



INSTITUTE OF ENGINEERING , CENTRAL CAMPUS,PULCHOWK

EMBEDDED SYSTEM

LAB #2

---

## Interfacing 7-Segment LED Display with 8051/8052 Micro-controller

---

**Submitted BY:**

Amrit Prasad Phuyal

Roll: PULL074BEX004

**Submitted To:**

Department of Electronics and  
Computer Engineering

November 6, 2020

Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Microcontroller . . . . .	1
1.2	8051 Microcontroller . . . . .	1
1.3	7-Segment LED Display . . . . .	2
1.4	Applications . . . . .	3
<b>2</b>	<b>Objective</b>	<b>3</b>
<b>3</b>	<b>Equipment Required</b>	<b>3</b>
<b>4</b>	<b>Circuit Description</b>	<b>4</b>
<b>5</b>	<b>LAB Problems</b>	<b>5</b>
5.1	Question -1 . . . . .	5
5.2	Question -2 . . . . .	6
5.3	Question -3 . . . . .	9
5.4	Question -4 . . . . .	11
5.5	Question -5 . . . . .	13
5.6	Question -6 . . . . .	14
<b>6</b>	<b>Discussion and Conclusion</b>	<b>16</b>

## List of Figures

1	Block diagram of 8051 microcontroller . . . . .	1
2	Common cathode vs Common anode 7 segment display . . . . .	2
3	Lookup table for Common anode and Common Cathode . . . . .	3
4	Circuit for Proteus Simulation . . . . .	4
5	Proteus output for count 0-9 and back to 0 . . . . .	6
6	Proteus output for count 00-20 . . . . .	8
7	Proteus output for count 20-00 . . . . .	9
8	Proteus output Fibonacci sequence of first 8 numbers . . . . .	11
9	Proteus output for Multiplication table of 7 . . . . .	13
10	Proteus output for Roll no. Display . . . . .	14
11	Proteus output for Scrolling roll no. . . . .	15

## List of Codes

1	Problem no. 1 Assembly . . . . .	5
2	Problem no. 1 C language . . . . .	5
3	Problem no. 2 Assembly . . . . .	7
4	Problem no. 2 C language . . . . .	7
5	Problem no. 3 Assembly . . . . .	10
6	Problem no. 3 C language . . . . .	11
7	Problem no. 4 Assembly . . . . .	12
8	Problem no. 4 C language . . . . .	12
9	Problem no. 5 Assembly . . . . .	13
10	Problem no. 5 C language . . . . .	14
11	Problem no. 6 Assembly . . . . .	15
12	Problem no. 6 C language . . . . .	15

# 1 Introduction

## 1.1 Microcontroller

A microcontroller is an integrated circuit (IC), usually via an MPU, memory and certain peripherals, to control other parts of an electronic system. These devices are optimized for embed-in applications that require agile and agile processing, digital, analog or electromechanical interactions.

## 1.2 8051 Microcontroller

In 1981, Intel introduced an 8-bit microcontroller called the 8051. It was referred as system on a chip because it had 128 bytes of RAM, 4K byte of on-chip ROM, two timers, one serial port, and 4 ports (8-bit wide), all on a single chip.

The different features of the 8051 microcontroller include:

- 4KB bytes on-chip program memory (ROM)
- 128 bytes on-chip data memory (RAM)
- Four register banks
- 128 user defined software flags
- 8-bit bidirectional data bus
- 16-bit unidirectional address bus
- 32 general purpose registers each of 8-bit
- 16 bit Timers (usually 2, but may have more or less)
- Three internal and two external Interrupts
- Four 8-bit ports,(short model have two 8-bit ports)
- 16-bit program counter and data pointer
- 8051 may also have a number of special features such as UARTs, ADC, Op-amp, etc.

