LABORATORY WORK REPORT NO. 1

MCU PORTS

Subject: Microprocessors and their Programming
Student: Halil Ibrahim Bekli
Target Device: PIC16F1518

1. Goal and Task

GOAL: Find out how the ports of the PIC16F1518 microcontroller are controlled.

TASK:
1. According to the selected variant (Variant 1), create and connect the circuit.
2. Program the microcontroller so that the LED connected to it lights up initially.
3. When the button is pressed, the LED should go out. It should light up again only when the button is released.
4. Use the microcontroller's internal pull-up resistors for the button circuit.

2. Task Variant Data

Selected Variant: 1

- LED Anode Connection: PORTB.4
- LED Cathode Connection: VSS (GND)
- Pushbutton Connection: PORTB.0
- Logic: LED turns ON with Logic High (1). Button reads Logic Low (0) when pressed (Pull-up).

3. Circuit Diagram Description

Based on Variant 1:

1. MCU: PIC16F1518
2. Power: VDD (+5V) to pin 20, VSS (GND) to pin 8 and 19.
3. MCLR: Pin 1 connected to +5V via 10k Resistor.
4. LED: Anode -> Pin 25 (RB4). Cathode -> Resistor (330R) -> GND.
5. Button: Pin 21 (RB0) -> Button -> GND. (Internal Pull-up Enabled).

4. Code with Comments

```
/* * File: LabWork1.c
 * Author: Halil Ibrahim Bekli
 * Variant: 1 (LED: RB4, BTN: RB0)
 */

#include <xc.h>

// CONFIG1 & CONFIG2 settings
#pragma config FOSC = INTOSC
#pragma config WDTE = OFF
#pragma config PWRTE = ON
#pragma config MCLRE = ON
```

LABORATORY WORK REPORT NO. 1

MCU PORTS

```
#pragma config CP = OFF
#pragma config BOREN = OFF
#pragma config CLKOUTEN = OFF
#pragma config IESO = OFF
#pragma config FCMEN = OFF
#pragma config WRT = OFF
#pragma config VCAPEN = OFF
#pragma config STVREN = ON
#pragma config BORV = LO
#pragma config LPBOR = OFF
#pragma config LVP = OFF

#define LED_LAT LATBbits.LATB4
#define LED_TRIS TRISBbits.TRISB4
#define BTN_PORT PORTBbits.RB0
#define BTN_TRIS TRISBbits.TRISB0
#define BTN_WPU WPUBbits.WPUB0
#define _XTAL_FREQ 4000000UL

void init_io();
void delay_ms_simple(int ms);

int main(void) {
    OSCCON = 0b01011010; // 4MHz Internal Osc
    init_io();

    LED_LAT = 1; // Start with LED ON

    while (1) {
        if (BTN_PORT == 0) { // Button Pressed (Active Low)
            __delay_ms(20); // Debounce
            if (BTN_PORT == 0) {
                LED_LAT = 0; // Turn LED OFF
                while (BTN_PORT == 0); // Wait for release
                __delay_ms(20); // Debounce release
                LED_LAT = 1; // Turn LED ON
            }
        }
    }
    return 0;
}

void init_io() {
    ANSELB = 0;   // Digital I/O
    LED_TRIS = 0; // Output
    BTN_TRIS = 1; // Input
    nWPUEN = 0;   // Enable Weak Pull-ups Global
    BTN_WPU = 1;  // Enable Pull-up for RB0
}
```

LABORATORY WORK REPORT NO. 1

MCU PORTS

5. Program Algorithm

1. Initialize System: Set Oscillator to 4MHz.
2. Configure GPIO: Set PORTB as digital. Set RB4 as Output, RB0 as Input.
3. Enable Pull-ups: Activate internal pull-up for RB0.
4. Initial State: Turn LED ON (Set RB4 High).
5. Loop:
   a. Read RB0 State.
   b. If RB0 is Low (Button Pressed):
      i. Debounce delay (20ms).
      ii. Turn LED OFF (Set RB4 Low).
      iii. Wait until RB0 becomes High (Button Released).
      iv. Debounce delay.
      v. Turn LED ON (Set RB4 High).
   c. Repeat Loop.

6. Conclusions

In this laboratory work, the control of PIC16F1518 ports was successfully implemented. We learned how to configure the TRIS and ANSEL registers for digital I/O. The use of internal pull-up resistors (nWPUEN, WPUB) allowed us to connect the button to ground without external resistors. The program logic successfully handled the requirement to turn the LED off only when the button is pressed, utilizing a polling method with debouncing.