Complex Engineering Project

Subject:	MICROPROCESSOR AND CONTROLLER
Subject & Lab Instructor:	MAM SABA MUMTAZ
Title:	4 WAY TRAFFIC CONTROL.
Group Members:	Muhammad Zeeshan (210711) Zainab Mazhar (210715) Muhammad Abdullah (210779)

TABLE OF CONTENTS:

INTRODUCTION:

Objective:

The goal of the 4-way traffic control project is to simulate a traffic signal system at a four-way intersection. The system should efficiently control the traffic flow, preventing collisions and ensuring the smooth movement of vehicles. We also have a feature for manual control in any emergency.

AIM:

The aim of the 4-way traffic control project using the 8051 microcontroller is to simulate a traffic signal system at a four-way intersection. The project seeks to achieve the following objectives:

Traffic Management:
Create a system that effectively manages the flow of traffic at a four-way intersection, preventing collisions and ensuring a smooth movement of vehicles.
Signal Sequencing:
Implement a control logic that sequences the traffic lights in a coordinated manner, simulating the real-world operation of traffic signals.
Sensor Integration:
Integrate sensors to detect the presence of vehicles in each direction, allowing the system to respond dynamically to the actual traffic conditions.
Safety and Efficiency:
Prioritize safety by ensuring that conflicting traffic movements are appropriately controlled to avoid accidents. At the same time, strive for efficiency by minimizing unnecessary waiting times for vehicles.
Microcontroller Utilization:
Demonstrate the capabilities of the 8051 microcontroller in managing a real-time system, showcasing its role in processing inputs, making decisions, and controlling outputs.

APPLICATIONS:

The application of 4-way traffic controls is as follows.
Urban Traffic Management:
City Intersections: 4-way traffic control is commonly used in urban areas at intersections where multiple roads converge. It helps regulate the movement of vehicles and pedestrians efficiently.
Road Safety:
Intersection Safety: Traffic signals at 4-way intersections enhance road safety by providing a systematic way to control traffic movements, reducing the risk of collisions.

Traffic Flow Optimization:
Efficient Traffic Movements: Properly designed and timed 4-way traffic control systems optimize traffic flow, minimizing delays and congestion.
Pedestrian Safety:
Crosswalks and Pedestrian Signals: 4-way traffic controls often include pedestrian signals and crosswalks, enhancing safety for pedestrians crossing the intersection. **Public**
Transportation:
Bus and Transit Stops: Traffic signals at intersections facilitate the smooth movement of public transportation, such as buses, through the intersection.

8051 PROGRAMMING:

8051-microcontroller uses several programming languages but we used C-language it is easy to understand and can be used on various platforms.

SOFTWARE USED:

The software that we used is Keilu Vision.

ADVANTAGES:

The advantages of using the 8051 microcontroller are as follows.

Versatile and Popular:

The 8051 microcontroller is well-known and widely used in various applications, making it easy to find support and resources.

Integrated Features:

It comes with built-in tools like timers and communication ports, simplifying the design of devices and systems.

Cost-Effective:

8051 microcontrollers are affordable, making them suitable for projects where cost is a significant consideration.

Easy to Program:

Programming the 8051 is straightforward, and plenty of tools are available for developers in languages like C and assembly.

Low Power and Legacy Support:

It consumes minimal power, making it suitable for battery-powered devices. Plus, it has been around for a long time, ensuring ongoing support and compatibility.

FEATURE:

Integrated Peripherals:

The 8051 includes built-in peripherals such as timers, counters, and communication ports, streamlining the design of embedded systems.

Harvard Architecture:

The microcontroller follows a Harvard architecture, separating program memory and data memory, which facilitates efficient and simultaneous access to both.

On-Chip Memory:

It comes with on-chip memory (RAM and ROM), reducing the need for external memory components and simplifying system design.

Flexible Interrupt System:

The 8051 features a versatile interrupt system, enabling the microcontroller to respond to external events promptly, making it suitable for real-time applications.

Low Power Consumption:

Many versions of the 8051 are designed for low power consumption, making them suitable for battery-operated devices and applications with power constraints.

COMMANDS:

1. **Sbit:**

   It is used to assign the name to a single port so that the code will become easy to understand and can be understood easily.

2. **#define:**

   It is used to give a name to the whole register so that the registers can be differentiated easily.

3. **Unsigned int:**

   The unsigned int is a 16-bit data type

   ➢ Takes a value of 0 to 65535 (0000– FFFH).

   ➢ Define 16-bit variables such as memory addresses.

   ➢ Set counter values of more than 256.

4. **If-else statement:**

   It is a conditional statement that is used to implement conditions if the statement is true the the following statement will be executed otherwise else if statement will be executed.

5. **For loop.**

   It is used to run the statement no of times as you want.

