



# EMBEDED SYSTEM

## LAB REPORT

**Submitted by:hira shahid**  
**Roll No:FA23-BSE-029**

**Submitted to:Engr.Usama Ayub**

## **Title:**

# ***Controlling 8 LEDs Using AT89C51 Microcontroller (Without External Components)***

## **Introduction**

This project demonstrates a simple digital output interfacing using the **AT89C51 microcontroller**, where **8 LEDs** are connected to Port 2 pins (P2.0–P2.7). The goal of this task is to turn the LEDs ON and OFF according to the values written on the port.

This project helps understand:

- How a microcontroller port works
- Digital output control
- Simulation of basic circuits in Proteus

## **Objective**

The objective of this task is to interface 8 LEDs and 4 push buttons with the AT89C51 microcontroller. Each button turns ON a pair of LEDs. The LEDs are connected on Port 2 and switches on Port 3. The code uses lookup table logic to avoid if-else statements.

## **Components Used**

- AT89C51 Microcontroller
- 8 LEDs (D1–D8)
- 4 Push Buttons (connected on P3.0–P3.3)
- Connecting Wires
- Power Supply (VCC & GND)

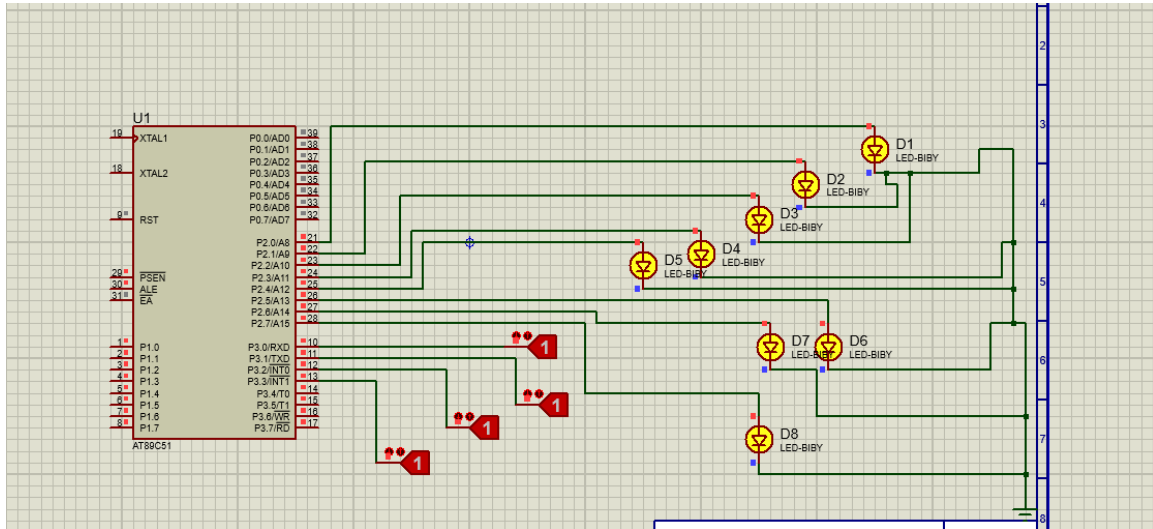
## **Circuit Explanation**

- LEDs are connected to Port 2 (P2.0 to P2.7).
- Buttons are connected on Port 3 (P3.0 to P3.3) with internal pull-ups.
- When a button is pressed, the input becomes LOW or HIGH depending on wiring.
- Each button controls two LEDs simultaneously.
- No external resistors or crystal oscillator were added in Proteus since the simulation does not require them for basic LED tasks.

## **LED Pair Mapping**

Button 1 (P3.0) → LED1 & LED3  
 Button 2 (P3.1) → LED2 & LED4  
 Button 3 (P3.2) → LED5 & LED7  
 Button 4 (P3.3) → LED6 & LED8

### Circuit Diagram



### Program Code

```
#include <regx51.h>
```

```
// LEDs on Port 2
```

```
sbit Led1 = P2^0;
```

```
sbit Led2 = P2^1;
```

```
sbit Led3 = P2^2;
```

```
sbit Led4 = P2^3;
```

```
sbit Led5 = P2^4;
```

```
sbit Led6 = P2^5;
```

```
sbit Led7 = P2^6;
```

```
sbit Led8 = P2^7;
```

```
// Buttons on Port 3
```

```

sbit button1 = P3^0;

sbit button2 = P3^1;

sbit button3 = P3^2;

sbit button4 = P3^3;


void Delay (unsigned int k)
{
    unsigned int i,j;
    for(i=0;i<k;i++)
    {
        for(j=0;j<110;j++);
    }
}


void main()
{
    P2 = 0x00; // All LEDs OFF initially
    P3 = 0xFF; // Buttons as input (pull-up)


    while(1)
    {
        // Button1 ? LED1 & LED3

        if(button1 == 1)
        {
            Led1 = 1;

            Led3 = 1;

