ISSN 2581-3463

AJMS

Asian Journal of Mathematical Sciences

RESEARCH ARTICLE

Simplified Traffic Lights Using 8051 Maxim DS89C4XX Embedded Controller (MDE)

Rotimi-Williams Bello[1], Daniel Adebiyi Olubummo[2]

[1]*Department of Mathematical Sciences, University of Africa, Toru-Orua, Bayelsa, Nigeria,* [2]*Department of Computer Science, Federal Polytechnic, Ekowe, Bayelsa, Nigeria*

Received: 20-05-2019; Revised: 10-06-2019; Accepted: 20-07-2019

ABSTRACT

Traffic lights are signaling devices positioned at road intersections, pedestrian crossings, and other locations to control flows of traffic. An actual traffic light alternates the right way of road users by displaying lights of a standard color (red, yellow/amber, and green), using a universal color code (and a precise sequence to enable comprehension by those who are color blind). In the typical sequence of colored light, (1) illumination of the green light allows traffic to proceed in the direction denoted; (2) illumination of the yellow/amber light denoting ready to proceed in the direction denoted; and (3) illumination of the red signal prohibits any traffic from proceeding. Usually, the red light contains some orange in its hue, and the green light contains some blue, for the benefit of people with red-green color blindness, and green lights in many areas are in fact blue lenses on a yellow light (which together appear green). Program was written for the MDE trainer kit to control the outputs of the microcontroller in a given sequence. Green and red light-emitting diodes (LEDs) are connected to the microcontroller outputs. First, the assembly language programs were written to turn on only one LED and then turn off the same LED. Next, the program is improved by making the LED blink, and then, the input switches are read. The walk push button and the other indicating a car at the crossing light are turned on. As switches are mechanical objects, some debounce time (timer programmed dead time of 50 ms) is also placed in the program. The light is controlled as long as each LED with one switch is pressed, and the LED is ON and when the switch is not depressed, the LED is OFF. Then, the LED will be made to blink once per second as long as the associated switch is ON. Finally, the program is improved when a subroutine is added where the traffic light controller is on green or red stays ON while the corresponding switch is ON. If more than one switch is activated, then the ON is for the red light. The LEDs simulate the traffic lights and switches simulate the walk push button and the car presence sensor at a crossroad.

Key words: Traffic light, red light, green light, yellow light, light-emitting diodes, switch, microcontroller

INTRODUCTION

The essence of traffic lights is to avert road accidents which can lead to catastrophic. The history behind the development of traffic light can be traced to the 18th century when the first gas-lit traffic lights earlier proposed by British railway engineer, J. P. Knight, were installed outside the houses of parliament in London. This was to control the traffic of horse carriages around the area and to ensure the safety of the pedestrians crossing the roads. The method of operating this gas-fuelled traffic lights involved manual operation where the controller (policeman) would have to either raise or lower the semaphore arms used for the operation during the daytime to signal moving vehicles to either stop or proceed. Gas-lit red and green lights were used at night instead of the semaphore arms where red signaled stoppage of vehicles and green signaled proceed. Red color was used to signal stop due to its danger and caution attributes while green was used to signal proceed because most cultures accepted it to be more reassuring danger-free color. Different

Address for correspondence:
Rotimi-Williams Bello,
E-mail: sirbrw@yahoo.com

negative incidents of light explosion were recorded with the use of gas-lit traffic lights; this caused so much injury to the operators and controllers of the gas-lit traffic lights making it not completely save to rely on.

The limitation of the traditional method of controlling traffic coupled with the growth in industrialization and automobiles which made cities to become crowded motivated the development of electric traffic light method of controlling traffic. This electric traffic light was conceived in 1912 by Lester Wire, an American policeman to replace the gas-lit traffic lights, but this technology lacked the yellow light, instead of the yellow light, it used buzzer sound to signal get ready status. The limitation found in the work of Lester Wire motivated another policeman named William Potts in Detroit, Michigan, to invent first four-way and three-colored traffic lights that had yellow/amber as the third color [Figure 1a].

