**Name :**

**Roll no :**

**Group B Lab Assignment:** 8

**Subject :**PSDL

**Title :** External interrupt input switch press, output at relay

**Assignment No: 8**

**Title :** Embedded C program for External interrupt input switch press, output at relay

**Aim :** To write a C program to interface PIC18F4550 to Relay and switch it ON/OFF using input from external switch. Use ISR programming for External Interrupt.

**Experimental Setup:** MicroPIC18F board, USB cable, Power supply adaptor, MPLABx IDE, PICLoader software.

**Objective:**

* To Perform interfacing of real-world input and output devices to PIC18FXXX microcontroller.
* To learn embedded C programming for External interrupt input switch press, output at relay.

**Theory:**

**1). Interrupt Handling in PIC18F4550**

**1.1). Interrupts:**

An interrupt is a signal to a processor, generated by hardware or software, of the occurrence of a particular event. The processor receiving the interrupt signal and requested to stop the current execution sequence and attend to a different code sequence called the interrupt vector or Interrupt Service Routine( ISR ). After executing the ISR, the previous execution of sequence is resumed. The processor may choose to ignore or act on the request according to the configuration set by the programmer with the help of registers. These interrupts are given to the microcontroller unit through external pins of the microcontroller

* Once the controller completes the routine, it returns to the location from where it had made a jump.
* In cases where interrupts are not used, the program would need to constantly poll the input signals to monitor external events for catching the pulses when they occurred. But in some cases, the program can miss an event.
* E.g. An infrared slot sensor trying to catch a coin drop. In this situation, using an interrupt can free the microcontroller to get some other work done while not missing the input. In such cases, external interrupts are used.

**1.2). Interrupts in PIC18F4550**

There are 2 types of interrupts based on origin

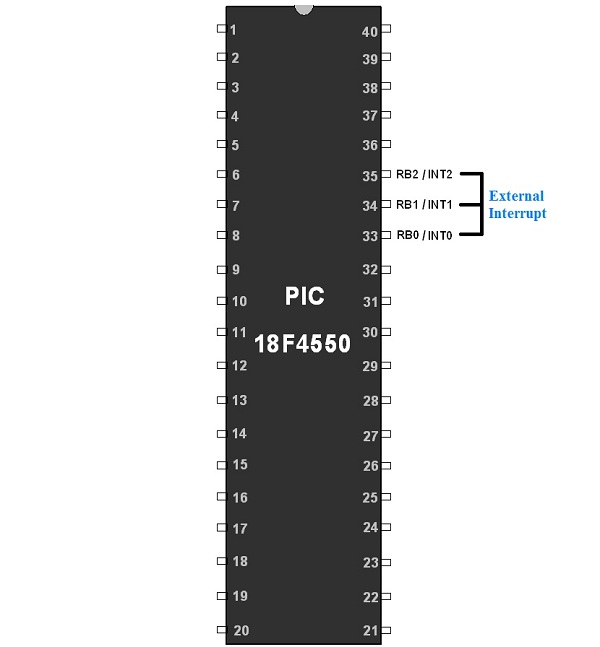
* Software Interrupt: It comes from a program that is executed by a microcontroller or by internal peripherals of the microcontroller.
* Hardware Interrupt: These interrupt requests are sent by external hardware devices connected to certain pins of the microcontroller.

Interrupts could also be classified based on their priority

* High priority Interrupt: These interrupts cannot be interrupted. A high priority interrupt vector is located at 0008h in the program memory.
* Low priority Interrupt: These interrupts itself could be interrupted by high priority interrupts and its interrupt vector is located at 0018h.
* The interrupt used in PIC18f4550 are edge triggered and the edge trigger could be configured as a rising edge or falling edge.

**1.3). External Interrupt Pins**

PIC18F4550 has three external hardware interrupts - INT0, INT1, and INT2. They are on PORTB pins RB0, RB1, and RB2 shown in the below image.



These interrupts are edge-triggered interrupts i.e. triggered by either a rising edge or by a falling edge.

**1.4). Registers associated with Interrupt**

**Reset control register**

**RCON**

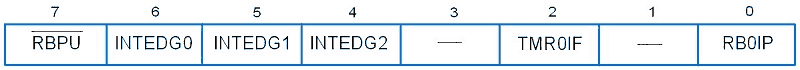
[Interrupt Handling in PIC18F4550](https://openlabpro.com/wp-content/uploads/2018/06/1.png)

* **IPEN:**Interrupt Priority Enable bit. Enables priority levels when se

The edge trigger bit is present in an INTCON2 register.

**INTCON2 Register:**

**INTCON2**: Interrupt Control Register



**INTCON2 Register**

**INTEDG0:** External Interrupt 0 Edge select bit

**1**= Interrupt on Rising Edge

**0**= Interrupt on Falling Edge

**INTEDG1:** External Interrupt 1 Edge select bit

**1**= Interrupt on Rising Edge

**0**= Interrupt on Falling Edge

**INTEDG2:** External Interrupt 2 Edge select bit

**1**= Interrupt on Rising Edge

**0**= Interrupt on Falling Edge

**RBPU:** PORTB Pull-up enable bit

* PORTB has a weak internal pull-up. So to enable them RBPU bit is used.

**1**= All PORTB Pull-ups are disabled

**0**= All PORTB Pull-ups are enabled

**TMR0IP:** TMR0 overflow interrupt Priority

**1**= High Priority

**0**= Low Priority

**RBIP:** RB PORT change Interrupt Priority

**1**= High Priority

**0**= Low Priority

**Procedure:**

**Step1:** Open MPLABX IDE on the PC for program development and create a new project and save it in a new folder.

**Step2:** Write the program in C language for interfacing Relay to PIC18F4550, using External Interrupt ISR. (in program properties make sure to add the 0x800 offset)

**Step3:** Build the program and create hex file. In case of errors correct program and rebuild to create hex file.

**Step4:** Prepare the experimental setup by connecting the MicroPIC18F board to the PC using USB cable. Power ON the Board. Check for the USBtoSerial COMx allocated by the PC**.**

**Step5:** Using the PICLoader Software flash the hex file in the PIC18F4550.

**Step6:** Press reset button and execute the program.

**.**

**Source code :**

#include <pic18f4550.h>

#define RELAY\_PIN LATAbits.LATA4

void interrupt extint\_isr(void)

{

unsigned int i;

if(INT1F)

{

INT1F = 0;

INT1IE = 0;

RELAY\_PIN = ~RELAY\_PIN;

for(i=0; i<10000; i++); //small delay for debouncing

INT1IE = 1;

}

}

int main()

{

ADCON1 = 0x0F; //set pins as Digital

TRISAbits.TRISA4 = 0; //set relay pin RA4 as output

TRISBbits.TRISB1 = 1; //Interrupt pin as input

RELAY\_PIN = 1;

INT1IE = 1; //Enable external interrupt INT1

INTEDG1 = 0; //Interrupt on falling edge

GIE = 1; // Enable global interrupt

while(1);

}

**Result:** Check if the Relay is switching ON/OFF when external interrupt switch is pressed and the ISR is getting executed .

**Conclusion:** Thus, we have studied embedded C program to interface PIC18F4550 to Relay and switch it ON/OFF using input from external switch using ISR programming for External Interrupt.