Date: 03/28/2025

Experiment no: 02

Student No: EC/2021/006

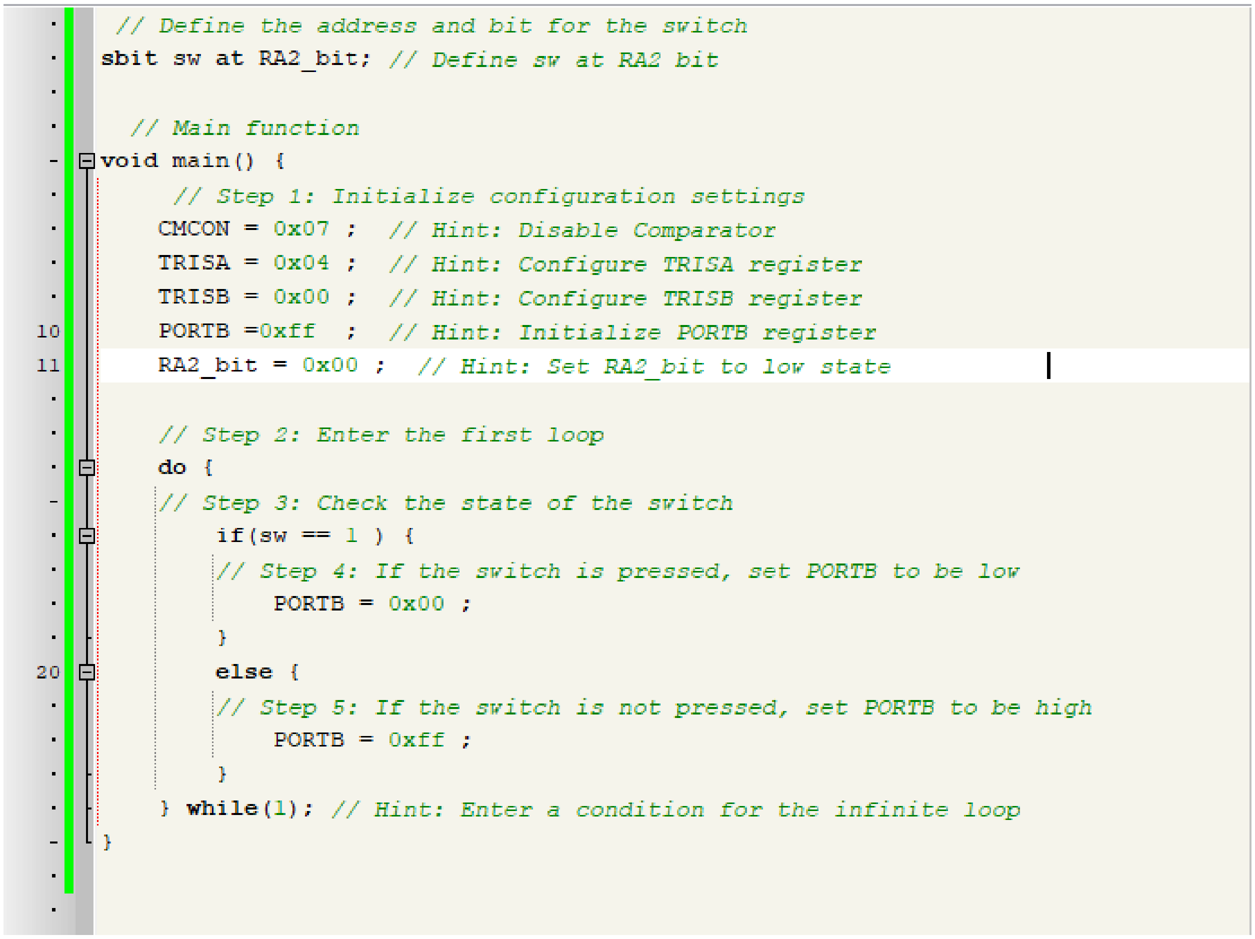
Discussion

In this experiment we learned fundamental concepts of configuring input output operation in PIC16F628A microcontroller programming. We clearly understand the TRIS register which determines whether a pin is set as an input or output. When bit of TRIS register is set to logic 1, the corresponding pin functions as an input. When it setting to logic 0 configures it as output. For this experiment, RA2 was assigned as an input (TRISA = 0b00000100), allowing it to read the state of a tactile switch, while PORTB was set as an output (TRISB = 0b00000000), controlling the state of LEDs.

A C program was written to monitor the state of the switch and control the LEDs. When the switch releases, it sends a high logic to RA2, triggering PORTB to output a high State by turning LEDs On. When the switch pressed RA2 sends low logic and PORTB trigger to output low state and all the LEDs off.