The automated traffic lights were a huge success toward the middle of the 19th century; lights were changed by themselves at a fixed interval though causing unnecessary queuing as the light would be red even in the absence of traffic. This also led to what motivated engineers and scientists to think of a better traffic monitoring system. Charles Adler Jr. had a brief contribution to traffic system with his invented machine that could detect vehicle's honking and change signals accordingly, but the noise generated by this invention during its operation due to the unnecessary honking of vehicles, thereby causing excruciating experience to passerby and people living around the place the machine was mounted, this led to its ban.[1]

Traffic lights started to become computer-aided during the early 1960s [Figure 1b] making all the limitations experienced in the early inventions perfected. Powerful software applications were designed that could predict and control the traffic of congested cities. This paper contributes to the traffic lights technology by writing program for the MDE trainer kit to control the outputs of the microcontroller in a given sequence, thereby simplifying traffic light. The remainder of the paper is structured as follows. In Section 2, the programs involved in application simplified traffic light are shown. Laboratory experimental results and discussion of application simplified traffic light are presented in Section 3. Section 4 concludes the paper.

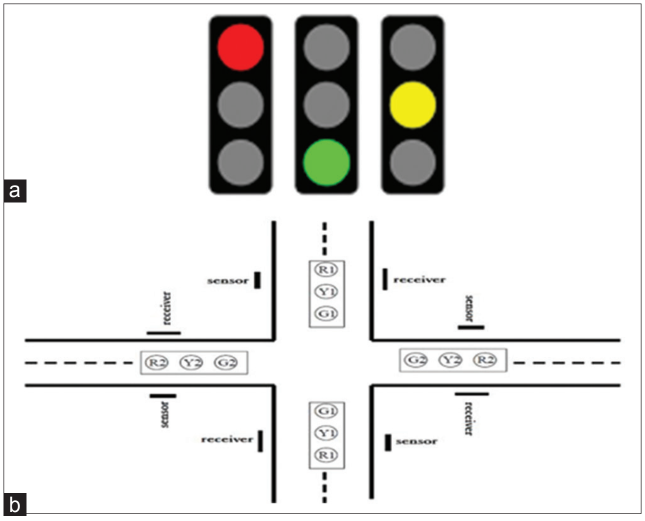

Figure 1: (a) Traffic lights, (b) Traffic light control module using sensors

APPLICATION PROGRAM TRAFFIC LIGHT

Presented in this section is a program we wrote for the application simplified traffic light. The application program traffic light is the foundation on which the application simplified traffic lights in Section 3 is based.

Simple traffic light program

- THIS PROGRAM WILL MAKE THE LED TURN OFF AND THEN ON. LED IS CONNECTED TO P1.1
  1. ORG 00H
     THE ORG OOH TELLS THE MICROCONTROLLER FROM WHERE IT SHOULD START
  2. AGAIN: CLR P1.1; NOW THE BIT P1.1 IS LOW. THIS MEANS THAT THE LED IS OFF
  3. MOV R1, #0FFH; MOVES FF HEXA I.E 255 (DECIMAL) TO R1
  4. HERE: DJNZ R1, HERE; DECREMENTS R1 AND THEN CHECKS IF R1 IS ZERO
- OTHERWISE JUMPS TO HERE AND DECREMENTS R1
  5. SETB P1.1; SETB SETS THE P1.1 AND THE LED IS NOW TURNED ON
  THE CODE BELOW IS TO ADD A LITTLE DELAY TO OBSERVE THE OUTPUT
  6. MOV R1, #0FFH
  7. HERE2: DJNZ R1, HERE2
- IMPROVE THE PROGRAM BY MAKING THE LED BLINK. THIS WILL MAKE

