Student Name:	蔡汉霖
Fuzhou University ID:	832002117

Student Name:	卿祺果
Fuzhou University ID:	832002127

EE302 Lab1

Introduction：

EQUIPMENT：MPLAB Simulator, PIC16F877A (Figure 1), PICkit 3.

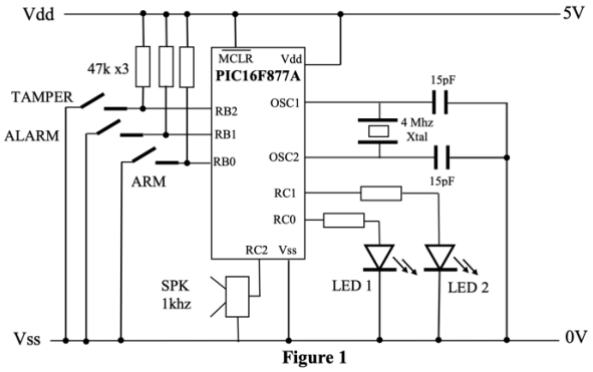


Figure 1 Operation of PIC16F877A

Part 1

In this question, we are required to analyze tables 1 and derive a simpler representation. Through a basic observe, we can figure out that when tamper, arm and alarm's status are 1, 0, 0, Led1, Led2 and SPK-1khz's statuses are off, on, on in respect. Despite that special case, when TAMPER's status is 1, the Led1, Led2 and SPK-1khz's statuses are ON, OFF, OFF. And when TAMPER's status is 0, the Led1, Led2 and SPK-1khz's statuses are ON, ON, ON.

Table 1 The desired function of the circuit

TAMPER	ARM	ALARM		LED1	LED2	SPK-1khz*
1	1	1		ON	OFF	OFF
1	1	0		ON	OFF	OFF
1	0	1		ON	OFF	OFF
1	0	0		OFF	ON	ON
0	1	1		ON	ON	ON
0	1	0		ON	ON	ON
0	0	1		ON	ON	ON
0	0	0		ON	ON	ON

Part 2

In this part, we are required to outline the design of the program by using pseudo code. And the Table 2 shows the pseudo code.

Table 2 Pseudo Code for the Program

Pseudo code:
If TAMPER = 1 and ARM = 0 and ALARM = 0 then LED1 ← OFF LED2 ← ON SPK-1khz ← ON Else If TAMPER = 1 and ARM = 0 and ALARM = 0 then LED1 ← ON LED2 ← OFF SPK-1khz ← OFF Else then LED1 ← ON LED2 ← ON SPK-1khz ← ON

Part 3

In this part, we are required to list the Special Function Registers associated with this program task and detail the bit configuration for each. As the Table 3 shown, the following are the special function registers we used in this program.

Table 3 Special Function Register

Number	Registers	Bit Configuration
01	PORTB	0x00
02	TRISB	0x03
03	PORTC	0x00
04	TRISC	0x00
05	TRISD	0x00
06	PORTD	0xFF

Part 4

The complete code of C program is shown in Table 4.

