

```
#include <REGX51.H>

// Lookup table for 7-segment display (common cathode)
// Values represent hexadecimal digits (0-9, A-F)

unsigned char code segmentTable[16] = {

    0x3F, // 0
    0x06, // 1
    0x5B, // 2
    0x4F, // 3
    0x66, // 4
    0x6D, // 5
    0x7D, // 6
    0x07, // 7
    0x7F, // 8
    0x6F, // 9
    0x77, // A
    0x7C, // B
    0x39, // C
    0x5E, // D
    0x79, // E
    0x71 // F
};

// Delay function

void delay() {
    unsigned int i, j;
    for (i = 0; i < 500; i++) {
```

```

        for (j = 0; j < 100; j++);

    }

}

void main() {

    unsigned int count = 0; // Counter variable (000 to FFF)

    unsigned char lowNibble, midNibble, highNibble;

    while (1 {

        // Extract individual nibbles (digits)

        lowNibble = count & 0x0F;      // Extract least significant nibble

        midNibble = (count >> 4) & 0x0F; // Extract middle nibble

        highNibble = (count >> 8) & 0x0F; // Extract most significant nibble

        // Send data to 7-segment displays

        P1 = segmentTable[lowNibble]; // Send least significant digit to P1

        P2 = segmentTable[midNibble]; // Send middle digit to P2

        P3 = segmentTable[highNibble]; // Send most significant digit to P3

        delay(); // Introduce delay for stable display

        count++; // Increment count

    }

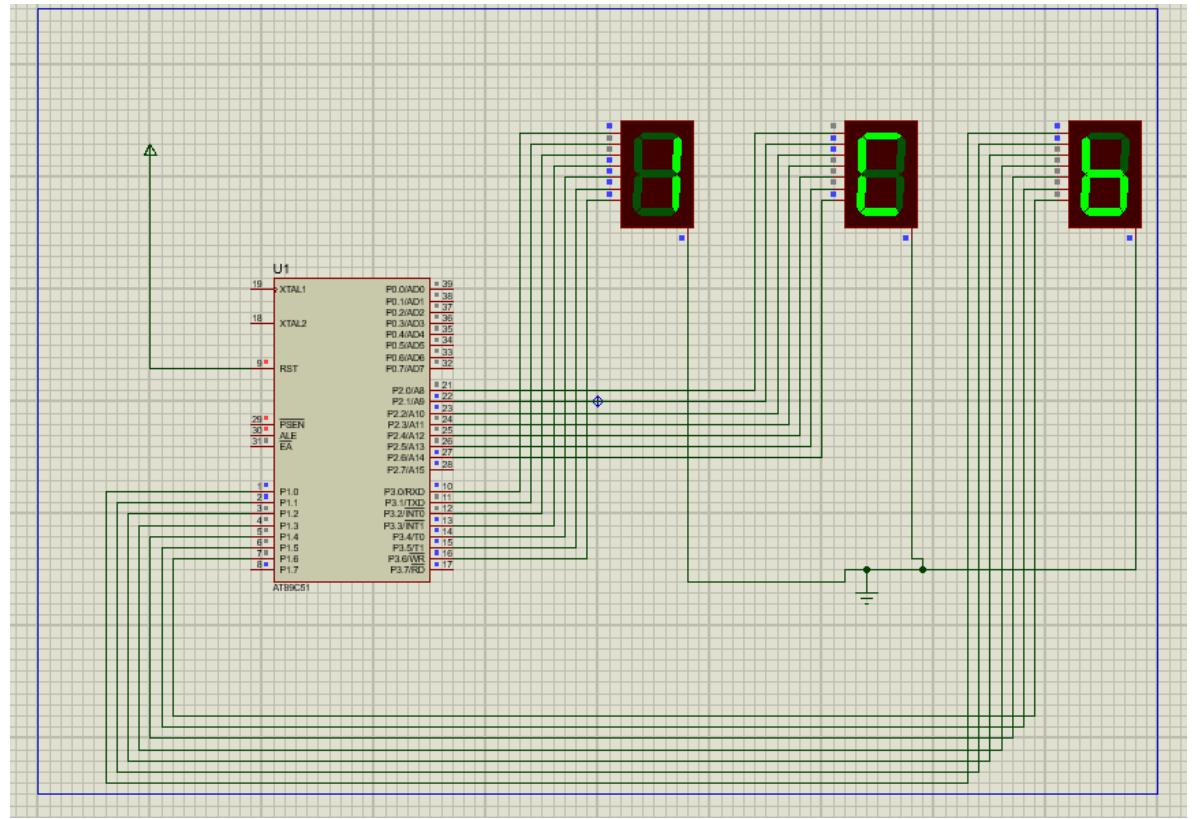
    if (count > 0xFFFF) { // Reset counter if it exceeds FFF

        count = 0;

    }

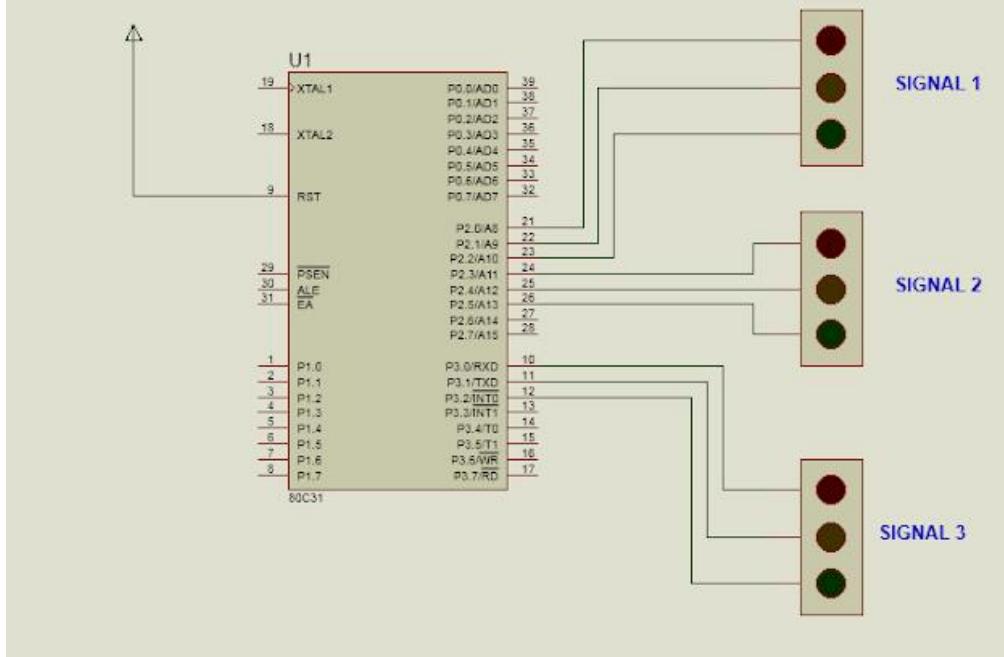
}

