**HACETTEPE UNIVERSITY COMPUTER ENGINEERING DEPARTMENT**

**Student Information:** Merve Müge Deliktaş - 21526896
Onur Cankur – 21526791

**Course Information:** BBM 434 - Embedded Systems Laboratory
Spring – 2019

**Report Information:** Lab-02 Experiment Report

**Short Brief of Lab-02 and Function Explanations**

In this lab, we made PF0(SW2) and PF4(SW1) inputs and PF1, PF2 and PF3(LEDs) outputs. Then, we turn the LEDs on in red-blue-green sequence. When SW1 is pressed, SOS signal is flashed. Finally, to get some bonus points, we flashed the sky blue red when SW2 is pressed.

```
int main(void){
  PortF_Init();  // make PF1 out (PF1 built-in LED)
  while(1){
    SW1 = GPIO_PORTF_DATA_R&0x10;
    SW2 = GPIO_PORTF_DATA_R&0x01;
    if(SW1 == 0x10){
      RedBlueGreen();
    }
    if(SW2 == 0x00){
      SW2_Pressed();
    }
    if(SW1 == 0x00) {
      FlashSOS();
    }
  }
}
```

*Figure 1-main()*

In main function, we assigned SW1 and SW2 and checked if they are pressed or not and called the functions according to which switch is pressed.

```
void RedBlueGreen(void){
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.5); // Red LED is on
  GPIO_PORTF_DATA_R &= ~0x02;             // Red LED is off
  GPIO_PORTF_DATA_R |= 0x04;  Delay(1);   // Blue LED is on
  GPIO_PORTF_DATA_R &= ~0x04;             // Blue LED is off
  GPIO_PORTF_DATA_R |= 0x08;  Delay(1.5); // Green LED is on
  GPIO_PORTF_DATA_R &= ~0x08;             // Green LED is off
  if(pressed_1 == 1){
    FlashSOS();
  }
  if(pressed_2 == 1){
    SW2_Pressed();
  }
}
```

*Figure 2-RedBlueGreen()*

In RedBlueGreen() function, we turned on and turned off the LEDs in the sequence of red, green, blue. We checked if any switch is pressed or not in the Delay() function but it will be explained later under the “Measuring Time” title.

```
void FlashSOS(void){
  pressed_1 = 0;
  //S
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  //O
  GPIO_PORTF_DATA_R |= 0x0E;  Delay(0.5);
  GPIO_PORTF_DATA_R &= ~0x0E; Delay(0.5);
  GPIO_PORTF_DATA_R |= 0x0E;  Delay(0.5);
  GPIO_PORTF_DATA_R &= ~0x0E; Delay(0.5);
  GPIO_PORTF_DATA_R |= 0x0E;  Delay(0.5);
  GPIO_PORTF_DATA_R &= ~0x0E; Delay(0.5);
  //S
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  GPIO_PORTF_DATA_R |= 0x02;  Delay(0.25);
  GPIO_PORTF_DATA_R &= ~0x02; Delay(0.25);
  if(pressed_2 == 1){
    SW2_Pressed();
  }
  if(pressed_1){
    FlashSOS();
  }
}
```

*Figure 3-FlashSOS()*

FlashSOS() function simply flashes SOS signal using red and white LEDs.

Some details and pictures about every part of the lab and explanations about measuring time, Delay function, bonus part and using Logic Analyzer is given in the next sections.

**Part 1) Run the lab in the simulator**

After running the program on Debug mode, we opened simulator from Peripherals.