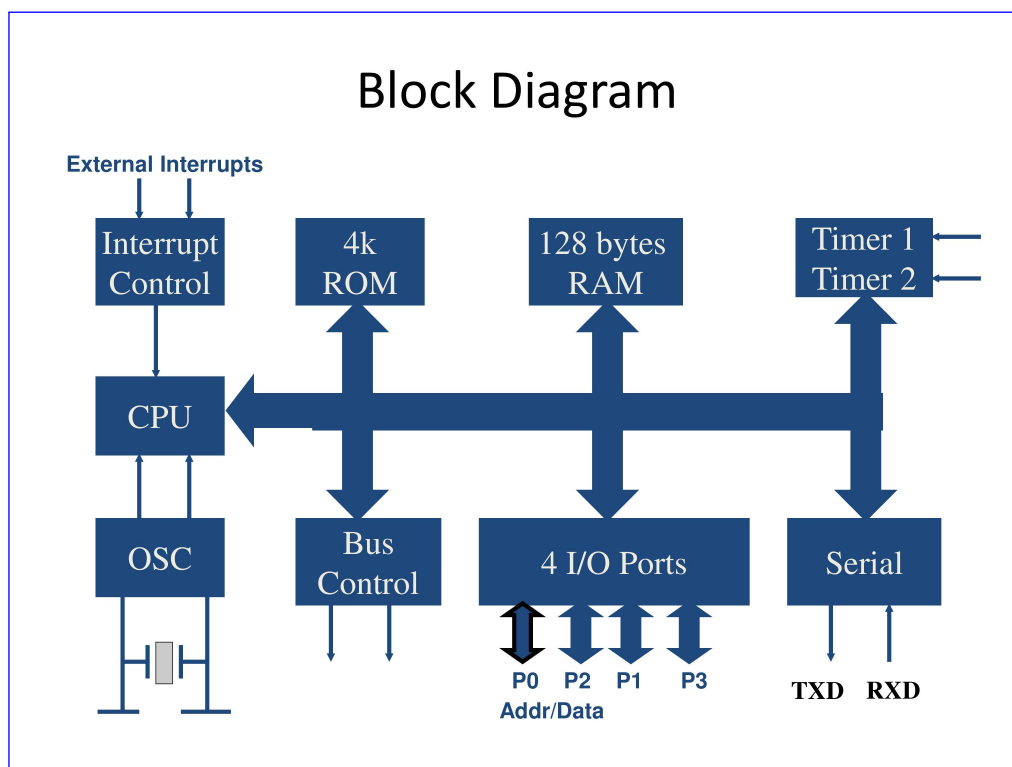


Figure 1: Block diagram of 8051 microcontroller

### 1.3 7-Segment LED Display

A seven segment display module is an electronic device used to display digital numbers and it is made up of seven LED segments. LEDs are PN-junction diodes which emit energy by a process called electroluminescence. 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. The light energy is emitted as 'photons' when it is forward biased by a voltage applied across its junctions. In a seven segment display module, seven LEDs 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.

Features of seven segment Display:-

- Available in two modes Common Cathode (CC) and Common Anode (CA)
- Available in many different sizes like 9.14mm,14.20mm,20.40mm,38.10mm,57.0mm and 100mm (Commonly used/available size is 14.20mm)
- Available colours: White, Blue, Red, Yellow and Green (Res is commonly used)
- Low current operation
- Better, brighter and larger display than conventional LCD displays.
- Current consumption : 30mA / segment
- Peak current : 70mA

The displays common pin is generally used to identify which type of 7-segment display it is. As each LED has two connecting pins, one called the "Anode" and the other called the "Cathode", there are therefore two types of LED 7-segment display called: Common Cathode (CC) and Common Anode (CA). The difference between the two displays, as their name suggests, is that the common cathode has all the cathodes of the 7-segments connected directly together and the common anode has all the anodes of the 7-segments connected together

