Objective:

The main purpose of this experiment is to implement an external input such as a push switch in an Arduino powered system and to observe the effects of push switch in the Microcontroller. Through this experiment is a simple yet effective system is made where a series of 6 LEDs flash in a specific sequence and upon pressing the switch, the sequence reverses it's direction. A common phenomenon known as bouncing occurs while pressing the switch. This generates multiple signals which creates uncertainty in the Microcontroller.

Apparatus:

i) Arduino IDE (any version)
ii) Arduino Mega
iii) Tilt sensor
iv) LED lights (six)
v) Regi Resistors (One 10kΩ and others are 220Ω).

Theory:

Until now, we have used delay when you needed something to happy on the Arduino at a specific time(). This is practical but somewhat restricting. When the arduino uses the delay() function, it's present state is preserved for the duration of the delay. Hence, while it is waiting, there can be neither input nor output. Moreover, delays are not very useful for keeping track on time. A delay of 10 seconds would be millis() method. It records the number of milliseconds that our Arduino has been running.

A tilt switch will change state when your Digital timer is turned over, which will cause another cycle of LEDs to light on.

The tilt switch functions as an on/off sensor exactly like a standard switch. It will serve a digital input for this. Tilt switches are distict because they recognize orientation. They often have a metal ball in a small chamber inside the housing. The switch is closed when the switch is properly slanted, causing the ball to slide to one side of the cavity and connect to the two leads on your breadboard. Your timer will operate for six ~~t~~minutes if it has six Lights

Procedure :

i) Firstly, we familiarized with the different type of Arduino family components. Arduino Mega Microcontroller is one of them.

ii) After that, we know the basic working principle of thinker cad software.

iii) Here, we need to open the thinker cad online software to implement the given circuit. Then we select the different types of components to implement it.

iv) After that we implement the circuit in our bread board and connect this experiment to the hardware.

Source code:


sketch_feb09a | Arduino 1.8.19
File Edit Sketch Tools Help
sketch_feb09a
const int switchPin = 8;
unsigned long previousTime = 0;
int switchState = 0;
int prevSwitchState = 0;
int led = 2;
long interval = 2000;
void setup() {
for(int x = 2;x<8;x++){
pinMode(x, OUTPUT);
}
pinMode(switchPin, INPUT);
}
void loop(){
unsigned long currentTime = millis();
if(currentTime - previousTime > interval) {
previousTime = currentTime;
digitalWrite(led, HIGH);
led++;
if(led == 7){
}
}
switchState = digitalRead(switchPin);
if(switchState != prevSwitchState){
for(int x = 2;x<8;x++){
digitalWrite(x, LOW);
}
led = 2;
previousTime = currentTime;
}
prevSwitchState = switchState;
}


Done compiling.
no_avr_uno_0b855f4dde85493394e91d005a5fd7f4.a
o -fuse-linker-plugin -Wl,--gc-sections -mmcu=atmega328p -o "C:\\Users\\User\\AppData\\Local\\Temp\\arduino_build_393303/sketch_feb09a.ino.elf
.eeprom --set-section-flags=.eeprom=alloc,load --no-change-warnings --change-section-lma .eeprom=0 "C:\\Users\\User\\AppData\\Local\\Temp\\ar
.eeprom "C:\\Users\\User\\AppData\\Local\\Temp\\arduino_build_393303/sketch_feb09a.ino.elf" "C:\\Users\\User\\AppData\\Local\\Temp\\arduino_b
\\User\\AppData\\Local\\Temp\\arduino_build_393303/sketch_feb09a.ino.elf"
Maximum is 2048 bytes.

Simulation:

