337356

Embedded C Programming

A INTERNSHIP REPORT

Submitted by

Tirth Parmar

200170111040

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

In

Electronics and Communication

Vishwakarma Government Engineering College, Chandkheda

Gujarat Technological University, Ahmedabad

July, 2023

337356

Vishwakarma Government Engineering College,
Nr. Visat three roads, Sabarmati-Koba highway, Chandkheda,
Ahmedabad, Gujarat, India-382424

CERTIFICATE

This is to certify that the internship report submitted along with the internship entitled **Embedded C Programming** has been carried out by **Tirth Parmar** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in Electronics and Communication, 7th Semester of Gujarat Technological University, Ahmadabad during the academic year 2023-24.

Date: 16/09/2023

Prof. Rahul. M. Patel
Internal Guide

Prof. Arun B. Nandurbarkar
Head of Department

Enriching Skills
An ISO 9001:2015 Company

Certificate of Internship

This certificate is presented to

Tirth Mukeshkumar Parmar

for completion of 15 days Industrial Internship on Embedded C Technology from 27th July 2023 to 10th August 2023

10th August 2023
Date of Issue

Signature
Head- Training & Project

Sofcon India Pvt Ltd
CIN NO. : U30007DL2002PTC115673
405, 4th Floor, Sukhsagar Complex,
Near Hotel Fortune Land Mark, Usmanpura,
Ashram Road, Ahmedabad - 380013 Gujarat
9227185900

Certificate No. : SIPL/AHM/23/ A114

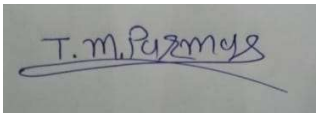
337356

Vishwakarma Government Engineering College,
Nr. Visat three roads, Sabarmati-Koba highway, Chandkheda,
Ahmedabad, Gujarat, India-382424

DECLARATION

We hereby declare that the Internship report submitted along with the Internship entitled **Embedded C Programming** submitted in partial fulfillment for the degree of Bachelor of Engineering in **Electronics and Communication** to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me at **Sofcon India Pvt. Ltd.** under the supervision of **Jigna Dave** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Tirth Parmar



ACKNOWLEDGEMENT

I wish to express my sincere gratitude to both of my internal and external mentors Prof. Rahul. M. Patel and Ms. Jigna Dave from Sofcon India Pvt. Ltd. and all the faculty members for helping us through my internship by giving me the necessary suggestions and advices along with their valuable co-ordination in completing this internship.

I also thank our parents, friends and all the members of the family for their precious support and encouragement which they had provided in completion of my Internship. In addition to that, I would also like to mention the college personals who gave me the permission to use and experience the valuable resources required for the Internship from the college premises.

Thus, In conclusion to the above said, I once again thank the faculties and members of Vishwakarma Government Engineering College for their valuable support in completion of the Internship.

Thank You,
Tirth Parmar.

ABSTRACT

This internship report presents a comprehensive overview of the experiences and learning gained during a July to August internship period focused on C and embedded C programming. The primary objective of the internship was to acquire practical skills and insights into software development, with a specific emphasis on programming for embedded systems.

The report begins with an introduction to the importance of C and embedded C programming in the context of modern technology and its applications in various industries. It highlights the significance of efficient and optimized programming for resource-constrained embedded systems.

Furthermore, the conclusions drawn from the internship experience underscore the significance of strong foundational knowledge in C programming and its extension to embedded systems. The hands-on exposure to debugging techniques, code optimization, and hardware-software integration has deepened the understanding of software development processes for resource-constrained environments.

In conclusion, this internship report serves as a documentation of the gained expertise and insights in C and embedded C programming. It underscores the practical relevance of these skills and their applicability in the development of efficient, responsive, and robust embedded systems.

ABOUT COMPANY

Sofcon Training is leading industrial training institutes in India providing 100% placement Assistance. Sofcon provides NSDC affiliated certificate to each successful participant, which is recognized across industries and helps in achieving one's career goal.

We have been delivering professional hands on training services for the last two decades. Sofcon Training has dedicated teams of placement professionals for catering placement support services to each participant. We provide companies well trained work force that helps them to improve productivity, quality and reduce maintenance cost.

Benefits of our Training programs:

a) 100% Job oriented Industrial trainings with Advanced lab.

b) 2/4/6 Week Summer / Winter / Project / In-Campus Training (with Live Project)

c) 3/6 months Industrial Training with Industrial Visit.

d) In-campus / In-house Customized training for Colleges / Institutions.

e) In-Plant / Corporate Customized Training for companies / working professionals

f) Live projects & Onsite Exposure (Industrial visit)

g) Fully equipped lab with advance technologies

h) Mock test and online assessment

i) Industry Aligned Industrial Training

j) Personality Development and soft skill session

Mission & Vision

- **Our Mission:** To be the most successful organization delivering best hands-on training in the market we serve.
- **Our Vision:**

  :• To be the Brand bridging gap between Industry & Academia.

  • To be a workplace where everyone is inspired to be the best they can be.

  • To inspire innovation, learning, creativity.

  • To be a responsible, effective, dynamic and fast-moving organization.

Company Website: https://www.sofcontraining.com/

Table of Contents

ACKNOWLEDGEMENT……5

ABSTRACT……6

ABOUT COMPANY……...7

Chapter 1 Introduction Of C and Embedded C……..…12

1.1 What is C Programming ??.....12

1.2 Application of C ……12

1.3 What is Embedded C ??.....13

1.4 Benefits of Embedded C ……14

1.5 Industrial Applications of Embedded C……..14

1.6 Comparison Of C and Embedded C……15

Chapter 2 C Programming……...16

2.1 Basics Of C……16

2.2 C Data types……17

2.3 C input/Output (I/O)……17

2.4 C Operators……18

2.5 C Statements ……...18

2.6 C Loops……...19

2.6.1 for loop……...19

2.6.2 While Loop……20

2.6.3 do while loop……..21

2.7 C break and continue……...22

2.7.1 C break……...22

2.7.2 C continue……..22

2.8 C functions……..22

2.9 C Arrays……..23

2.10 C Pointers……..23

2.11 C strings……24

Chapter 3 Embedded C ……...……25
3.1 LED Blinking……..……25
3.1.1 Display LED linking with 0&1……..……25
3.1.2 Display combination of LEDs with Blink,Toggle,on off and Sequential ……..26
3.2 Seven-Segment Display……..30
3.2.1 Display Number 0 To 9……..30
3.2.2 Display Two digit fix number (MPX2-CC)……31
3.2.3 Display 00 TO 99 Counting on SSD (MPX2-CC)……..33
3.3 LCD…… 35
3.3.1 Display Single character transmit on LCD……37
3.3.2 Display Full Name On LCD……..39
3.3.3 Display Full Name On LCD using Function……..41
3.3.4 Number show On LCD……..44
3.3.5 Number show On LCD Using Function……46
3.3.6 Display 00 to 99 Counting On LCD……..49
3.4 DC Motor……52
3.4.1 Forward DC Motor Rotation……52
3.4.2 Reverse DC Motor Rotation……..53
3.5 Stepper Motor……54
3.5.1 Full 360 Forward Rotation……54
3.5.2 Full 360 Reverse Rotation……56
3.5.3 To Rotate 90 degree forward rotation and stop it……..57
3.5.4 Full 360 Forward Rotation Using Array……59
References……...60

List Of Figures

Figure 1: C Operators……..18
Figure 2: C Statements……18
Figure 3: Working of for loop……..19
Figure 4: Working of While loop……20
Figure 5: Working of do-while loop……21
Figure 6: Display codes for seven-segment……29
Figure 7: LCD Pin Diagram……35
Figure 8: LCD Pin Function……36

Chapter 1 Introduction Of C and Embedded C

1.1 What is C Programming ??

- C is a general-purpose programming language created by Dennis Ritchie at the Bell Laboratories in 1972.
- It is a very popular language, despite being old.
- C is strongly associated with UNIX, as it was developed to write the UNIX operating system.
- The C language is a high-level language.
- It provides a straightforward, consistent, powerful interface for programming systems. That's why the C language is widely used for developing system software, application software, and embedded systems.

1.2 Application of C

- Operating systems: C is widely used for developing operating systems such as Unix, Linux, and Windows.
- Embedded systems: C is a popular language for developing embedded systems such as microcontrollers, microprocessors, and other electronic devices.
- System software: C is used for developing system software such as device drivers, compilers, and assemblers.
- Networking: C is widely used for developing networking applications such as web servers, network protocols, and network drivers.
- Database systems: C is used for developing database systems such as Oracle, MySQL and PostgreSQL.

- Gaming: C is often used for developing computer games due to its ability to handle low-level hardware interactions.
- Artificial Intelligence: C is used for developing artificial intelligence and machine learning applications such as neural networks and deep learning algorithms.
- Scientific applications: C is used for developing scientific applications such as simulation software and numerical analysis tools.
- Financial applications: C is used for developing financial applications such as stock market analysis and trading systems.
- Please write comments if you find anything incorrect, or if you want to share more information about the topic discussed above.

1.3 What is Embedded C ??

- Embedded C is most popular programming language in software field for developing electronic gadgets. Each processor used in electronic system is associated with embedded software.
- Embedded C programming plays a key role in performing specific function by the processor.
- In day-to-day life we used many electronic devices such as mobile phone, washing machine, digital camera, etc. These all device working is based on microcontroller that are programmed by embedded C.
- In embedded system programming C code is preferred over other language because it is easy to understand, High Reliability, Scalability and Portability.

1.4 Benefits of Embedded C

- It is effortless to understand.
- It executes a similar task continually, so there is no requirement for changing hardware like additional memory, otherwise storage space.
- It performs merely a single task at once
- The cost of the hardware used in the embedded c is typically so much low.
- The applications of embedded are incredibly appropriate in industries.
- It takes less time to develop an application program.
- Embedded C can run pre-defined programming.
- Embedded C is the speed of entering code which provides more accelerated results.

1.5 Industrial Applications of Embedded C

- Embedded C programming is used in industries for different purposes
- The programming language used in the applications is speed controller on the highway, commanding of traffic lights, controlling of street lights, pursuing the vehicle, artificial intelligence, home automation, and auto intensity control.

1.6 Comparison Of C and Embedded C

C	Embedded C
• It is a structural and general purpose programming language used by the developers to build desktop-based applications. • C is a high-level programming language. • This programming language is hardware independent. • The traditional or standard compilers are used to run the program	• Embedded C is generally used to develop microcontroller-based applications. • Embedded C is just the extension variant of the C language. • Embedded C language is truly hardware dependent. • Here, we need a specific compiler that can help in generating micro-controller based results.

Chapter 2 C Programming

2.1 Basics Of C

Special Characters in C Programming

,	<	>	.	_
(	)	;	$	:
%	[	]	#	?
'	&	{	}	"
^	!	*	/	|
-	\	~	+	

C Keywords

`auto`	`double`	`int`	`struct`
`break`	`else`	`long`	`switch`
`case`	`enum`	`register`	`typedef`
`char`	`extern`	`return`	`union`
`continue`	`for`	`signed`	`void`
`do`	`if`	`static`	`while`
`default`	`goto`	`sizeof`	`volatile`
`const`	`float`	`short`	`unsigned`

2.2 C Data types

Type	Size (bytes)	Format Specifier
`int`	at least 2, usually 4	`%d`, `%i`
`char`	1	`%c`
`float`	4	`%f`
`double`	8	`%lf`
`short int`	2 usually	`%hd`
`unsigned int`	at least 2, usually 4	`%u`
`long int`	at least 4, usually 8	`%ld`, `%li`
`long long int`	at least 8	`%lld`, `%lli`
`unsigned long int`	at least 4	`%lu`
`unsigned long long int`	at least 8	`%llu`
`signed char`	1	`%c`
`unsigned char`	1	`%c`
`long double`	at least 10, usually 12 or 16	`%Lf`

2.3 C input/Output (I/O)

In C programming, printf() is one of the main output function. The function sends formatted output to the screen.

In C programming, scanf() is one of the commonly used function to take input from the user. The scanf() function reads formatted input from the standard input such as keyboards.

