Laboratory Work No. 2 Report

Microcontroller Interrupts and Timers (PIC16F1518)

Student Information

Subject: Microprocessors and their Programming
Student: Halil Ibrahim Bekli
Target Device: PIC16F1518
Variant: 1

1. Goal and Task

GOAL: To figure out how the interrupts of the PIC16F1518 microcontroller are managed and executed, and to master the operation of special registers: INTCON, PIE1, PIE2, PIR1, PIR2, IOCBP, IOCBN, as well as OPTION, TMR0, TMR1, and TMR2.

TASK: According to the assigned variant number (1), design and connect the circuit and program the microcontroller so that:
1. When a button is pressed, a light-emitting diode (LED) lights up immediately, and if it is already lit, it goes out.
2. Simultaneously, the program must control a 7-segment indicator to sequentially show the symbols specified in the task variant with a set period T.

2. Task Variant Data

Variant No.: 1
7-Segment Indicator Port: PORTB
Button Input: PORTC.0
LED Output: PORTC.1
Period (T): 700 ms
Symbols: 1, 2, 3, 4, 5

Timing Formula Used:
T = (255 - TMR0_initial) * Prescaler * k (microseconds)

3. Circuit Diagram

[Circuit Diagram Placeholder]

Description:
- PIC16F1518 Microcontroller.
- 7-Segment Display (Common Anode) connected to PORTB.
- Push Button connected to RC0 (with pull-down/up resistor as needed).
- Discrete LED connected to RC1.

4. Code with Comments

```
/* File: LabWork2.c */
#include <xc.h>

// CONFIG: Internal Osc, WDT Off, MCLR On
#pragma config FOSC = INTOSC
```

Laboratory Work No. 2 Report

Microcontroller Interrupts and Timers (PIC16F1518)

```
#pragma config WDTE = OFF
#pragma config MCLRE = ON

#define _XTAL_FREQ 4000000

// Variant 1 Definitions
#define BUTTON_PIN     PORTCbits.RC0
#define LED_PIN        PORTCbits.RC1
#define TMR0_START     100
#define K_LIMIT        18  // Calc for 700ms

unsigned char sequence[] = {
    0b11111001, // 1
    0b10100100, // 2
    0b10110000, // 3
    0b10011001, // 4
    0b10010010  // 5
};
volatile unsigned char k_counter = 0;
volatile unsigned char seq_index = 0;

void __interrupt() ISR(void) {
    if (INTCONbits.TMR0IF) {
        k_counter++;
        TMR0 = TMR0_START;
        if (k_counter >= K_LIMIT) {
            k_counter = 0;
            PORTB = sequence[seq_index];
            seq_index++;
            if (seq_index >= 5) seq_index = 0;
        }
        INTCONbits.TMR0IF = 0;
    }
}

void main(void) {
    OSCCON = 0b01101010; // 4MHz
    TRISB = 0x00;        // Display Output
    TRISC = 0xFF;        // Inputs
    TRISCbits.TRISC1 = 0; // LED Output

    OPTION_REGbits.T0CS = 0;
    OPTION_REGbits.PSA = 0;
    OPTION_REGbits.PS = 0b111; // 1:256

    INTCONbits.TMR0IE = 1;
    INTCONbits.GIE = 1;

    while(1) {
        if (BUTTON_PIN == 1) {
            __delay_ms(20);
            if (BUTTON_PIN == 1) {
                LED_PIN = ~LED_PIN;
                while(BUTTON_PIN == 1);
            }
        }
    }
}
```

Laboratory Work No. 2 Report

Microcontroller Interrupts and Timers (PIC16F1518)

5. Program Algorithm

1. Initialization: Set internal oscillator to 4MHz. Configure PORTB as output (Display) and PORTC bits for Button/LED. Configure Timer0 with 1:256 prescaler.
2. Interrupt Service Routine (ISR): Triggered on Timer0 overflow. Increments counter 'k'. When 'k' reaches the limit (calculated for 700ms), update PORTB with the next symbol and reset 'k'.
3. Main Loop: Polls the button state. If pressed, performs debounce, toggles the LED, and waits for release.

6. Conclusions

In this laboratory work, we successfully configured the PIC16F1518 Timer0 interrupts to control a display timing task independently of the main program loop. We verified the calculation of interrupt timing periods and successfully implemented simultaneous tasks (display updating and button polling).