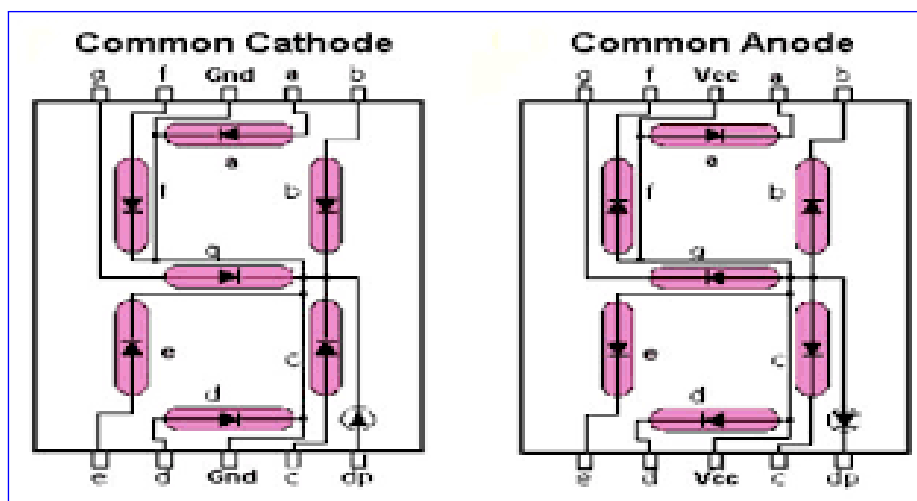


Figure 2: Common cathode vs Common anode 7 segment display

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 3: Lookup table for Common anode and Common Cathode

## 1.4 Applications

- Used in applications where font size is required to be bigger
- Microcontroller Independent, hence used in small circuit projects
- Used in combination with four segments to display measurement/sensor value with four characters
- Has bright illumination, hence used where display are required to work in low light or dark conditions

## 2 Objective

To enable us to write assembly language code for the 8051/8052 micro-controller capable of:

- Displaying non-multiplexed and multiplexed output on 7-segment LED units
- Displaying static and scrolling output on 7-segment LED units

## 3 Equipment Required

- Hardware: 8051 or 8052 micro-controller development board, Jumper cables
- Simulation Software: KEIL, Vision-Embedded development tool, Proteus Design Suite – Professional PCB layout, circuit design and simulation tool
- In-System Programming (ISP) Software: ProgISP – An in-system-programmable tool to load HEX files in to micro-controller
- Device Drivers: LibUSB – Application controlling data transfer to/from USB devices