6. **While loop.**

   The same is the case with the while loop in this project we use while(1) which means an infinity loop that will never stop

QUESTION STATEMENT:

Your task is to design an Intelligent Variable Time 4-way Traffic Controller, a microcontroller-based system that efficiently manages traffic at a 4-way intersection. Unlike traditional traffic controllers with fixed time intervals, your project should dynamically adjust signal timings based on real-time traffic intensity in each lane and optimize traffic flow.

4-WAY TRAFFIC CONTROL:

USES:

1. Efficient Urban Traffic Management:

- Optimizes traffic flow at 4-way intersections in urban areas by dynamically adjusting signal timings based on real-time conditions.

2. Intersection Efficiency:

- Improves intersection efficiency by adapting signal timings to varying traffic volumes from different directions.

3. Enhanced Road Safety:

- Prioritizes road safety through intelligent signal control, reducing the risk of accidents at intersections.

4. Public Transportation Support:

- Facilitates smooth public transportation flow with priority access at intersections.

5. Emergency Vehicle Response:

Prioritizes emergency vehicles, ensuring faster response times during critical situations.

6. Pedestrian and Cyclist Safety:

- Includes dedicated signal timings for safe pedestrian and cyclist crossings.

7. Adaptive Traffic Planning:

- Respond to changing traffic patterns, supporting adaptive traffic planning.

8. Environmental Impact Reduction:

- Reduces congestion, fuel consumption, and emissions, contributing to environmental sustainability.

9. Optimized Travel Experience:

- Minimizes waiting times at intersections for improved commute predictability.

10. Smart City Integration:

- Integrates with smart city infrastructure for data-driven decision-making.

ADVANTAGES:

1. Dynamic Traffic Optimization:

- Adjusts signal timings in real-time, minimizing congestion and optimizing traffic flow.

2. Reduced Accidents:

- Intelligent signal control lowers the risk of accidents through prioritized traffic movements.

3. Quick Emergency Response:

- Enables faster response times for emergency vehicles by providing priority at intersections.

4. Efficient Public Transportation:

- Facilitates seamless transit flow with prioritized access at intersections.

5. Pedestrian and Cyclist Safety:

- Dedicated signal timings ensure safe crossings, enhancing overall road safety.

6. Predictable Commute Times:

Minimizes waiting times, providing commuters with more predictable travel experiences.

7. Environmental Sustainability:

Reduces fuel consumption and emissions, contributing to a more sustainable urban environment.

8. Adaptive Traffic Planning:

-Responds to changing traffic patterns, supporting adaptive and efficient traffic planning.

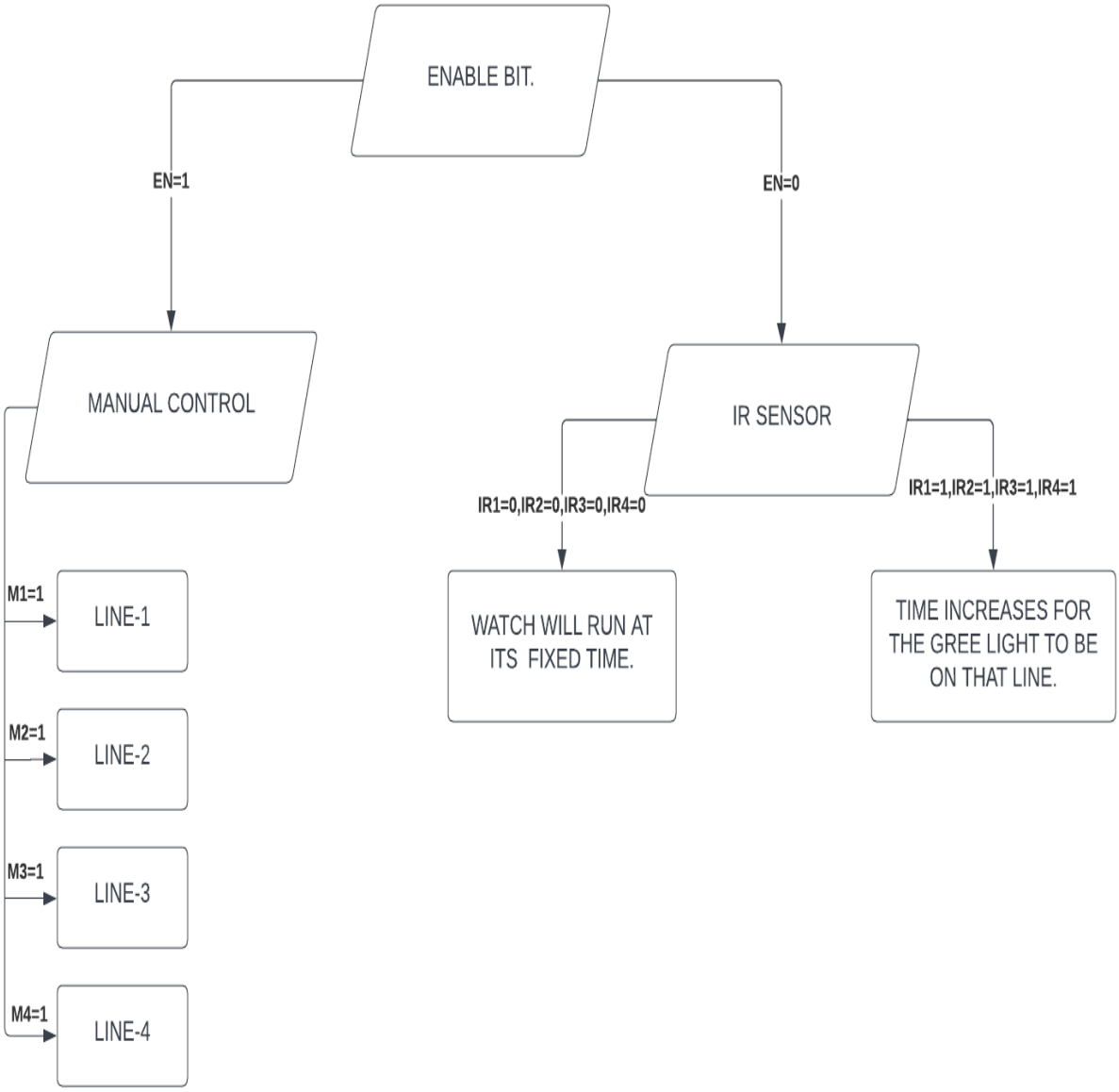
9. Efficient Road Capacity Use:

- Maximizes road capacity by preventing unnecessary delays and gridlock.

10. Smart City Integration:

- Integrates seamlessly with smart city infrastructure for data-driven decision-making.

FLOW DIAGRAM:



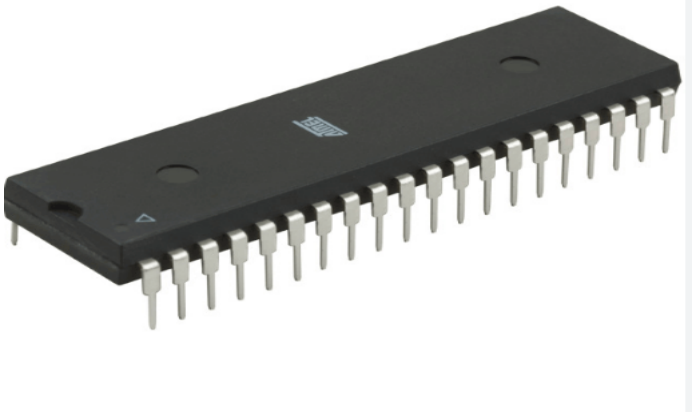
HARDWARE:

COMPONENTS:

1. 8051 MICROCONTROLLER.
2. TRAFFIC LIGHTS.
3. IR SENSORS.
4. 7-SEGMENT DISPLAY.
5. LCD.
6. 7447.
7. RESISTORS.
8. SWITCH.

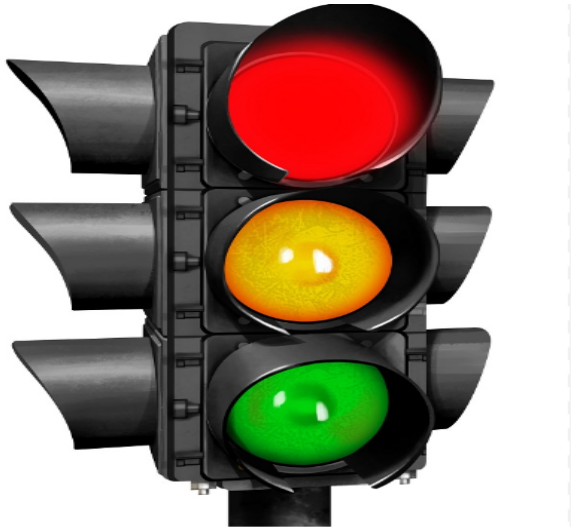
1) 8051 MICROCONTROLLER.

The 8051 microcontroller is popular and widely used in embedded systems and electronic projects. The 8051 microcontroller is an 8-bit microcontroller with a Harvard architecture, which means it has separate program and data memory spaces.



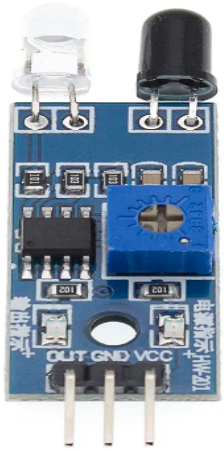
2) TRAFFIC LIGHTS.

Traffic lights, also known as traffic signals or stoplights, play a crucial role in controlling the flow of vehicular and pedestrian traffic at intersections. The system typically consists of three colored lights: red, yellow (or amber), and green. These lights are arranged vertically or horizontally to convey specific instructions to drivers and pedestrians.



3) IR SENSORS.