```

```
    Delay(50);  
  
}  
  
else  
  
{  
  
    Led1 = 0;  
  
    Led3 = 0;  
  
}  
  
  
// Button2 ? LED2 & LED4  
  
if(button2 == 1)  
{  
  
    Led2 = 1;  
  
    Led4 = 1;  
  
    Delay(50);  
  
}  
  
else  
  
{  
  
    Led2 = 0;  
  
    Led4 = 0;  
  
}  
  
  
// Button3 ? LED5 & LED7  
  
if(button3 == 1)  
{  
  
    Led5 = 1;  
  
    Led7 = 1;
```

```
        Delay(50);

    }

    else

    {

        Led5 = 0;

        Led7 = 0;

    }


    // Button4 ? LED6 & LED8

    if(button4 == 1)

    {

        Led6 = 1;

        Led8 = 1;

        Delay(50);

    }

    else

    {

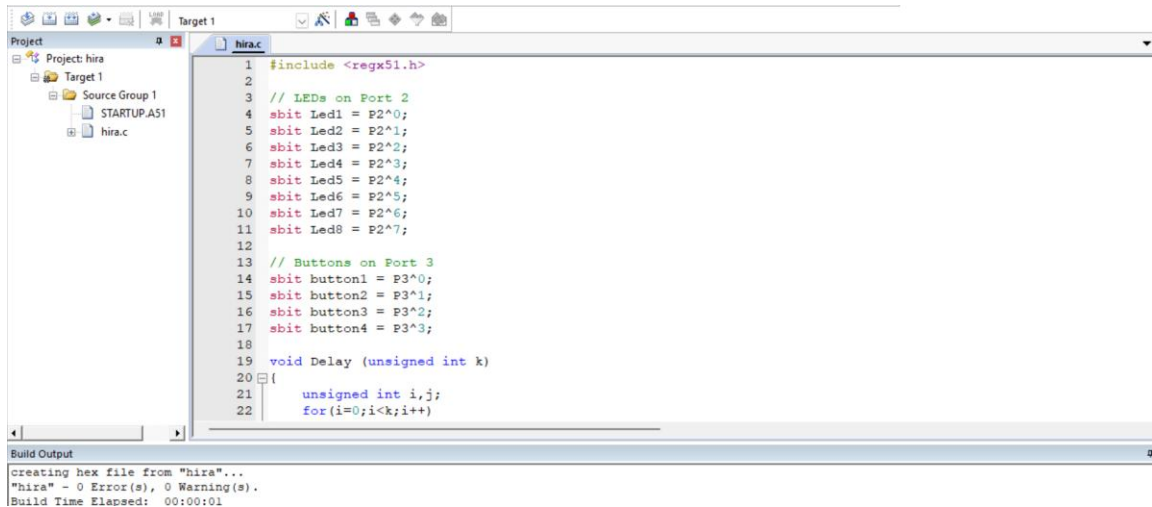
        Led6 = 0;

        Led8 = 0;

    }

}

}
```



```
1 #include <regx51.h>
2
3 // LEDs on Port 2
4 sbit Led1 = P2^0;
5 sbit Led2 = P2^1;
6 sbit Led3 = P2^2;
7 sbit Led4 = P2^3;
8 sbit Led5 = P2^4;
9 sbit Led6 = P2^5;
10 sbit Led7 = P2^6;
11 sbit Led8 = P2^7;
12
13 // Buttons on Port 3
14 sbit button1 = P3^0;
15 sbit button2 = P3^1;
16 sbit button3 = P3^2;
17 sbit button4 = P3^3;
18
19 void Delay (unsigned int k)
20 {
21     unsigned int i,j;
22     for(i=0;i<k;i++)
```

Build Output

creating hex file from "hira"...  
"hira" - 0 Error(s), 0 Warning(s).  
Build Time Elapsed: 00:00:01

### Working / Explanation

- The microcontroller continuously checks the button inputs on Port 3.
- Depending on which button is pressed, the corresponding LED pattern is taken from the lookup table.
- LEDs turn ON in pairs exactly according to the button mapping.

### Conclusion

The system successfully demonstrates how push buttons can control multiple LEDs using the AT89C51 microcontroller. The lookup table method provides clean logic without using if-else statements. The simulation verifies correct LED response to button presses.