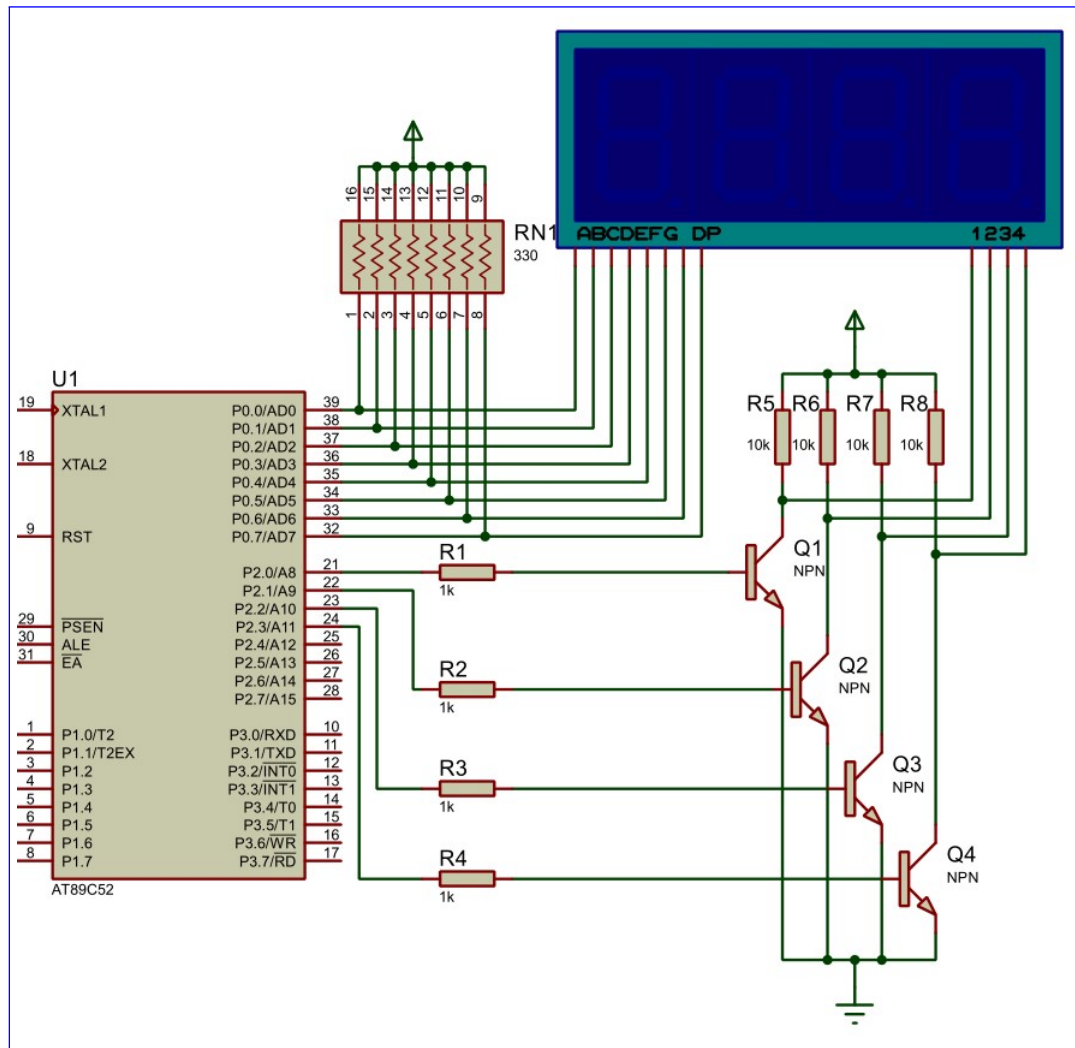


Figure 4: Circuit for Proteus Simulation

## 4 Circuit Description

The circuit diagram, consisting of micro-controller AT89C52 and four common cathode 7 segment display, used for simulation for this lab is shown below:

Figure shows that the common data lines, from the array of four seven segment display, are connected to the PORT 0 of microcontroller with array of 8 pull-up resistor. Here data line A is connected to P0.0 (LSB) whereas DP is connected to P0.7 (MSB). The control pins 1, 2, 3 and 4 are indirectly connected to P2.0, P2.1, P2.2 and P2.3 respectively. Since logic low should be applied to control pin to trigger corresponding segment, here transistor is used to invert the logic. It means in order to trigger a certain segment, let's say 1, logic high is applied to the connected pin, here P2.0, so that there will be logic low across the transistor where the control pin 1 is connected.

Now in order to display digits on more than one segment then illusion technique must be used. It means we have to give an illusion that multiple values are displayed at once on multiple 7-segment LED units using shared data lines. This illusion is created due to the persistence of vision as we know that human brain cannot differentiate between the two events occurring at a time difference of less than 40 milliseconds. Hence the data must be passed to the common data lines at a rate of about 60 to 100 times per second in order to avoid flickering. At the same time corresponding 7-segment units need to be turned ON or OFF.



## 5 LAB Problems

### 5.1 Question -1

Write a code to design a single digit decimal counter that counts up from 0 to 9 and back to 0. This process should repeat indefinitely.

#### Assembly

```

1  ORG 00H
2  ;HEX values for digits 0 to 9
3  MOV 40H,#3FH
4  MOV 41H,#06H
5  MOV 42H,#5BH
6  MOV 43H,#4FH
7  MOV 44H,#66H
8  MOV 45H,#6DH
9  MOV 46H,#7DH
10 MOV 47H,#07H
11 MOV 48H,#7FH
12 MOV 49H,#6FH
13
14 MOV P2,#01H
15 AGAIN: MOV R0,#40H
16         MOV R2,#0AH
17 C_INC:  MOV P0,@R0
18         INC R0
19
20         ACALL DELAY
21         DJNZ R2,C_INC
22         DEC R0 ;display from 8 to 0
23         MOV R2,#08H
24 C_DEC:  DEC R0
25         MOV P0,@R0
26         ACALL DELAY
27         DJNZ R2,C_DEC
28         AJMP AGAIN
29
30 DELAY:  MOV R3,#5
31         HERE1: MOV R4,#255
32         HERE2: MOV R5,#255
33         HERE3: DJNZ R5,HERE3
34         DJNZ R4,HERE2
35         DJNZ R3,HERE1
36         RET
37
38         END