2.4 C Operators

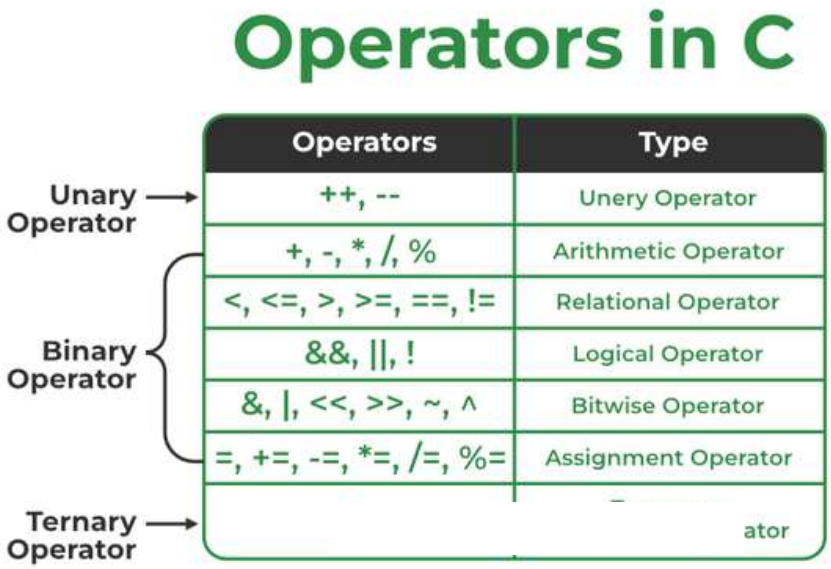


Figure 9: C Operators[1]

2.5 C Statements

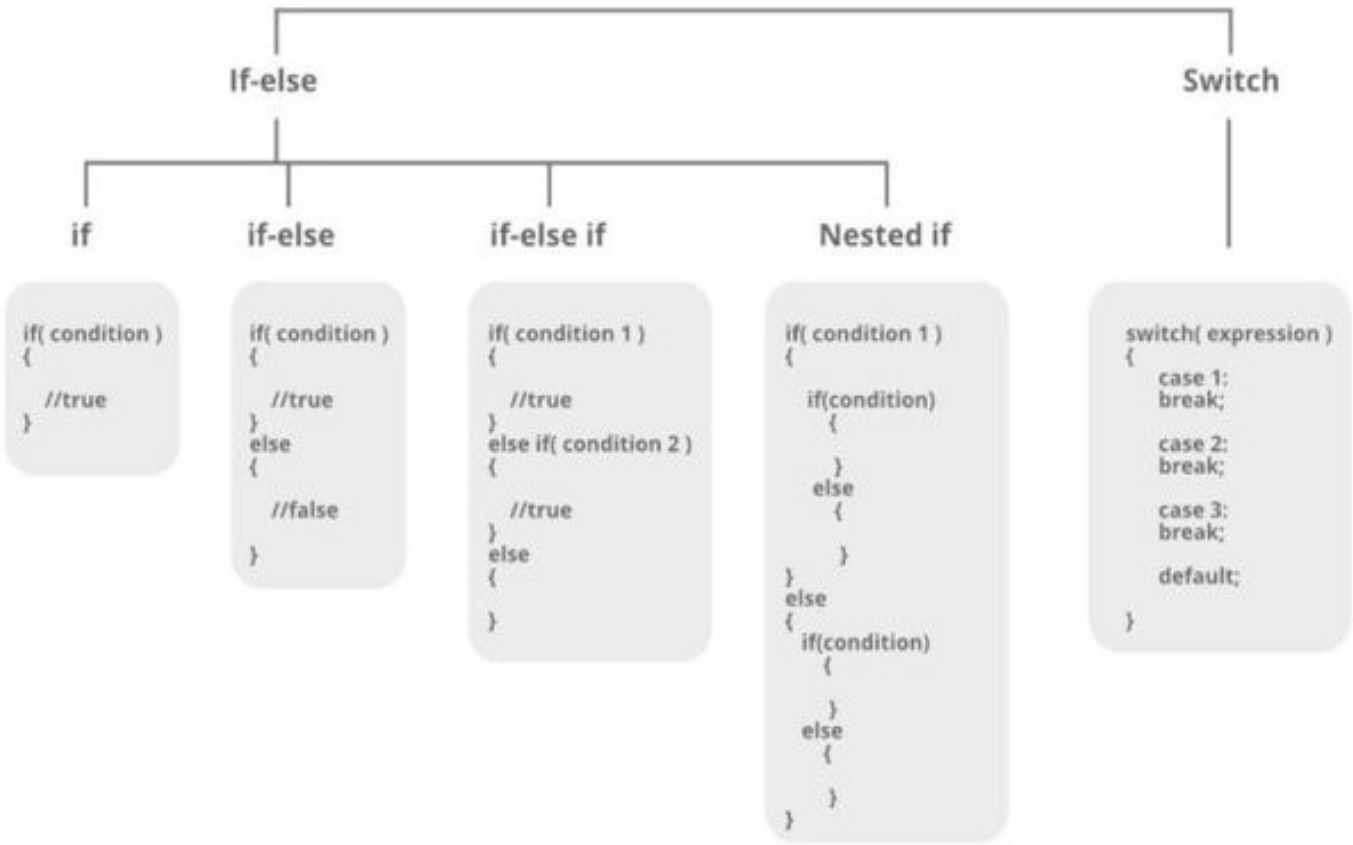


Figure 10: C Statements[2]

2.6 C Loops

2.6.1 for loop

The syntax of the for loop is:

```
for (initializationStatement; testExpression; updateStatement)
{
   // statements inside the body of loop
}
```

How for loop works?

- The initialization statement is executed only once.
- Then, the test expression is evaluated. If the test expression is evaluated to false, the for loop is terminated.
- However, if the test expression is evaluated to true, statements inside the body of the for loop are executed, and the update expression is updated.
- Again the test expression is evaluated.
- This process goes on until the test expression is false. When the test expression is false, the loop terminates.

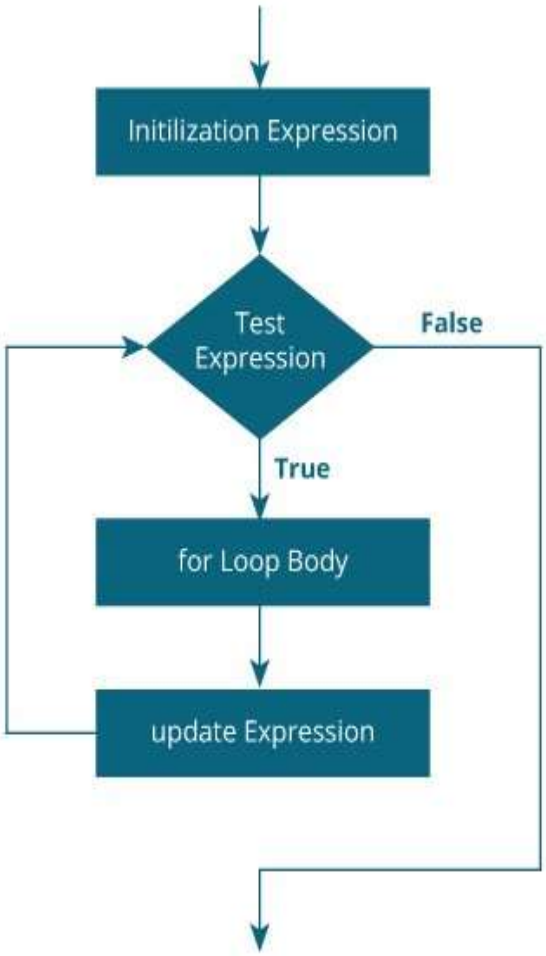


Figure 11: Working of for loop[3]

2.6.2 While Loop

The syntax of the while loop is:

```
while (testExpression) {
  // the body of the loop
}
```

How while loop works?

- The while loop evaluates the testExpression inside the parentheses ().
- If testExpression is true, statements inside the body of while loop are executed. Then, testExpression is evaluated again.
- The process goes on until testExpression is evaluated to false.
- If testExpression is false, the loop terminates (ends).
- To learn more about test expressions (when testExpression is evaluated to true and false), check out relational and logical operators.

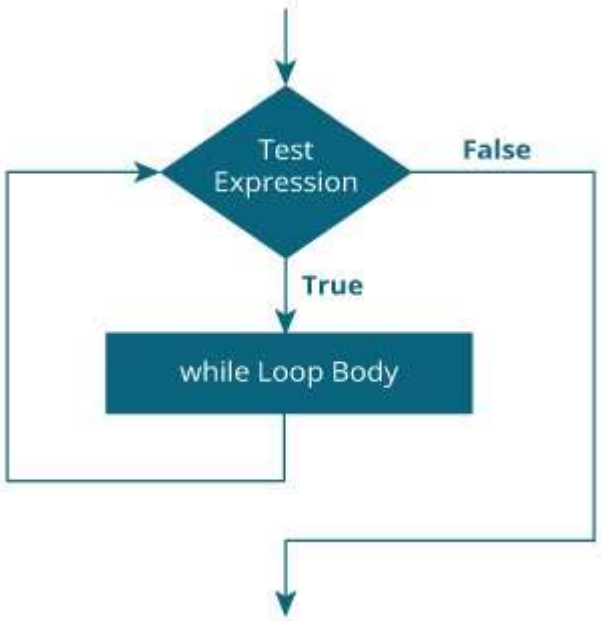


Figure 12: Working of While loop[4]

2.6.3 do while loop

The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed at least once. Only then, the test expression is evaluated.

The syntax of the do...while loop is:

```
do {
  // the body of the loop
}
while (testExpression)
```

How do...while loop works?

- The body of do...while loop is executed once. Only then, the testExpression is evaluated.
- If testExpression is true, the body of the loop is executed again and testExpression is evaluated once more.
- This process goes on until testExpression becomes false.
- If testExpression is false, the loop ends.

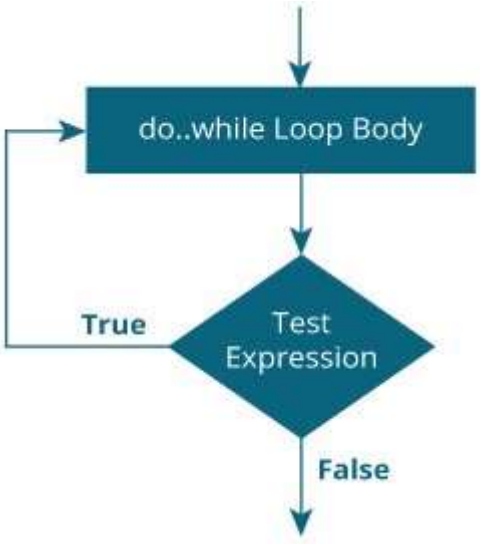


Figure 13: Working of do-while loop[5]

2.7 C break and continue

2.7.1 C break

The break statement ends the loop immediately when it is encountered. Its syntax is:

break;

The break statement is almost always used with if...else statement inside the loop.

2.7.2 C continue

The continue statement skips the current iteration of the loop and continues with the next iteration. Its syntax is:

continue;

The continue statement is almost always used with the if...else statement.

2.8 C functions

A function is a block of code that performs a specific task.

Suppose, you need to create a program to create a circle and color it. You can create two functions to solve this problem:

- create a circle function
- create a color function
- Dividing a complex problem into smaller chunks makes our program easy to understand and reuse.

Types of function :

There are two types of function in C programming:

1) Standard library functions
2) User-defined functions

2.9 C Arrays

An array is a variable that can store multiple values. For example, if you want to store 100 integers, you can create an array for it.

```
int data[100];
```

How to declare an array?

dataType arrayName[arraySize];

For example,

float mark[5];

Here, we declared an array, mark, of floating-point type. And its size is 5. Meaning, it can hold 5 floating-point values.

It's important to note that the size and type of an array cannot be changed once it is declared.

2.10 C Pointers

Pointers (pointer variables) are special variables that are used to store addresses rather than values.

Pointer Syntax

Here is how we can declare pointers.

int* p;

Here, we have declared a pointer p of int type.

You can also declare pointers in these ways.

int *p1;

int * p2;

2.11 C strings

When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default then it is a string.

How to declare a string?

Here's how you can declare strings:

```
char s[5];
```

How to initialize strings?

