Lab 3 Report
CSE 4560 Embedded System

Name: Daniel Delgado Acosta Coyote ID:006896598

I Goals

Goals of this Lab

• Get to know the MSP-EXP432P401R
• Learn how to use registers for configuration
• Get to know and use library functions
• Implement simple GPIO with peripherals
• Understand and implement polling
• Establish a simple UART communication

II Lab Questions, Processes and Program

Clicker Questions

1.) To which General Purpose Input Output (GPIO) Port are the LED1 and LED2 connected? (Hint: Check the Schematic Pages 1 and 2)
(c) Port 1 and Port 2

2.) Using C, which of the following operation allows toggling of the LSB of an 8-bit integer X?
(a) X ^ 0x01

3.) What is the declaration of the function I2C_initSlave()? (Hint: DriverLib Users Guide Section 12.6.3)

```
(d) void I2C_initSlave(
        uint32_t moduleInstance, uint_fast16_t slaveAddress,
        uint_fast8_t slaveAddressOffset, uint32_t
        slaveOwnAddressEnable
    )
```

4.) What is the UART module baud rate divider (BRDIV) for a baud rate of 4800 at low frequency baud rate generation, using the default setting of SMCLKas Clock Source? (Hint: LaunchPad Users Guide Section 2)
(b) 625

5.) Which of the following statements is not zero, when for the pins 5 or 7 of Port 2 the primary or tertiary module function is enabled? (Hint: Datasheet Table 6-1 & Technical Reference Manual Chapter 10 on PxSEL0 and PxSEL1)
(c) HWREG8(0x40004C00 + 0x0B) & 0xA0

Task 1: Flashing, Library Usage and Simple I/O

Task 1.1: Flashing an Application

What is the purpose of the application?

The purpose of the application is to make LED 1 blink.

Task 1.2: Using Library Functions instead of Hard-Coded Register Access

What is the advantage of using library functions and defines instead of direct register access?

Advantages include a more clear and concise code that is easier to change in the future.

Task 1.3: Adding a blinking LED

Do you need to set any register specifically to make sure the LEDs blink alternating in any case, when using the toggle output pin function?

Yes, initially LED 1 should be set to on and the red RGB led should be set to off before toggling each of them.

Task 2: GPIO Pins as Inputs with Polling

Task 2.1: Identifying the GPIO Configuration

Buttons S1 and S2 are P1.1 and P1.4 respectively. A Pull-Up resistor is used to take inputs of the buttons.

Task 2.2: Implement Button Polling

Can you observe any difference in terms of blinking frequency compared to task_1_3.c? What about the reaction time of the buttons S1 and S2?

The blinking frequency is about the same but the reaction time seems a little slow.

Task 2.3: When should we Poll?

What are the upsides of the implementations in Task 2.2 and Task 2.3?

Both tasks use loops to implement the code however the frequency of task 2.3 is a lot slower than task 2.2.

Can you think of a way to implement polling such that there is no influence on the blinking frequency and the button reaction time is minimized?

Yes, the frequency should stay the same as task 2.2 if the only thing in the for loop is the polling of the buttons.

Task 3: Simple UART output

Task 3.1: Calculating the UART Parameters

Only the eUSCI_A modules can be used for UART communication. The default setting for the SMCLK as clock source is 3MHz.

Task 3.2: Implementing UART Output

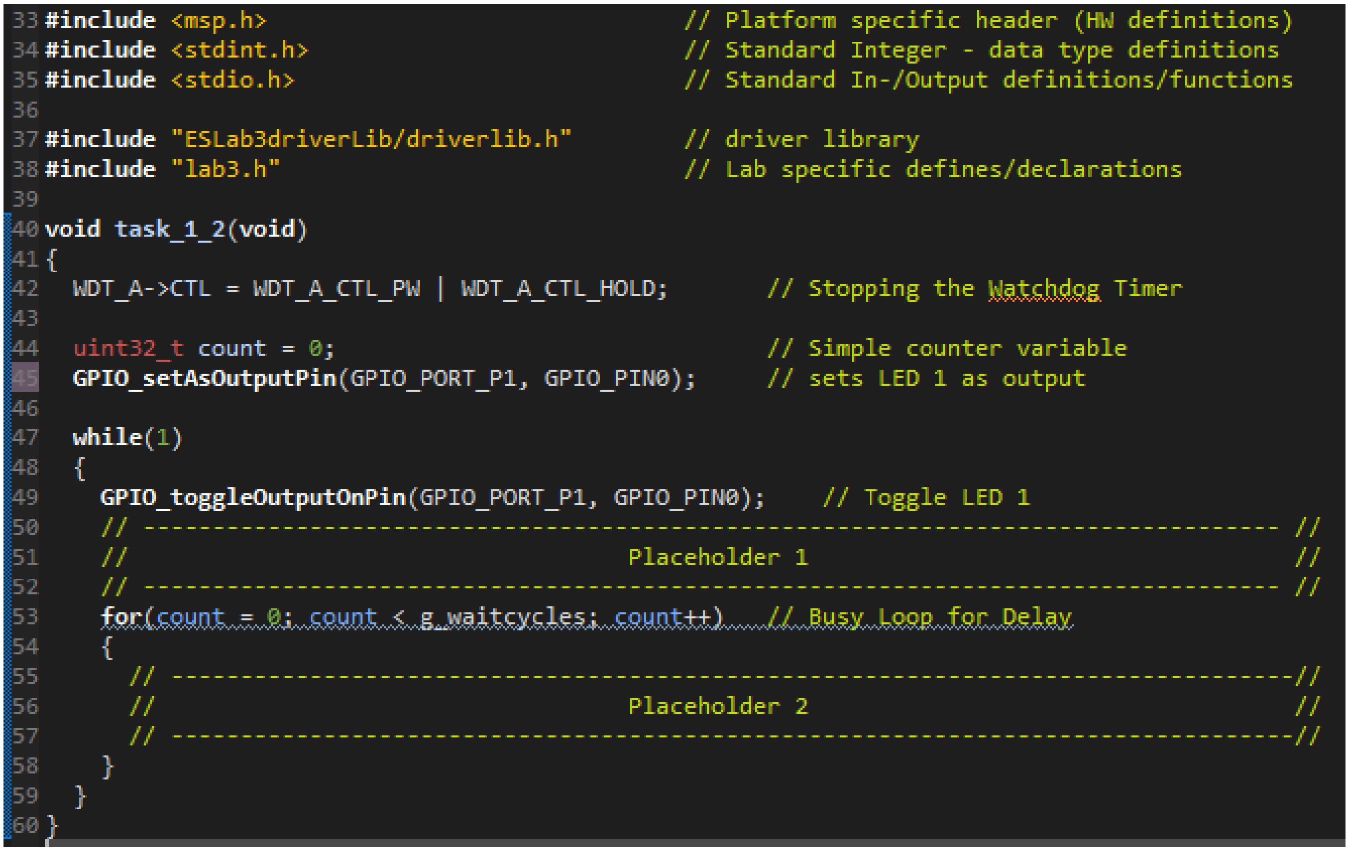
Can you think of a better method to register button presses instead of polling to save resources and avoid duplicated output messages?

