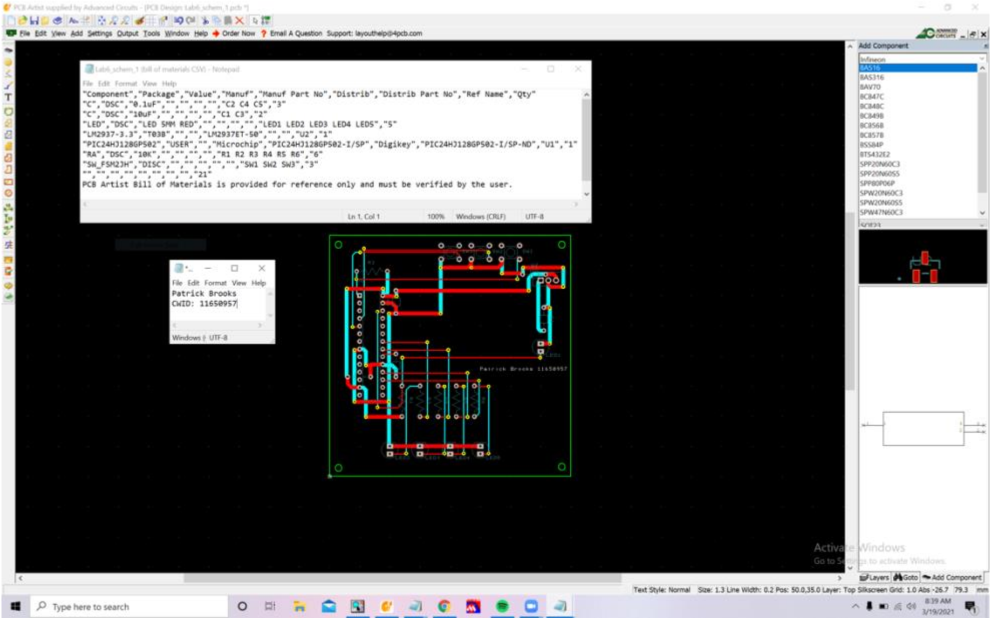
Figure 2. Deliverable 2


Lab6_schem_1 (PCB - Design Rule Check Report) - Note...
File Edit Format View Help
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
File Edit Format View Help
Windows (
UTF-8
Patrick Brooks 11650957
Activa
Go to Se

```
Design Title:
Created:            3/17/2021 1:24:54 PM
Last Saved:         3/19/2021 8:26:13 AM
Editing Time:       2582 min
Units:              mm (precision 1)


Results
=======


No errors found


Settings
========


Spacings

---------
```

```
Patrick Brooks
CWID: 11650957
```

