Lab 3 Report

Kerim Turak

Introduction

- **Laboratory Work 1**
- **Laboratory Work 2**
- **Laboratory Work 3**
- **Laboratory Work 4**

Youtube link: https://youtu.be/_vCwuzEwd1Q

Laboratory Work 1-2-3

Actually, I wrote all of the lab work as separate codes, but since we combine labwork 1 and 2 in lab work 3, I will explain lab work 1 and 2 by explaining only lab work 3 so that the lab report is not too long.

On purpose of doing led ligthing operation , I use my code that I use in lab 2 work. I add needed port addresses and library as following.

```
#include <stdint.h>
#include <stdlib.h>
#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))
```

I manage the problem of build the algorithm, for building algorithm and gaining more simplicity I write fucntion to all operation in assessment.

```
// Function prototypes
void PortEInit(void);  // port E initial fuction
void delay(int sec);   // delay function
void red(void);        // red led on-off dunction
volatile int k = 1;    // delay time controller variable
int cond;              // condition checker variable
```

To make simpler and easier the structure of 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.

```
65    // Initializes port F pins for input and output.
66  void PortFInit(void) {
67      SYSCTL_RCGC2_R    |= 0x00000020;     // 1) F clock
68      GPIO_PORTF_LOCK_R  = 0x4C4F434B;     // 2) unlock Port F
69      GPIO_PORTF_CR_R   |= 0x1F;           // allow changes to PF4-PF0
70      GPIO_PORTF_AMSEL_R = 0x00;           // 3) disable analog function
71      GPIO_PORTF_PCTL_R  = 0x00;           // 4) GPIO clear bit PCTL
72      GPIO_PORTF_DIR_R  |= 0x0E;           // 5) PF4, PF0 inputs. PF3, PF2, PF1 outputs
73      GPIO_PORTF_AFSEL_R = 0x00;           // 6) no alternate function
74      GPIO_PORTF_PUR_R  |= 0x11;           // enable pullup resistors on PF4, PF0
75      GPIO_PORTF_DEN_R  |= 0x1F;           // 7) enable digital pins PF4-PF0
76  }
77
```

Functions **red** controls the case of on-off about red led.

```
void red(void){
  if(!switch_1){
      GPIO_PORTE_DATA_R = 0x02;
      delay(20000);
      GPIO_PORTE_DATA_R = 0x00;
      delay(80000);
  }
```

Then all this function are called in main funtion to complete the lab work1-2-3.

```
int  main(void) {
    PortEInit();
    while (1) {
        switch_1 = GPIO_PORTE_DATA_R & 0x10;
        if(cond == 0){  // cond = 0 red on
          red();
        }
    }
    return 0;
}
```

Lastly, our main code block that perform the main work breathing and other process in red function. Delay time equal 0,00001 second for get very sensitive clock to breathing process.

```
int time = 5000;
int op = 5000;
int cl=0;
int ctrl = 0;
void red(void){
  if(!switch_1){
      GPIO_PORTF_DATA_R = 0x02;
      delay(25000);
      GPIO_PORTF_DATA_R = 0x00;
      delay(100000);
  }
```

Time variable holds the LED on time at startup, op is open time at beginning of while loop, cl is closed time at beginning while loop. And ctrl is the variable that declare for setting sensitivy of the flashing. After first if block controls the if switch on then turn on led %20 duty cycle after turn off %80 duty cycle.