Maybe by using the header file.

III Answers and Results

Task 1.2: Using Library Functions instead of Hard-Coded Register Access

Changed Code



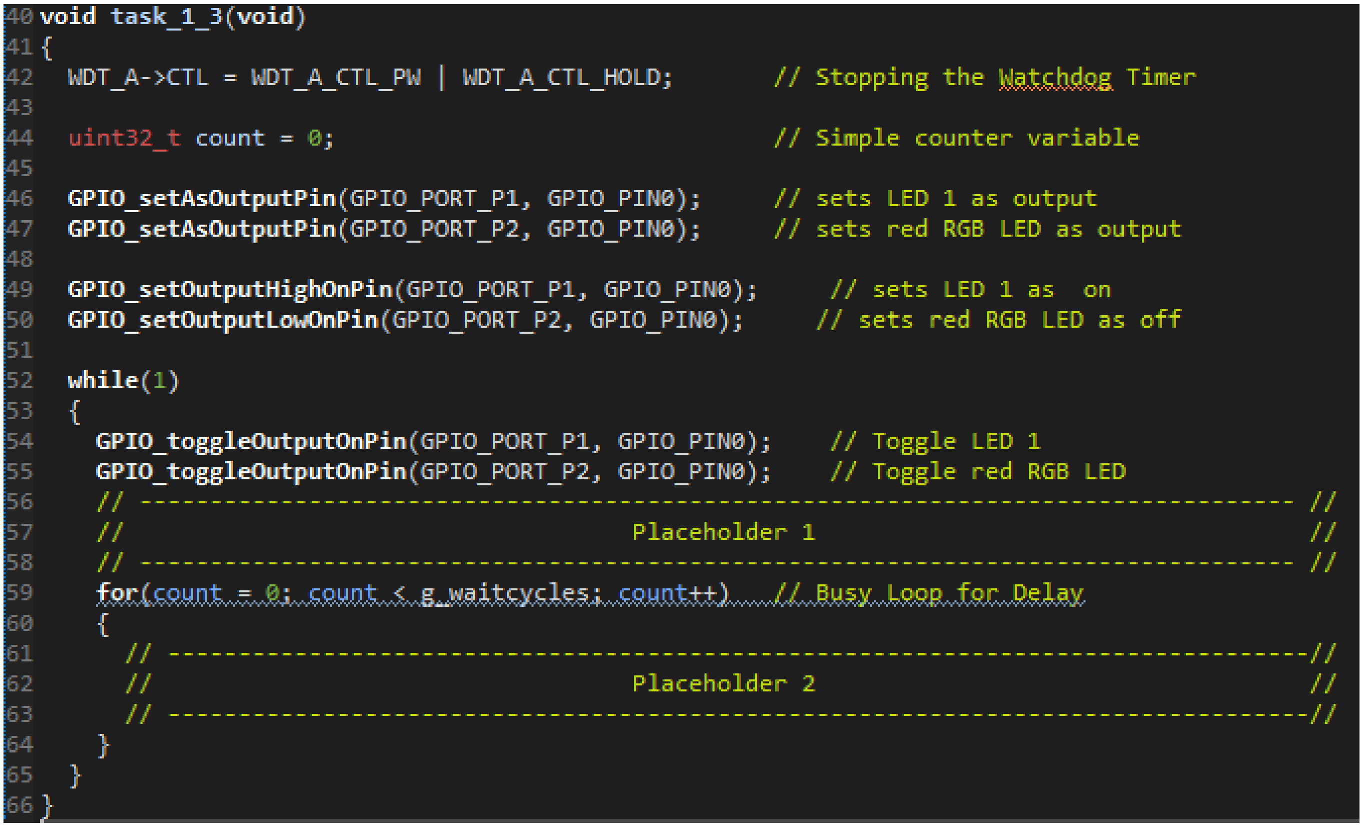
```
#include <msp.h>                               // Platform specific header (HW definitions)
#include <stdint.h>                            // Standard Integer - data type definitions
#include <stdio.h>                             // Standard In-/Output definitions/functions

#include "ESLab3driverLib/driverlib.h"         // driver library
#include "lab3.h"                              // Lab specific defines/declarations

void task_1_2(void)
{
  WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;        // Stopping the Watchdog Timer

  uint32_t count = 0;                                // Simple counter variable
  GPIO_setAsOutputPin(GPIO_PORT_P1, GPIO_PIN0);      // sets LED 1 as output

  while(1)
  {
    GPIO_toggleOutputOnPin(GPIO_PORT_P1, GPIO_PIN0);    // Toggle LED 1
    // ------------------------------------------------------------------------------ //
    //                                  Placeholder 1                                  //
    // ------------------------------------------------------------------------------ //
    for(count = 0; count < g_waitcycles; count++)    // Busy Loop for Delay
    {
      // ---------------------------------------------------------------------------//
      //                                Placeholder 2                               //
      // ---------------------------------------------------------------------------//
    }
  }
}
```