THE LED BLINK INDEFINITELY. LED IS CONNECTED TO P1.1

8. ORG 30H; THIS ORG 30H TELLS THE MICROCONTROLLER FROM WHERE IT SHOULD START
9. CLR P1.1; NOW THE BIT P1.1 IS LOW. THIS MEANS THAT THE LED IS OFF
10. MOV R1, #0FFH; MOV COMMAND MOVES FF HEXA I.E 255 (DECIMAL) TO R1
11. DJNZ R1, HERE; DJNZ DECREMENTS R1 AND THEN CHECKS IF R1 IS ZERO OTHERWISE JUMPS TO HERE WHERE IT DECREMENTS R1 AGAIN
12. SETB P1.1; SETB SETS THE P1.1 AND THE LED IS NOW TURNED ON
13. MOV R1, #0FFH
14. DJNZ R1, HERE2; DELAY TO OBSERVE THE OUTPUT
15. JMP AGAIN; JUMPS THE PROGRAM BACK TO LINE 07 AND THE PROGRAM CONTINUES.IN THIS III. WAY WE CAN MAKE THE LED BLINK
16. END

- THE NEXT PART OF THE LABORATORY IS TO READ THE INPUT SWITCHES, ONE FROM THE WALK PUSH BUTTON AND THE OTHER INDICATING A CAR AT THE CROSSING. AS SWITCHES ARE MECHANICAL OBJECTS; SOME DEBOUNCE TIME (DEAD TIME) WILL ALSO BE PLACED IN THE PROGRAM. EACH LED WILL BE CONTROLLED WITH ONE SWITCH; AS LONG AS THE SWITCH IS ACTIVE; THE RESPECTIVE LED IS ON AND WHEN SWITCH IS INACTIVE, THE CORRESPONDING LED IS OFF. THEN, THE LED WILL BE MADE TO BLINK AS LONG AS SWITCH IS ON
- THIS PROGRAM WILL TAKE THE INPUT FROM A SWITCH AND CONTROL THE LED

17. SW1=P1.1;SW2=P1.3; LED1=P3.1; LED2=P3.2
18. EQU; TAKES THE INPUT AND ASSIGNS IT A NAME
19. SW1 EQU P1.1
20. SW2 EQU P1.3
21. LED1 EQU P3.1
22. LED2 EQU P3.3
23. ORG 00H
24. MOV P1, #0FFH; MAKES THE PORT 1 AS INPUT PORT SO IT BEHAVE AS INPUT
25. MOV P3, #00H; MOVES ALL ZEROS TO PORT 3 MAKES THEM TO BEHAVE AS OUTPUTS
26. CHKSW1: JNB SW1, CHKSW2; IF SWITCH 1 IS OFF P1.1 IS LOW THEN CHECK IF SWITCH 2 IS ON
27. CLR LED2; TO SWITCH OFF LED IF IT WAS ON
28. CPL LED1; MAKE THE LED BLINK. CPL IS THE COMMAND FOR COMPLIMENT WILL INVERT THE LED FROM ITS PREVIOUS STATE IF THE LED IS ON THEN NOW IT WILL BE OFF AND VICE VERSA
29. MOV R1, #0A0H
30. DJNZ R1, HERE

- CALL DEADTIME; CALL A SUBROUTINE "DEADTIME." AS THIS WILL BE USED A COUPLE OF TIMES SO IT IS BETTER TO MAKE SUBROUTINE RATHER THAN WRITING THE CODE AGAIN AND AGAIN