Table 4

The C program based on MPLAB

```
/*****************************************************************************
;
; Title:          demo-for-lab-1.c
; Platform:       PICmicro PIC16F877 @ 4 Mhz
; Modified by:    Hanlin Cai (蔡汉霖)
; Date:           Sep. 2022
;
; Function: Outputs a square wave tones RC2 depending on the state of
; two push buttons (RB0 and RB1)
;
*****************************************************************************/


#ifndef _XTAL_FREQ
 // Unless already defined assume 4MHz system frequency
 // This definition is required to calibrate __delay_us() and __delay_ms()
 #define _XTAL_FREQ 4000000
#endif

//Configuration Bits
#pragma config FOSC = XT        // Oscillator Selection bits (XT oscillator)
#pragma config WDTE = OFF       // Watchdog Timer Enable bit (WDT disabled)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT
disabled)
#pragma config BOREN = OFF      // Brown-out Reset Enable bit (BOR
disabled)
#pragma config LVP = OFF        // Low-Voltage (Single-Supply) In-Circuit
Serial Programming Enable bit (RB3 is digital I/O, HV on MCLR must be used
for programming)
#pragma config CPD = OFF        // Data EEPROM Memory Code Protection bit
(Data EEPROM code protection off)
#pragma config WRT = OFF        // Flash Program Memory Write Enable bits
(Write protection off; all program memory may be written to by EECON
control)
#pragma config CP = OFF         // Flash Program Memory Code Protection bit
(Code protection off)
```

```
// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

#include <xc.h>

//Prototype Declarations
void Tone();
void Init();
void test();

//Constant Declarations
#define BUTTON1 RB0 //bit 0 of PORTB
#define BUTTON2 RB1 //bit 1 of PORTB
#define BUTTON3 RB2 //bit 2 of PORTB
#define LED1     RC0      //bit 0 of PORTD
#define LED2     RC1      //bit 1 of PORTD
#define SPK1     RC2      //bit 2 of PORTC
#define OPEN     1        //Give a name to the open state of a button
#define PRESSED  0        //Give a name to the pressed state of a button

//Variables
int RepeatCount; //Variable used in repeat loops

// Main Program
void main()
{
        Init();          //Do initialization
        for(;;)          //Continuous Superloop
        {
        // TwoTone();
   test();
        }
}

//Initialization
void Init()
{
        PORTB = 0x00;    //Set PORTB outputs to 0
        TRISB = 0x03;    //Set RB0 and RB1 as inputs
        PORTC = 0x00;    //Set PORTC outputs to 0
        TRISC = 0x00;    //RC2 set as an output along with the rest of
PORTC
   TRISD = 0x00; //Set PORTD as outputs
   PORTD = 0xFF; //Set PORTD outputs to 1, switches off all LEDs
```

```
}

//Functions

void Tone()       // 1 cycle of 2khz squarewave output
{
         SPK1 = 1;
         __delay_us(250);          //250us delay
         SPK1 = 0;
         __delay_us(250);          //250us delay

}

void test() {
    if ((BUTTON3 = PRESSED)) //If BUTTON3 PRESSED
    {
        LED1 = 1;
        LED2 = 1;
        Tone();    // 1 cycle of 500hz
    } else if ((BUTTON3 = OPEN) && (BUTTON2 = PRESSED) && (BUTTON2 =
PRESSED)) //If only BUTTON3 OPEN
    {
        LED1 = 0;
        LED2 = 1;
        Tone();    // 1 cycle of 500hz
    } else {
        LED1 = 1;
        LED2 = 0;
        Tone();    // 1 cycle of 500hz
    }
}

// Modified by:  Hanlin Cai
// Date:         Sep.25th 2022
```

Part 5

In this part, we are required to verify the function of the program step by step using the MPLAB Simulator. So, we should set up a break point in Inin(), as the Figure 2 shown.

```
53   // Main Program
54   void main()
55   {
         Init();      //Do initialization
57       for(;;)      //Continuous Superloop
58       {
59       // TwoTone();
60       test();
61       }
62   }
```

Figure 2 Break Point

Then we used the function in MPLAB (Debugging, Watch, Stimulus) to verify the program. Firstly, we set the Watch function to PORTB, TRISB, PORTC, TRISC, TRISD and PORID, as shown in Figure 3 and Figure 4.