Explanation: LED 1 is set as output and is then toggled in the while loop.

Board blinks LED 1

Task 1.3: Adding a blinking LED

Changed Code



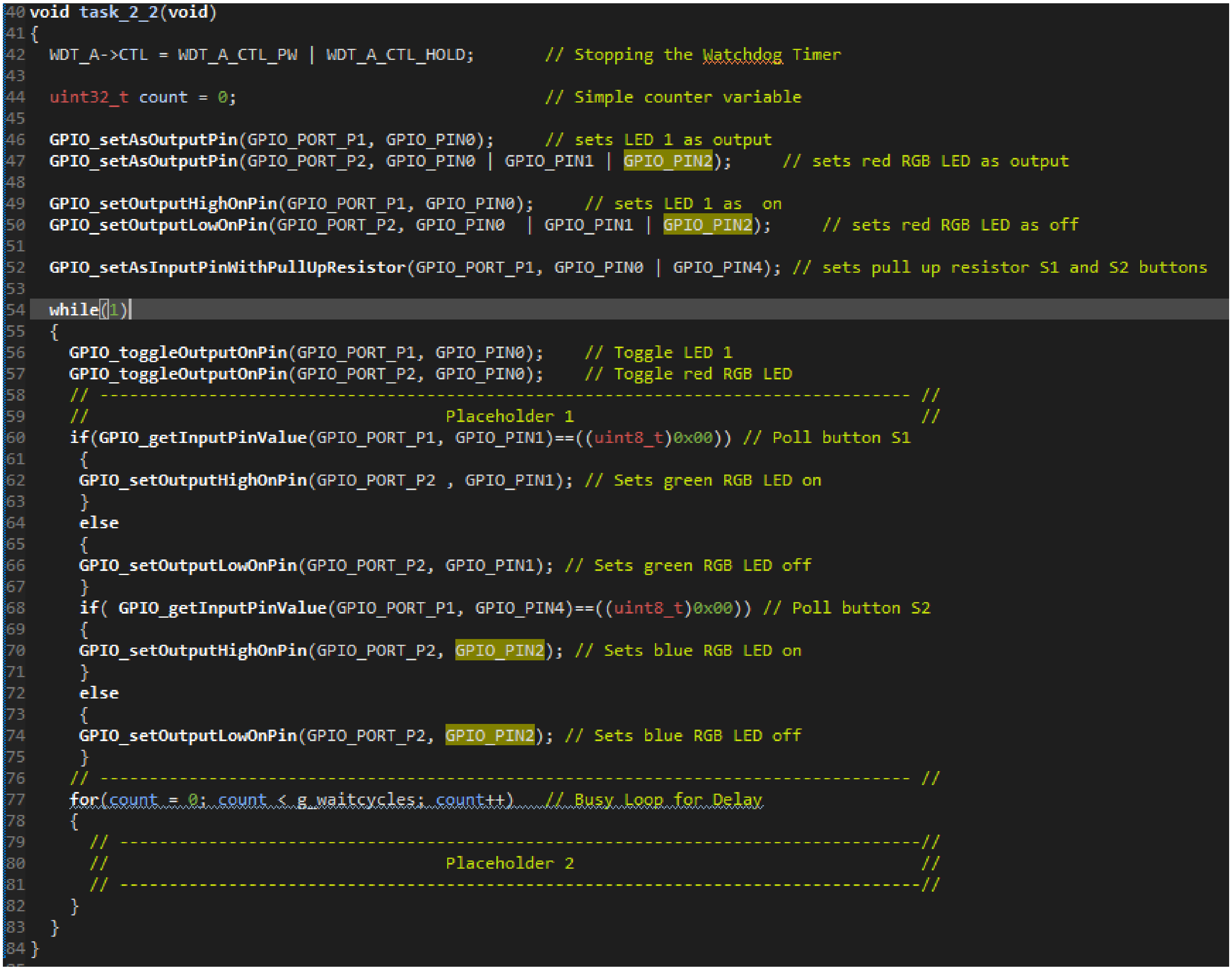
```
void task_1_3(void)
{
  WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;       // Stopping the Watchdog Timer

  uint32_t count = 0;                                // Simple counter variable

  GPIO_setAsOutputPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as output
  GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0);     // sets red RGB LED as output

  GPIO_setOutputHighOnPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as  on
  GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN0);     // sets red RGB LED as off

  while(1)
  {
    GPIO_toggleOutputOnPin(GPIO_PORT_P1, GPIO_PIN0);    // Toggle LED 1
    GPIO_toggleOutputOnPin(GPIO_PORT_P2, GPIO_PIN0);    // Toggle red RGB LED
    // ----------------------------------------------------------------------------------- //
    //                                    Placeholder 1                                   //
    // ----------------------------------------------------------------------------------- //
    for(count = 0; count < g_waitcycles; count++)   // Busy Loop for Delay
    {
      // ---------------------------------------------------------------------------------//
      //                                  Placeholder 2                                   //
      // ---------------------------------------------------------------------------------//
    }
  }
}
```