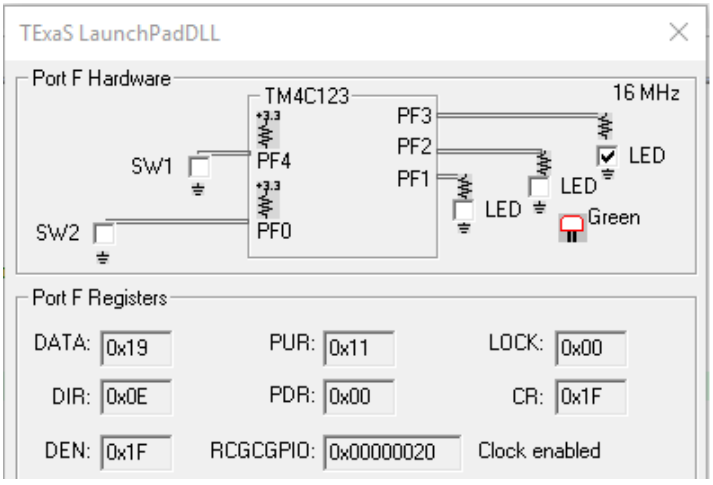


*Figure 4 - RedBlueGreen simulator*

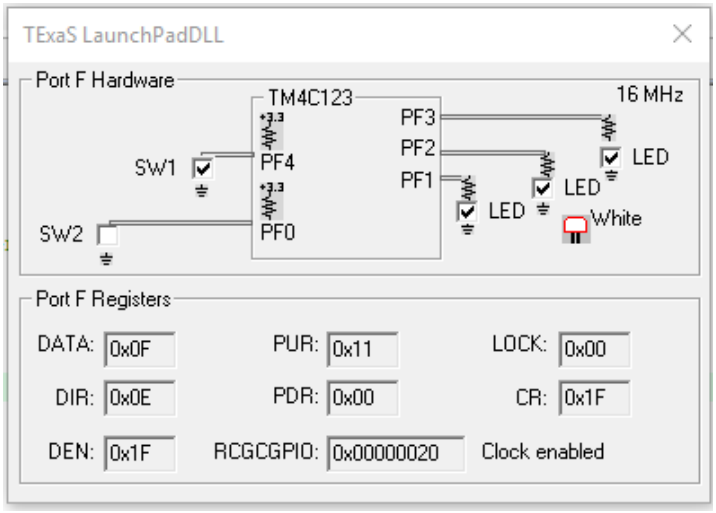


*Figure 5 - SOS simulator*

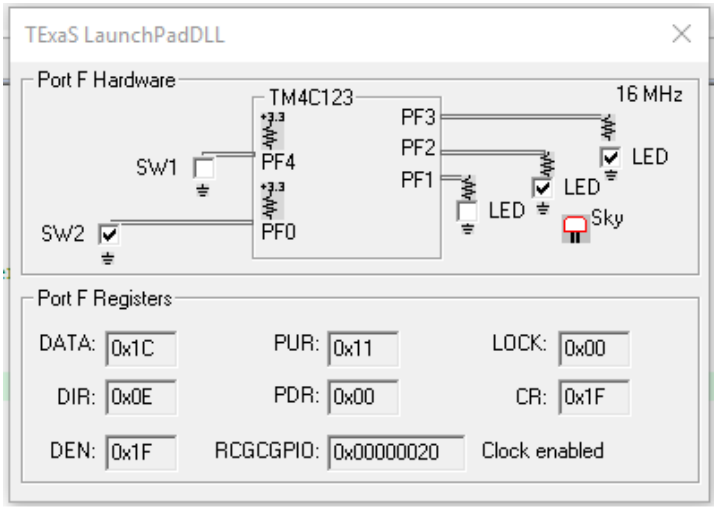


*Figure 6 - SW2 simulator*

Figure 4 shows a part of what happens when no switch is pressed. Figure 5 shows what happens when SW1 is pressed in a certain time. Figure 6 shows what happens when SW2 is pressed in a certain time.

**Part 2) Run the lab on the LaunchPad**

You can see some pictures of the board while code is running.

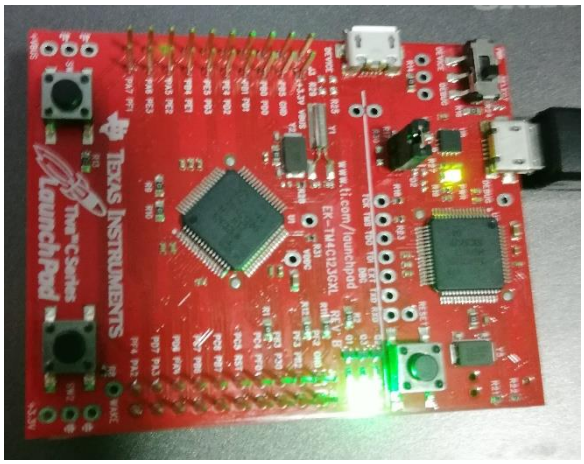


*Figure 7 - RedBlueGreen on board*


*Figure 8 - SOS on board*

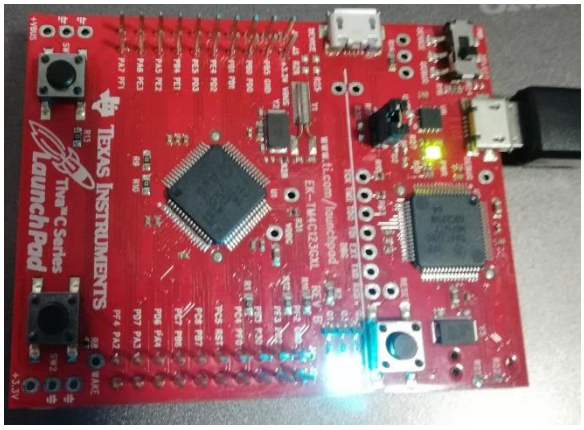


*Figure 9 - SW2 on board*

As in the previous section, Figure 7,8, and 9 show what happened when switches are pressed in certain times.