```

Code 1: Problem no. 1 Assembly

#### C language

```

1 #include <reg51.h>
2 //HEX values for digits 0 to 9
3 unsigned char led_pattern[10] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f, 0x6f};
4
5 void delay(int time)
6 {
7     unsigned int i,j;
8     for (i=0;i<time;i++)
9         for (j=0;j<125;j++);
10 }
11
12 void display(int i)
13 {
14     P0 = led_pattern[i];
15     delay(1000);
16 }
17
18 void main(void)
19 {
20     unsigned int i;
21     P2 = 0x01;
22
23     while(1)
24     {
25         for(i=0; i<10; i++)
26             display(i);
27         for(i=8; i>0; i--) //display from 8 to 0
28             display(i);
29     }
30 }

```

Code 2: Problem no. 1 C language

#### OUTPUT:

Each output is taken from Proteus Simulator using delay in code.

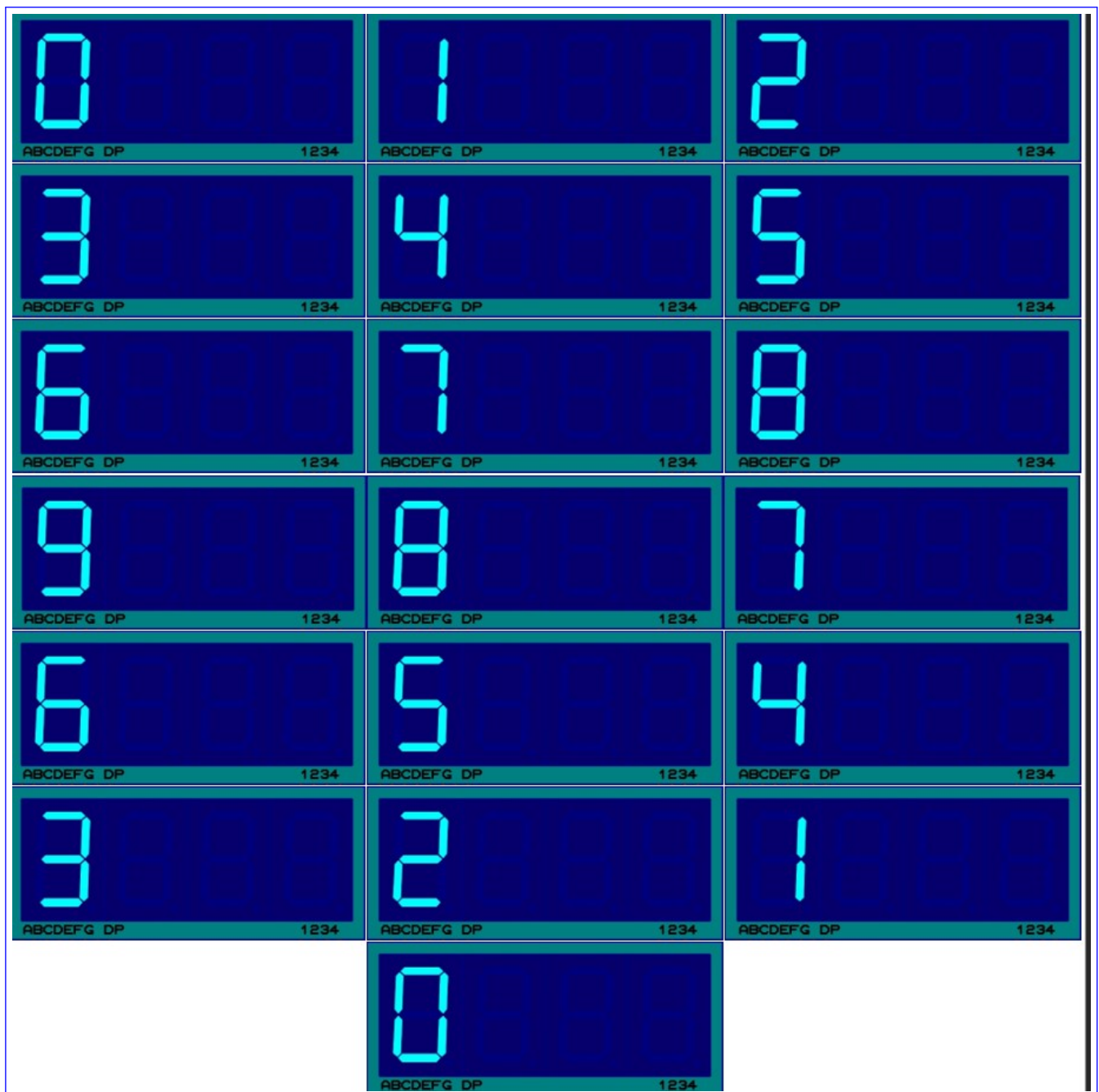


Figure 5: Proteus output for count 0-9 and back to 0

## 5.2 Question -2

Write a code to design a double digit decimal counter that counts up from 00 to 20 and back to 00 indefinitely.

### Assembly

```

1      ORG 00H
2
3      MOV 40H,#3FH
4      MOV 41H,#06H
5      MOV 42H,#5BH
6      MOV 43H,#4FH
7      MOV 44H,#66H
8      MOV 45H,#6DH
9      MOV 46H,#7DH
10     MOV 47H,#07H

```

```

11     MOV 48H,#7FH
12     MOV 49H,#6FH
13     MOV 4AH,#3FH
14
15     MOV 50H,40H
16     MOV 51H,41H
17     MOV 52H,42H
18
19     AGAIN: MOV R1,#50H
20

```

```

21      MOV R6,#02H
22 LOOP2: MOV R0,#40H
23      MOV R5,#0AH
24 LOOP1: MOV R7,#255
25 MAIN:  MOV A,@R1
26      MOV P2,#01H
27      MOV P0,A
28      ACALL DELAY
29      MOV A,@R0
30      MOV P2,#02H
31      MOV P0,A
32      ACALL DELAY
33      DJNZ R7,MAIN
34      INC R0
35      DJNZ R5,LOOP1
36      INC R1
37      DJNZ R6,LOOP2
38
39      MOV R7,#255
40 LOP:   MOV A,@R1
41      MOV P2,#01H