31. JMP CHKSW1; THIS CHECKS WHETHER THE SW1 IS STILL ON OR NOT IF YES IT WIL MAKE THE LED BLINK
32. CHKSW2: JNB SW2, CHKSW1
33. CLR LED1; TO WITCH THE LED1 OFF IT WAS STILL ON
34. CPL LED2; MAKE THE LED BLINK. CPL IS THE COMMAND FOR COMPLIMENT WILL INVERT THE LED FROM ITS PREVIOUS STATE IF THE LED IS ON THEN NOW IT WILL BE OFF AND VICE VERSA
35. MOV R1, #0A0H
36. DJNZ R1, HERE2
37. CALL DEADTIME; CALL A SUBROUTINE "DEADTIME." AS THIS WILL BE USED A COUPLE OF TIMES SO IT IS BETTER TO MAKE SUBROUTINE RATHER THEN WRITING THE CODE AGAIN AND AGAIN
38. JMP CHKSW2; THIS CHECKS WHETHER THE SW2 IS STILL ON OR NOT IF YES IT WIL MAKE THE LED BLINK
39. DEADTIME

```
MOV R1, #0AH; MOVES 10 HEXA TO R1
MOV R2, #0FFH; MOVES FFHEXA TO R2
BACK: DJNZ R2, BACK; THIS IS CALLED THE INNER LOOP IT WILL DECREMENT R2 255 TIMES BEFORE GOING TO THE NEXT STATEMENT
DJNZ R1, AGAIN; THIS IS THE OUTER LOOP ONCE THE INNER LOOP IS COMPLETED IT DECREMENTS BY 1. THIS MAKES THE INNER LOOP RUN 10 TIMES
RET; THIS COMMAND TELLS THE MICROCONTROLLER TO RETURN TO THE LAST COMMAND
```

- CHECK: CHECKING INPUT OF SWITCHES.

```
ORG 0000H
MOV R0, #SW1; MOVING SWITCH1 INPUT IN R0
MOV R1, #SW2; MOVING SWITCH2 INPUT IN R1
CPL SW1
CPL SW2
JMP RED
JMP GREEN
JMP CHECK
GREEN
LOOP1
CLR P1.1
CALL DELAY
SETB P1.1
CALL DELAY
JMP LOOP1
RED
LOOP
CLR P1.3
CALL DELAY
SETB P1.3
CALL DELAY
JMP LOOP
DELAY
MOV R7, #100
L1_DELAY
DJNZ R7, L1_DELAY
RET
END.
```

Simple traffic light program

```
#include<reg51.h>
void timer()
{
TF0=0;
TMOD=0X01;
TH0=0X3C;
TL0=0X60;
TR0=1;
while(TF0==0);
TF0=0;
}
void delay(unsigned int sec)
{
unsigned int i,j;
for(i=0;i<=sec;i++)
for(j=0;j<=10;j++)
{
timer();
}
}
void disp(unsigned int a)
{
int j;
unsigned int i[]={0XC0,0XF9,0XA4, 0XB0, 0X99, 0X92,0X82, 0XF8,0X80,0X98};
for(j=a;j>=0;j--)
{
P1=i[j];
delay(1);
}
}
void main()
{
while(1)
{
P2=0x87;
P3=0xFF;
disp(9);
P3=0x30;
disp(2);
P2=0x4b;
P3=0xFF;
disp(9);
P3=0x90;
disp(2);
P2=0x2D;
P3=0xFF;
disp(9);
P3=0xC0;
disp(2);
P2=0x1E;
P3=0xFF;
disp(9);
P3=0x60;
54. disp(2);
55. }
56. }
```

EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we present the application simplified traffic light. Microprocessor and its various electronic components used in developing the traffic light were diagrammatically illustrated and discussed. A light-emitting diode (LED), a semiconductor light source was used in carrying out the experiment [Figure 2]. The color of the light is determined by the energy gap of the semiconductor. When a LED is forward biased, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons. This effect is called electroluminescence (EL). EL is an optical and electrical phenomenon, in which a material emits light in response to the passage of an electric current or to a strong electric field. The wavelength of the light emitted, and thus, its color depends on the band gap energy of the materials forming the p-n junction. The materials used for the LED have a direct band gap with energies corresponding to near-infrared, visible, or near-ultraviolet light.

Seven-segment display

A seven-segment display is an electronic display device for displaying decimal numerals. A seven-segment display is composed of seven elements. Individually on or off, they can be combined to produce simplified representations of the Arabic numerals.
The set values and the selected time intervals are shown on the seven-segment display [Figure 3]. There are two types of displays available. One is common anode type display and the other is common cathode type display. In common cathode type display, all the cathodes of the segments are tied together and connected to ground [Figure 4]. The supply will be given to the required segment from the decoder or driver.
In common anode type display, the anodes of all the segments are tied together and connected to supply and the required segments will be connected to ground from the decoder or driver [Figure 5].