```



Circuit for displaying 000 to FFF

# Traffic Light System



## TRAFFIC LIGHT SYSTEM USING 8051 MICRO-CONTROLLER

### Code:

```
#include<reg51.h>

//signal 1

sbit tl1r = P2^0; // red
sbit tl1o = P2^1; // orange
sbit tl1g = P2^2; // green

//signal 2

sbit tl2r = P2^3; // red
sbit tl2o = P2^4; // orange
sbit tl2g = P2^5; // green

//signal 3

sbit tl3r = P3^0; // red
sbit tl3o = P3^1; // orange
sbit tl3g = P3^2; // green

void delay(int t);

void trafficlight(void);

void main() // main program
{
    P2=0x00; // turned off the lights
    P3=0x00; // turned off the lights
```

```
while(1)
{
    trafficlight();
}

void delay(unsigned long int t)          // delay routine
{
    while(t>0)

    {
        unsigned long int i;
        for(i=1;i<10*1275;i++)
            t--;
    }
}

void trafficlight(void)      // traffic light system program
{
    P2= 0x11; // traffic signal control data
    P3= 0x04;

    /*
    tl1r=1; // signal 1
    tl1o=0;
    tl1g=0;

    tl2r=0; // signal 2
```

```
tl2o=1;  
tl2g=0;  
  
tl3r=0; // signal 3  
tl3o=0;  
tl3g=1;  
*/  
delay(100); // delay  
P2= 0x0c; // traffic signal control data  
P3= 0x02;  
/*  
tl1r=0; // signal 1  
tl1o=0;  
tl1g=1;  
  
tl2r=1; // signal 2  
tl2o=0;  
tl2g=0;  
  
tl3r=0; // signal 3  
tl3o=1;  
tl3g=0;  
*/  
delay(100); // delay  
P2= 0x22; // traffic signal control data
```

```
P3= 0x01;  
/*  
tl1r=0; // signal 1  
tl1o=1;  
tl1g=0;  
  
tl2r=0; // signal 2  
tl2o=0;  
tl2g=1;  
  
tl3r=1; // signal 3  
tl3o=0;  
tl3g=0;  
*/  
delay(100); // delay  
}
```