You can initialize strings in a number of ways.

```
char c[] = "abcd";

char c[50] = "abcd";

char c[] = {'a', 'b', 'c', 'd', '\0'};

char c[5] = {'a', 'b', 'c', 'd', '\0'};
```

Chapter 3 Embedded C

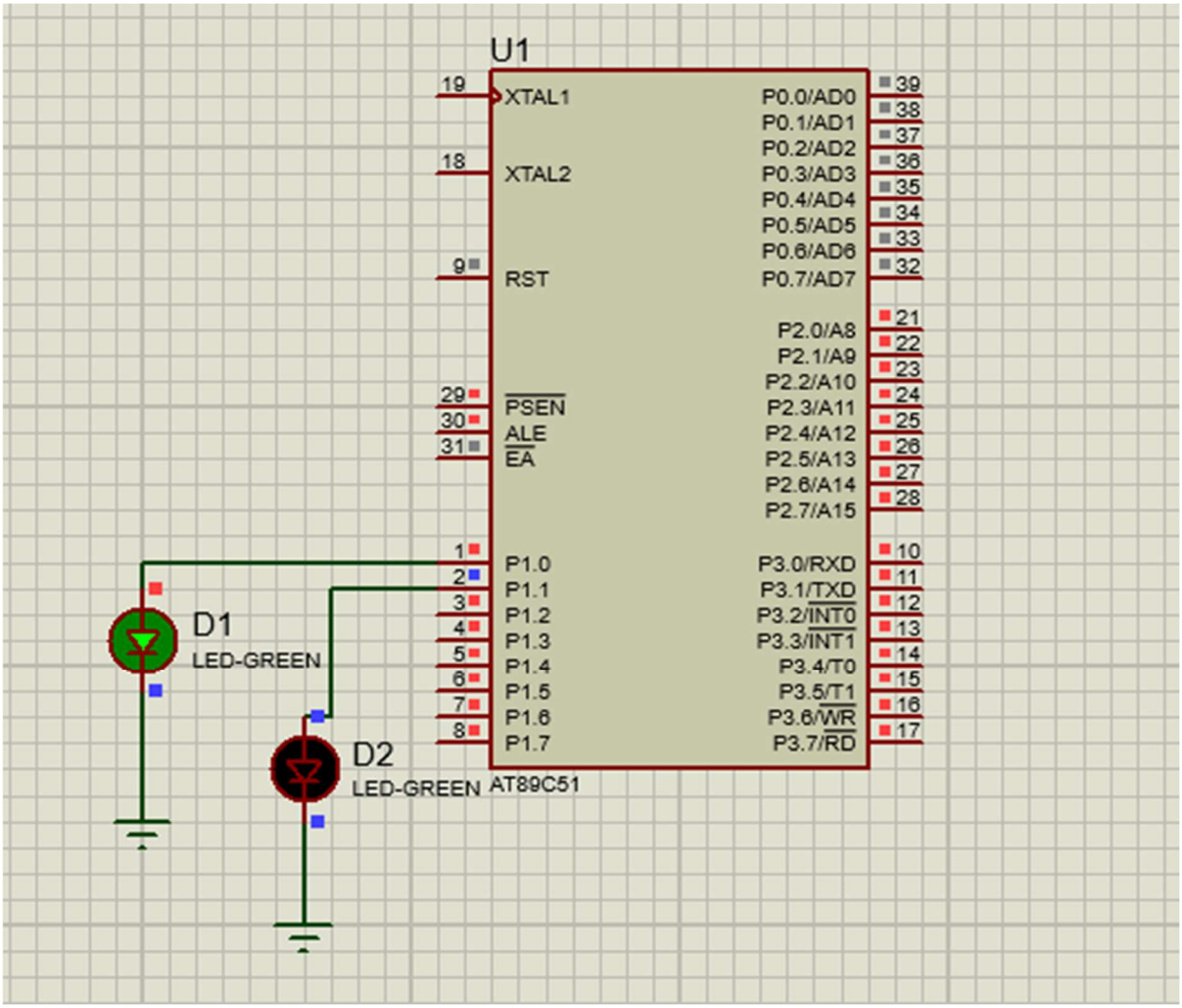
3.1 LED Blinking

LED is a semiconductor device used in many electronic devices, mostly used for indication purposes. It is used widely as indicator during test for checking the validity of results at different stages.

It is very cheap and easily available in variety of shape, color and size. The LEDs are also used in designing of message display boards and traffic control signal lights etc.

3.1.1 Display LED linking with 0&1

```
#include<reg51.h>
void delay(unsigned int y)
{
        int i;
        for(i=0;i<=y;i++){
        }
}
void main()
{
        P2=0XFF;
        delay(100);
        P2=0X00;
        delay(100);
}
```

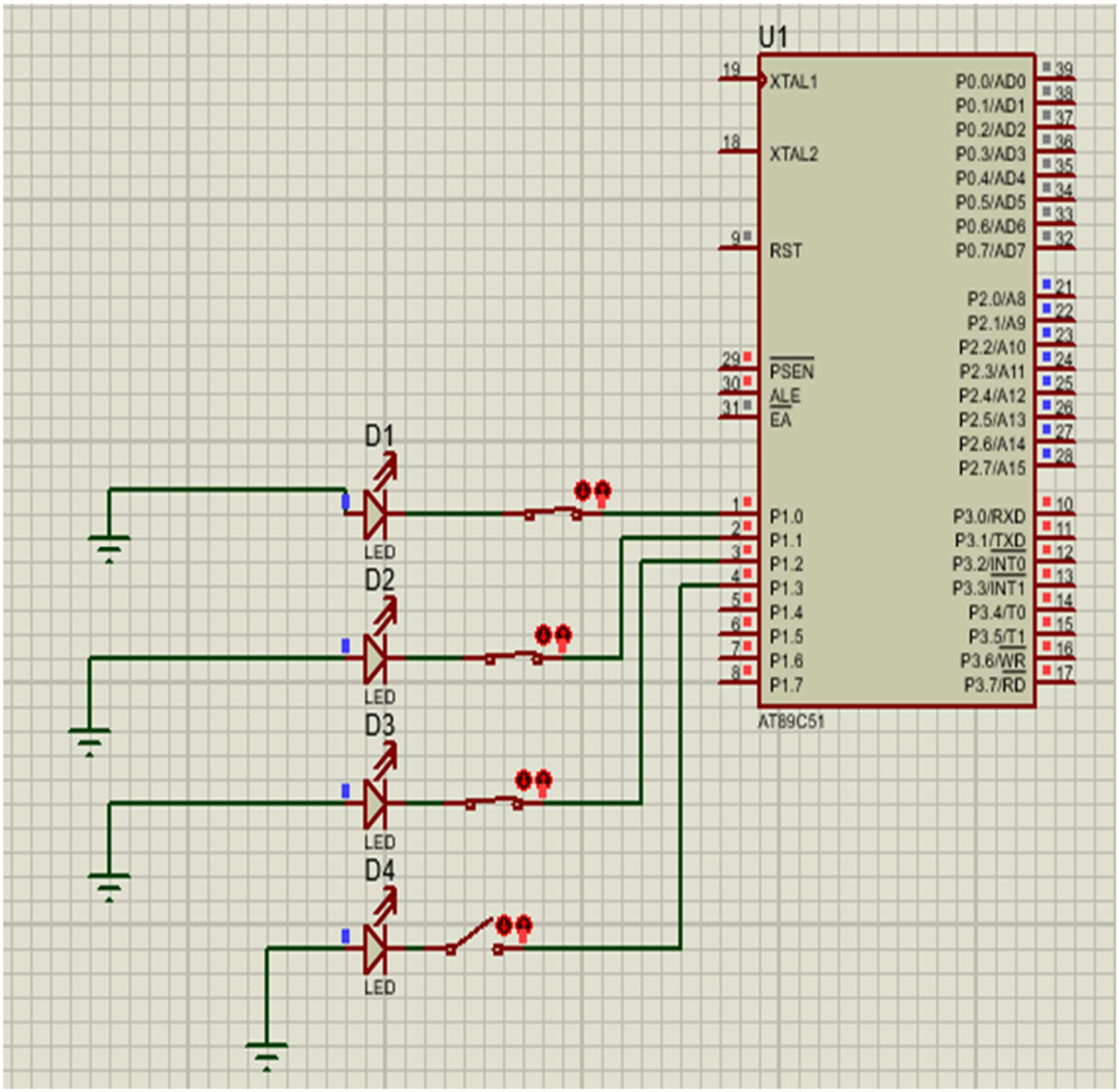


3.1.2 Display combination of LEDs with Blink,Toggle,on off and Sequential

```
#include<reg51.h>
#define LED P2
sbit sw0=P1^0;//led blinking
sbit sw1=P1^1;//led toggle
sbit sw2=P1^2;//led on/off
sbit sw3=P1^3;//led sequence
void delay(unsigned int y)
{
        int x,z;
        for(x=0;x<y;x++)
        for(z=0;z<1275;z++);
}
void main()
```

```
{
        unsigned int a[8]={0x01,0x03,0x07,0x0F,0x1F,0x3F,0x7F,0xFF},i;
         while(1)
         {
                  if(sw0==0)//led blinking
                        {
                                LED=0xFF;
                                delay(10000);
                                LED=0x00;
                                delay(10000);

                        }
                        else if(sw1==0)//led toggle
                 {
                         LED=0x55;
                         delay(3000);
                         LED=0xAA;
                         delay(3000);
                 }
                else if(sw2==0)//led on/off
                 {
                         LED=0xFF;
                         delay(3000);

                 }
                else if(sw3==0)//led sequ
                 {
                         for(i=0;i<8;i++)
                         {
                                  LED=a[i];
                                 delay(3000);
                         }
                         for(i=7;i>0;i--)
                         {
                                 LED=a[i];
                                 delay(3000);
                         }
```

```
            }
                else
                {
                    LED=0x00;
                }
            }
        }
```



3.2 Seven-Segment Display

A seven-segment display module is an electronic device used to display digital numbers and it is made up of seven LED segments. Because of the small size of the LEDs, it is really easy for a number of them to be connected together to make a unit like seven segment display. In the seven-segment display module, seven LED s are arranged in a rectangle. Sometimes, an additional LED is seen in a seven-segment display unit which is meant for displaying a decimal point.

Each LED segment has one of its pins brought out of the rectangular package. Other pins are connected together to a common terminal. Seven segment displays can only display 0 to 9 numbers. These seven LEDs indicate seven segments of the numbers and a dot point.

Seven segment displays are seen associated with a great number of devices such as clocks, digital home appliances, signal boards on roads etc.

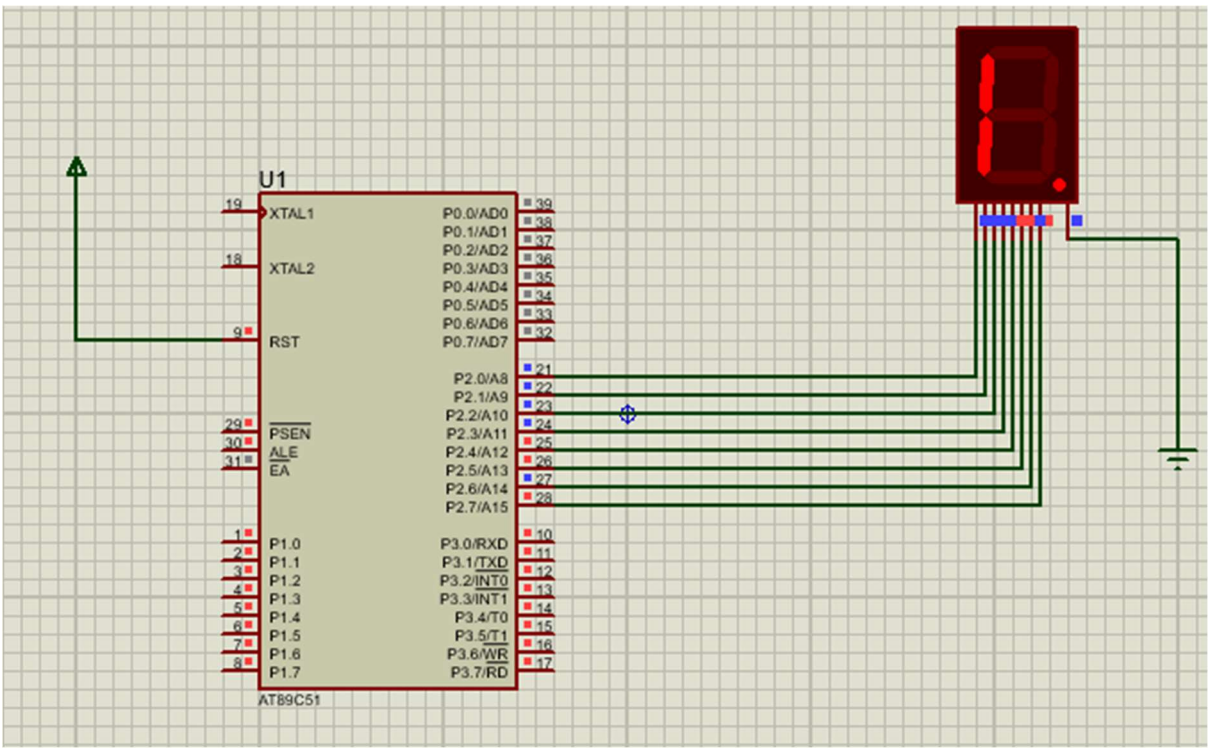
Numbers	Common Cathode		Common Anode	
	(DP)GFEDCBA	HEX Code	(DP)GFEDCBA	HEX Code
0	00111111	0x3F	11000000	0xC0
1	00000110	0x06	11111001	0xF9
2	01011011	0x5B	10100100	0xA4
3	01001111	0x4F	10110000	0xB0
4	01100110	0x66	10011001	0x99
5	01101101	0x6D	10010010	0x92
6	011111101	0x7D	10000010	0x82
7	00000111	0x07	11111000	0xF8
8	01111111	0x7F	10000000	0x80
9	01101111	0x6F	10010000	0x90