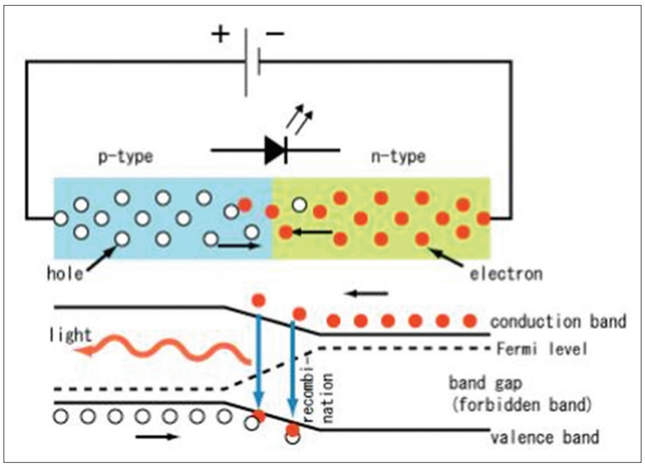

Figure 2: Traffic light application using light-emitting diode

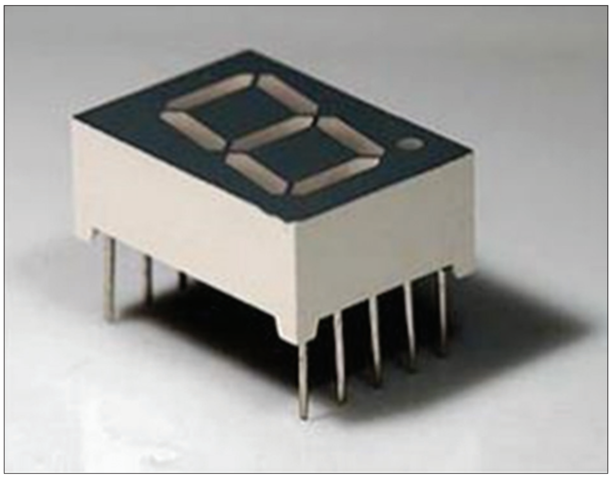
Figure 3: A seven-segment display

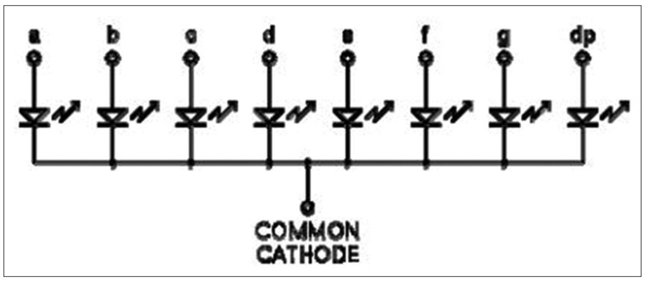

Figure 4: Cathode type display

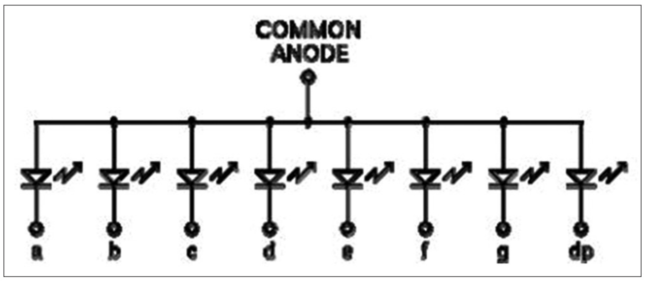

Figure 5: Anode type display

Port 1 is used for the seven-segment data. The seven segments are arranged as a rectangle of two vertical segments on each side with one horizontal segment on the top, middle, and bottom. In addition, the seventh segment bisects the rectangle horizontally. In a simple LED package, typically, all of the cathodes (negative terminals) or all of the anodes (positive terminals) of the segment LEDs are connected together and brought out to a common pin; this is referred to as a "common cathode" or "common anode" device. Hence, a seven-segment plus decimal point package will only require nine pins.[2-5]

A single byte can encode the full state of a seven-segment display. The most popular bit encodings are gfedcba and abcdefg – both usually assume 0 is off and 1 is on [Table 1]. Figure 6 gives the hexadecimal encodings for displaying the digits 0–9.