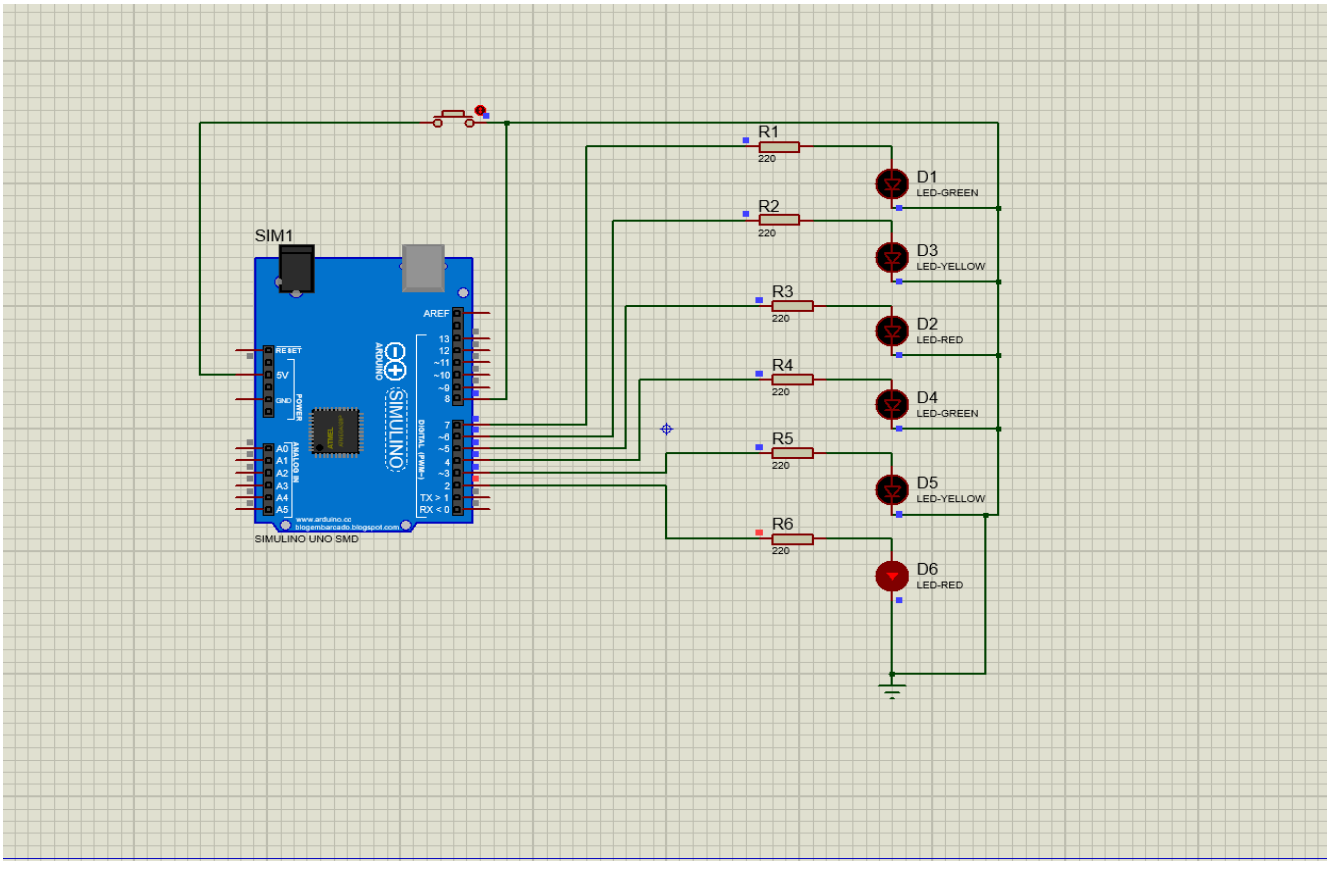

Figure: 1st LED is On

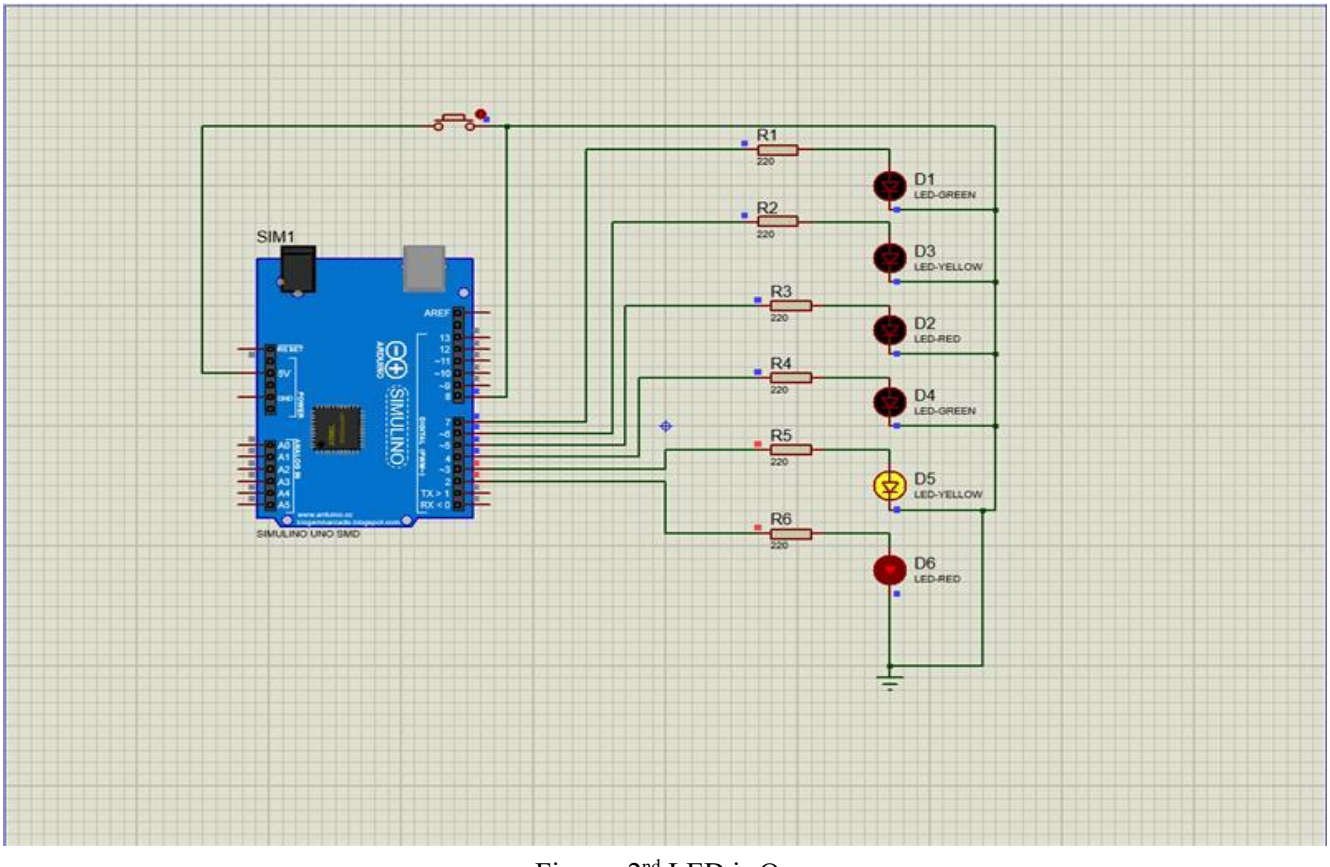