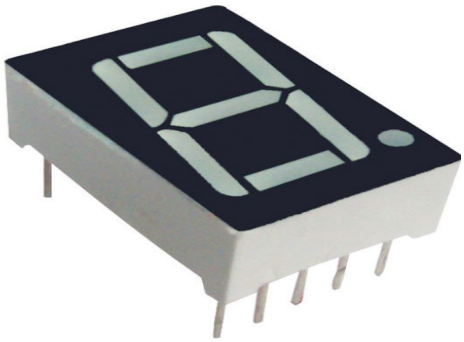
Infrared (IR) sensors are devices that use infrared radiation to detect the presence of an object or measure its distance. These sensors are commonly used in various applications, including proximity sensing, object detection, and distance measurement.



4) 7-SEGMENT DISPLAY.

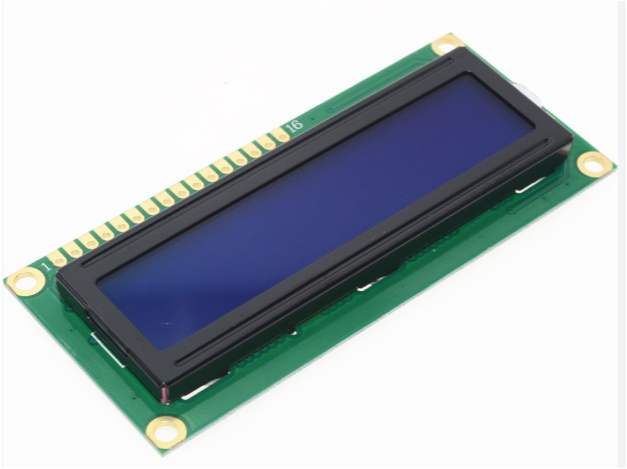
A 7-segment display is a type of electronic display device commonly used to represent decimal numbers. It consists of seven individual LED (Light Emitting Diode) segments

arranged in the shape of the digit "8," with each segment capable of being individually controlled. The segments are labeled from 'a' to 'g' and are used to display numeric characters (0-9) and some additional characters (such as letters A-F in hexadecimal).



5) LCD:

A Liquid Crystal Display (LCD) is a flat-panel display technology that uses liquid crystals to modulate the passage of light. LCDs are commonly used in a variety of electronic devices, including computer monitors, television screens, digital watches, calculators, and more.



6) 7447.

The 7447 is a BCD-to-Seven-Segment Decoder/Driver IC (Integrated Circuit). This IC is commonly used in electronic circuits to convert binary codes. Coded Decimal (BCD) input into signals that can drive a seven-segment display.

7) RESISTORS.

Resistors are fundamental electronic components that restrict the flow of electric current. They are widely used in electronic circuits for various purposes, such as controlling the amount of current, dividing voltage, providing bias, and limiting current to LEDs.

8) SWITCH.

A switch is a simple yet fundamental electronic component that controls the flow of electric current within a circuit. Its primary function is to make or break an electrical connection, allowing or interrupting the current flow.

CODE:

1ST CONTROLLER:

1) FIRST CODE STEPS:

1. In the first code main function we have set the input-output ports.
2. Then we used an infinity-while loop to run the lights continuously.
3. Then we have an if-else statement that detects the input that is the switch if it is on then the lights will be controlled manually.
4. If it is zero then it will run on its set value.
5. The output from the IR sensor is detected at any line(L-1-L-2) of the function with the larger delay will be called.

```
#include <reg51.h>

#define display_port P0
sbit rs = P3^6;
sbit rw = P3^7;
sbit e = P1^6;

sbit o1 = P1^0;
sbit o2 = P1^1;
sbit o3 = P1^2;
sbit o4 = P1^4;
sbit m1 = P1^3;
sbit en = P1^5;
```

```
sbit a = P3^0;
sbit b = P3^1;
sbit c = P3^4;
sbit d = P3^5;
sbit f = P3^6;
sbit i = P3^7;

sbit a1 = P2^0;
sbit b1 = P2^1;
sbit c1 = P2^2;
sbit d1 = P2^4;
sbit a2 = P2^3;
sbit b2 = P2^5;
sbit c2 = P1^6;
sbit d2 = P1^7;

unsigned int milliseconds = 0;
unsigned int seconds = 0;

void msdelay(unsigned char time) {
   unsigned int i;
   for (i = 0; i < time; i++) {
      TMOD = 0x10;
      TH1 = 0xFC;
      TL1 = 0x66;
      TR1 = 1;
      while (TF1 == 0);
      TR1 = 0;
      TF1 = 0;
   }
}

void icd_cmd(unsigned char command) {
   display_port = command;
   rs = 0;
   rw = 0;
   e = 1;
   msdelay(1);
   e = 0;
```

```
}

void icd_data(unsigned char disp_data) {
   display_port = disp_data;
   rs = 1;
   rw = 0;
   e = 1;
   msdelay(1);
   e = 0;
}

void icd_init() {
   icd_cmd(0x38);
   icd_cmd(0x0F);
   icd_cmd(0x01);
   icd_cmd(0x81);
}

void displayTime() {
   icd_cmd(0xC7);
   icd_data('0' + seconds / 10);
   icd_data('0' + seconds % 10);
   icd_data(':');
   icd_data('0' + milliseconds / 100);
   icd_data('0' + (milliseconds / 10) % 10);
   icd_data('0' + milliseconds % 10);
}

void string(char *p) {
   while (*p) {
      icd_data(*p++);
   }
}

void sw(unsigned int x) {
   unsigned int y = x;
         unsigned int
seg[]={0x00,0x01,0x02,0x03,0x04,0x05,0x07,0x08,0x09,0x10,0x11,0x12,0x13,0x14,0x15,0x
16,0x17,0x18,0x19,0x20};
   seconds = ((x / 1000) - 1);
```

```
   milliseconds = 1000;
   while (seconds > 0 || milliseconds > 0) {
      msdelay(100);  // Wait for 100 ms
      milliseconds -= 100;  // Increment milliseconds

      if (milliseconds >= 1000) {
         milliseconds = 900;
         seconds--;
      }
      y -= 100;
      displayTime();
                                    P2=seg[seconds];
   }
}

void ext_int0() interrupt 0 {

}

void ext_int2() interrupt 2 {
//count2=count2+1;
//    counter2(count2);
}

void control1() {
   m1 = 0;
   icd_cmd(0x01);
   icd_cmd(0x80);
   string("TRAFFIC SIGNAL");
   icd_cmd(0xC0);
   string("M-1");
   o1 = 0;
   o2 = 0;
   o3 = 0;
   o4 = 0;
   sw(12000);
   o1 = 1;
   o2 = 0;
   o3 = 0;
   o4 = 0;
```

```
    sw(2000);
}

void control2() {
    icd_cmd(0x01);
    icd_cmd(0x80);
    string("TRAFFIC SIGNAL");
    icd_cmd(0xC0);
    string("L-1");
    m1 = 0;
    o1 = 0;
    o2 = 1;
    o3 = 0;
    o4 = 0;
    sw(8000);
    o1 = 1;
    o2 = 1;
    o3 = 0;
    o4 = 0;
    sw(2000);
}

void control3() {
    icd_cmd(0x01);
    icd_cmd(0x80);
    string("TRAFFIC SIGNAL");
    icd_cmd(0xC0);
    string("M-2");
    o1 = 0;
    o2 = 0;
    o3 = 1;
    o4 = 0;
    sw(12000);
    o1 = 1;
    o2 = 0;
    o3 = 1;
    o4 = 0;
    sw(2000);
}
```

```
void control4() {
   icd_cmd(0x01);
   icd_cmd(0x80);
   string("TRAFFIC SIGNAL");
   icd_cmd(0xC0);
   string("L-2");
   m1 = 0;
   o1 = 0;
   o2 = 1;
   o3 = 1;
   o4 = 0;
   sw(8000);
   o1 = 1;
   o2 = 1;
   o3 = 1;
   o4 = 0;
   sw(2000);
}

void control5() {
   icd_cmd(0x01);
   icd_cmd(0x80);
   string("TRAFFIC SIGNAL");
   icd_cmd(0xC0);
   string("M-3");
   o1 = 0;
   o2 = 0;
   o3 = 0;
   o4 = 1;
   sw(12000);
   o1 = 1;
   o2 = 0;
   o3 = 0;
   o4 = 1;
   sw(2000);
}

void control6() {
   icd_cmd(0x01);
   icd_cmd(0x80);
```

```
    string("TRAFFIC SIGNAL");
    icd_cmd(0xC0);
    string("L-3");
    m1 = 0;
    o1 = 0;
    o2 = 1;
    o3 = 0;
    o4 = 1;
    sw(8000);
    o1 = 1;
    o2 = 1;
    o3 = 0;
    o4 = 1;
    sw(2000);
}

void control7() {
    icd_cmd(0x01);
    icd_cmd(0x80);
    string("TRAFFIC SIGNAL");
    icd_cmd(0xC0);
    string("M-4(5)");
    o1 = 0;
    o2 = 0;
    o3 = 1;
    o4 = 1;
    sw(12000);
    o1 = 1;
    o2 = 0;
    o3 = 1;
    o4 = 1;
    sw(2000);
}

void control8() {
    icd_cmd(0x01);
    icd_cmd(0x80);
    string("TRAFFIC SIGNAL");
    icd_cmd(0xC0);
    string("L-4(4)");
```

```
    o1 = 0;
    o2 = 1;
    o3 = 1;
    o4 = 1;
    sw(8000);
    o1 = 1;
    o2 = 1;
    o3 = 1;
    o4 = 1;
    sw(2000);
}

void main() {
          msdelay(100);
    icd_init();
    P1 = 0xFF;
    P0 = 0x00;
    P2 = 0x00;
    o1 = 0;
    o2 = 0;
    o3 = 0;
    o4 = 0;
    m1 = 0;
    i = 1;
    f = 1;
    EX0 = 1;
    EA = 1;
    EX1 = 1;

    while (1) {
       if (en == 1) {
          m1 = 1;
          icd_init();
          icd_cmd(0x01);
          icd_cmd(0x80);
          string("TRAFFIC SIGNAL");
          icd_cmd(0xC0);
          string("EMERGENCY");
          msdelay(100);
       }
```

```
                                    else if (en == 0) {
        m1 = 0;
        if (a == 1) {
           control1();
        } else {
           control2();
        }
        if (b == 1) {
           control3();
        } else {
           control4();
        }
        if (c == 1) {
           control5();
        } else {
           control6();
        }
        if (d == 1) {
           control7();
        } else {
           control8();
        }
     }
  }
}
```