## Measuring Time

I will explain measuring time in this part. Before measuring the time, you can see our Delay() function below.

```
void Delay(double sec){
  unsigned long volatile time;
  time = 613704*sec;  // When sec==1, it represents 1 sec
  while(time){
    time--;
    SW1 = GPIO_PORTF_DATA_R&0x10;
    SW2 = GPIO_PORTF_DATA_R&0x01;
    if(SW1 == 0x00){
      pressed_1 = 1;
    }
    if(SW2 == 0x00){
      pressed_2 = 1;
    }
  }
}
```

*Figure 10 - Delay function*

To be able to check if a switch is pressed or not while the code is running, we decided to add these check statements in the Delay function. As you can see from the Figure 10, if SW1 is pressed it assigns 1 to pressed_1 variable and if SW2 is pressed, it assigns 1 to pressed_2 variable. The other important thing in here is the line **"time = 613704*sec".** It means that when we multiply 613704 with 1, it should be 1 second. However, when we check it using Logic Analyzer, we can see that it is not 1 sec now.

From the Logic Analyzer, we can see that every time when LED comes to the red, the signal is 1 and as you can see from the Figure 2, we have three Delay() function which should delay for 3 seconds at total. However, it is seen on the Logic Analyzer, it lasts for 4.96 seconds. After finding this value, using simple math, we can calculate what should be the time variable. Approximately, if 3*613704 is equal to 4.96 seconds, 371192 is equal to 1 second. Therefore, it should be **time = 371192*sec**.

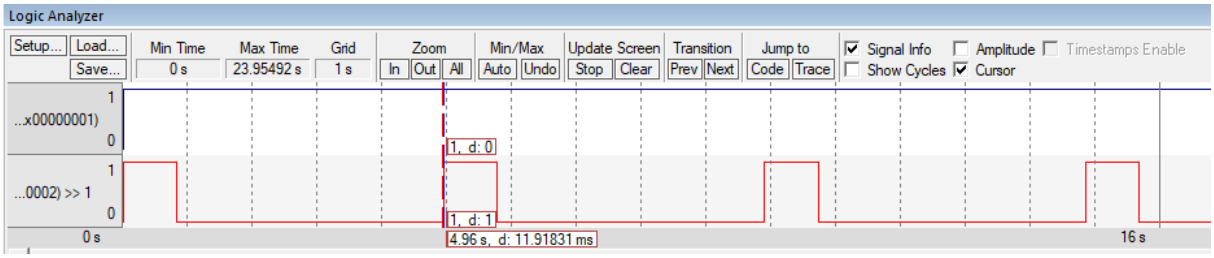


*Figure 11 - Logic Analyzer*

**Bonus Part**

When we click to the SW2, sky blue LED is on for 2 seconds. If we press to the SW1, it calls FlashSOS() function and if we press to the SW2, again, it simply turns the sky blue LED on. If we press nothing, it goes back to the red-blue-green. You can see it in simulation mode in Figure 6, and on board in Figure 9.

```
void SW2_Pressed(void){
  pressed_2 = 0;
  GPIO_PORTF_DATA_R |= 0x0C;  Delay(2); // Sky blue LED is on
  GPIO_PORTF_DATA_R &= ~0x0C;
  if(pressed_1 == 1){
    FlashSOS();
  }
  if(pressed_2 == 1){
    SW2_Pressed();
  }
}
```

**Part 3) Experimenting with the code**

1) **In this lab, you are detecting the pressing of a button via polling. Do you think if it is a good practice? Are there any disadvantages?**

   It is **not** a very **good** practice because to be able to react to switch operation, we must **constantly check** if button is pressed or not. Therefore, it causes **extra work** on software. For example, in this lab, we have to check it in Delay function to be able to catch if the button is pressed or not.

2) **In this lab, you introduced delay via looping. Do you think if it is a good practice? Are there any disadvantages?**

   It is not a good practice if we do not want blocking because it is in a while loop and we cannot do anything while delay function is running. Instead of delay function, we can use millis(). It calculates how much time passed, instead of waiting for time to pass. Therefore, we can do multiple things instead of waiting delay function using millis() function.

3) **How long does it take for a single iteration of the 1-second delay loop on the development board?**

   As I explained in the Measuring Time section, we should make 371192 iterations for 1 second. Therefore, one iteration equals to $\frac{1}{371192} = 2.69\text{x}10^{-6}$