Lab 2 Report

Kerim Turak

Introduction

To complete this lab assessment I follow these steps;

- Writing the codes specified in the lab assessment
- Thinking about logical part of the algorithm
- Finalizing the task by running the code

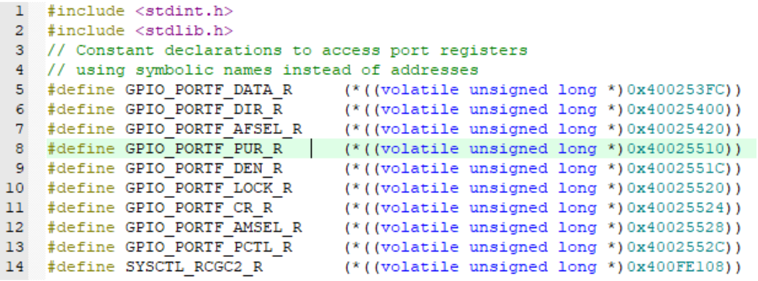
Youtube link: https://youtu.be/tL8OI0WLuk8

My Code File: https://drive.google.com/drive/folders/1in8fHzKSvaSW8t448_KWpFyIGBJeMNdN?usp=sharing

a) Writing the codes specified in the lab assessment

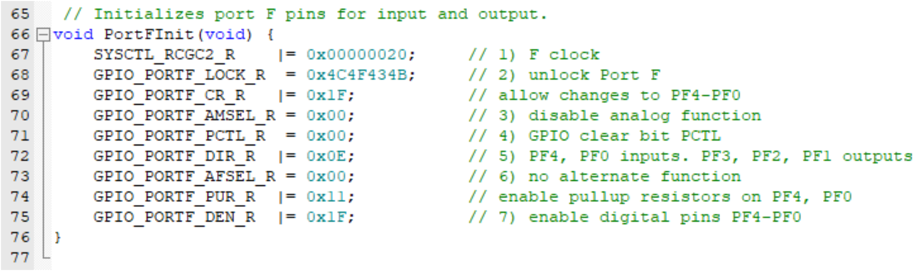
In this part I write necessary frame work which given by lab teaching assistant.

Defining symbolic variables;



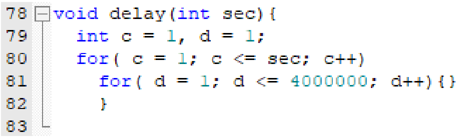
```c
#include <stdint.h>
#include <stdlib.h>
// Constant declarations to access port registers
// using symbolic names instead of addresses
#define GPIO_PORTF_DATA_R      (*((volatile unsigned long *)0x400253FC))
#define GPIO_PORTF_DIR_R       (*((volatile unsigned long *)0x40025400))
#define GPIO_PORTF_AFSEL_R     (*((volatile unsigned long *)0x40025420))
#define GPIO_PORTF_PUR_R       (*((volatile unsigned long *)0x40025510))
#define GPIO_PORTF_DEN_R       (*((volatile unsigned long *)0x4002551C))
#define GPIO_PORTF_LOCK_R      (*((volatile unsigned long *)0x40025520))
#define GPIO_PORTF_CR_R        (*((volatile unsigned long *)0x40025524))
#define GPIO_PORTF_AMSEL_R     (*((volatile unsigned long *)0x40025528))
#define GPIO_PORTF_PCTL_R      (*((volatile unsigned long *)0x4002552C))
#define SYSCTL_RCGC2_R         (*((volatile unsigned long *)0x400FE108))
```