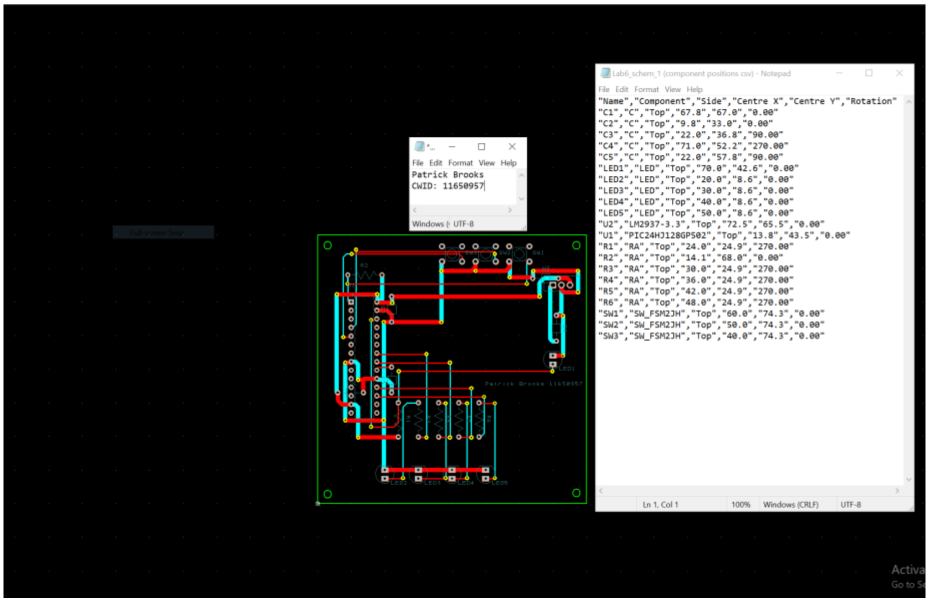
Figure 3. Deliverable 3



```
"Component","Package","Value","Manuf","Manuf Part No","Distrib","Distrib Part No","Ref Name","Qty"
"C","DSC","0.1uF","","","","","C2 C4 C5","3"
"C","DSC","10uF","","","","","C1 C3","2"
"LED","DSC","LED 5MM RED","","","","","LED1 LED2 LED3 LED4 LED5","5"
"LM2937-3.3","T03B","","","LM2937ET-50","","","U2","1"
"PIC24HJ128GP502","USER","","Microchip","PIC24HJ128GP502-I/SP","Digikey","PIC24HJ128GP502-I/SP-ND","U1","1"
"RA","DSC","10K","","","","","R1 R2 R3 R4 R5 R6","6"
"SW_FSM2JH","DISC","","","","","","SW1 SW2 SW3","3"
"","","","","","","","","21"
PCB Artist Bill of Materials is provided for reference only and must be verified by the user.
```

```
Patrick Brooks
CWID: 11650957
```

Figure 4. Deliverable 4



```
Patrick Brooks
CWID: 11650957
```

```
"Name","Component","Side","Centre X","Centre Y","Rotation"
"C1","C","Top","67.8","67.0","0.00"
"C2","C","Top","9.8","33.0","0.00"
"C3","C","Top","22.0","36.8","90.00"
"C4","C","Top","71.0","52.2","270.00"
"C5","C","Top","22.0","57.8","90.00"
"LED1","LED","Top","70.0","42.6","0.00"
"LED2","LED","Top","20.0","8.6","0.00"
"LED3","LED","Top","30.0","8.6","0.00"
"LED4","LED","Top","40.0","8.6","0.00"
"LED5","LED","Top","50.0","8.6","0.00"
"U2","LM2937-3.3","Top","72.5","65.5","0.00"
"U1","PIC24HJ128GP502","Top","13.8","43.5","0.00"
"R1","RA","Top","24.0","24.9","270.00"
"R2","RA","Top","14.1","68.0","0.00"
"R3","RA","Top","30.0","24.9","270.00"
"R4","RA","Top","36.0","24.9","270.00"
"R5","RA","Top","42.0","24.9","270.00"
"R6","RA","Top","48.0","24.9","270.00"
"SW1","SW_FSM2JH","Top","60.0","74.3","0.00"
"SW2","SW_FSM2JH","Top","50.0","74.3","0.00"
"SW3","SW_FSM2JH","Top","40.0","74.3","0.00"
```