Explanation: LED 1 and RGB LED are set as outputs with initialization for LED 1 as on and RGB LED as off and each are then toggled in the while loop.

Board blinks red leds alternating between LED 1 and the RGB LED.

Task 2.2: Implement Button Polling

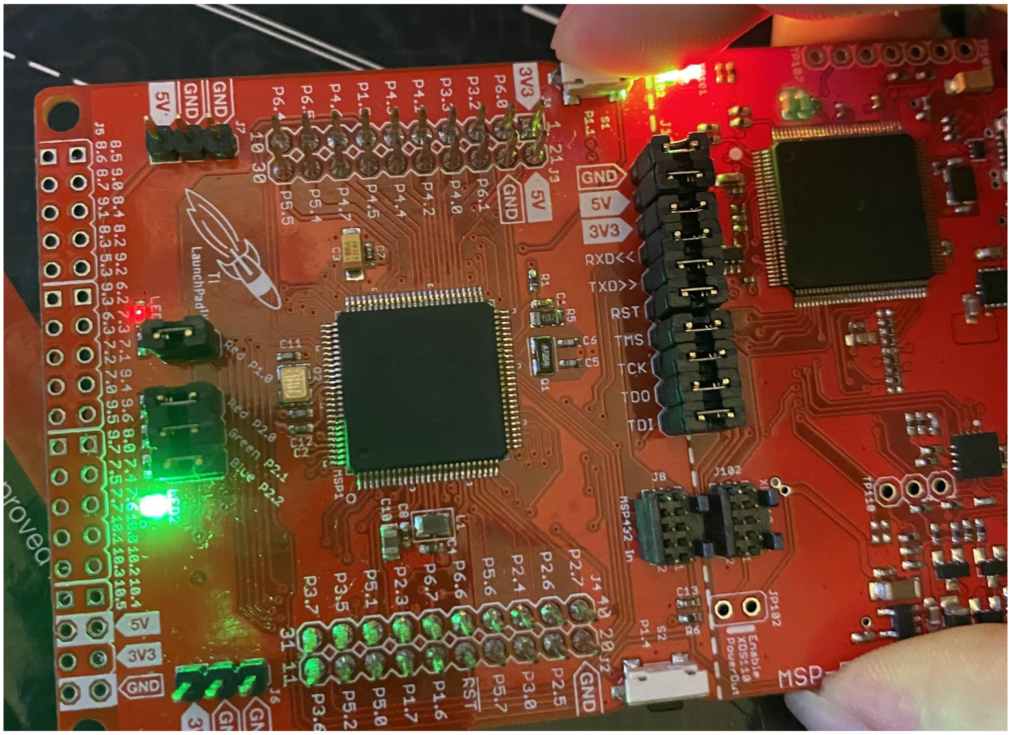
Changed code

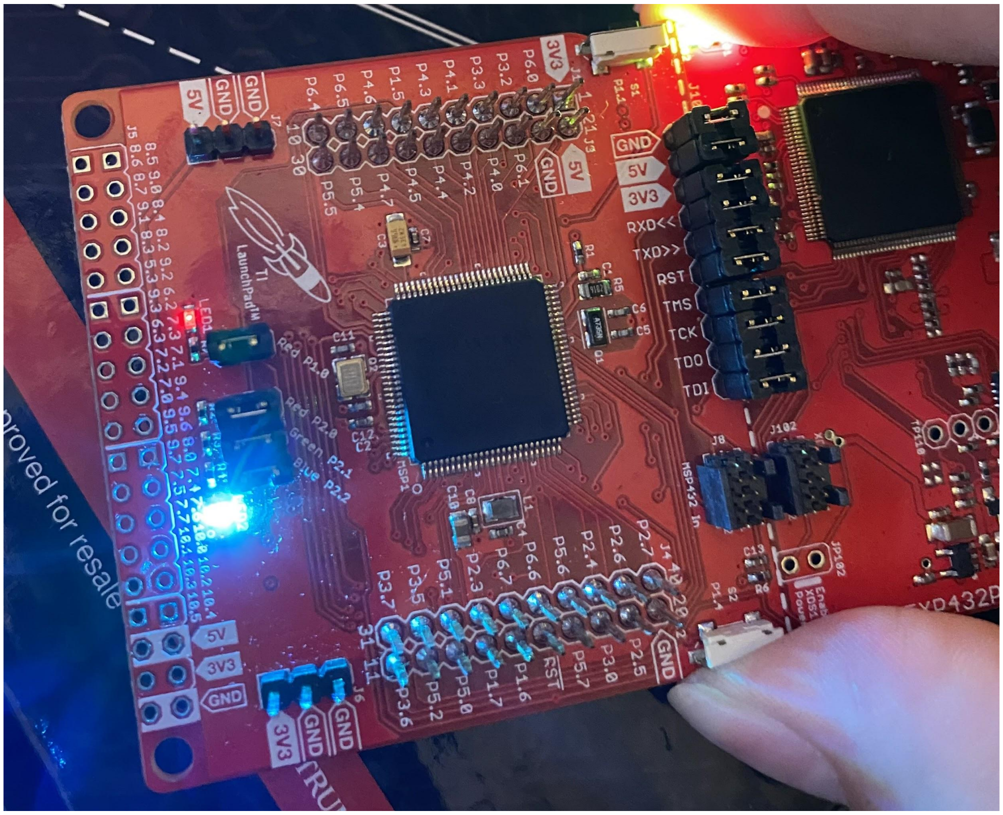


```
void task_2_2(void)
{
  WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;       // Stopping the Watchdog Timer

  uint32_t count = 0;                               // Simple counter variable

  GPIO_setAsOutputPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as output
  GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0 | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED as output

  GPIO_setOutputHighOnPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as  on
  GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN0  | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED as off

  GPIO_setAsInputPinWithPullUpResistor(GPIO_PORT_P1, GPIO_PIN0 | GPIO_PIN4); // sets pull up resistor S1 and S2 buttons

  while(1)
  {
    GPIO_toggleOutputOnPin(GPIO_PORT_P1, GPIO_PIN0);    // Toggle LED 1
    GPIO_toggleOutputOnPin(GPIO_PORT_P2, GPIO_PIN0);    // Toggle red RGB LED
    // ---------------------------------------------------------------------------------- //
    //                                  Placeholder 1                                     //
    if(GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN1)==((uint8_t)0x00)) // Poll button S1
     {
     GPIO_setOutputHighOnPin(GPIO_PORT_P2 , GPIO_PIN1); // Sets green RGB LED on
     }
     else
     {
     GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN1); // Sets green RGB LED off
     }
     if( GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN4)==((uint8_t)0x00)) // Poll button S2
     {
     GPIO_setOutputHighOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED on
     }
     else
     {
     GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED off
     }
    // ---------------------------------------------------------------------------------- //
    for(count = 0; count < g_waitcycles; count++)   // Busy Loop for Delay
    {
      // --------------------------------------------------------------------------------//
      //                                Placeholder 2                                    //
      // --------------------------------------------------------------------------------//
    }
  }
}
```

Explanation: LED 1 and RGB LED are set as outputs with initialization for LED 1 as on and RGB LED as off and each are then toggled in the while loop. Pull up resistors are set to buttons s1 and s2. In the while loop, button s1 toggles green and button s2 toggles blue.