Figure 14: Display codes for seven-segment

3.2.1 Display Number 0 To 9

```
#include<reg51.h>
#define ssd P2
void delay(unsigned int y)
{
        unsigned int x;
        for(x=0;x<y;x++);
}
void main()
{
        ssd=0xc0;
        delay(30000);
        ssd=0xF9;
        delay(30000);
        ssd=0xA4;
        delay(30000);
        ssd=0xB0;
        delay(30000);
        ssd=0x99;
        delay(30000);
        ssd=0x92;
        delay(30000);
        ssd=0x82;
        delay(30000);
        ssd=0xF8;
        delay(30000);
        ssd=0x80;
```

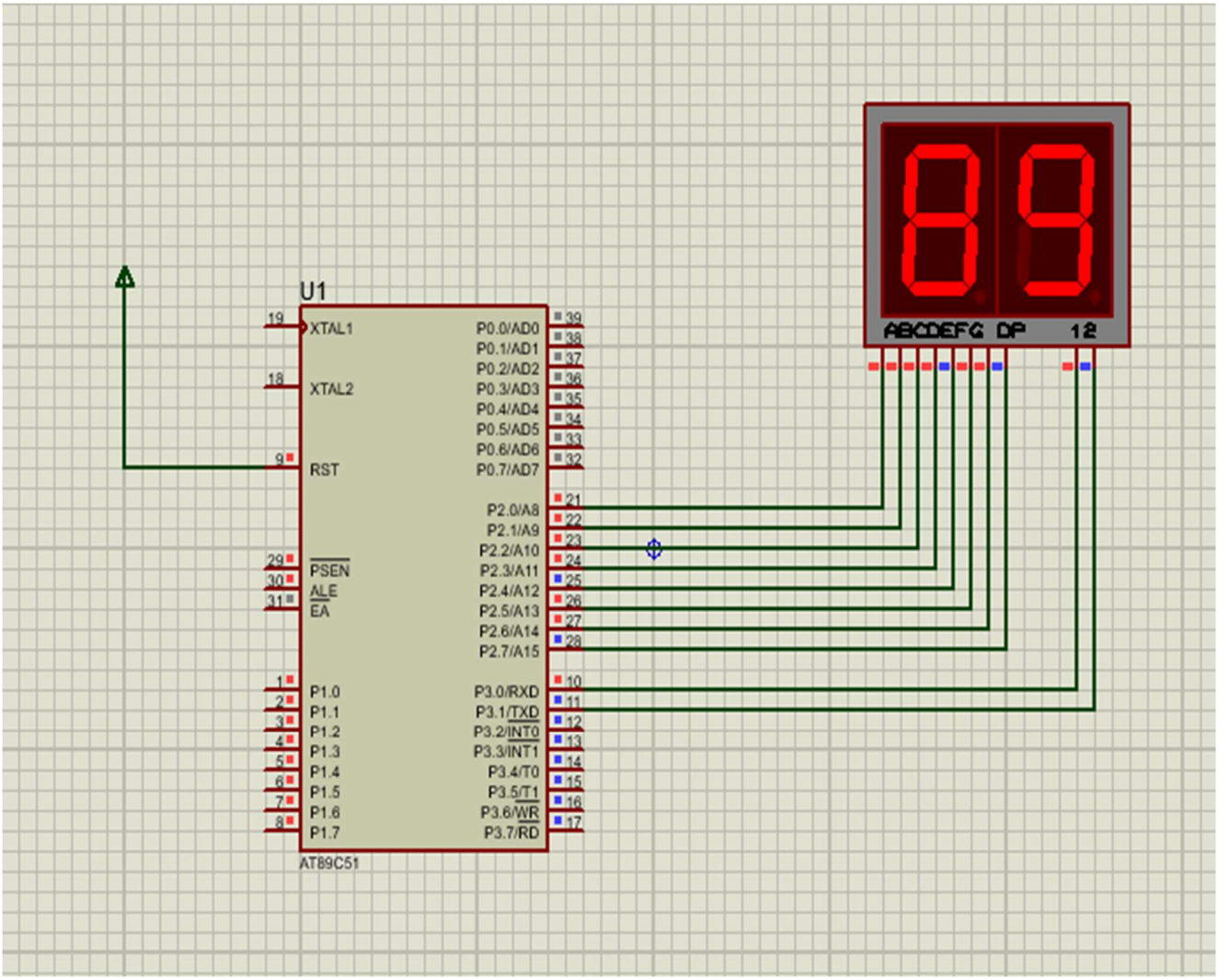
```
        delay(30000);
        ssd=0x90;
        delay(30000);
}
```



3.2.2 Display Two digit fix number (MPX2-CC)

```
#include<reg51.h>
#define ssd P2
#define num P3

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}
```

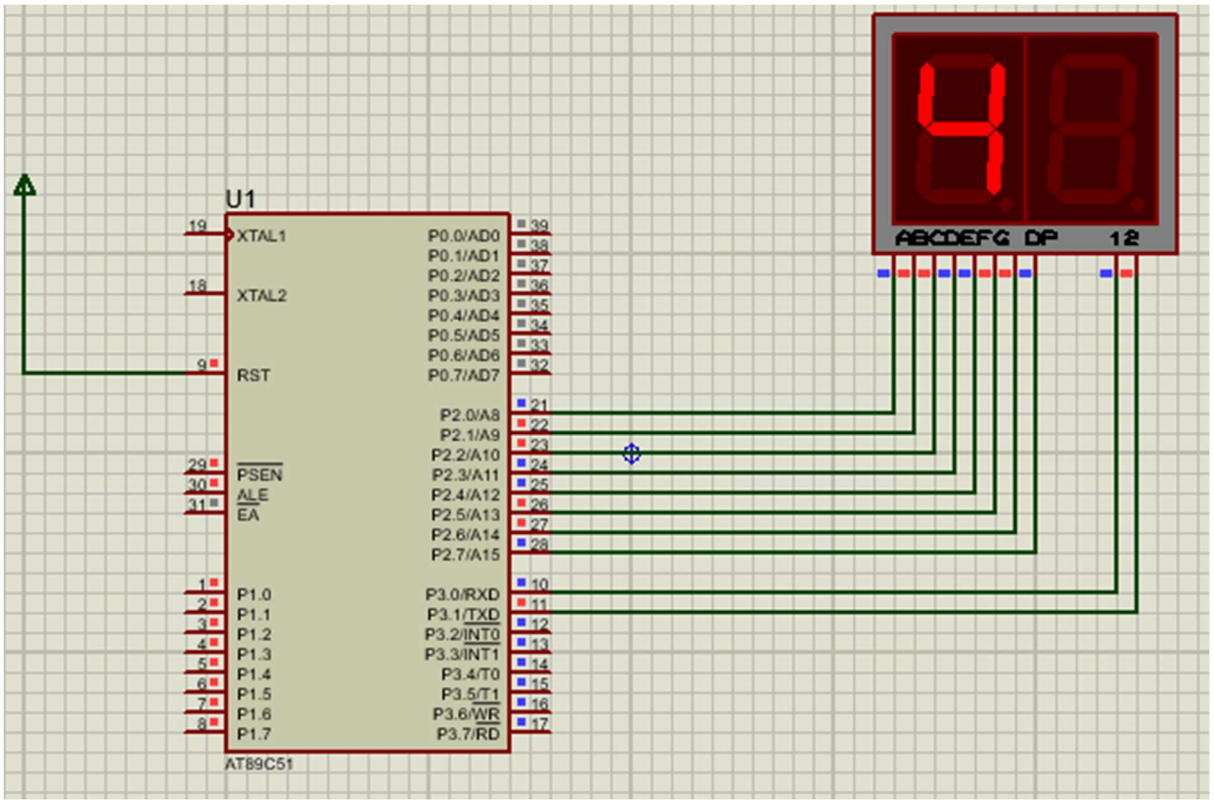
```
void display(unsigned char y)
{
 unsigned char x[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};
 ssd=x[y];
}

void main()
{
        int a,b,c;
        while(1)
        {
                c=98;
                a=c%10;
                b=c/10;
                num=0x02;
                display(a);
                delay(300000);
                num=0x01;
                display(b);
                delay(300000);
        }
}
```



3.2.3 Display 00 TO 99 Counting on SSD (MPX2-CC)

```
#include<reg51.h>
#define ssd P2
#define num P3

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
```

```
}

void display(unsigned char y)
{
 unsigned char x[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};
 ssd=x[y];
}

void main()
{
        int a,b,c,i;
        while(1)
        {
          for(c=0;c<100;c++)
          {
           a=c%10;
           b=c/10;
            for(i=0;i<10;i++)
             {
                num=0x02;
                display(a);
                delay(90000);
                num=0x01;
                display(b);
                delay(90000);
                }
             }
        }
}
```