2ND CONTROLLER:

2) SECOND CODE STEPS:

1. In this code we only have to call the function based on the inputs that are the output of the first controller.
2. In these functions led lights are controlled.

```
#include <reg51.h>

sbit s0=P3^0;
sbit s1=P3^1;
```

```
sbit s2=P3^4;
sbit s3=P3^5;
sbit s4=P3^6;

sbit g1=P0^0;
sbit g2=P0^1;
sbit g3=P0^2;
sbit g4=P0^3;
sbit y1=P0^4;
sbit y2=P0^5;
sbit y3=P0^6;
sbit y4=P0^7;

sbit r1=P2^0;
sbit r2=P2^1;
sbit r3=P2^2;
sbit r4=P2^3;

sbit z1=P2^4;
sbit z2=P2^5;
sbit z3=P2^6;
sbit z4=P2^7;

void off()
        {
         g1=0;
         g2=0;
         g3=0;
         g4=0;
         y1=0;
         y2=0;
         y3=0;
         y4=0;
         r1=1;
         r2=1;
         r3=1;
         r4=1;

 }
 void line11()
```

```
 {
        g1=1;
        g2=0;
        g3=0;
        g4=0;
        y1=0;
        y2=0;
        y3=0;
        y4=0;
        r1=0;
        r2=1;
        r3=1;
        r4=1;
 }
void line12()
 {
        g1=0;
        g2=0;
        g3=0;
        g4=0;
        y1=1;
        y2=1;
        y3=0;
        y4=0;
        r1=0;
        r2=0;
        r3=1;
        r4=1;
 }
 void line21()
 {
        g1=0;
        g2=1;
        g3=0;
        g4=0;
        y1=0;
        y2=0;
        y3=0;
        y4=0;
        r1=1;
```

```
        r2=0;
        r3=1;
        r4=1;
}
void line22()
{
        g1=0;
        g2=0;
        g3=0;
        g4=0;
        y1=0;
        y2=1;
        y3=1;
        y4=0;
        r1=1;
        r2=0;
        r3=0;
        r4=1;
}
void line31()
{
        g1=0;
        g2=0;
        g3=1;
        g4=0;
        y1=0;
        y2=0;
        y3=0;
        y4=0;
        r1=1;
        r2=1;
        r3=0;
        r4=1;
}
void line32()
{
        g1=0;
        g2=0;
        g3=0;
        g4=0;
```

```
        y1=0;
        y2=0;
        y3=1;
        y4=1;
        r1=1;
        r2=1;
        r3=0;
        r4=0;
 }
 void line41()
 {
        g1=0;
        g2=0;
        g3=0;
        g4=1;
        y1=0;
        y2=0;
        y3=0;
        y4=0;
        r1=1;
        r2=1;
        r3=1;
        r4=0;
 }
void line42()
 {
        g1=0;
        g2=0;
        g3=0;
        g4=0;
        y1=1;
        y2=0;
        y3=0;
        y4=1;
        r1=0;
        r2=1;
        r3=1;
        r4=0;
 }
```

```
void main() {
        unsigned char x;
        P3=0x00;
P3=0xFF;
P2=0x00;
P1=0xFF;
        while(1){

if(s4==1)
                        {

        if(z1==0 && z2==0 && z3==0 && z4==0)
        {
        off();
        }
        if(z1==1)
        {
        line11();
        }
        if(z2==1)
        {
        line21();
        }
        if(z3==1)
        {
        line31();
        }
        if(z4==1)
        {
        line41();
        }

        else
        {}
                        }
else

{

if(s0==0 && s1==0 && s2==0 && s3==0)
```

```
                {
                        line11();
                }
else if(s0==1 && s1==0 && s2==0 && s3==0)
                {
line12();
}
else if(s0==0 && s1==1 && s2==0 && s3==0)
                {
                        line11();

                }
else if(s0==1 && s1==1 && s2==0 && s3==0)
                {
line12();
}


else if(s0==0 && s1==0 && s2==1 && s3==0)
                {
line21();
                }
                else if(s0==1 && s1==0 && s2==1 && s3==0)
                {
line22();
                }

                else if(s0==0 && s1==1 && s2==1 && s3==0)
                {
line21();
                }
                else if(s0==1 && s1==1 && s2==1 && s3==0)
                {
line22();
                }


                else if(s0==0 && s1==0 && s2==0 && s3==1)
                {
line31();
```

```
                }
                else if(s0==1 && s1==0 && s2==0 && s3==1)
                {
  line32();
                }

                else if(s0==0 && s1==1 && s2==0 && s3==1)
                {
line31();
                }
                else if(s0==1 && s1==1&& s2==0 && s3==1)
                {
  line32();
                }


                else if(s0==0 && s1==0 && s2==1 && s3==1)
                {
line41();
                }

                else if(s0==1 && s1==0 && s2==1 && s3==1)
                {
line42();
                }
          else if(s0==0 && s1==1 && s2==1 && s3==1)
                {
line41();
                }
                                else if(s0==1 && s1==1 && s2==1 && s3==1)
                {
line42();
                }
          }
}}
```

