Lab Manual for Embedded System Design

Lab No. 8

Interrupt Enabling/ Disabling

Objectives

Understanding the basic concept of Interrupt Enabling/ Disabling and implementation of concepts on Arduino UNO and IDE

LAB # 8
Interrupt Enabling/ Disabling

Introduction

Interrupts are useful for making things happen automatically in microcontroller programs and can help solve timing problems. Good tasks for using an interrupt may include reading a rotary encoder, or monitoring user input.

Generally, an ISR should be as short and fast as possible. If your sketch uses multiple ISRs, only one can run at a time, other interrupts will be executed after the current one finishes in an order that depends on the priority they have. millis() relies on interrupts to count, so it will never increment inside an ISR. Since delay() requires interrupts to work, it will not work if called inside an ISR. micros() works initially but will start behaving erratically after 1-2 ms. delayMicroseconds() does not use any counter, so it will work as normal.

Arduino Uno has digital I/O pin 2, 3 that can be used for external interrupts.

Syntax

attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)

Syntax

To turn off the external interrupts use:

detachInterrupt(digitalPinToInterrupt(pin))

Timer Interrupts:

Timer interrupts allows designer to perform tasks at specific time intervals, for example by using timer interrupts, incoming signal from sensor can be measured at specified time intervals.

Arduino UNO uses atmega328p as microcontroller and it has three timers

Timer0 - 8 Bit

Timer1 - 16 Bit

Timer2 - 8 Bit

In timer interrupts when timer overflows (i.e. time completes), an ISR is called.

Time Boxing

Activity Name	Activity Time	Total Time
Login Systems + Setting up Proteus & Arduino Environment	3 mints + 5 mints	8 mints
Walk through Theory & Tasks	60 mints	60 mints
Implement Tasks	80 mints	80 mints
Evaluation Time	30 mints	30 mints
	Total Duration	178 mints

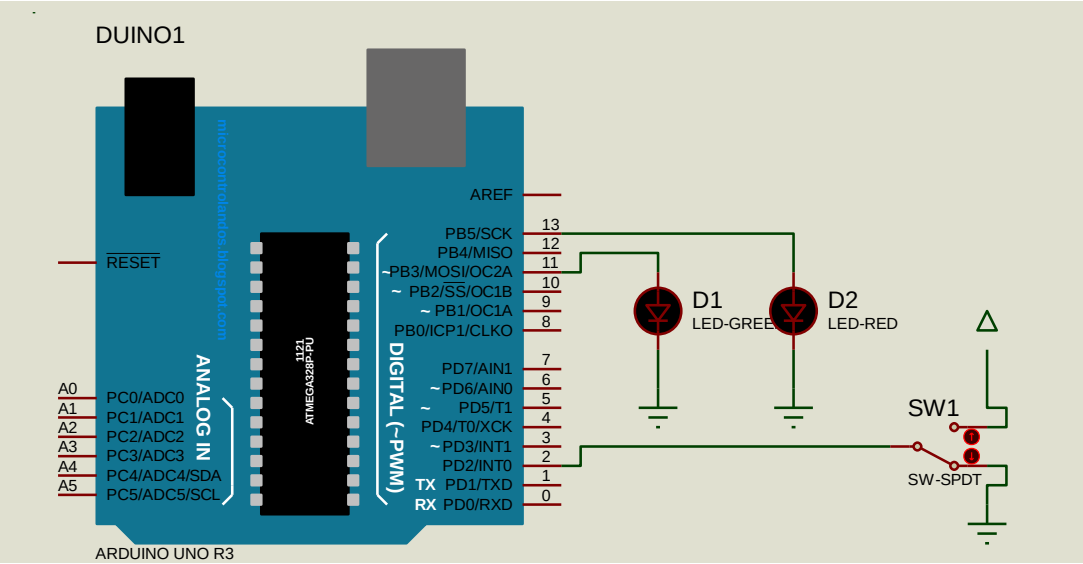
Objectives/Outcomes

This Lab exercise delivers the idea/concept of:

- Understand use of interrupts Enabling/Disabling in Arduino.
-

Lab Tasks/Practical Work

Task # 01: In this lab, we will use external interrupt in Arduino UNO.



Code:

void setup() {

// put your setup code here, to run once:

pinMode(11,OUTPUT);

pinMode(13,OUTPUT);

pinMode(2,INPUT);

attachInterrupt(digitalPinToInterrupt(2),routine,CHANGE);

}

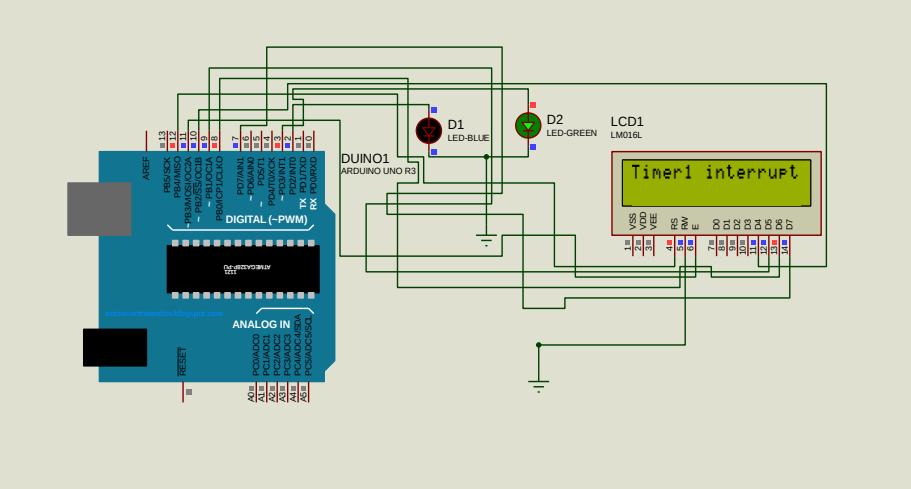
```
void loop() {
  // put your main code here, to run repeatedly:
digitalWrite(11,HIGH);
delay(1000);
digitalWrite(11,LOW);
delay(1000);

}

void routine()
{
  digitalWrite(13,HIGH);

}
```

Task # 02: Write a program to use timer interrupt in Arduino Uno.



Code:

```
#include <LiquidCrystal.h>
#include <TimerOne.h>
int led = 2;            // led connect to pin 2 of arduino
int led_i = 3;        // led_i connect to pin 3 of arduino
int x =0;
LiquidCrystal lcd(12, 11, 10, 9, 8, 7);  // arduino pins for lcd connections
void setup() {

pinMode(led,OUTPUT);
pinMode(led_i,OUTPUT);
pinMode(4,INPUT);
lcd.begin(16,2);
  Timer1.initialize(4000000);   /// 4 sec timer
  Timer1.attachInterrupt(interrupt);
}

void interrupt()
{
  lcd.setCursor(0, 0);
  lcd.print("Timer1 interrupt");     // ISR
  if (x==LOW)
  { digitalWrite(3,HIGH);
  x = HIGH;
  }
  else
  {digitalWrite(3,LOW);
    x =LOW;
  }
}
```

```
void loop() {
delay(500);
lcd.clear();
digitalWrite(2 ,HIGH);
delay(1000);
digitalWrite(2,LOW);
delay(1000);
}
```

Task # 03: Using the concept of interrupts you have learnt, develop a program that uses interrupt.