3.3 LCD

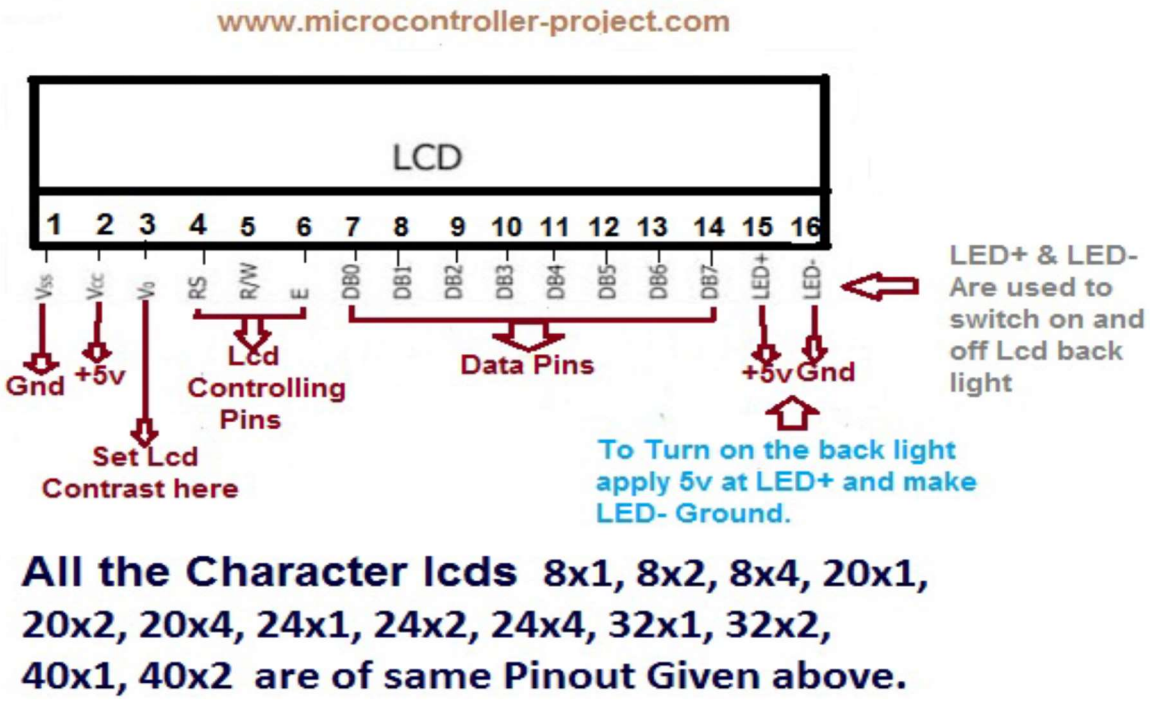


Figure 15: LCD Pin Diagram

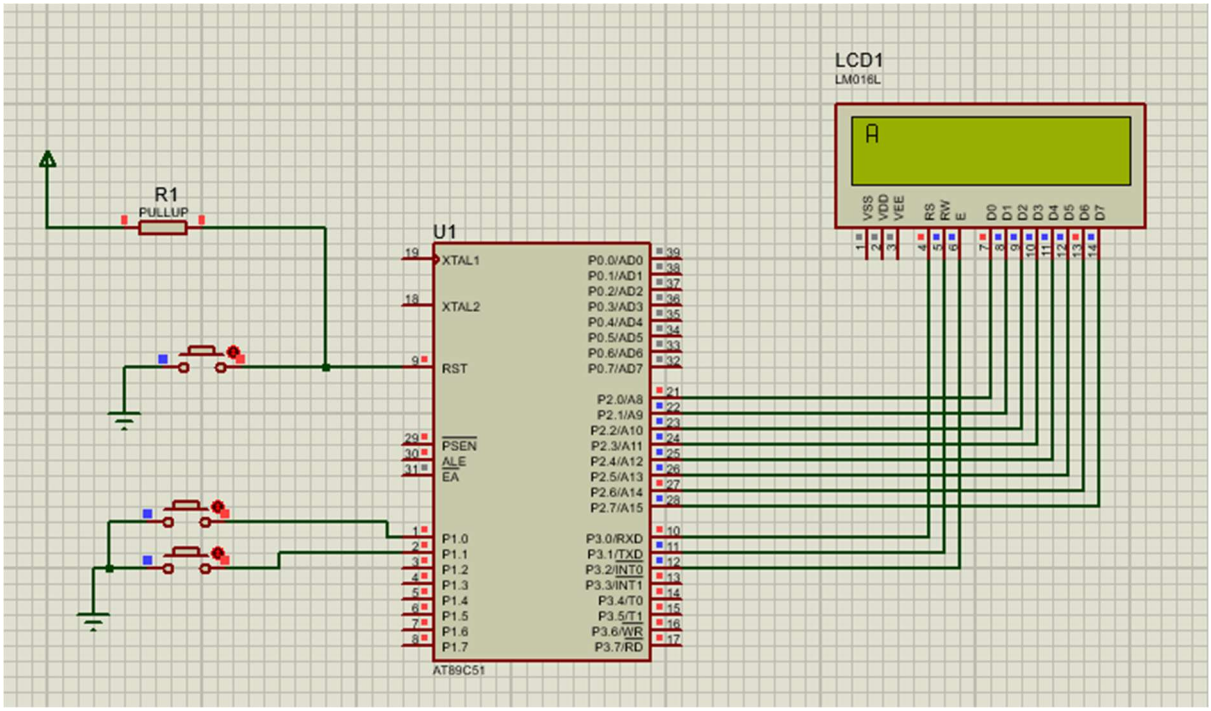
Pin number	Name	Function
1	VSS	Ground voltage
2	VEE	+5V
3	VCC	Contrast voltage
4	RS	Register select 1-Data register 0-Instruction register
5	R/W	Read/Write mode, to select read/write mode 0-write mode 1-read mode
6	E	Enable 0-Start to latch data to LCD character 1-Disable
7	DB0	Data bit 0 (LSB BIT)
8	DB1	Data bit 1
9	DB2	Data bit 2
10	DB3	Data bit 3
11	DB4	Data bit 4
12	DB5	Data bit 5
13	DB6	Data bit 6
14	DB7	Data bit 7 (MSB)
15	BPL	Black Plane Light (+5V) or lower (optional)
16	GND	Ground voltage (optional)

Figure 16: LCD Pin Function

3.3.1 Display Single character transmit on LCD

```
#include<reg51.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}

void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}

void dat(unsigned char y)
{
        lcd=y;
        rs=1;
      rw=0;
```

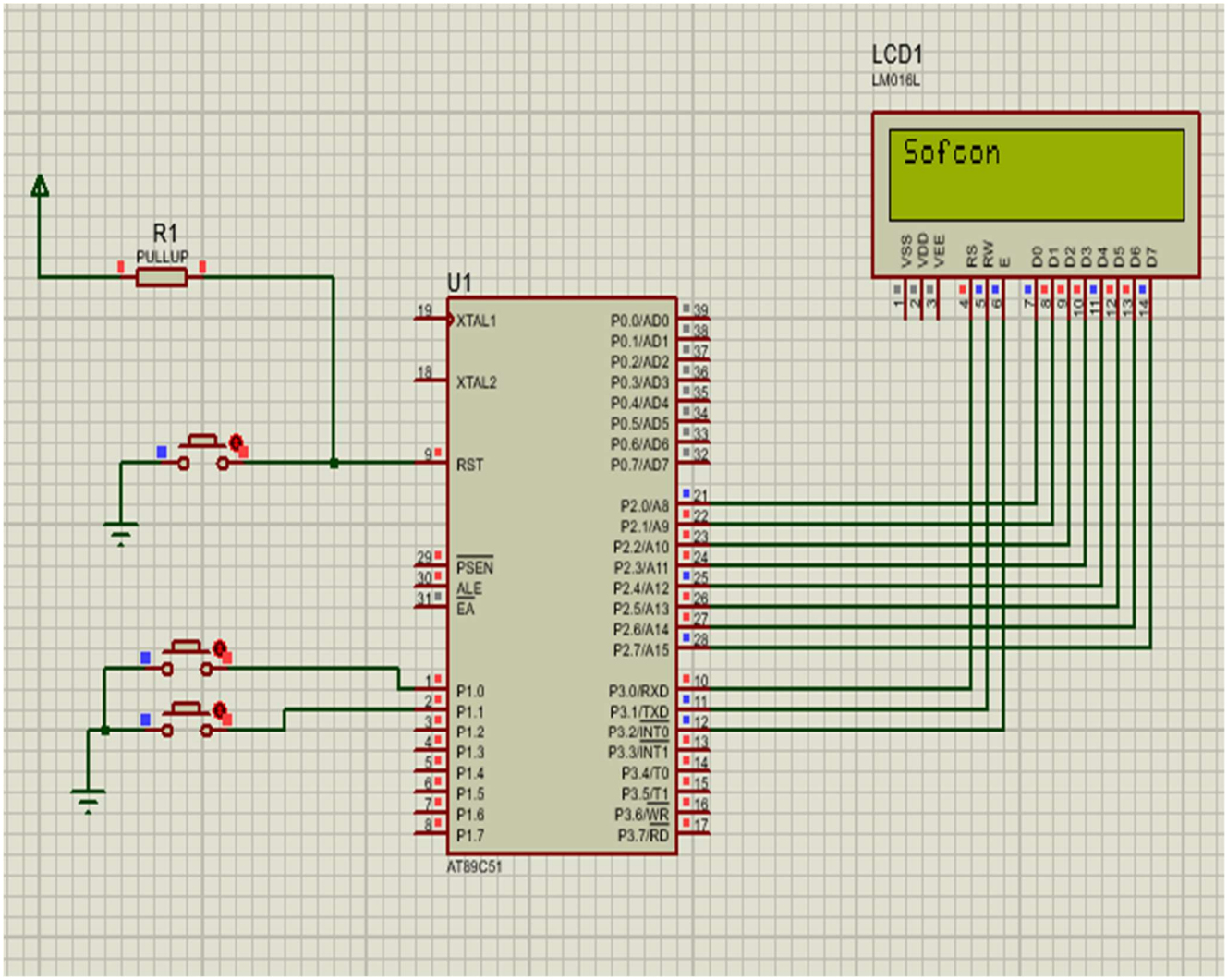
```
        en=1;
        delay(1000);
        en=0;
}


void main()
{
        cmd(0x01);
        cmd(0x0c);
        cmd(0x06);
        cmd(0x38);
        cmd(0x80);
        dat('A');
        while(1);
}
```



3.3.2 Display Full Name On LCD

```
#include<reg51.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}

void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}

void dat(unsigned char y)
{
        lcd=y;
        rs=1;
```

```
        rw=0;
          en=1;
          delay(1000);
          en=0;
}


void main()
{
          cmd(0x01);
          cmd(0x0c);
          cmd(0x06);
          cmd(0x38);
          cmd(0x80);
          dat('S');
          dat('o');
          dat('f');
          dat('c');
          dat('o');
          dat('n');
          while(1);
}
```



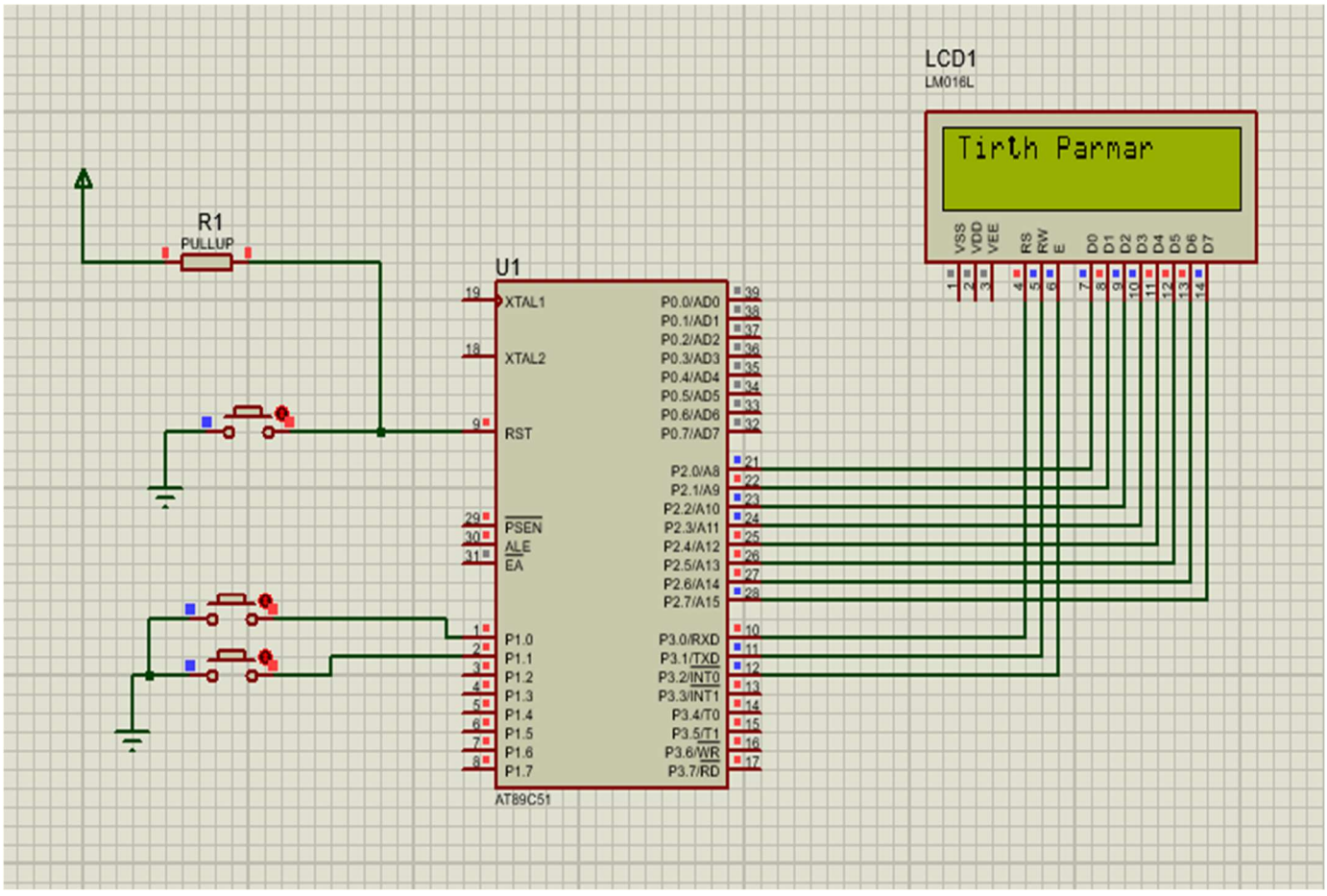
3.3.3 Display Full Name On LCD using Function

```
#include<reg51.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
```

```
        for(i=0;i<y;i++);
}
void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
       rw=0;
        en=1;
        delay(1000);
        en=0;
}
void dat(unsigned char y)
{
        lcd=y;
        rs=1;
       rw=0;
        en=1;
        delay(1000);
        en=0;
}
void display(unsigned char k[])
{
        unsigned char j=0;
        while(k[j]!='\0')
        {
                dat(k[j]);
                j++;
```