Figure 5. Deliverable 5

Task 3: Basic LED Problem

a. Start MPLAB IDE

i. Use project wizard to set up your project using the lab instructions

ii. Use program in lab instructions to create results with your PIC24 setup

Task 3 was demoed for TA in designated lab time

Figure 7. Deliverable 7

Task 3 was demoed for TA in designated lab time

Figure 8. Deliverable 8

Task 4: Software-Based Finite State Machine for LED/Switch I/O

a. Replicate the schematic on your breadboard

i. Create code to implement what is asked for in lab instructions

ii. Use PIC24 and MPLAB to run the code you wrote

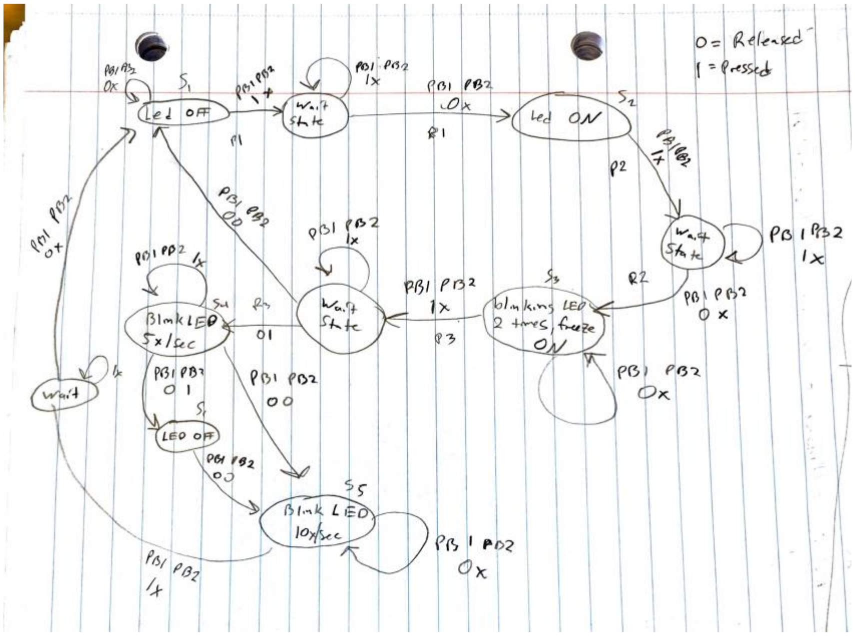
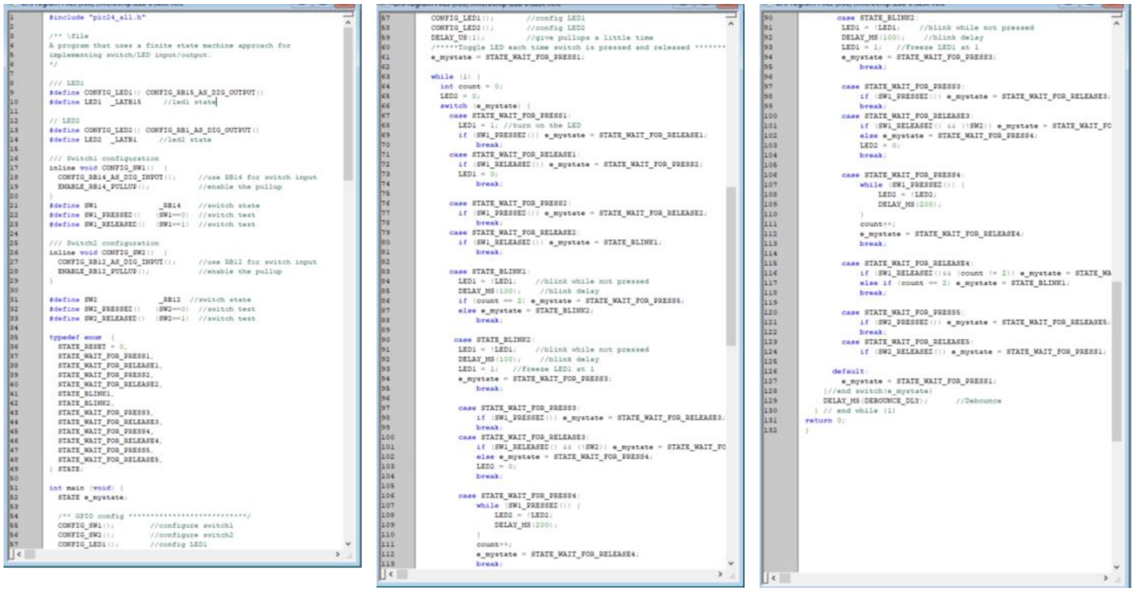