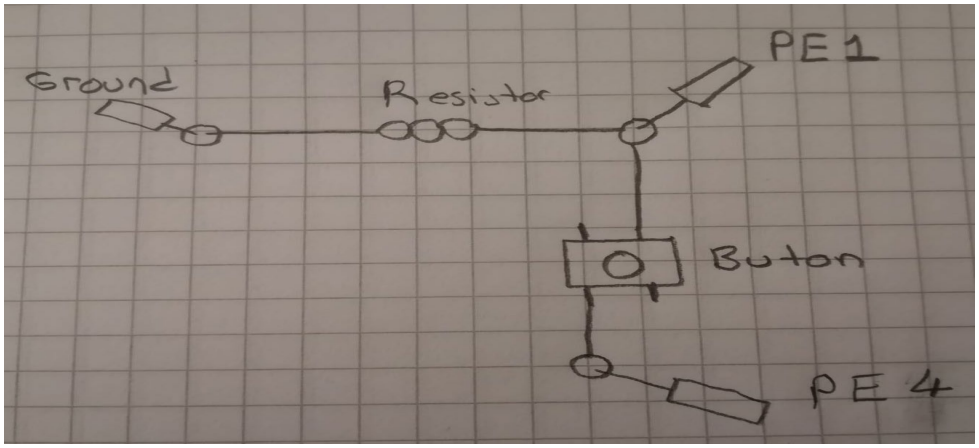
```
  else{
    while(switch_1){
      GPIO_PORTF_DATA_R = 0x02;
      delay(op);
      if(ctrl%2 ==0){
      op = op -120;
      }
      GPIO_PORTF_DATA_R = 0x00;
      delay(cl);
      if(ctrl%2 ==0){
      cl = cl+120;
      }
      if(cl>=time){
        delay(200000);
        break;
      }
      ctrl++;
    }
```

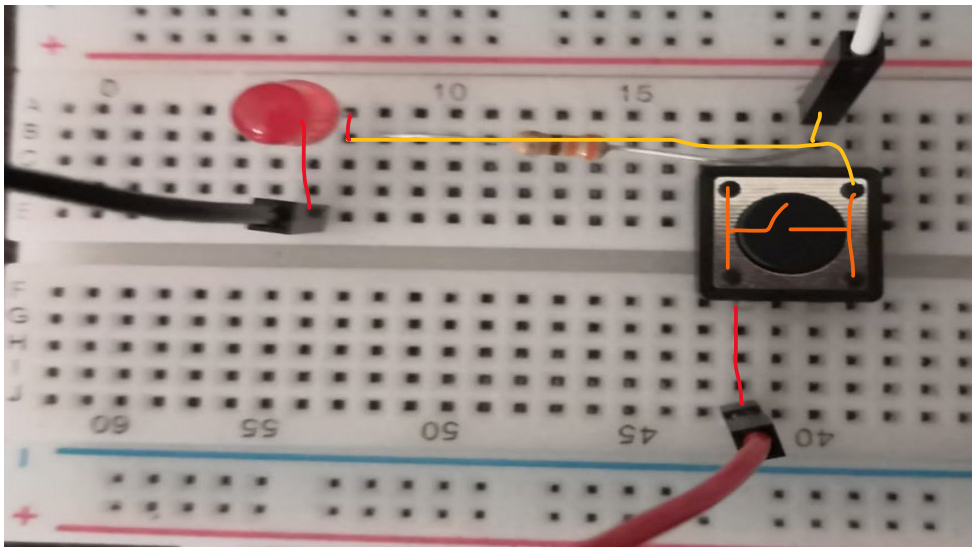
In else block, that means switch is off, first led on 5000*delaytime after led off 0* delaytime. After first loop op decrease by 5 and cl increase by 5. ctrl% 2 makes this increase and decrease every two cycles. Lastly cl>=time check if cl equal begining value of op then break the loop. This first while performs turning off process. The second cycle performs the turning on process by performing the reverse of the operations in the first cycle.

```
while(switch_1){
  GPIO_PORTF_DATA_R = 0x02;
  delay(op);
  if(ctrl%2 ==0){
  op = op +120;
  }
  GPIO_PORTF_DATA_R = 0x00;
  delay(cl);
  if(ctrl%2 ==0){
  cl = cl-120;
  }
  if(op>=time){
    break;
  }
  ctrl++;
}
```

Laboratory Work 4

In this part port E is configured to perform the task on breadbord. I only change port addresses and by deleting lock and cr register all port become ready for this operation after I build the circuit on breadbord. I draw the circuit diagram below then I built the circuit accordingly. I expain all the code algorithm in lab video.





I use same codes for port E except GPIO_LOCK and GPIO_CR because port E has no locked pin. For lab work4. And I explain the code algorithm in video.