337356

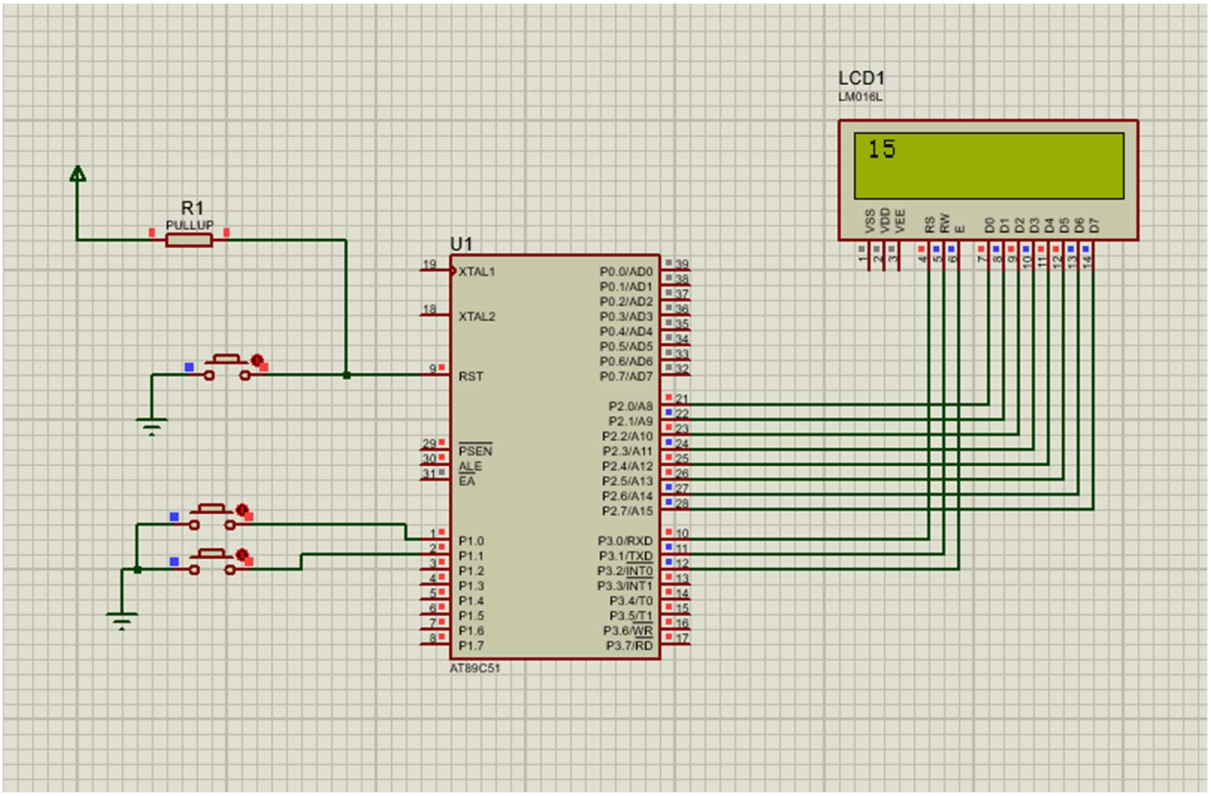
```
        }
}
void main()
{
        cmd(0x01);
        cmd(0x0c);
        cmd(0x06);
        cmd(0x38);
        cmd(0x80);
        display("Tirth Parmar");
        while(1);
}
```



3.3.4 Number show On LCD

```
#include<reg51.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}

void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
       rw=0;
        en=1;
        delay(1000);
        en=0;
}

void dat(unsigned char y)
{
        lcd=y;
```

```
        rs=1;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}

void main()
{
        int a=15,b,c;
        cmd(0x01);
        cmd(0x0c);
        cmd(0x06);
        cmd(0x38);
        cmd(0x80);

        b=a%10;
        c=a/10;

        cmd(0x80);
        dat(c+48);

        cmd(0x81);
        dat(b+48);

        while(1);
}
```



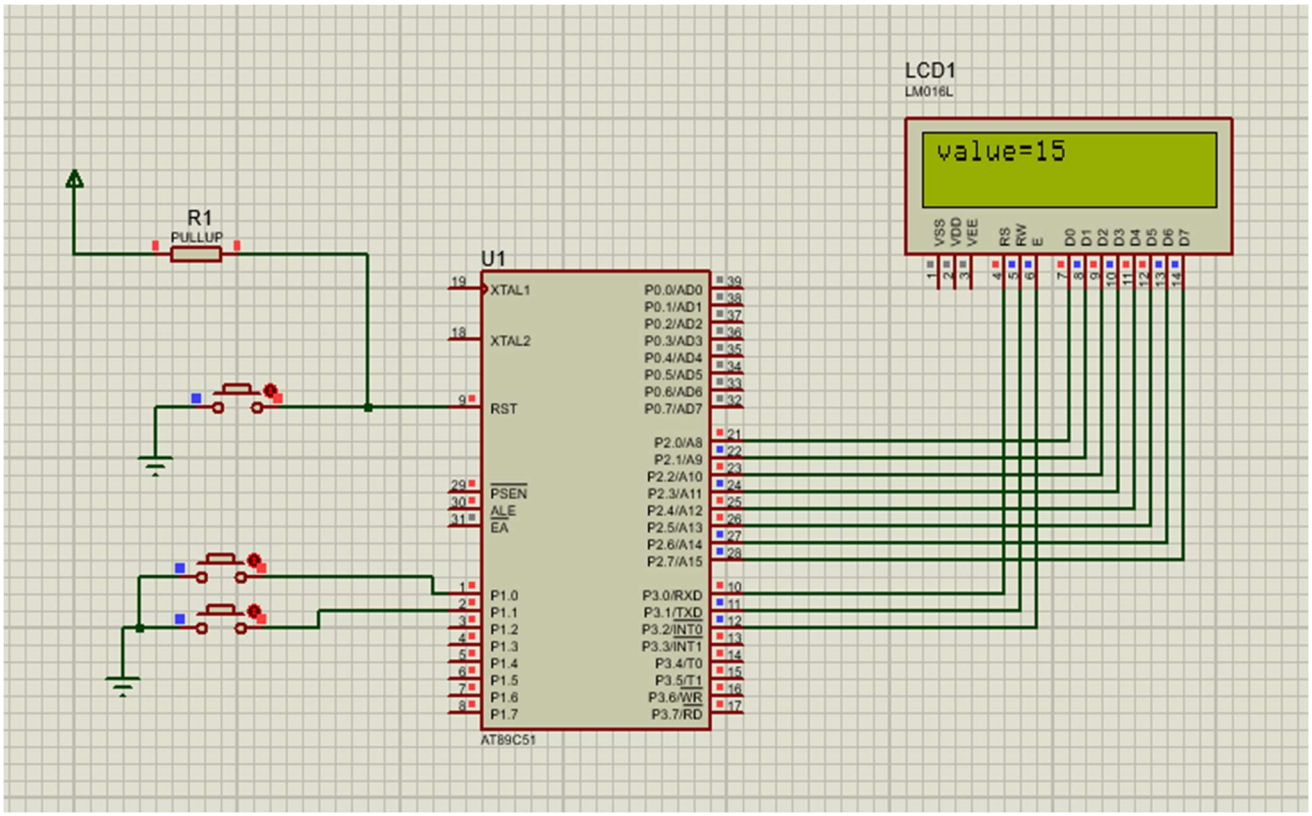
3.3.5 Number show On LCD Using Function

```
#include<reg51.h>
#include<stdio.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}
```

```
void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}
void dat(unsigned char y)
{
        lcd=y;
        rs=1;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}
void display(unsigned char k[])
{
        unsigned char j=0;
        while(k[j]!='\0')
        {
                dat(k[j]);
                j++;
        }
}
```

```
void main()
{
        int a=15;
        char z[20];
        cmd(0x01);
        cmd(0x0c);
        cmd(0x06);
        cmd(0x38);
        cmd(0x80);
      sprintf(z,"value=%d",a);
        display(z);
        while(1);
}
```

sprintf does integer to string conversion. and its header file is #include<stdio.h>

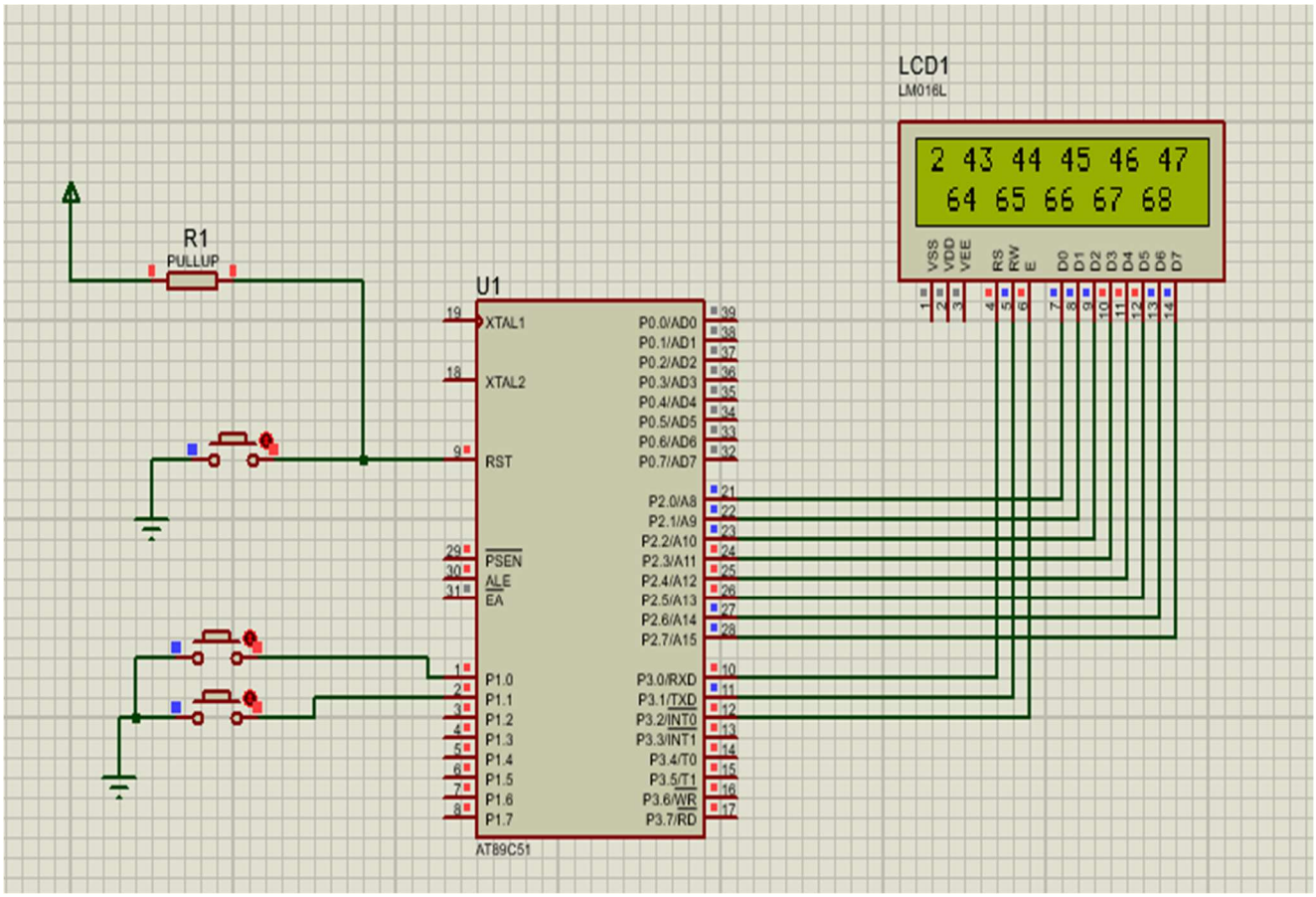


3.3.6 Display 00 to 99 Counting On LCD

```
#include<reg51.h>
#include<stdio.h>
#define lcd P2
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void delay(unsigned int y)
{
        unsigned int i;
        for(i=0;i<y;i++);
}

void cmd(unsigned char y)
{
        lcd=y;
        rs=0;
      rw=0;
        en=1;
        delay(1000);
        en=0;
}

void dat(unsigned char y)
{
```

```
        lcd=y;
        rs=1;
       rw=0;
        en=1;
        delay(1000);
        en=0;
}

void display(unsigned char k[])
{
        unsigned char j=0;
        while(k[j]!='\0')
        {
                dat(k[j]);
                j++;
        }
}

void main()
{
        int a;
        char z[20];
        cmd(0x01);
        cmd(0x0c);
        cmd(0x06);
        cmd(0x38);
        cmd(0x80);
```

```
        while(1)
        {
         for(a=0;a<=100;a++)
         {
        sprintf(z," 3d",a);
          display(z);
          }
    }
}
```