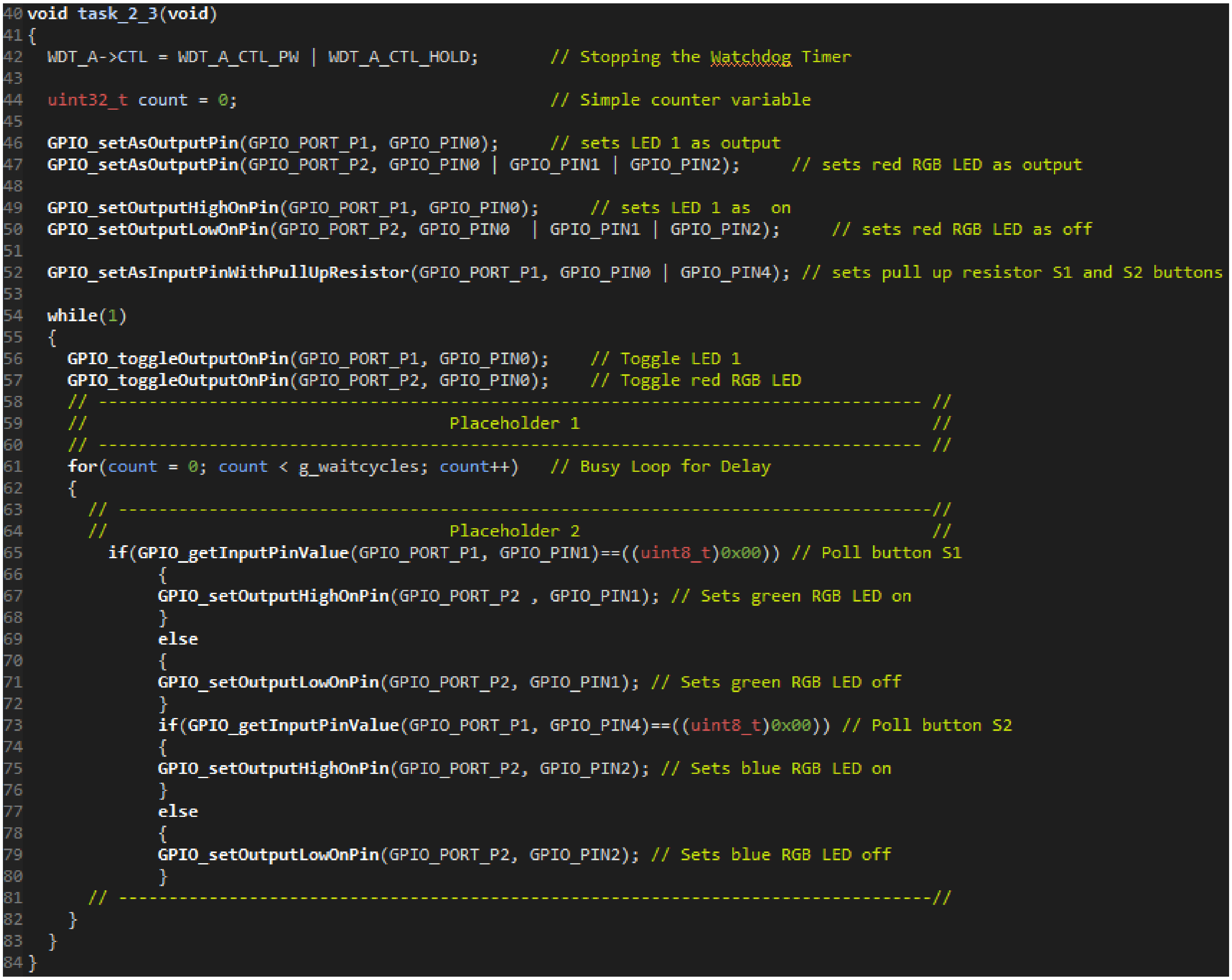
Board


3V3
5V
GND
RXD<<
TXD>>
RST
TMS
TCK
TDO
TDI
LaunchPad
Red P1.0
Red P2.0
Green P2.1
Blue P2.2
MSP432 In
J8
J102
JP102
Enable XDS110 PowerOut
S1
S2
P1.1
P1.4
MSP-


3V3
5V
GND
RXD<<
TXD>>
RST
TMS
TCK
TDO
TDI
LaunchPad
Red P1.0
Red P2.0
Green P2.1
Blue P2.2
MSP432 In
J8
J102
JP102
S1
S2
P1.1
P1.4
XP432P
proved for resale

Task 2.3: When should we Poll?