To validate the implementation, we first simulated the circuit in PROTEUS, where the HEX file of the compiled program was loaded onto the virtual PIC16F628A microcontroller. The simulation confirmed the expected LED behavior in response to switch actuation. Following this, the program was transferred to the physical microcontroller using Micro Pik Kit 3, and the circuit was assembled on a breadboard using the necessary components, including resistors, power supply, and an LM7805 voltage regulator.

Source Code

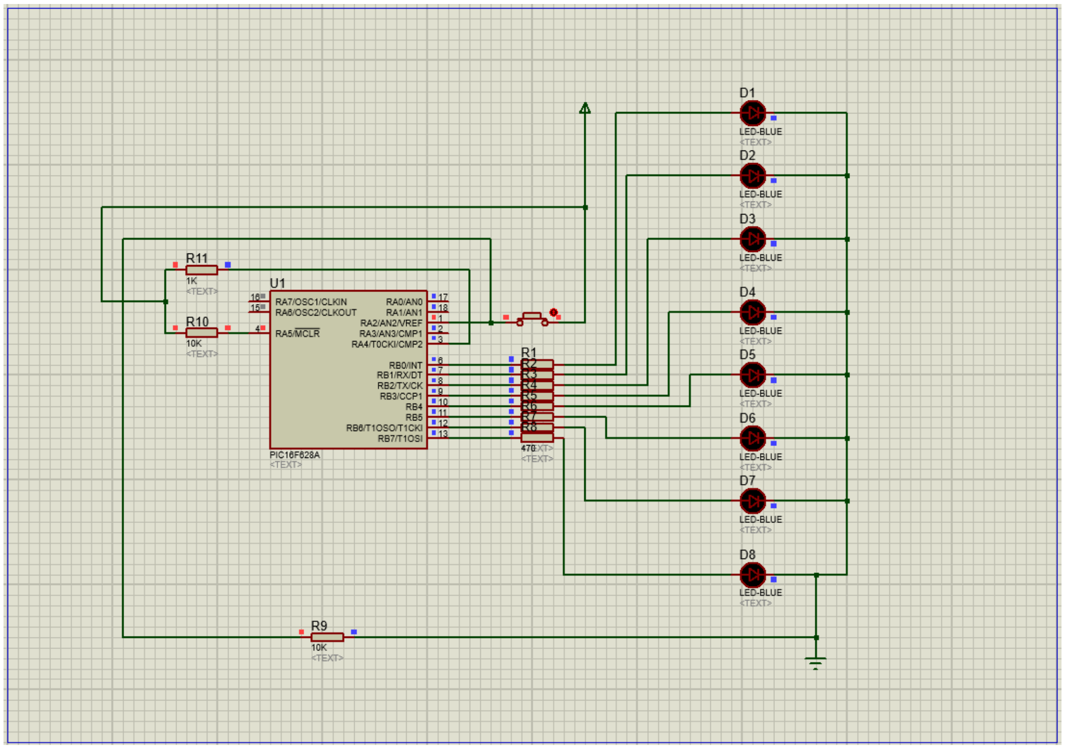


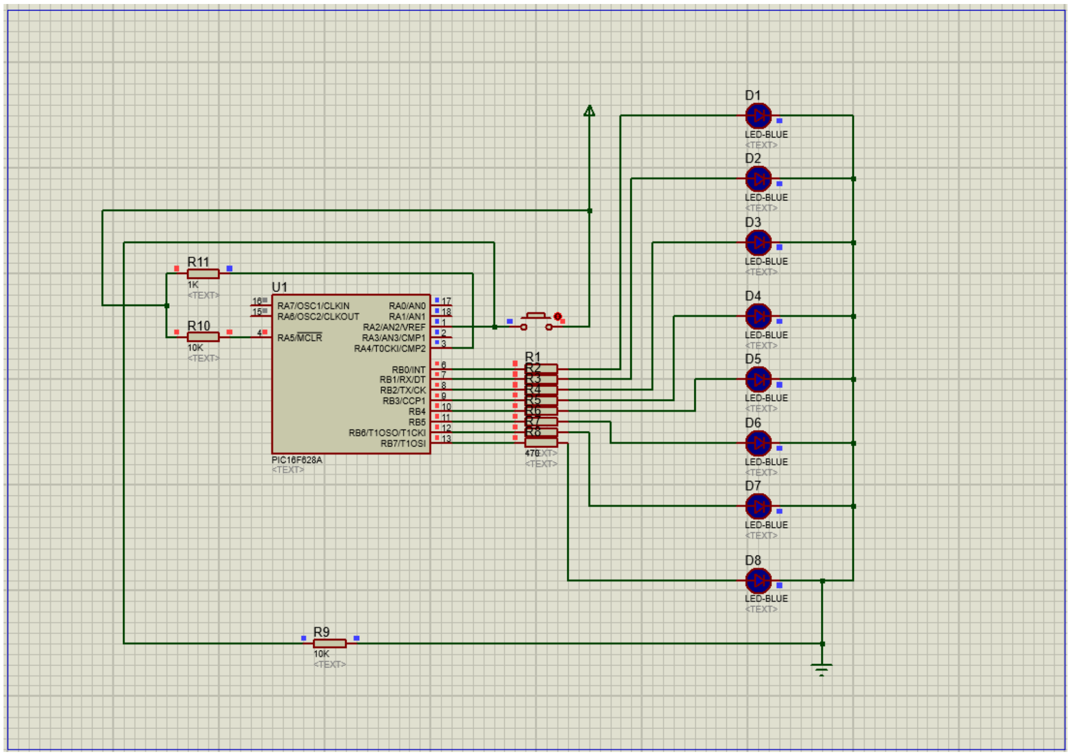
```
 // Define the address and bit for the switch
sbit sw at RA2_bit; // Define sw at RA2 bit

  // Main function
void main() {
     // Step 1: Initialize configuration settings
    CMCON = 0x07 ;  // Hint: Disable Comparator
    TRISA = 0x04 ;  // Hint: Configure TRISA register
    TRISB = 0x00 ;  // Hint: Configure TRISB register
    PORTB =0xff  ;  // Hint: Initialize PORTB register
    RA2_bit = 0x00 ;  // Hint: Set RA2_bit to low state

    // Step 2: Enter the first loop
    do {
    // Step 3: Check the state of the switch
        if(sw == 1 ) {
        // Step 4: If the switch is pressed, set PORTB to be low
            PORTB = 0x00 ;
        }
        else {
        // Step 5: If the switch is not pressed, set PORTB to be high
            PORTB = 0xff ;
        }
    } while(1); // Hint: Enter a condition for the infinite loop
}
```

```
// DEFINE THE ADDRESS AND BIT FOR THE SWITCH