To make simple and easier the main function I set all the symbolic variables in a function named **PortFInit,** in this way, there will be less code in the main function and it will gain simplicity.



```c
 // Initializes port F pins for input and output.
void PortFInit(void) {
    SYSCTL_RCGC2_R     |= 0x00000020;     // 1) F clock
    GPIO_PORTF_LOCK_R  = 0x4C4F434B;      // 2) unlock Port F
    GPIO_PORTF_CR_R    |= 0x1F;           // allow changes to PF4-PF0
    GPIO_PORTF_AMSEL_R = 0x00;            // 3) disable analog function
    GPIO_PORTF_PCTL_R  = 0x00;            // 4) GPIO clear bit PCTL
    GPIO_PORTF_DIR_R   |= 0x0E;           // 5) PF4, PF0 inputs. PF3, PF2, PF1 outputs
    GPIO_PORTF_AFSEL_R = 0x00;            // 6) no alternate function
    GPIO_PORTF_PUR_R   |= 0x11;           // enable pullup resistors on PF4, PF0
    GPIO_PORTF_DEN_R   |= 0x1F;           // 7) enable digital pins PF4-PF0
}
```

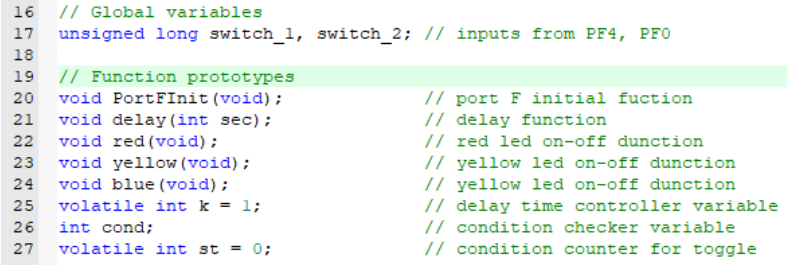
Then I add **delay** function.



```c
void delay(int sec){
  int c = 1, d = 1;
  for( c = 1; c <= sec; c++)
    for( d = 1; d <= 4000000; d++){}
    }
```

b) Thinking about logical part of the algorithm

In this part, I manage the problem and build the algorithm, for build algorithm and gain more simplicity I write fucntion to all operation in assessment.



```
// Global variables
unsigned long switch_1, switch_2; // inputs from PF4, PF0

// Function prototypes
void PortFInit(void);             // port F initial fuction
void delay(int sec);              // delay function
void red(void);                   // red led on-off dunction
void yellow(void);                // yellow led on-off dunction
void blue(void);                  // yellow led on-off dunction
volatile int k = 1;               // delay time controller variable
int cond;                         // condition checker variable
volatile int st = 0;              // condition counter for toggle
```

In the picture above, variable **switch1** and **switch1** store the coming data from **sw1** and **sw2** button. **PortFInit** is initialized the necessary launchpad operation that is;

- F clock setting
- unlock Port F to activate PF0
- allow changes to PF4-PF0 by configuring the GPIO_PORTF_CR_R register
- disable analog function
- GPIO clear bit PCTL
- PF4, PF0 inputs. PF3, PF2, PF1 outputs
- no alternate function
- enable pullup resistors on PF4, PF0
- enable digital pins PF4-PF0

Some of this are default but I set it up anyway.

Functions **red**, **yellow**, **blue** controls the case of on-off about leds.

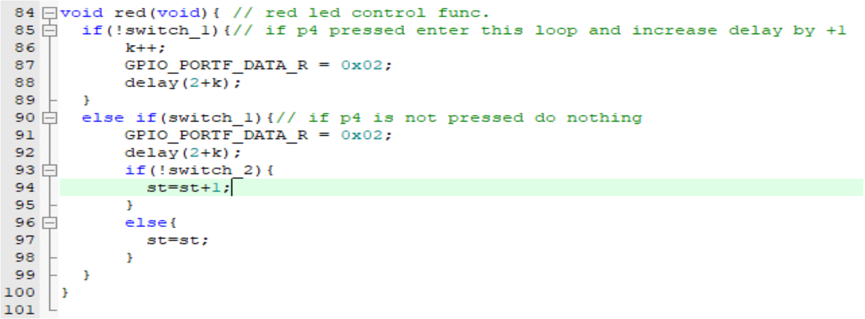
K is just any variable to increase the delay.