Figure: 2nd LED is On

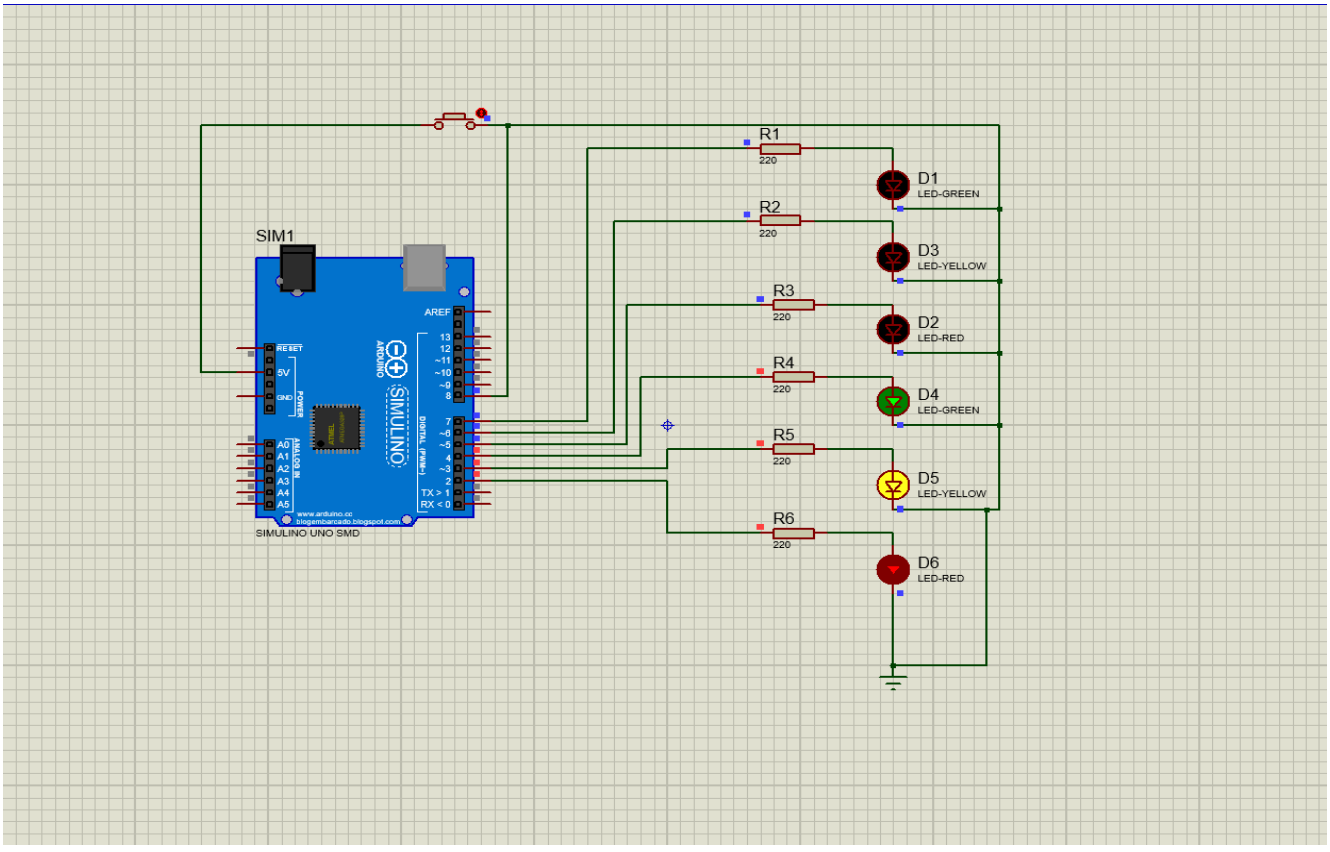

Figure: 3rd LED is On