SBIT SW AT RA2_BIT; // DEFINE SW AT RA2 BIT

  // MAIN FUNCTION
VOID MAIN() {
     // STEP 1: INITIALIZE CONFIGURATION SETTINGS
    CMCON = 0X07 ;  // HINT: DISABLE COMPARATOR
    TRISA = 0X04 ;  // HINT: CONFIGURE TRISA REGISTER
    TRISB = 0X00 ;  // HINT: CONFIGURE TRISB REGISTER
    PORTB =0XFF  ;  // HINT: INITIALIZE PORTB REGISTER
    RA2_BIT = 0X00 ;  // HINT: SET RA2_BIT TO LOW STATE

    // STEP 2: ENTER THE FIRST LOOP
    DO {
```

```
    // STEP 3: CHECK THE STATE OF THE SWITCH
        IF(SW == 1 ) {
        // STEP 4: IF THE SWITCH IS PRESSED, SET PORTB TO BE LOW
            PORTB = 0X00 ;
        }
        ELSE {
        // STEP 5: IF THE SWITCH IS NOT PRESSED, SET PORTB TO BE HIGH
            PORTB = 0XFF ;
        }
    } WHILE(1); // HINT: ENTER A CONDITION FOR THE INFINITE LOOP
}
```

Porteous Simulation


D1
LED-BLUE
D2
LED-BLUE
D3
LED-BLUE
D4
LED-BLUE
D5
LED-BLUE
D6
LED-BLUE
D7
LED-BLUE
D8
LED-BLUE
R11
1K
R10
10K
U1
RA7/OSC1/CLKIN
RA6/OSC2/CLKOUT
RA5/MCLR
RA0/AN0
RA1/AN1
RA2/AN2/VREF
RA3/AN3/CMP1
RA4/T0CKI/CMP2
RB0/INT
RB1/RX/DT
RB2/TX/CK
RB3/CCP1
RB4
RB5
RB6/T1OSO/T1CKI
RB7/T1OSI
PIC16F628A
R1
R2
R3
R4
R5
R6
R7
R8
470
R9
10K


D1
LED-BLUE
D2
LED-BLUE
D3
LED-BLUE
D4
LED-BLUE
D5
LED-BLUE
D6
LED-BLUE
D7
LED-BLUE
D8
LED-BLUE
R11
1K
R10
10K
U1
RA7/OSC1/CLKIN
RA6/OSC2/CLKOUT
RA5/MCLR
RA0/AN0
RA1/AN1
RA2/AN2/VREF
RA3/AN3/CMP1
RA4/T0CKI/CMP2
RB0/INT
RB1/RX/DT
RB2/TX/CK
RB3/CCP1
RB4
RB5
RB6/T1OSO/T1CKI
RB7/T1OSI
PIC16F628A
R1
R2
R3
R4
R5
R6
R7
R8
470
R9
10K