St is variable which perform the toggle operation.

c)Finalizing the task by running the code

In this section, I write the logical function which are defined in the part of **b.**

All the led controls function do same operation as following image;

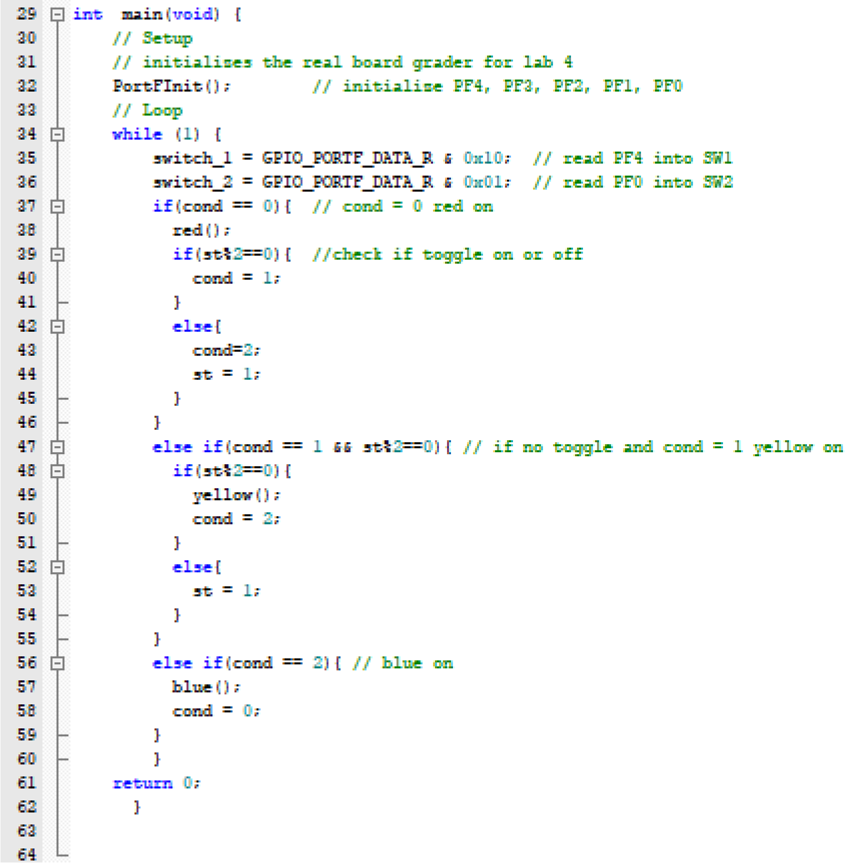


```
void red(void){ // red led control func.
  if(!switch_1){// if p4 pressed enter this loop and increase delay by +1
      k++;
      GPIO_PORTF_DATA_R = 0x02;
      delay(2+k);
  }
  else if(switch_1){// if p4 is not pressed do nothing
      GPIO_PORTF_DATA_R = 0x02;
      delay(2+k);
      if(!switch_2){
        st=st+1;
      }
      else{
        st=st;
      }
  }
}
```

Function controls if switch 1 pressed, then increase the delay, if not continue.

Second if-else block checks the switch2 if switch2 is pressed, then increase the st by +1.

Finally, in main code block code execute other logical process as following;



```
int  main(void) {
    // Setup
    // initializes the real board grader for lab 4
    PortFInit();        // initialize PF4, PF3, PF2, PF1, PF0
    // Loop
    while (1) {
        switch_1 = GPIO_PORTF_DATA_R & 0x10;  // read PF4 into SW1
        switch_2 = GPIO_PORTF_DATA_R & 0x01;  // read PF0 into SW2
        if(cond == 0){  // cond = 0 red on
          red();
          if(st%2==0){  //check if toggle on or off
            cond = 1;
          }
          else{
            cond=2;
            st = 1;
          }
        }
        else if(cond == 1 && st%2==0){ // if no toggle and cond = 1 yellow on
          if(st%2==0){
            yellow();
            cond = 2;
          }
          else{
            st = 1;
          }
        }
        else if(cond == 2){ // blue on
          blue();
          cond = 0;
        }
        }
    return 0;
      }
```

It is so clear that, switchs are get their value in while loop to read data continuesly. **st** is initially 0 if switch2 is presse done time then st will increase by +1 in the led functions and if its mod. İs not equal zero this mean perform toggle operation and pass the yellow led.