FUNCTIONS USED:

TIMMER:

CALCULATION:

1 clock cycle=11.0592MHz.

Machine Cycle=$\frac{12}{11.0592M} = 1.085us$

Max delay=$2^{16} \times 1.085us$**=71.1ms.**

We will choose 1ms

No of Timmer increments =$\frac{1ms}{1.085us}$**=921.659.**

Starting value= $2^{16} - 921.659$**=64614.34.**

Convert into Hexadecimal= FC66

TH1=0xFC.

TL1=0x66.

1. This timer function will give us a 1 ms delay.
2. The loop is used so we can use it for the generic delay that we want to generate.

```
void ms delay(unsigned char time) {
   unsigned int i;
   for (i = 0; i < time; i++) {
      TMOD = 0x10;
      TH1 = 0xFC;
      TL1 = 0x66;
      TR1 = 1;
      while (TF1 == 0);
      TR1 = 0;
      TF1 = 0;
   }
}
```

LCD COMMAND:

1. This is the command function for LCD that is used to give command to act on LCD.
2. **Commands.**

No.	Instruction	Hex	Decimal
1	Function Set: 8-bit, 1 Line, 5x7 Dots	0x30	48
2	Function Set: 8-bit, 2 Line, 5x7 Dots	0x38	56
3	Function Set: 4-bit, 1 Line, 5x7 Dots	0x20	32
4	Function Set: 4-bit, 2 Line, 5x7 Dots	0x28	40
5	Entry Mode	0x06	6
6	Display off Cursor off (clearing display without clearing DDRAM content)	0x08	8
7	Display on Cursor on	0x0E	14
8	Display on Cursor off	0x0C	12
9	Display on Cursor blinking	0x0F	15
10	Shift entire display left	0x18	24
12	Shift entire display right	0x1C	30
13	Move cursor left by one character	0x10	16
14	Move cursor right by one character	0x14	20
15	Clear Display (also clear DDRAM content)	0x01	1

```
void icd_cmd(unsigned char command) {
   display_port = command;
   rs = 0;
   rw = 0;
   e = 1;
   ms delay(1);
   e = 0;
}
```

rs	Register to select. 0 for command and 1 for data
rw	Read/write and is set to always zero
e	Enable the LCD to start when 1.

LCD DATA:

1. This data function is used to transfer the data to the LED screen.
2. To send a string we will write another function.

```
void icd_data(unsigned char disp_data) {
   display_port = disp_data;
   rs = 1;
   rw = 0;
   e = 1;
   ms delay (1);
   e = 0;
}
```

INITIAL LCD:

1. It is the function that sets the LCD to its initial condition.
2. 0x38 is for setting LCD for two lines.
3. 0x0F is for the star the cursor to blink on the LCD.
4. 0x01 is to clear the display.
5. 0x81 to set the cursor at the first line 1 column.

```
void icd_init() {
   icd_cmd(0x38);
   icd_cmd(0x0F);
   icd_cmd(0x01);
   icd_cmd(0x81);
}
```

DISPLAY DATA:

1. This function is used to display the character or string on the screen.
2. We have used pointers to get the address of the input that is given.
3. Then while loop is used because the loop should run until the whole of the value is not displayed on the screen.
4. We use *p++ in the data function because every time we have to display the next digit on the screen.

```
void string(char *p) {
   while (*p) {
      icd_data(*p++);
   }}
```

COUNTER DOWN:

1. It is used to display the value from the given value to 0.
2. It changes value after 100 ms.
3. This is done to show the time on the LCD indicating that the green light will be closed in this much time.
4. In this function we have also shown the value of seconds on the 7-segment.
5. To display the value on seven segments we use an array.

```
void sw(unsigned int x)
 {
   unsigned int y = x;
         unsigned int

seg[]={0x00,0x01,0x02,0x03,0x04,0x05,0x07,0x08,0x09,0x10,0x11,0x12,0x13,0x14,0
x15,0x16,0x17,0x18,0x19,0x20};
   seconds = ((x / 1000) - 1);
   milliseconds = 1000;
   while (seconds > 0 || milliseconds > 0) {
      ms delay(100);  // Wait for 100 ms
      milliseconds -= 100;  // Increment milliseconds

      if (milliseconds >= 1000) {
         milliseconds = 900;
         Seconds--;

      }
      y -= 100;
      display time();
    P2=seg[seconds];
   }
}
```

DISPLAY COUNTER VALUES:

1. In this function we are converting the integer value of the counter into the ASCII value.
2. We are converting the value of every place and then adding by char zero.
3. In ASCII 0 value is 48 which is added to the integer because of starting it from zero.