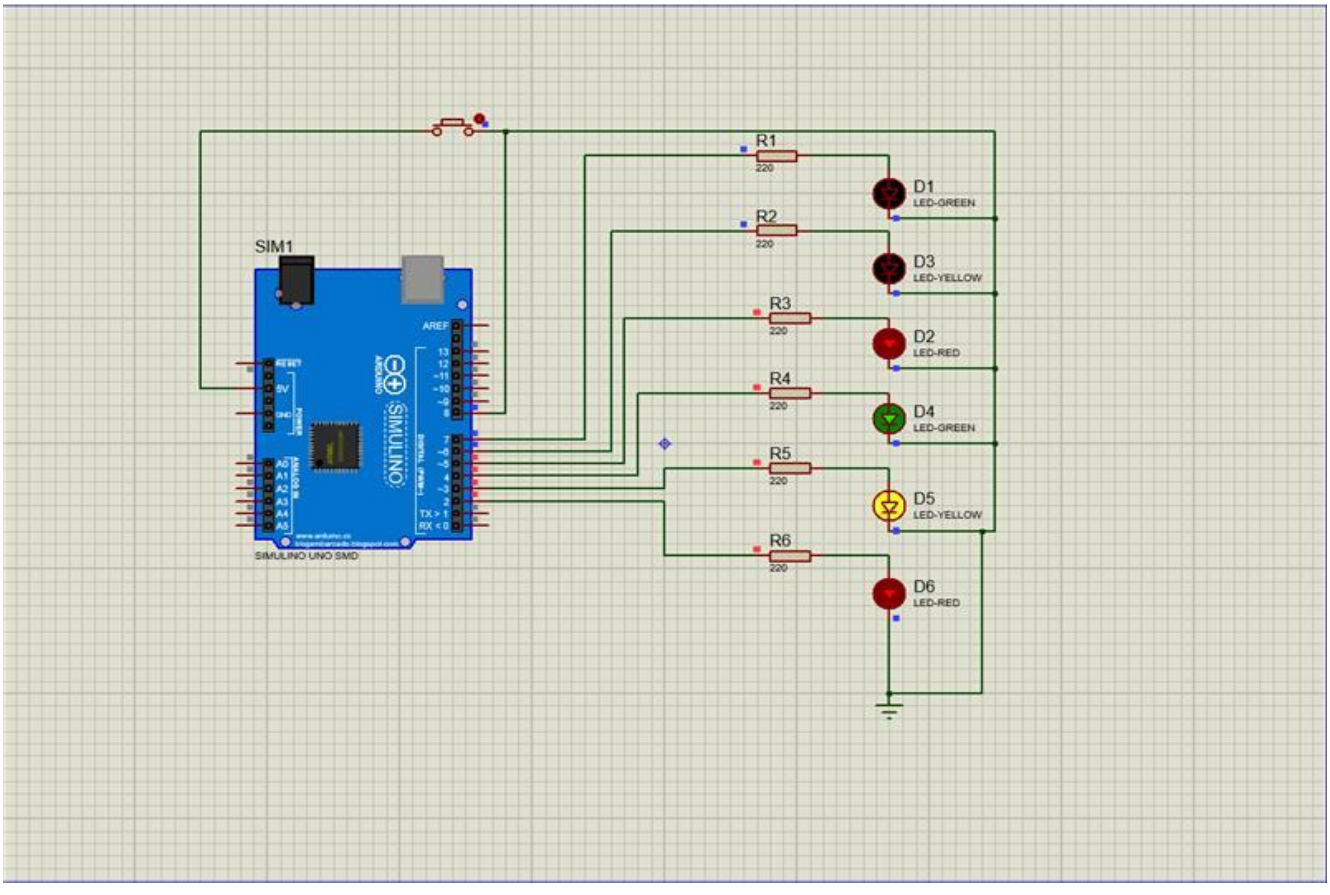

Figure: 4th LED is On

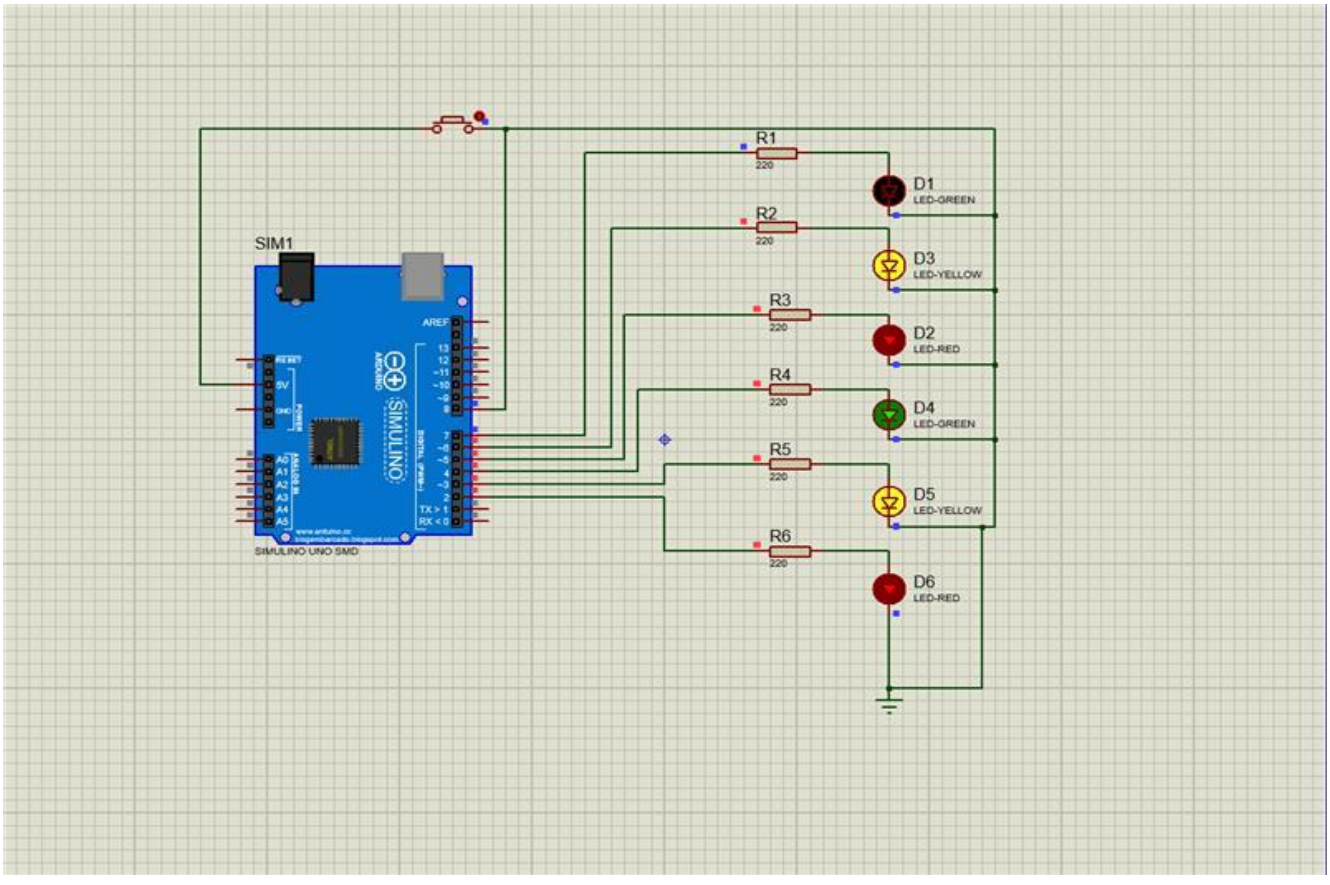