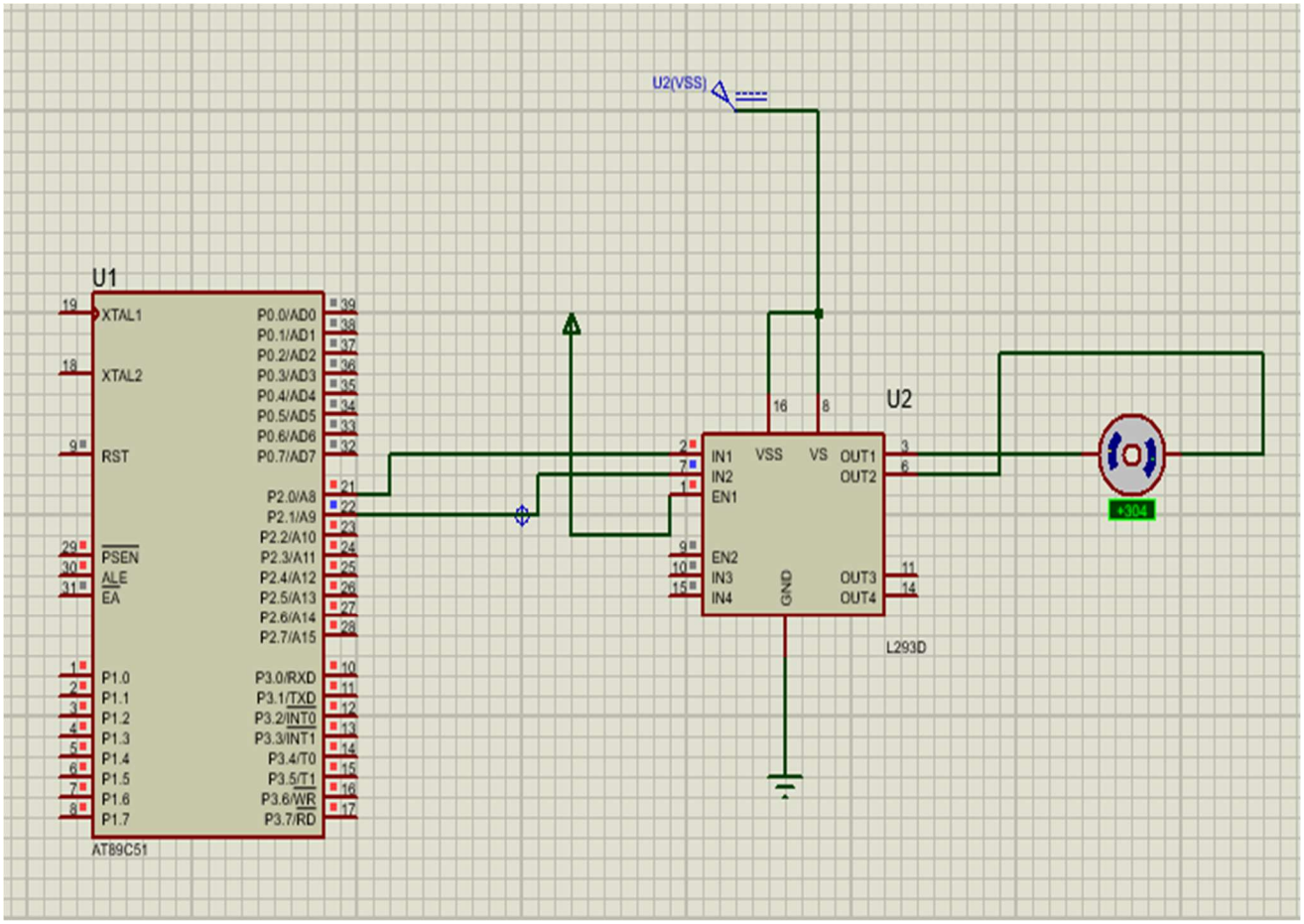
3.4 DC Motor

Application of dc motor:

- Elevators
- Lifts
- Air compressor
- Cranes
- Fans
- Washing machine
- Rolling mills

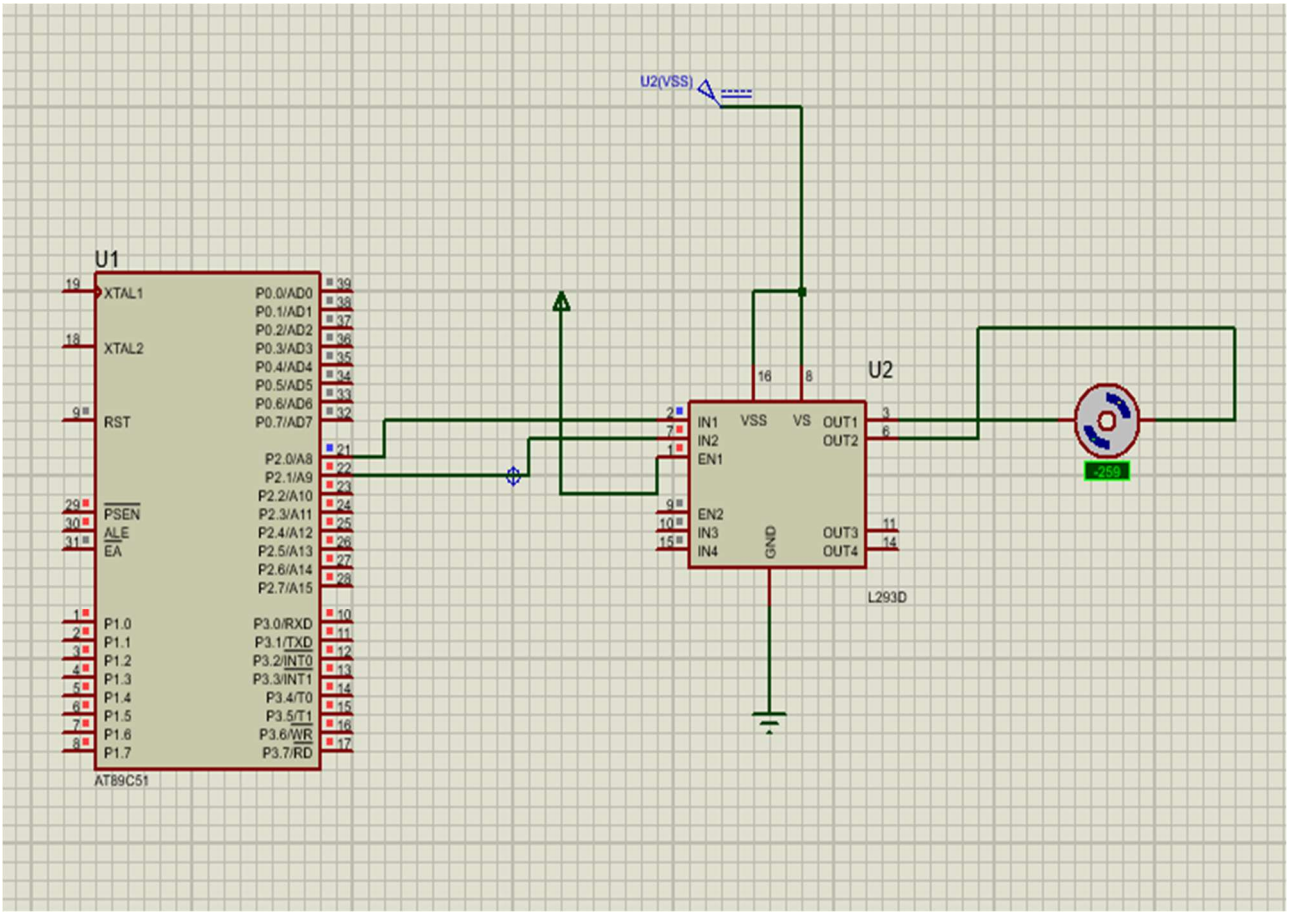
3.4.1 Forward DC Motor Rotation

```
#include<reg51.h>
sbit a=P2^0;
sbit b=P2^1;

void main()
{
        while(1)
        {
        a=1;
        b=0;
  }
}
```



3.4.2 Reverse DC Motor Rotation

```
#include<reg51.h>
sbit a=P2^0;
sbit b=P2^1;
void main()
{
        while(1)
        {
        a=0;
        b=1;
  }
}
```



3.5 Stepper Motor

Applications of Stepper motor:

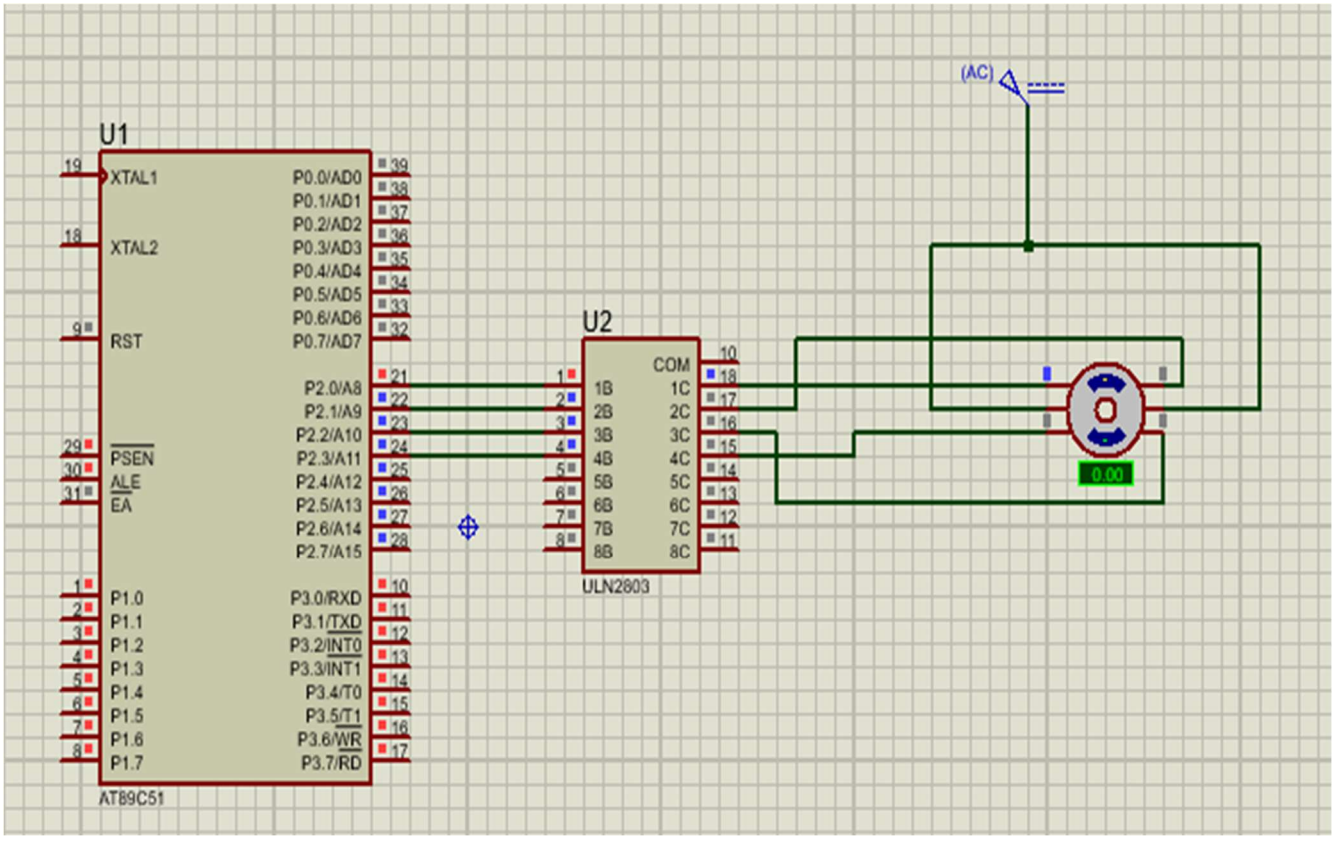
- floppy disk drives
- computer printers
- image scanners
- intelligent lighting
- camera lenses
- 3D printers

3.5.1 Full 360 Forward Rotation

```
#include<reg51.h>
#define stpr P2
void delay(unsigned int n)
{
        unsigned int i;
        for(i=0;i<n;i++);
```