Figure 9. Deliverable 9



```
#include "pic24_all.h"

/** \file
A program that uses a finite state machine approach for
implementing switch/LED input/output.
*/

/// LED1
#define CONFIG_LED1() CONFIG_RB15_AS_DIG_OUTPUT()
#define LED1  _LATB15     //led1 state

// LED2
#define CONFIG_LED2() CONFIG_RB1_AS_DIG_OUTPUT()
#define LED2  _LATB1     //led1 state

/// Switch1 configuration
inline void CONFIG_SW1()  {
  CONFIG_RB14_AS_DIG_INPUT();     //use RB14 for switch input
  ENABLE_RB14_PULLUP();           //enable the pullup
}
#define SW1              _RB14    //switch state
#define SW1_PRESSED()    (SW1==0)  //switch test
#define SW1_RELEASED()   (SW1==1)  //switch test

/// Switch2 configuration
inline void CONFIG_SW2()  {
  CONFIG_RB12_AS_DIG_INPUT();     //use RB12 for switch input
  ENABLE_RB12_PULLUP();           //enable the pullup
}

#define SW2              _RB12  //switch state
#define SW2_PRESSED()    (SW2==0)  //switch test
#define SW2_RELEASED()   (SW2==1)  //switch test

typedef enum  {
  STATE_RESET = 0,
  STATE_WAIT_FOR_PRESS1,
  STATE_WAIT_FOR_RELEASE1,
  STATE_WAIT_FOR_PRESS2,
  STATE_WAIT_FOR_RELEASE2,
  STATE_BLINK1,
  STATE_BLINK2,
  STATE_WAIT_FOR_PRESS3,
  STATE_WAIT_FOR_RELEASE3,
  STATE_WAIT_FOR_PRESS4,
  STATE_WAIT_FOR_RELEASE4,
  STATE_WAIT_FOR_PRESS5,
  STATE_WAIT_FOR_RELEASE5,
} STATE;

int main (void) {
  STATE e_mystate;

  /** GPIO config ***************************/
  CONFIG_SW1();         //configure switch1
  CONFIG_SW2();         //configure switch2
  CONFIG_LED1();        //config LED1
```

```
  CONFIG_LED1();        //config LED1
```

```
  CONFIG_LED2();        //config LED2
  DELAY_US(1);          //give pullups a little time
  /*****Toggle LED each time switch is pressed and released *******
  e_mystate = STATE_WAIT_FOR_PRESS1;

  while (1) {
    int count = 0;
    LED2 = 0;
    switch (e_mystate) {
      case STATE_WAIT_FOR_PRESS1:
        LED1 = 1; //turn on the LED
        if (SW1_PRESSED()) e_mystate = STATE_WAIT_FOR_RELEASE1;
            break;
      case STATE_WAIT_FOR_RELEASE1:
        if (SW1_RELEASED()) e_mystate = STATE_WAIT_FOR_PRESS2;
        LED1 = 0;
            break;

      case STATE_WAIT_FOR_PRESS2:
        if (SW1_PRESSED()) e_mystate = STATE_WAIT_FOR_RELEASE2;
            break;
      case STATE_WAIT_FOR_RELEASE2:
        if (SW1_RELEASED()) e_mystate = STATE_BLINK1;
            break;

      case STATE_BLINK1:
        LED1 = !LED1;    //blink while not pressed
        DELAY_MS(100);    //blink delay
        if (count == 2) e_mystate = STATE_WAIT_FOR_PRESS5;
        else e_mystate = STATE_BLINK2;
            break;

       case STATE_BLINK2:
        LED1 = !LED1;    //blink while not pressed
        DELAY_MS(100);    //blink delay
        LED1 = 1;   //Freeze LED1 at 1
        e_mystate = STATE_WAIT_FOR_PRESS3;
            break;

        case STATE_WAIT_FOR_PRESS3:
            if (SW1_PRESSED()) e_mystate = STATE_WAIT_FOR_RELEASE3;
            break;
        case STATE_WAIT_FOR_RELEASE3:
            if (SW1_RELEASED() && (!SW2)) e_mystate = STATE_WAIT_FO
            else e_mystate = STATE_WAIT_FOR_PRESS4;
            LED2 = 0;
            break;

        case STATE_WAIT_FOR_PRESS4:
            while (SW1_PRESSED()) {
                LED2 = !LED2;
                DELAY_MS(200);
            }
            count++;
            e_mystate = STATE_WAIT_FOR_RELEASE4;
            break;
```

```
       case STATE_BLINK2:
        LED1 = !LED1;    //blink while not pressed
        DELAY_MS(100);    //blink delay
        LED1 = 1;   //Freeze LED1 at 1
        e_mystate = STATE_WAIT_FOR_PRESS3;
            break;

        case STATE_WAIT_FOR_PRESS3:
            if (SW1_PRESSED()) e_mystate = STATE_WAIT_FOR_RELEASE3;
            break;
        case STATE_WAIT_FOR_RELEASE3:
            if (SW1_RELEASED() && (!SW2)) e_mystate = STATE_WAIT_FO
            else e_mystate = STATE_WAIT_FOR_PRESS4;
            LED2 = 0;
            break;

        case STATE_WAIT_FOR_PRESS4:
            while (SW1_PRESSED()) {
                LED2 = !LED2;
                DELAY_MS(200);
            }
            count++;
            e_mystate = STATE_WAIT_FOR_RELEASE4;
            break;
```

```

        case STATE_WAIT_FOR_RELEASE4:
            if (SW1_RELEASED()&& (count != 2)) e_mystate = STATE_WA
            else if (count == 2) e_mystate = STATE_BLINK1;
            break;

        case STATE_WAIT_FOR_PRESS5:
            if (SW2_PRESSED()) e_mystate = STATE_WAIT_FOR_RELEASE5;
            break;
        case STATE_WAIT_FOR_RELEASE5:
            if (SW2_RELEASED()) e_mystate = STATE_WAIT_FOR_PRESS1;

      default:
        e_mystate = STATE_WAIT_FOR_PRESS1;
    }//end switch(e_mystate)
    DELAY_MS(DEBOUNCE_DLY);       //Debounce
  } // end while (1)
return 0;
}
```