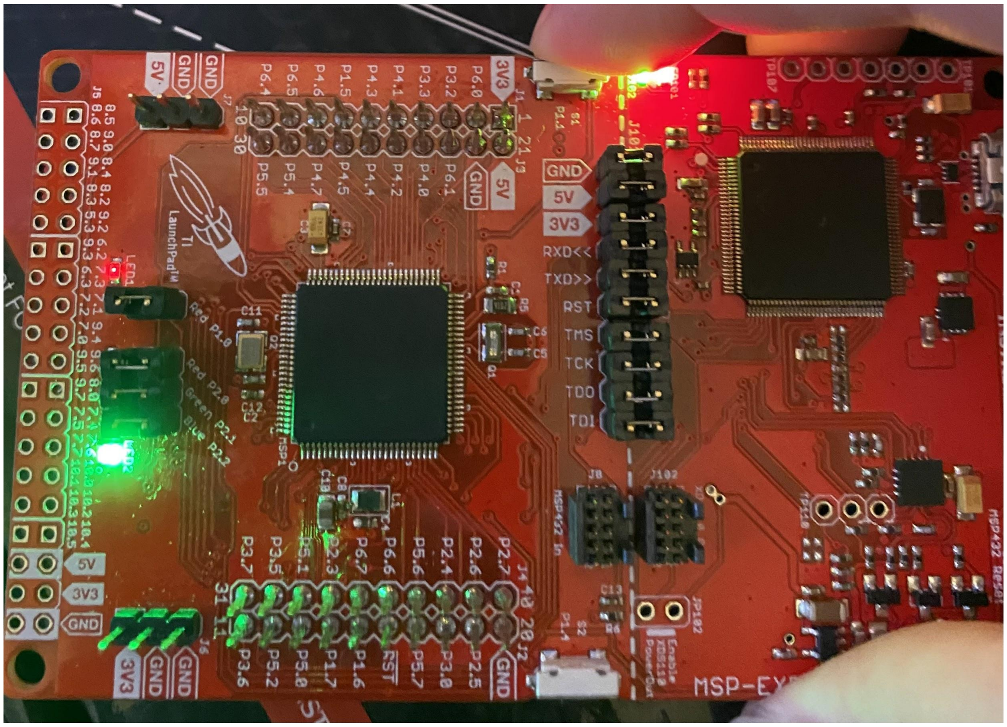
Changed code

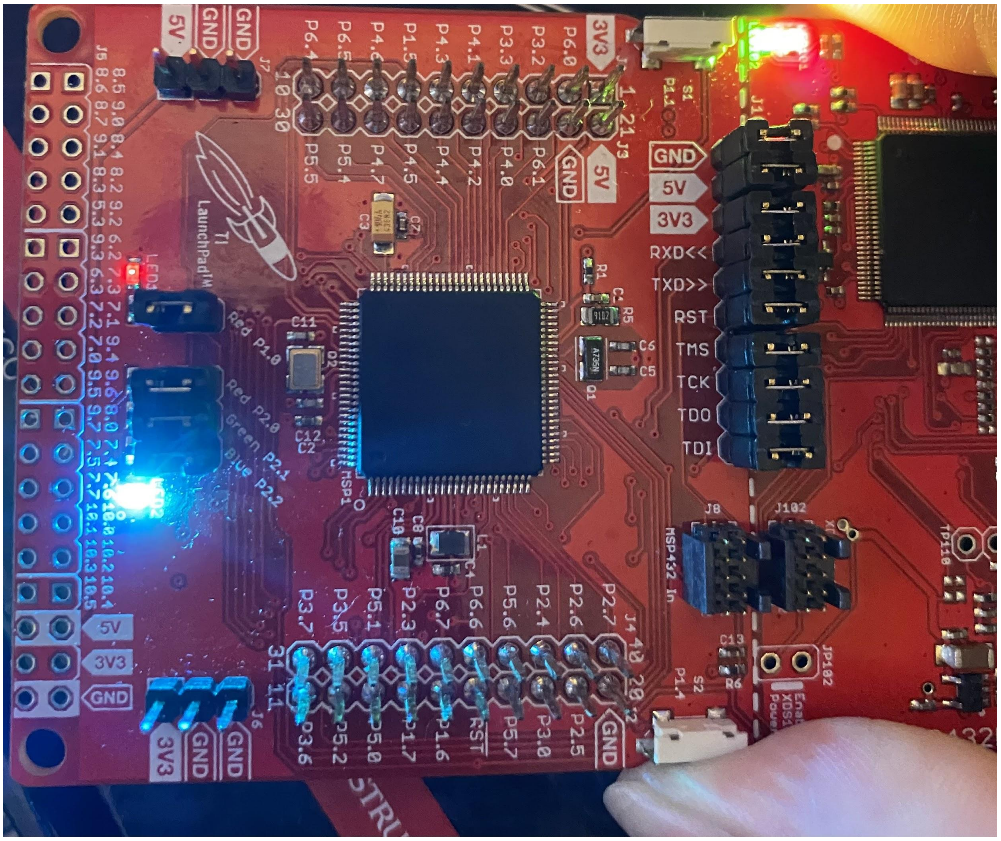


```
void task_2_3(void)
{
  WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;       // Stopping the Watchdog Timer

  uint32_t count = 0;                               // Simple counter variable

  GPIO_setAsOutputPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as output
  GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0 | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED as output

  GPIO_setOutputHighOnPin(GPIO_PORT_P1, GPIO_PIN0);     // sets LED 1 as  on
  GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN0  | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED as off

  GPIO_setAsInputPinWithPullUpResistor(GPIO_PORT_P1, GPIO_PIN0 | GPIO_PIN4); // sets pull up resistor S1 and S2 buttons

  while(1)
  {
    GPIO_toggleOutputOnPin(GPIO_PORT_P1, GPIO_PIN0);    // Toggle LED 1
    GPIO_toggleOutputOnPin(GPIO_PORT_P2, GPIO_PIN0);    // Toggle red RGB LED
    // ---------------------------------------------------------------------------------- //
    //                                    Placeholder 1                                   //
    // ---------------------------------------------------------------------------------- //
    for(count = 0; count < g_waitcycles; count++)   // Busy Loop for Delay
    {
      // --------------------------------------------------------------------------------//
      //                                  Placeholder 2                                   //
        if(GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN1)==((uint8_t)0x00)) // Poll button S1
             {
             GPIO_setOutputHighOnPin(GPIO_PORT_P2 , GPIO_PIN1); // Sets green RGB LED on
             }
             else
             {
             GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN1); // Sets green RGB LED off
             }
             if(GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN4)==((uint8_t)0x00)) // Poll button S2
             {
             GPIO_setOutputHighOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED on
             }
             else
             {
             GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED off
             }
      // --------------------------------------------------------------------------------//
    }
  }
}
```

Explanation: LED 1 and RGB LED are set as outputs with initialization for LED 1 as on and RGB LED as off and each are then toggled in the while loop. Pull up resistors are set to buttons s1 and s2. In the for loop, button s1 toggles green and button s2 toggles blue.