The timer of microcontroller is interfaced with seven-segment display to display the delay of light. The decoder enhances the capability of accommodation for more number of seven-segment displays with the same number of port pins. The current-limiting resistor associated with each segment limits the current at the cost of illumination. The drop across each segment will be 2 V approximately. The maximum current that the segment can handle is 10 mA.

Current drawn by segment = (Supply voltage–Drop across segment)/Resistance = (5 v–2 v)/ 1k = 3 mA (1)

This proximity detector using an infrared detector shown in Figure 5 can be used in various equipment like alarm devices. The circuit primarily consists of an infrared transmitter and an infrared receiver. The transmitter section consists of a 555 timer IC functioning in a stable mode. It is wired as shown in Figure 7. The output from a stable is fed to an infrared LED through resistor R4, which limits its operating current. This circuit provides a frequency output of 38 kHz at 50% duty cycle, which is required for the infrared detector/receiver module [Figure 8].

The receiver section comprises an infrared receiver module, a 555 monostable multivibrator, and an LED indicator. On reception of infrared signals, 555 timer (mono) turns on and remains on as long as infrared signals are received. When the signals are interrupted, the mono goes off after a few seconds (period = 1.1 R7 × C6) depending on the value of R7–C6 combination. Thus, if R7 = 470 kΩ and C6 = 4.7 µF, the mono period will be around 2.5 s.[6,7]

Both the transmitter and the receiver parts can be mounted on a single breadboard or printed circuit board. The infrared receiver [Figure 9] must be placed behind the infrared LED to avoid false indication due to infrared leakage. An object moving nearby actually reflects the infrared rays emitted by the infrared LED.

The infrared receiver has a sensitivity angle (lobe) of 0–60°, hence, when the reflected IR ray is sensed, the mono in the receiver part is triggered. The output from the mono may be used in any desired fashion. For example, it can be used to turn on a

Table 1: Hexadecimal reference for seven segments LED light on/off functions

Digit	gfedcba	abcdefg	a	b	c	d	e	f	g
0	0X3F	0X7E	On	On	On	On	On	On	Off
1	0X06	0X30	Off	On	On	Off	Off	Off	Off
2	0X5B	0X6D	On	On	Off	On	On	Off	On
3	0X4F	0X79	On	On	On	On	Off	Off	On
4	0X66	0X33	Off	On	On	Off	Off	On	On
5	0X6D	0X5B	On	Off	On	On	Off	On	On
6	0X7D	0X5F	On	Off	On	On	On	On	On
7	0X07	0X70	On	On	On	Off	Off	Off	Off
8	0X7F	0X7F	On	On	On	On	On	On	On
9	0X6F	0X7B	On	On	On	On	Off	On	On

