EE 215 Microprocessors LAB #5s

Student name：Li Xianzhe
Student ID number：2022214880

Explanation of Approach

NO QUESTIONS

Code

Lab5sa

```
#include <msp430.h>

#define CPU_F ((double)1000000)
#define delay_ms(x) __delay_cycles((long)(CPU_F*(double)x/1000.0))

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;  // Stop watchdog timer

    P8DIR |= BIT1;            // Set P8.1 as output for LED
    P8OUT &= ~BIT1;           // Initially turn off the LED
    P1REN |= BIT2;            // Enable pull-up resistor for P1.2 (Switch)
    P1OUT |= BIT2;            // Set pull-up resistor as active for P1.2
    P1IES = BIT2;             // Set interrupt edge select to trigger on
falling edge
    P1IFG &= ~BIT2;           // Clear P1.2 interrupt flag

    while(1)
    {
        if(P1IN & BIT2)
        {
            delay_ms(20);         // Debounce delay
            if(P1IN & BIT2)
            {
                P1IE |= BIT2;     // Enable interrupt on P1.2
                __bis_SR_register(LPM4_bits + GIE);  // Enter Low Power Mode
4 with interrupts enabled
            }
        }
    }
}

#pragma vector = PORT1_VECTOR
```

```
__interrupt void PORT1_ISR (void)
{
    P1IFG &= 0x00;           // Clear P1.2 interrupt flag
    P8OUT ^= BIT1;           // Toggle the LED state
    P1IES ^= BIT2;           // Toggle interrupt edge select for the next
rising/falling edge
}
```

Lab5sb

```
#include <msp430.h>

#define LED2_PIN BIT7
#define S2_PIN BIT3

#define CPU_F ((double)1000000)
#define delay_ms(x) __delay_cycles((long)(CPU_F * (double)x / 1000.0))

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;   // Stop watchdog timer

    P3DIR |= LED2_PIN;          // Set P3.7 as output for LED2
    P3OUT &= ~LED2_PIN;         // Initially turn off LED2
    P1REN |= S2_PIN;            // Enable pull-up resistor for P1.3 (S2)
    P1OUT |= S2_PIN;            // Set pull-up resistor as active for P1.3
    P1IES |= S2_PIN;            // Set interrupt edge select to trigger on
falling edge (button release)
    P1IFG &= ~S2_PIN;           // Clear P1.3 interrupt flag
    P1IE |= S2_PIN;             // Enable interrupt for S2 (P1.3)

    while (1)
    {
        if (!(P1IN & S2_PIN))// Button S2 is pressed
        {
            delay_ms(20); // Debounce delay
            if (!(P1IN & S2_PIN))
            {
                P3OUT ^= LED2_PIN; // Toggle LED2 state
                while (!(P1IN & S2_PIN)); // Wait for the button to be
released
            }
        }
```

```
    }
    return 0;
}

#pragma vector = PORT1_VECTOR
__interrupt void PORT1_ISR(void)
{
    P1IFG &= ~S2_PIN;  // Clear P1.3 interrupt flag
    P1IES ^= S2_PIN; // Toggle interrupt edge select for the next
rising/falling edge
}
```