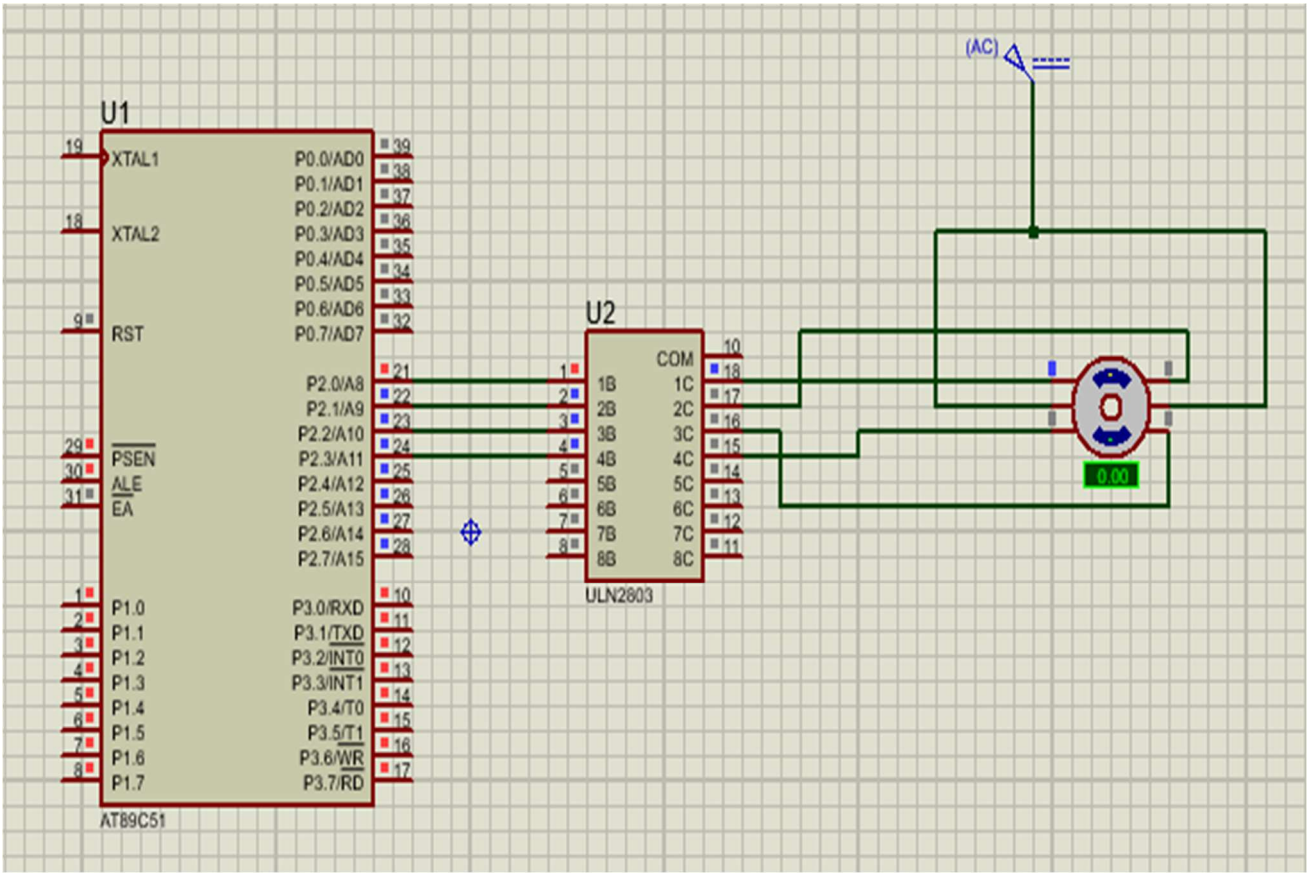
337356

```
}
void main()
{
        while(1)
        {
                stpr=0x01;
                delay(20000);
                stpr=0x02;
                delay(20000);
                stpr=0x04;
                delay(20000);
                stpr=0x08;
                delay(20000);
        }
}
```



3.5.2 Full 360 Reverse Rotation

```
#include<reg51.h>
#define stpr P2
void delay(unsigned int n)
{
	unsigned int i;
	for(i=0;i<n;i++);
}
void main()
{
	while(1)
	{
		stpr=0x08;
		delay(20000);
		stpr=0x04;
		delay(20000);
		stpr=0x02;
		delay(20000);
		stpr=0x01;
		delay(20000);
	}
}
```

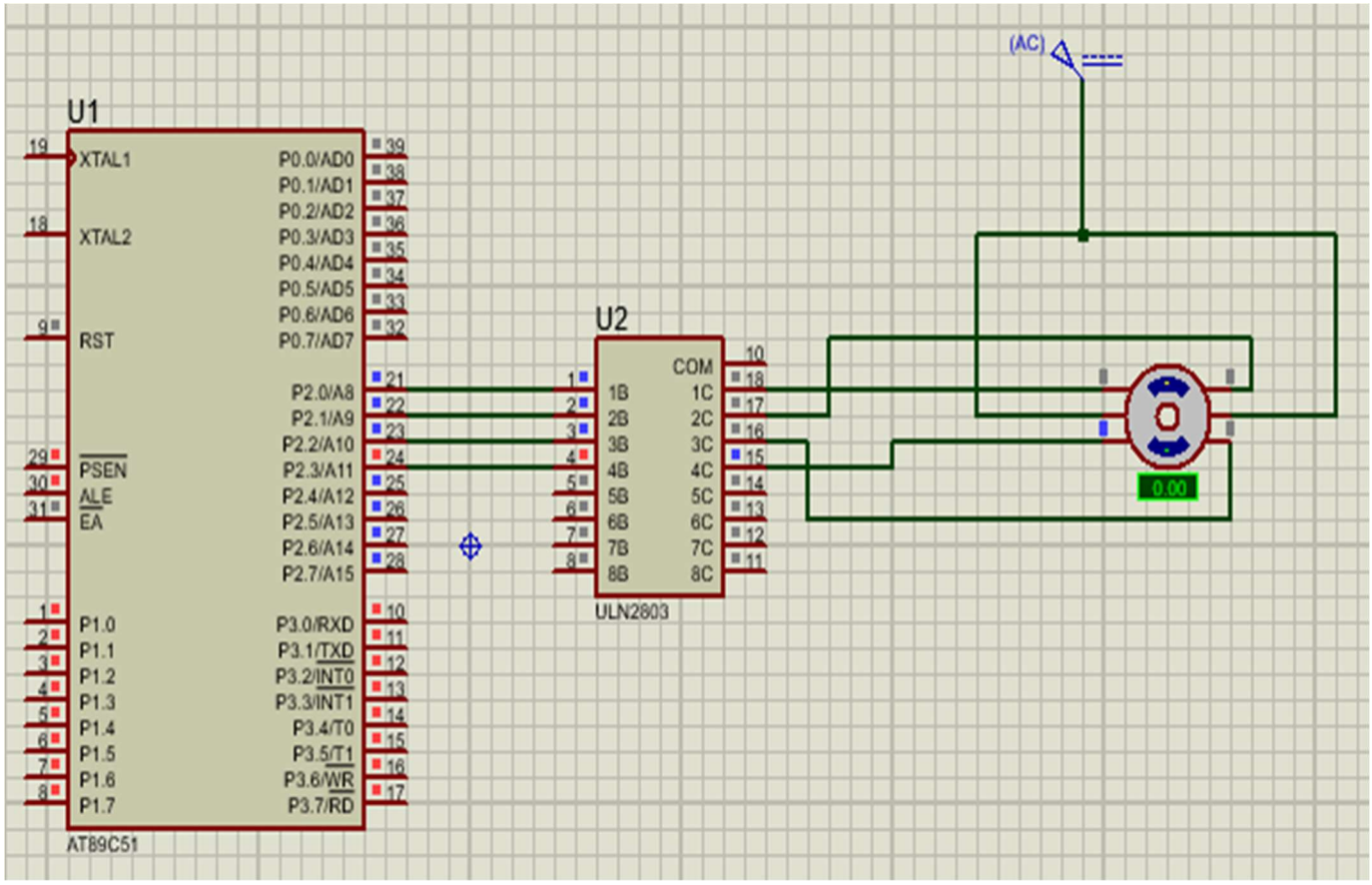


3.5.3 To Rotate 90 degree forward rotation and stop it

//If we want to rotate stepper motor 90 Degree than basic calculation is step angle is 1.8 then for 1 full rotation it took 7.2 i.e.(1.8*2), So now divide 90/7.2=12.5. So take a loop of 12.

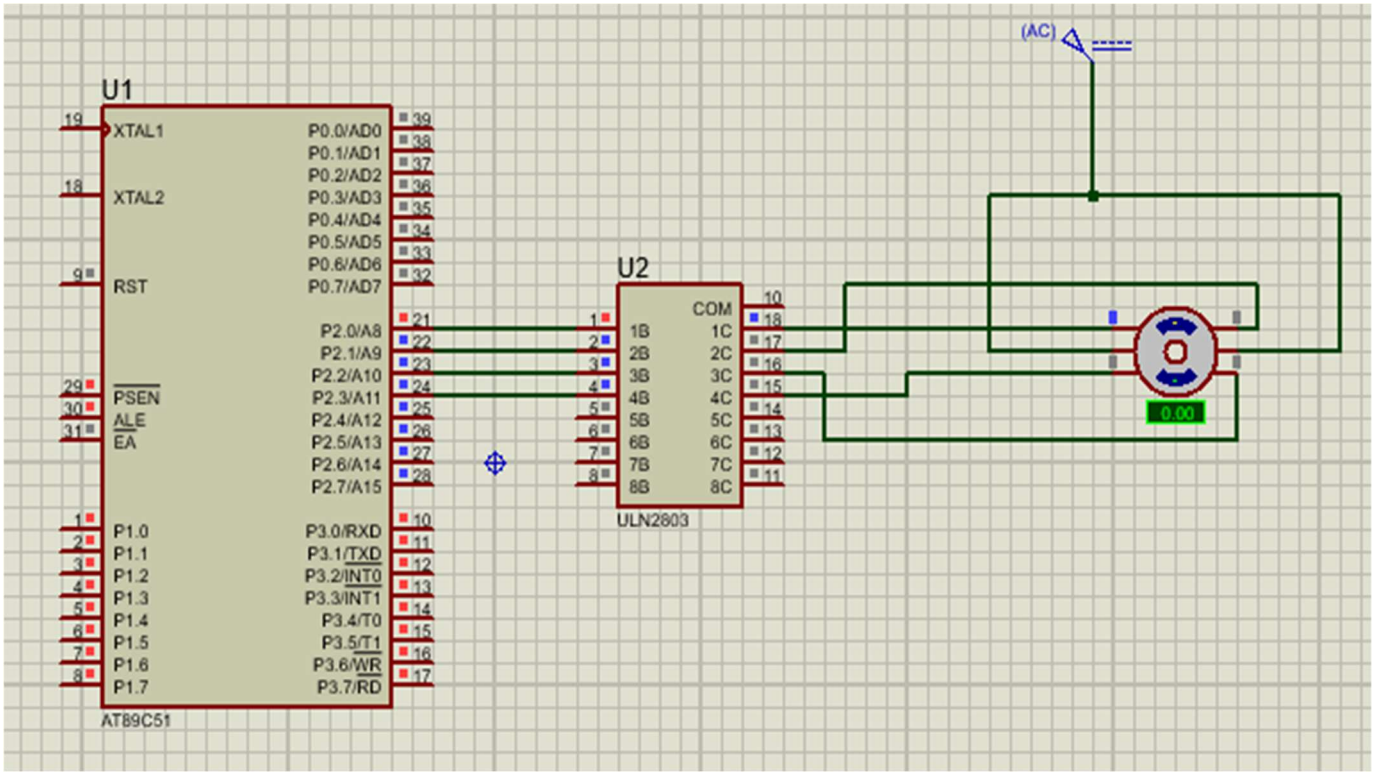
```
#include<reg51.h>
#define stpr P2

void delay(unsigned int n)
{
        unsigned int i;
        for(i=0;i<n;i++);
}
```

```
void main()
{
int i;
for(i=0;i<13;i++) // here we can keep 12,in proteus
                  // it will rotate 87 degree so we can keep 13.
{
        stpr=0x01;
        delay(30000);
        stpr=0x02;
        delay(30000);
        stpr=0x04;
        delay(30000);
        stpr=0x08;
        delay(30000);
        }
        while(1);
}
```



3.5.4 Full 360 Forward Rotation Using Array

```
#include<reg51.h>
#define stpr P2

void delay(unsigned int n)
{
        unsigned int i;
        for(i=0;i<n;i++);
}

void main()
{
 while(1)
 {
unsigned char x[]={0x01,0x02,0x04,0x08},i;
  for(i=0;i<4;i++)
  {
         stpr=x[i];
        delay(30000);
  }
 }
}
```



References

(1) https://media.geeksforgeeks.org/wp-content/uploads/20230302114603/Operators-in-C-(1)-768.png

(2) https://qph.cf2.quoracdn.net/main-qimg-b97ab7f502bb261744b0fa0d0cf99609-lq

(3) https://cdn.programiz.com/sites/tutorial2program/files/c-for-loop.jpg

(4) https://cdn.programiz.com/sites/tutorial2program/files/c-while-loop_0.jpg

(5) https://cdn.programiz.com/sites/tutorial2program/files/c-do-while-loop_0.jpg