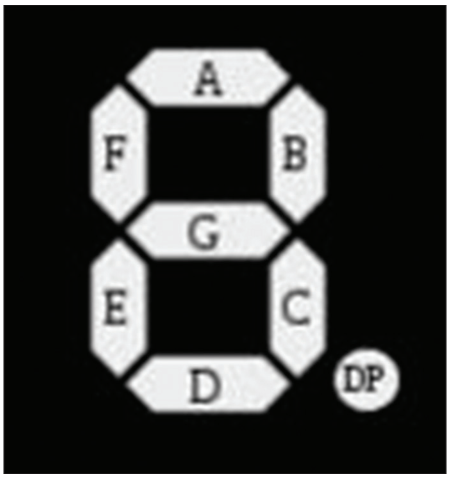

Figure 6: A seven-segment display

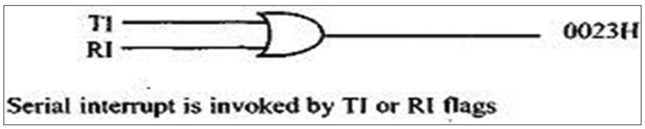

Figure 7: Serial interrupt example

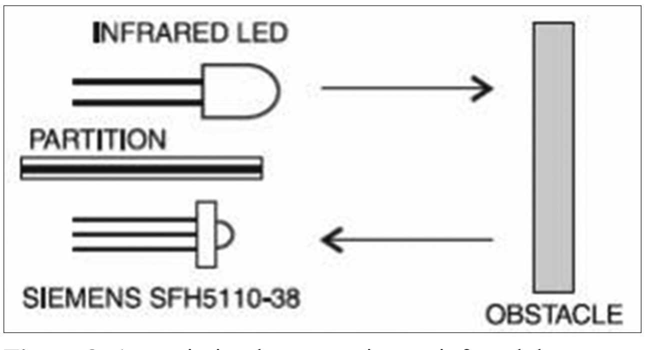

Figure 8: A proximity detector using an infrared detector

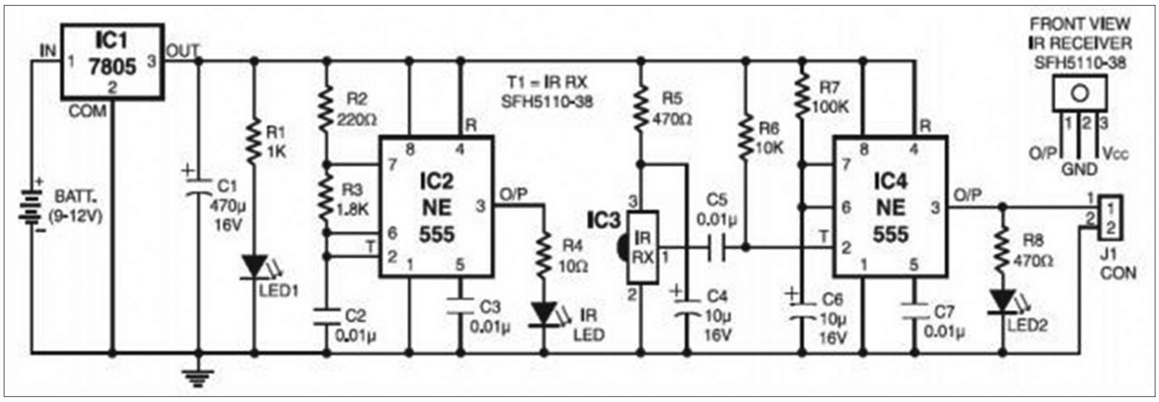

Figure 9: IR receiver circuit

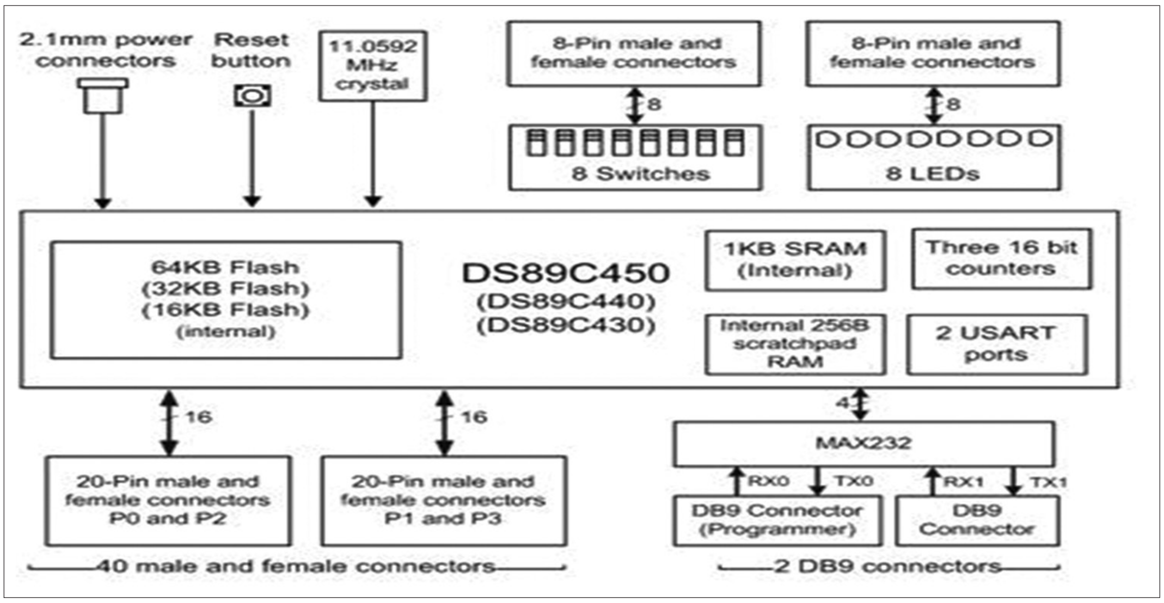
2.1mm power connectors
Reset button
11.0592 MHz crystal
8-Pin male and female connectors
8 Switches
8-Pin male and female connectors
8 LEDs
64KB Flash (32KB Flash) (16KB Flash) (internal)
DS89C450 (DS89C440) (DS89C430)
1KB SRAM (Internal)
Three 16 bit counters
Internal 256B scratchpad RAM
2 USART ports
MAX232
RX0
TX0
RX1
TX1
20-Pin male and female connectors P0 and P2
20-Pin male and female connectors P1 and P3
DB9 Connector (Programmer)
DB9 Connector
40 male and female connectors
2 DB9 connectors

Figure 10: 8051 MDE trainer board architecture descriptions

light when a person comes nearby by energizing a relay. The light would automatically turn off after sometime as the person moves away and the mono pulse period is over. The sensitivity of the detector depends on current-limiting resistor R4 in series with the infrared LED. Range is approximately 40 cm. For 20-ohm value of R4, the object at 25 cm can be sensed, while for 30-ohm value of R4, the sensing range reduces by 22.5 cm.

CONCLUSION

This article has exposed the use of microprocessor and various electronic components used in developing an embedded system found in traffic lights. We studied the application of 8051 microcontroller to traffic lights. The basic 8051 has two on-chip timers [Figure 10] that can be used