Lab-4 Group -4 report

(210020050,210020007)

Aim:

- The goal of this lab is to write a program that performs the following tasks using the Tiva C microcontroller:
  a. The **main loop** continuously monitors the state of a **GPIO input pin** (SW1). When SW1 is pressed, the **RED LED** turns on, and when SW1 is released, the RED LED turns off.
  b. The **SysTick interrupt handler** toggles the **BLUE LED** every 500ms, creating a 1Hz blink rate.

Approach:

GPIO Input and Output Control:

- GPIO pins on the Tiva C board can be configured as either input or output. In this lab, **SW1** (pin PF4) is configured as an input, and **LEDs** connected to **PF1 (RED LED)** and **PF2 (BLUE LED)** are configured as outputs.
- **SW1** is pulled up internally, meaning it reads as 1 when unpressed and 0 when pressed.

SysTick Timer for Interrupts:

- The **SysTick timer is used to generate periodic interrupts. In this lab, it is configured to generate an interrupt every 500 ms**, toggling the BLUE LED within the interrupt service routine (ISR).
- The SysTick timer is set up to count based on the system clock (16 MHz), and the reload value is calculated to achieve the desired delay (500 ms).

GPIO and SysTick Timer Configuration:

- The GPIO pins **PF1 (RED LED)**, **PF2 (BLUE LED)**, and **PF4 (SW1)** are initialized. PF4 is configured as input with a pull-up resistor, and PF1 and PF2 are set as outputs for controlling the LEDs.
- The SysTick timer is configured to trigger an interrupt every 500 ms by calculating the correct reload value based on the system clock frequency.

Main Loop Logic:

- The main loop monitors the state of SW1 by continuously reading the value of PF4. When the switch is pressed (PF4 reads 0), the RED LED (PF1) is turned on, when it is released (PF4 reads 1), the RED LED is turned off.

```
int main(void)
{
```

```
    GPIO_INIT();
    STCURRENT = 0;
    STRELOAD = 500 * 16000;              // reload value for 'ms' milliseconds
    STCTRL |= (CLKINT | ENABLE | INTERRUPT); // set internal clock
    int mask = 1 << 1;
#if 1
    int clicked = 0;
    while (1)
    {
        clicked = GPIO_PORTF_DATA_R & 0x10;
        if (clicked == 0x0)
            GPIO_PORTF_DATA_R |= mask;
        else
            GPIO_PORTF_DATA_R &= ~mask;
    }
    return 0;
#endif
}
```

SysTick Interrupt Handler:

- The SysTick interrupt handler toggles the BLUE LED (PF2) every time the interrupt is triggered, creating a 1Hz blinking pattern.

```
void SysTickHandler()
{
    GPIO_PORTF_DATA_R ^= 0x4;
    return;
}
```

Observation:

- The program was successfully implemented on the Tiva C board. The following behaviors were observed:
  - The **RED LED** turned on when **SW1** was pressed and turned off when **SW1** was released, as expected.
  - The **BLUE LED** blinked at a frequency of **1Hz**, toggling on and off every 500 ms, indicating that the SysTick timer was functioning correctly.

Conclusion:

- The program demonstrated how to use GPIO input to control an LED and how to configure the SysTick timer to generate periodic interrupts in a polled mode. By combining both functionalities, the system was able to monitor a button press to control one LED and use interrupts to create a regular blinking pattern for another LED.