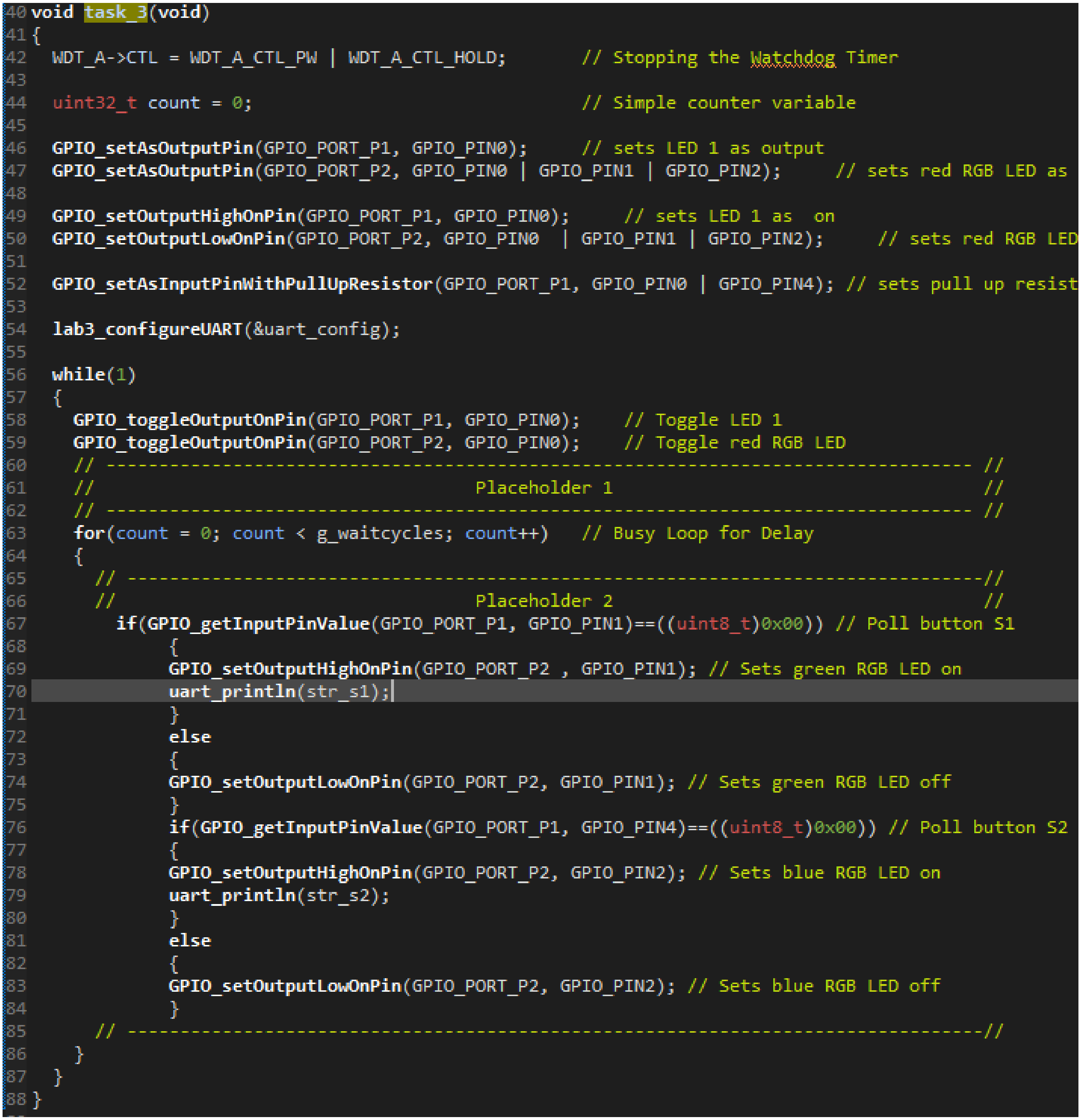
Board


5V
GND
GND
3V3
GND
5V
3V3
RXD<<
TXD>>
RST
TMS
TCK
TDO
TDI
TI
LaunchPad™
Red P1.0
Red P2.0
Green P2.1
Blue P2.2
J8
J102
MSP432 In
MSP-EX


5V
GND
GND
3V3
GND
5V
3V3
RXD<<
TXD>>
RST
TMS
TCK
TDO
TDI
TI
LaunchPad™
Red P1.0
Red P2.0
Green P2.1
Blue P2.2
J8
J102
MSP432 In

Task 3.2: Implementing UART Output

Changed code



```
void task_3(void)
{
  WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;       // Stopping the Watchdog Timer

  uint32_t count = 0;                              // Simple counter variable

  GPIO_setAsOutputPin(GPIO_PORT_P1, GPIO_PIN0);    // sets LED 1 as output
  GPIO_setAsOutputPin(GPIO_PORT_P2, GPIO_PIN0 | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED as

  GPIO_setOutputHighOnPin(GPIO_PORT_P1, GPIO_PIN0);    // sets LED 1 as  on
  GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN0  | GPIO_PIN1 | GPIO_PIN2);     // sets red RGB LED

  GPIO_setAsInputPinWithPullUpResistor(GPIO_PORT_P1, GPIO_PIN0 | GPIO_PIN4); // sets pull up resist

  lab3_configureUART(&uart_config);

  while(1)
  {
    GPIO_toggleOutputOnPin(GPIO_PORT_P1, GPIO_PIN0);    // Toggle LED 1
    GPIO_toggleOutputOnPin(GPIO_PORT_P2, GPIO_PIN0);    // Toggle red RGB LED
    // --------------------------------------------------------------------------------- //
    //                                    Placeholder 1                                   //
    // --------------------------------------------------------------------------------- //
    for(count = 0; count < g_waitcycles; count++)   // Busy Loop for Delay
    {
      // --------------------------------------------------------------------------------//
      //                                  Placeholder 2                                   //
        if(GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN1)==((uint8_t)0x00)) // Poll button S1
             {
             GPIO_setOutputHighOnPin(GPIO_PORT_P2 , GPIO_PIN1); // Sets green RGB LED on
             uart_println(str_s1);
             }
             else
             {
             GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN1); // Sets green RGB LED off
             }
             if(GPIO_getInputPinValue(GPIO_PORT_P1, GPIO_PIN4)==((uint8_t)0x00)) // Poll button S2
             {
             GPIO_setOutputHighOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED on
             uart_println(str_s2);
             }
             else
             {
             GPIO_setOutputLowOnPin(GPIO_PORT_P2, GPIO_PIN2); // Sets blue RGB LED off
             }
      // --------------------------------------------------------------------------------//
    }
  }
}
```

IV Problem

I had the same issue as last time, I kept having to find which comm the board was. I solved this issue by finding the comm number in the device manager on windows. One other issue I had was on task 3, I changed the code how the question asked me to but I'm not certain I understood what was supposed to happen.

V What have you learned

During this lab, I learned about setting pins and ports as outputs and how to use polling. What we've been learning in class has helped see how this lab works.