for timing durations or for counting external events. Interval timing allows the programmer to perform operations at specific instants in time. For example, in our LED flashing program, the LED was turned on for a specific length of time and then turned off for a specific length of time. We achieved this through the use of time delays. Since the microcontroller operates at a specific frequency, we could work out exactly how many iterations of the time delay were needed to give us the desired delay. However, this is cumbersome and prone to error. Moreover, there is another disadvantage; the central processing unit (CPU) is occupied, stepping through the loops. If we use the on-chip timers, the CPU could be off doing something more useful while the timers take on the menial task of keeping track of time.
For the control and program of the serial port of the microcontroller in a given sequence, we left that as future work.

REFERENCES

1. Mazidi MA, Mazidi JG, McKinlay RD. The 8051 Microcontroller and Embedded Systems: Using

Assembly and C. Vol. 626. Upper Saddle River, NJ: Pearson/Prentice Hall; 2006.
2. Ayala KJ. The 8051 Microcontroller. San Francisco, CA: Cengage Learning; 2004.
3. Singh BP, Singh R. Advanced Microprocessors and Microcontrollers. New Delhi: New Age International; 2008.
4. Yashwant K. Let us C. Array and Pointers. 7th ed. BPB Publication; 1999.
5. Mano MM. Digital Design. United States: EBSCO Publishing, Inc.; 2002.
6. Deshmukh AV. Microcontrollers: Theory and Applications. New York: Tata McGraw-Hill Education; 2005.
7. Predko M. Programming and Customizing the 8051 Microcontroller. New York: McGraw-Hill, Inc.; 1999.