Figure: 5th LED is On

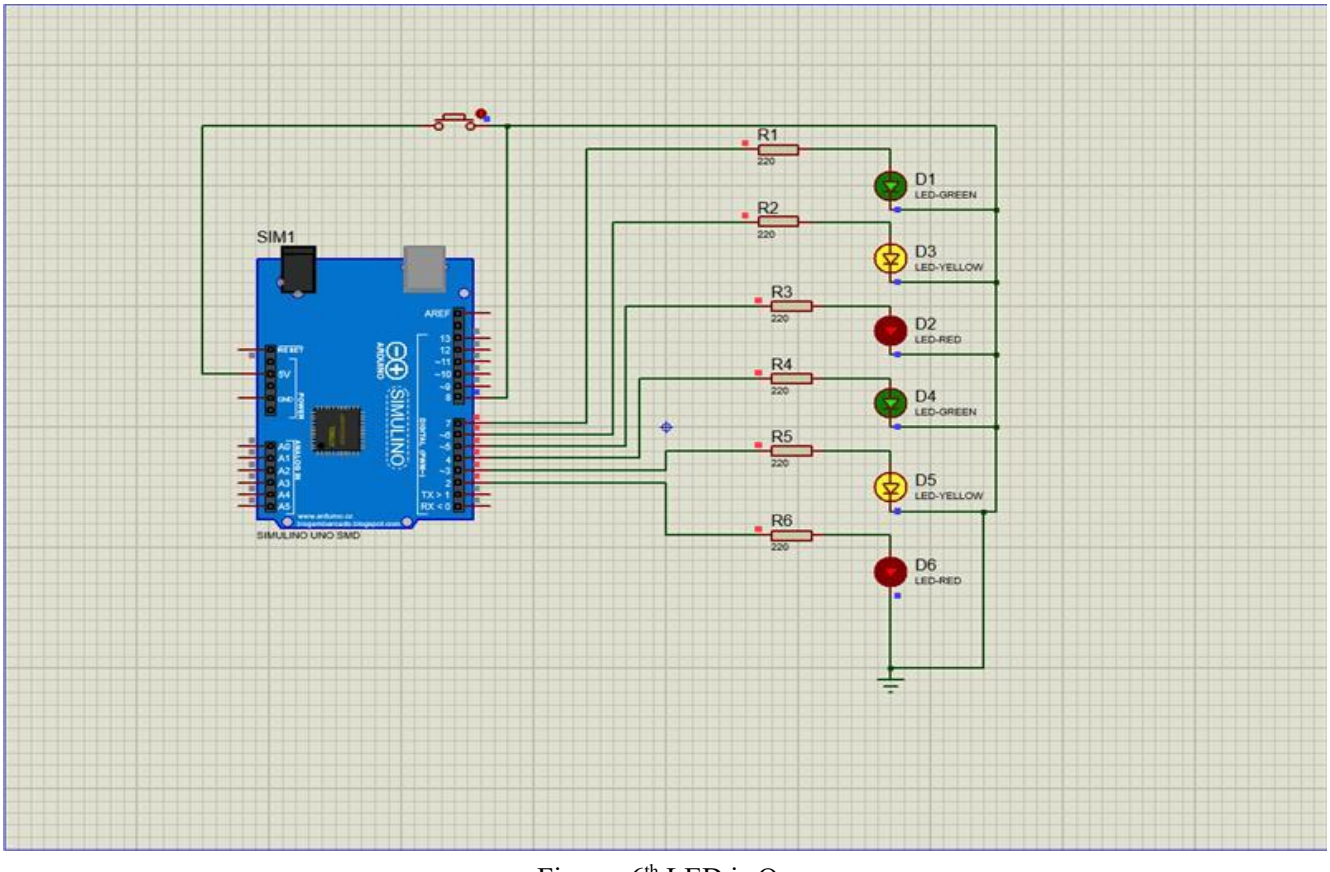

Figure: 6th LED is On

sHardware Implementation:

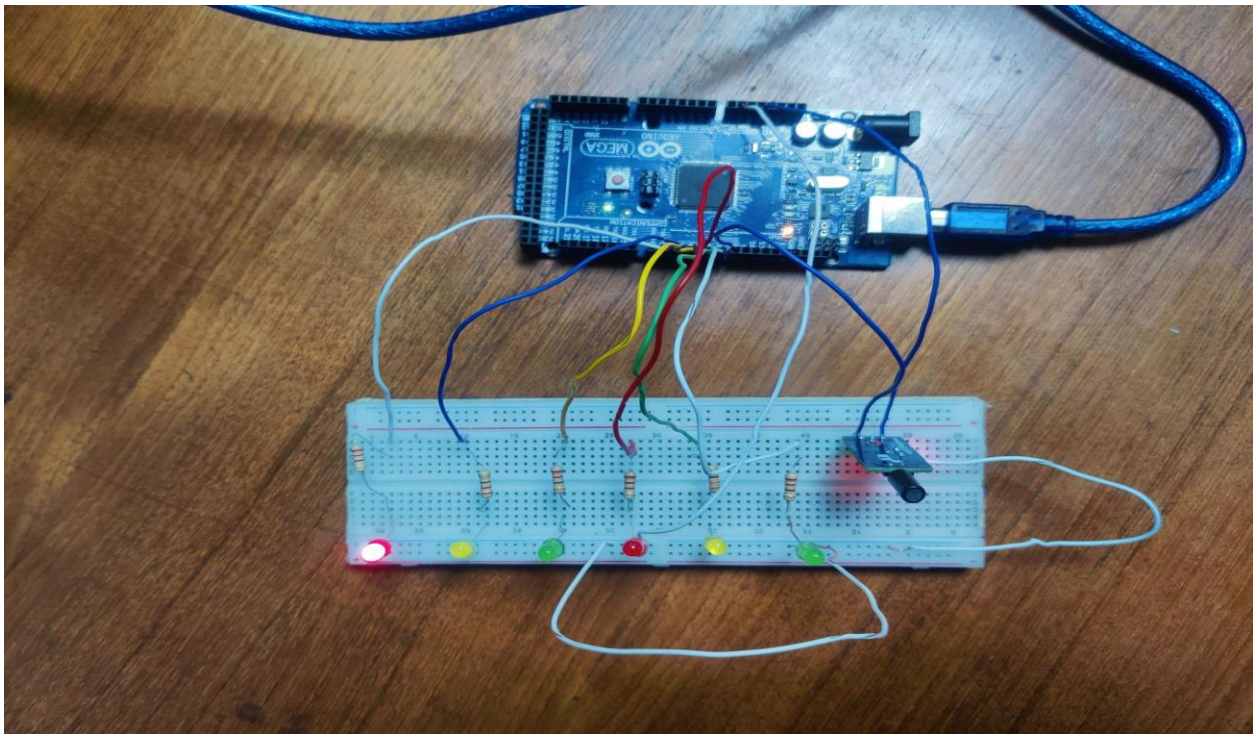

Figure: 1st LED is On

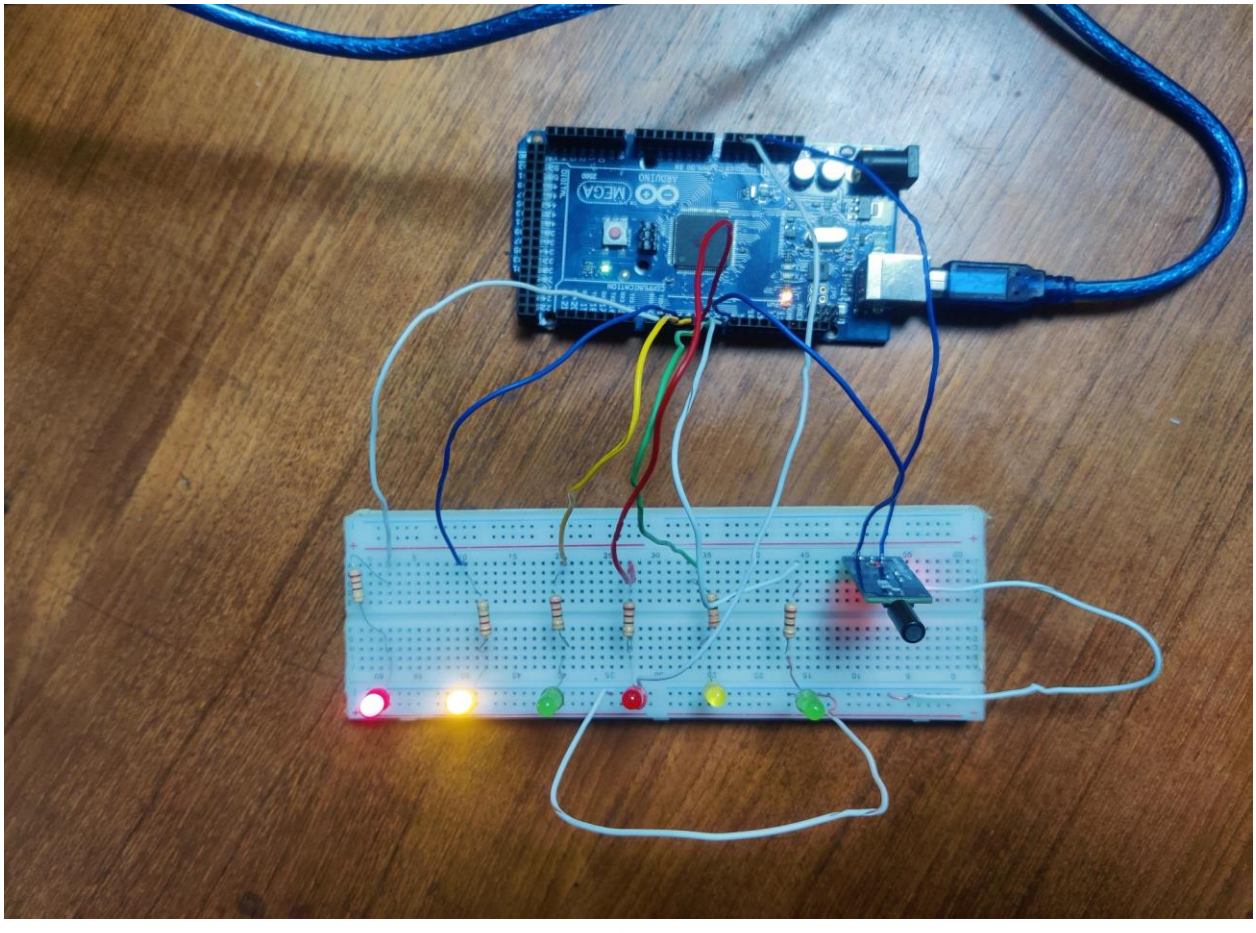

Figure: 2nd LED is On

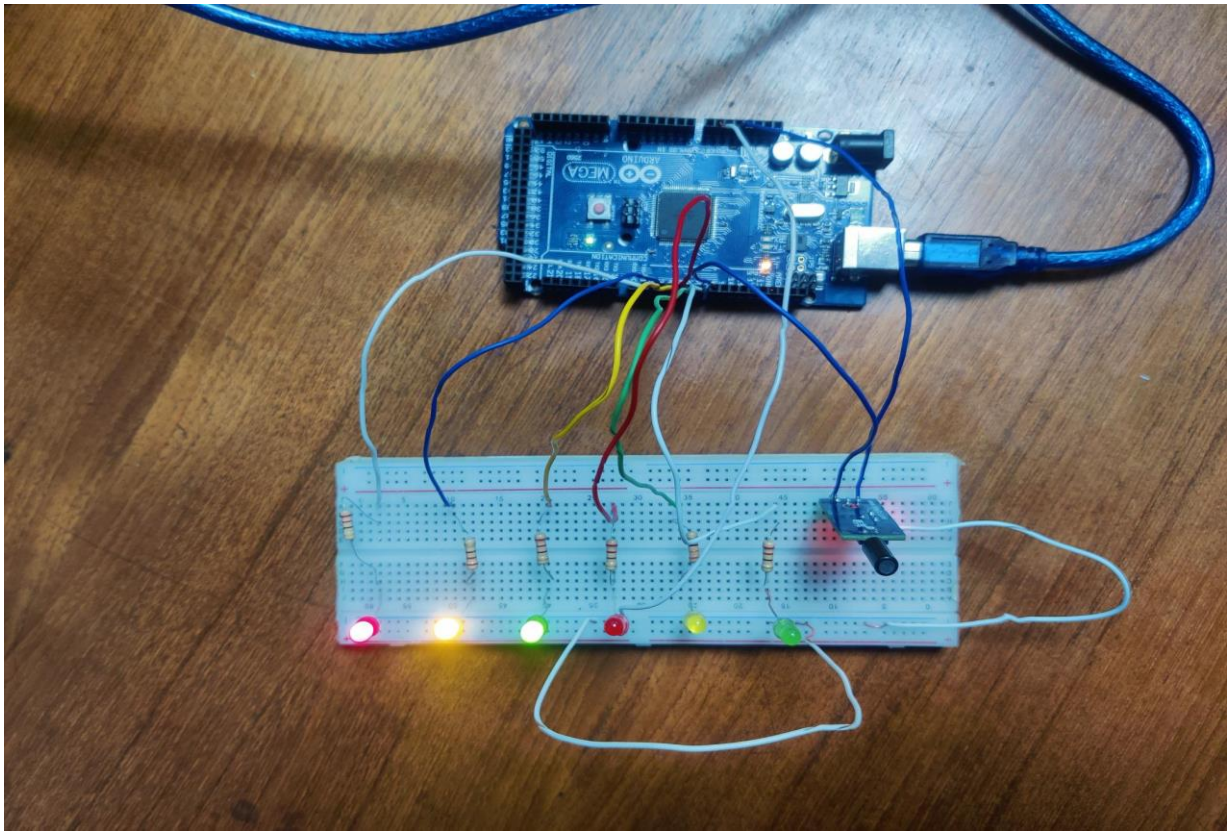

Figure: 3rd LED is On

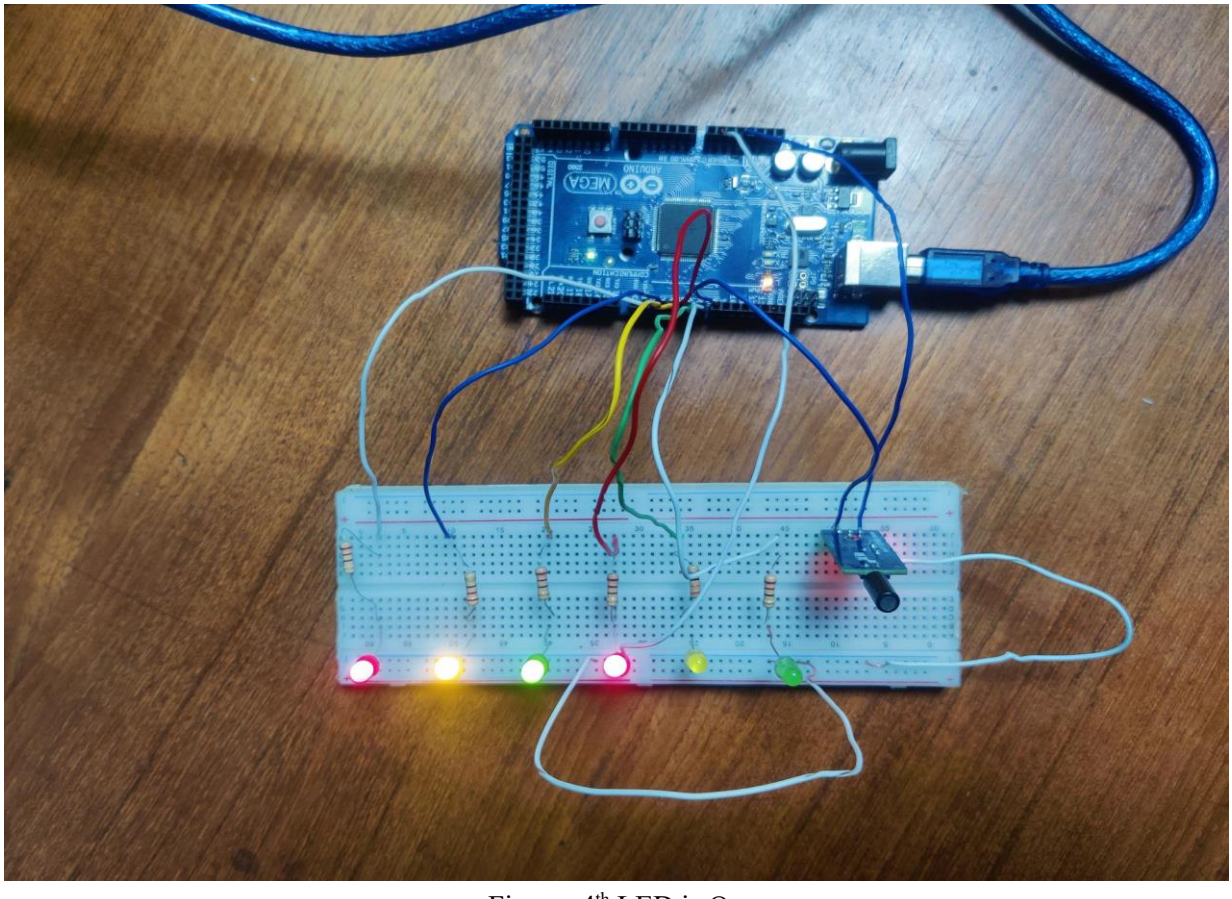

Figure: 4th LED is On

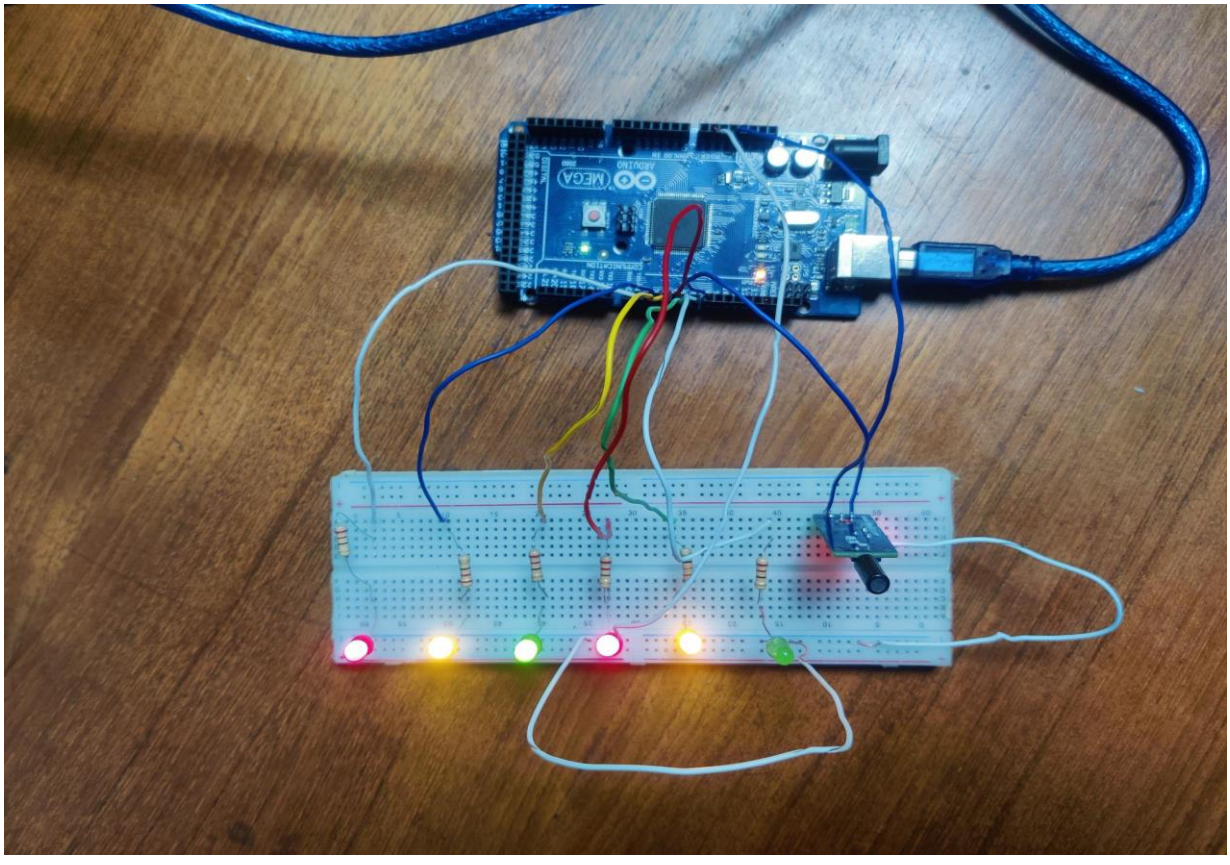

Figure: 5th LED is On

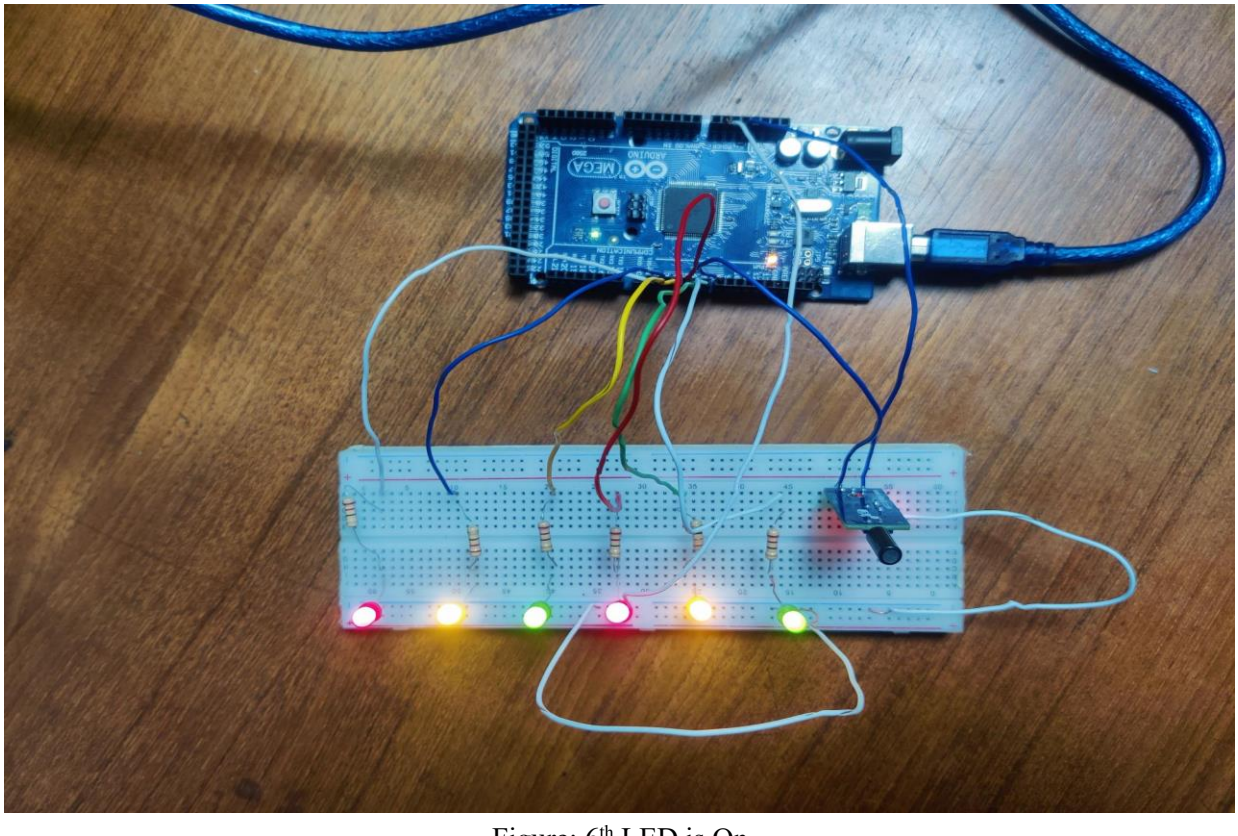

Figure: 6th LED is On

Report:

All codes, scripts and proteus simulation of the blink program and traffic light system is attached above.

Discussions :

Conducting the experiment, help us to get familiarize with the implementation of push button. In this case the push button functions as the external input. The code of the runway approach light system is written in a way that instructs the Microcontroller to reverse the light sequence when the push button is pressed. But as a consequence of bouncing, the button is generating multiple signals which is creating uncertainty in the Microcontroller. Thus, when pressing the button, light sequence does not reverse immediately. Sometimes it reverses after a long delay and other times is it does not even reverse. Performing this experiment aided to get a broaded understanding of bouncing and why

debouncing is a necessity when it comes to any kind of electronic device with physical buttons.

Conclusion :

When the switch is pressed, a set of 6 LED's will flash in a particular order, and the order will then reverse. An I/O pin will be used to connect the switch, and it will be configured as an input. The microcontroller will read the change in state of the I/O pin (here set as an input) upon dressing the switch and carry out instructions to reverse the flash sequence.