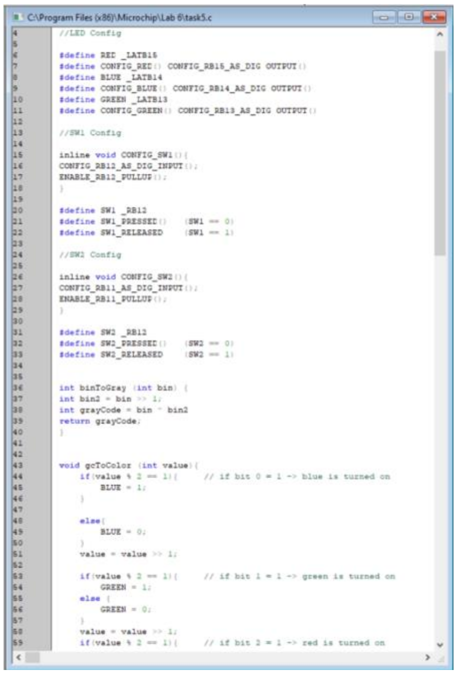
Figure 10. Deliverable 10

Task 5: Variable Rotating LED

a. Replicate the schematic on your breadboard

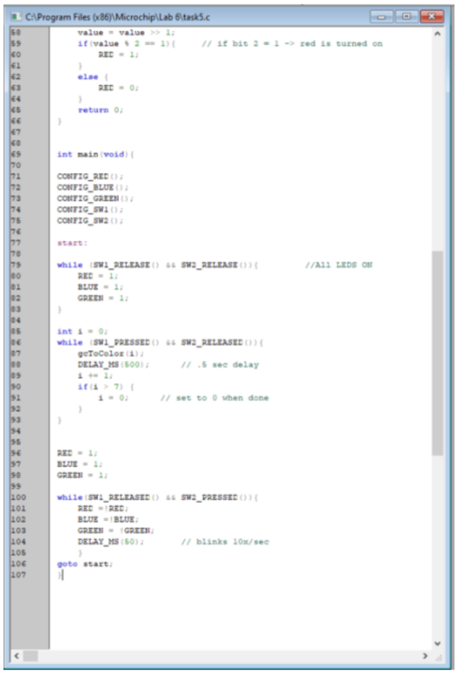
i. Create code to implement what is asked for in lab instructions