4. And then this value is displayed on LCD.
5. We created this function because the controller does not read the integer value in reads the ASCII value.

```
void display time () {
   icd_cmd(0xC7);
   icd_data('0' + seconds / 10);
   icd_data('0' + seconds % 10);
   icd_data(':');
   icd_data('0' + milliseconds / 100);
   icd_data('0' + (milliseconds / 10) % 10);
   icd_data('0' + milliseconds % 10);
}
```

OUTPUT CONTROL:

1. These functions are used to control the output of the first controller used.
2. With this output we have controlled the 2nd controller so that both the controllers work at the same time.
3. We have 8 functions similar to this with different outputs.

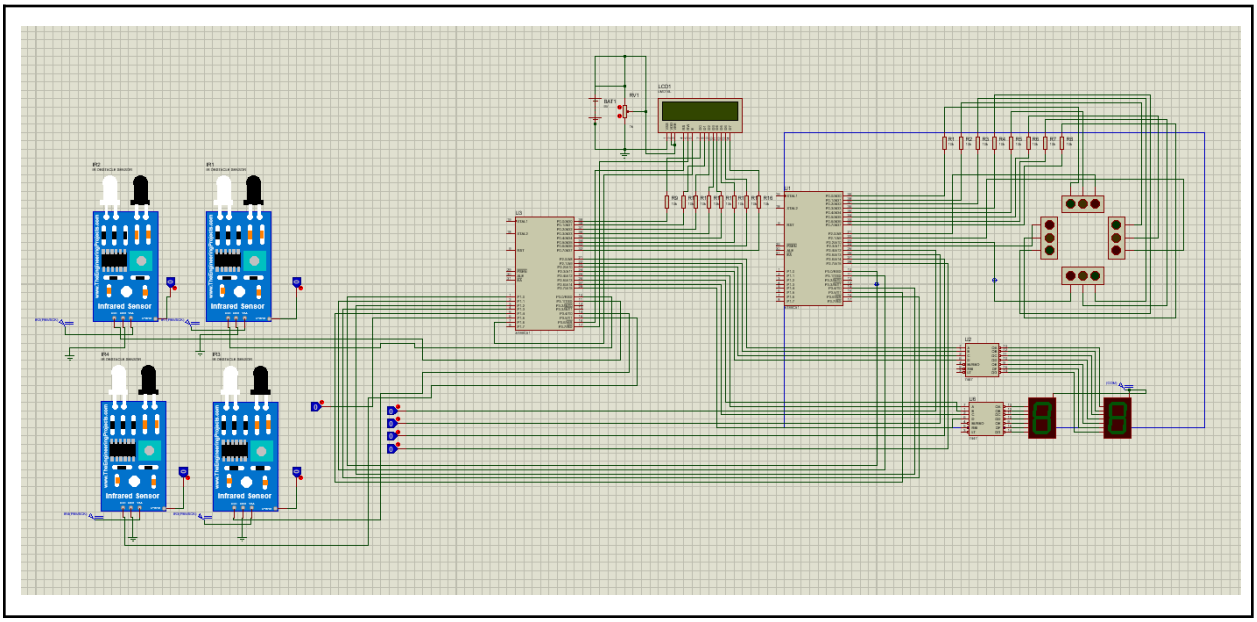
```
void control1() {
   m1 = 0;
   icd_cmd(0x01);
   icd_cmd(0x80);
   string("TRAFFIC SIGNAL");
   icd_cmd(0xC0);
   string("M-1");
   o1 = 0;
   o2 = 0;
   o3 = 0;
   o4 = 0;
   sw(12000);
   o1 = 1;
   o2 = 0;
   o3 = 0;
   o4 = 0;
   sw(2000);
}
```

LED CONTROL:

1. This is the function that is used to control the output on LEDS.
2. We have 8 functions based on a similar concept.

```
void off()
        {
         g1=0;
         g2=0;
         g3=0;
         g4=0;
         y1=0;
         y2=0;
         y3=0;
         y4=0;
         r1=1;
         r2=1;
         r3=1;
         r4=1;
 }
```

PROTEUS SIMULATION:



HARDWARE DESIGN:

Conclusion:

In conclusion, using a microcontroller enables us to create various projects. Our 4-way traffic control system adjusts signal times based on traffic intensity. We employed functions for specific outputs, loops for program repetition, and if statements to adapt signal times to sensor input. This project showcases the versatility of microcontrollers and the effectiveness of fundamental programming elements.

THE END