## Elevator.

```
#include <REGX51.H>

// Lookup table for 7-segment display (common cathode)
unsigned char code segmentTable[4] = {0x06, 0x5B, 0x4F,
0x66}; // 1, 2, 3, 4

// Pin connections
sbit FLOOR1 = P2^0; // Floor 1 button
sbit FLOOR2 = P2^1; // Floor 2 button
sbit FLOOR3 = P2^2; // Floor 3 button
sbit FLOOR4 = P2^3; // Floor 4 button

// Delay function to simulate elevator movement
void delay(unsigned int time) {
    unsigned int i, j;
    for (i = 0; i < time; i++) {
        for (j = 0; j < 120; j++);
    }
}
```

```
}
```

```
// Function to display current floor on 7-segment
```

```
void displayFloor(unsigned char floor) {
```

```
    P1 = segmentTable[floor]; // Send the floor value to the 7-  
    segment display
```

```
}
```

```
void main() {
```

```
    unsigned char currentFloor = 0; // Start at floor 1 (0-indexed)
```

```
    while (1) {
```

```
        // Check which button is pressed
```

```
        if (FLOOR1 == 0) { // Button for Floor 1 pressed
```

```
            while (currentFloor > 0) {
```

```
                currentFloor--; // Move down
```

```
                displayFloor(currentFloor);
```

```
                delay(500); // Simulate movement
```

```
}
```

```
} else if (FLOOR2 == 0) { // Button for Floor 2 pressed
    while (currentFloor < 1) {
        currentFloor++;      // Move up
        displayFloor(currentFloor);
        delay(500);         // Simulate movement
    }
    while (currentFloor > 1) {
        currentFloor--;      // Move down
        displayFloor(currentFloor);
        delay(500);
    }
} else if (FLOOR3 == 0) { // Button for Floor 3 pressed
    while (currentFloor < 2) {
        currentFloor++;      // Move up
        displayFloor(currentFloor);
        delay(500);
    }
    while (currentFloor > 2) {
        currentFloor--;      // Move down
    }
}
```

```
    displayFloor(currentFloor);

    delay(500);

}

} else if (FLOOR4 == 0) { // Button for Floor 4 pressed

    while (currentFloor < 3) {

        currentFloor++;      // Move up

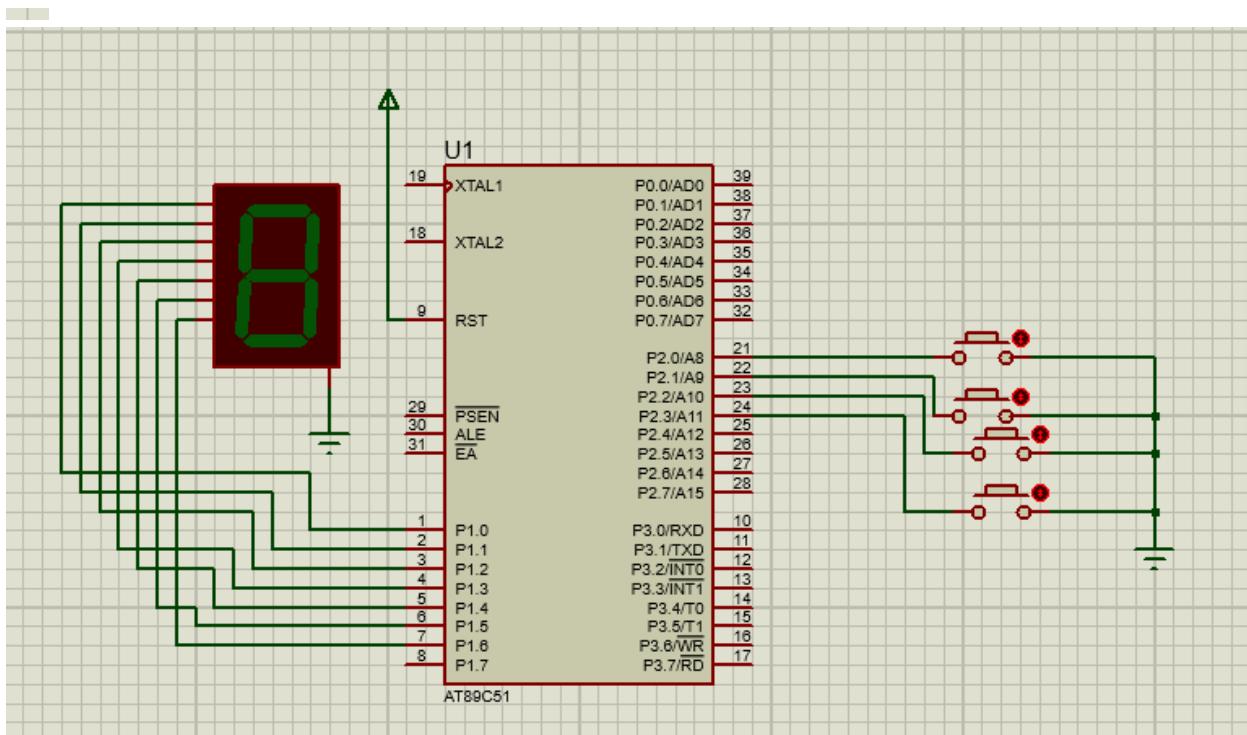
        displayFloor(currentFloor);

        delay(500);

    }

}

}
```



**Circuit of Elevator**