ii. Use PIC24 and MPLAB to run the code you wrote



C:\Program Files (x86)\Microchip\Lab 6\task5.c

```
//LED Config

#define RED _LATB15
#define CONFIG_RED() CONFIG_RB15_AS_DIG OUTPUT()
#define BLUE _LATB14
#define CONFIG_BLUE() CONFIG_RB14_AS_DIG OUTPUT()
#define GREEN _LATB13
#define CONFIG_GREEN() CONFIG_RB13_AS_DIG OUTPUT()

//SW1 Config

inline void CONFIG_SW1(){
CONFIG_RB12_AS_DIG_INPUT();
ENABLE_RB12_PULLUP();
}

#define SW1 _RB12
#define SW1_PRESSED()     (SW1 == 0)
#define SW1_RELEASED      (SW1 == 1)

//SW2 Config

inline void CONFIG_SW2(){
CONFIG_RB11_AS_DIG_INPUT();
ENABLE_RB11_PULLUP();
}

#define SW2 _RB12
#define SW2_PRESSED()     (SW2 == 0)
#define SW2_RELEASED      (SW2 == 1)


int binToGray (int bin) {
int bin2 = bin >> 1;
int grayCode = bin ^ bin2
return grayCode;
}


void gcToColor (int value){
    if(value % 2 == 1){      // if bit 0 = 1 -> blue is turned on
        BLUE = 1;
    }

    else{
        BLUE = 0;
    }
    value = value >> 1;

    if(value % 2 == 1){      // if bit 1 = 1 -> green is turned on
        GREEN = 1;
    else {
        GREEN = 0;
    }
    value = value >> 1;
    if(value % 2 == 1){      // if bit 2 = 1 -> red is turned on
```



C:\Program Files (x86)\Microchip\Lab 6\task5.c

```
    value = value >> 1;
    if(value % 2 == 1){      // if bit 2 = 1 -> red is turned on
        RED = 1;
    }
    else {
        RED = 0;
    }
    return 0;
}


int main(void){

CONFIG_RED();
CONFIG_BLUE();
CONFIG_GREEN();
CONFIG_SW1();
CONFIG_SW2();

start:

while (SW1_RELEASE() && SW2_RELEASE()){          //All LEDS ON
    RED = 1;
    BLUE = 1;
    GREEN = 1;
}

int i = 0;
while (SW1_PRESSED() && SW2_RELEASED()){
    gcToColor(i);
    DELAY_MS(500);       // .5 sec delay
    i += 1;
    if(i > 7) {
        i = 0;       // set to 0 when done
    }
}


RED = 1;
BLUE = 1;
GREEN = 1;

while(SW1_RELEASED() && SW2_PRESSED()){
    RED =!RED;
    BLUE =!BLUE;
    GREEN = !GREEN;
    DELAY_MS(50);        // blinks 10x/sec
    }
goto start;
}
```

Figure 12. Deliverable 12

Conclusion:

Unfortunately, my partner and I were never able to get our breadboard build to start working. We came to both lab sessions but were not able to get it to work. After completion of the lab students learned how to extend the reference PIC24 schematic, revisited how to create a PCB from schematic, create C code that implements pushing of buttons linked to an action of LED and learned how to critically think through these problems. Students also became familiar with the pins on the PIC24 and how they can be used to perform different tasks. We also revisited useful software tools implemented in both PCB Artist and MPLAB. Students also deepened their understanding of the lab report format which will be used throughout their academic endeavors.