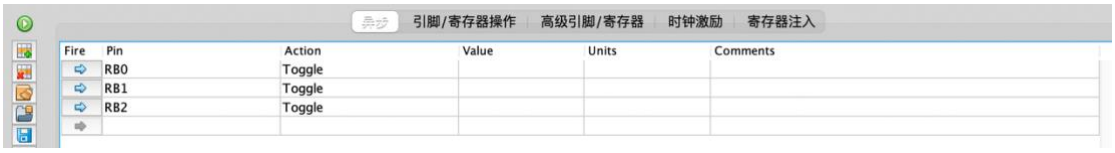


Figure 3 Stimulus Function

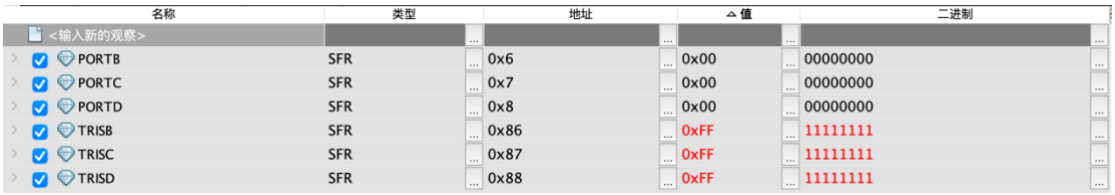
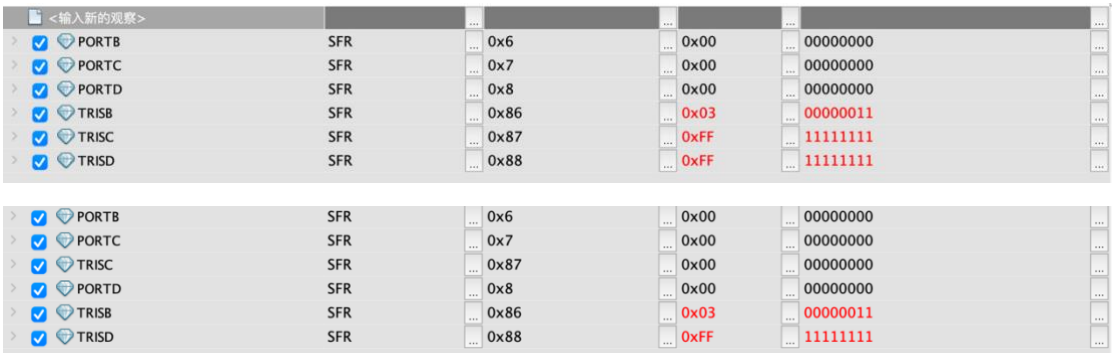


Figure 4 Watch Function I

The following is step by step demonstration:



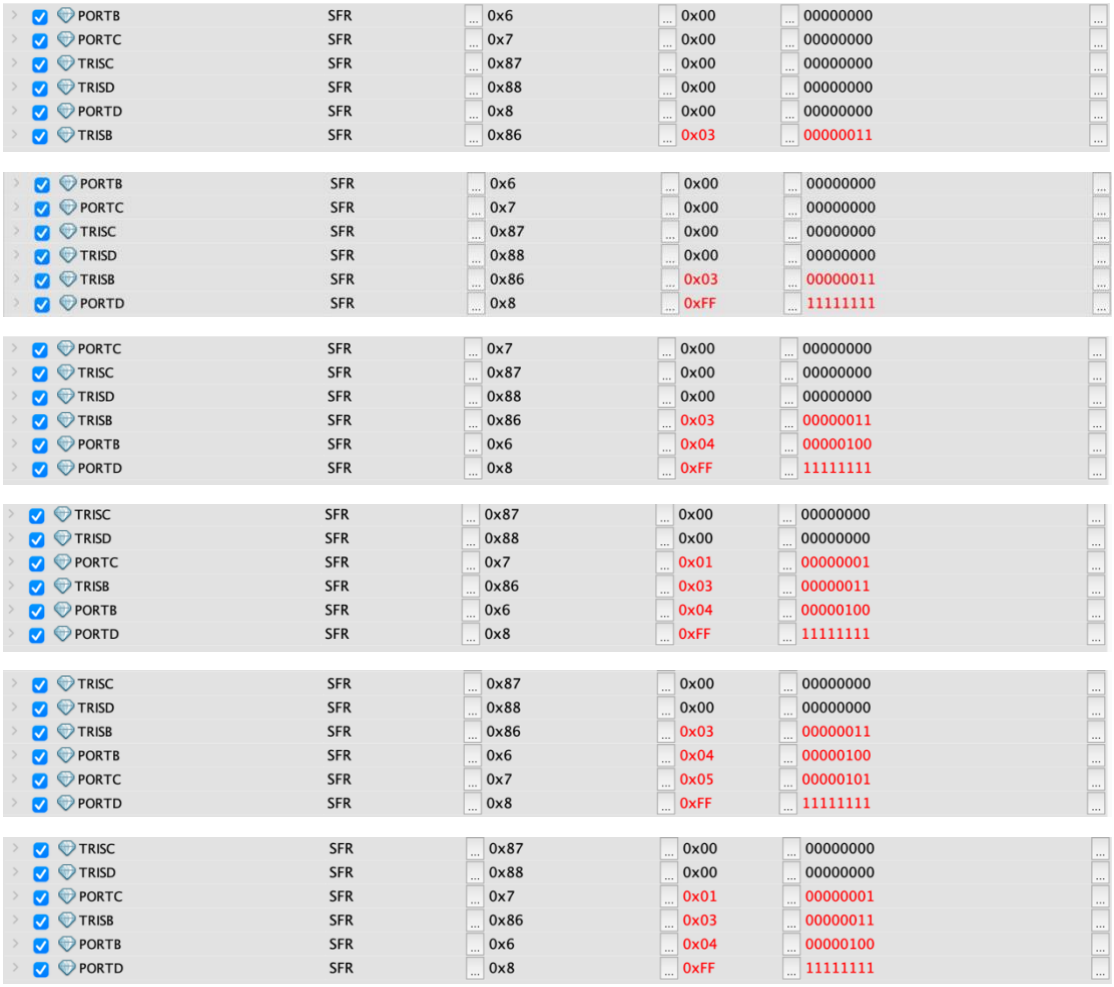


Figure 5 Watch Function II

Finally, the bit configuration and value go into loop, as the Figure 4 and Figure 5 shown.

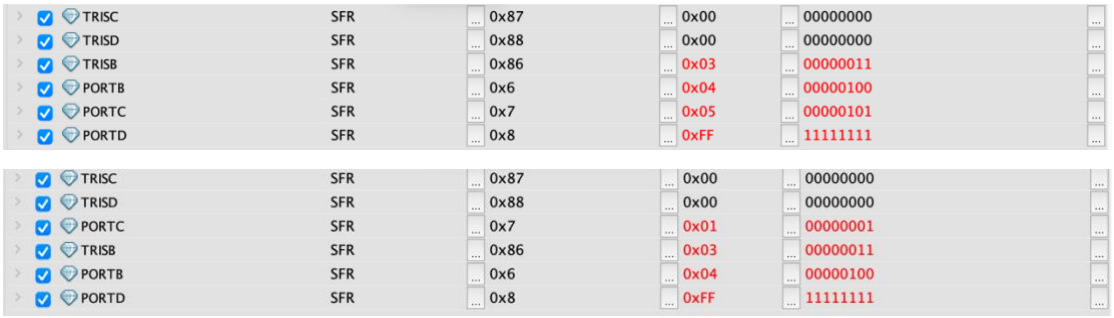


Figure 6 Watch Function III

As the Figure 5 and Figure 6 shown, we have used the MPLAB simulator to verify our program. So, that's the end of our exervise.

Summary for this Lab 1

In this Lab, we have learned the basic knowledge of PIC16F877A and how to use the MPLAB to write the program to the hardware. Thanks to our tutor, Yanxiang Wang, for his patient explanation and guidance.

By Hanlin Cai and Qiguo Qing.

In 2022/9/27.