42      MOV P0,A
43      ACALL DELAY
44      MOV A,@R0
45      MOV P2,#02H
46      MOV P0,A
47      ACALL DELAY
48      DJNZ R7,LOP
49      DEC R1
50
51      MOV R6,#02H
52 LOOP22: MOV R0,#49H
53      MOV R5,#0AH
54 LOOP11: MOV R7,#255
55 MAIN_D: MOV A,@R1
56      MOV P2,#01H
57      MOV P0,A
58      ACALL DELAY
59      MOV A,@R0
60      MOV P2,#02H
61      MOV P0,A
62      ACALL DELAY
63      DJNZ R7,MAIN_D
64      DEC R0
65      DJNZ R5,LOOP11
66      DEC R1
67      DJNZ R6,LOOP22
68      AJMP AGAIN
69
70 DELAY: MOV R3,#02H
71 DEL1:  MOV R2,#0FAH
72 DEL2:  DJNZ R2,DEL2
73      DJNZ R3,DEL1
74      RET
75
76      END
77

```

Code 3: Problem no. 2 Assembly

### C language

```

1  #include <reg51.h>
2  unsigned char led_pattern[10] = { 0x3f, 0x06,
3                                     0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
4                                     0x6f};
5
6  void delay(int time)
7  {
8      unsigned int i,j;
9      for (i=0;i<time;i++)
10         for (j=0;j<125;j++);
11 }
12
13 void display(unsigned int i)
14 {
15     unsigned int j, led[2];
16     led[0] = i / 10;
17     led[1] = i % 10;
18     for(j=0; j<10; j++) //Delay between two
19         consecutive numbers
20         for(i=0;i<2;i++)
21
22     {
23         P2 = 0x1 * (i + 1);
24         P0 = led_pattern[led[i]];
25         delay(40); //selected to avoid
26         flickering
27     }
28 }
29
30 void main(void)
31 {
32     unsigned int i;
33     while(1)
34     {
35         for(i=0; i<20; i++)
36             display(i);
37         for(i=20; i>0; i--)
38             display(i);
39     }
40 }

```

Code 4: Problem no. 2 C language

### OUTPUT:

Here two part of total four 7-segment LEDs is used to count from 00-20 and then back to 00

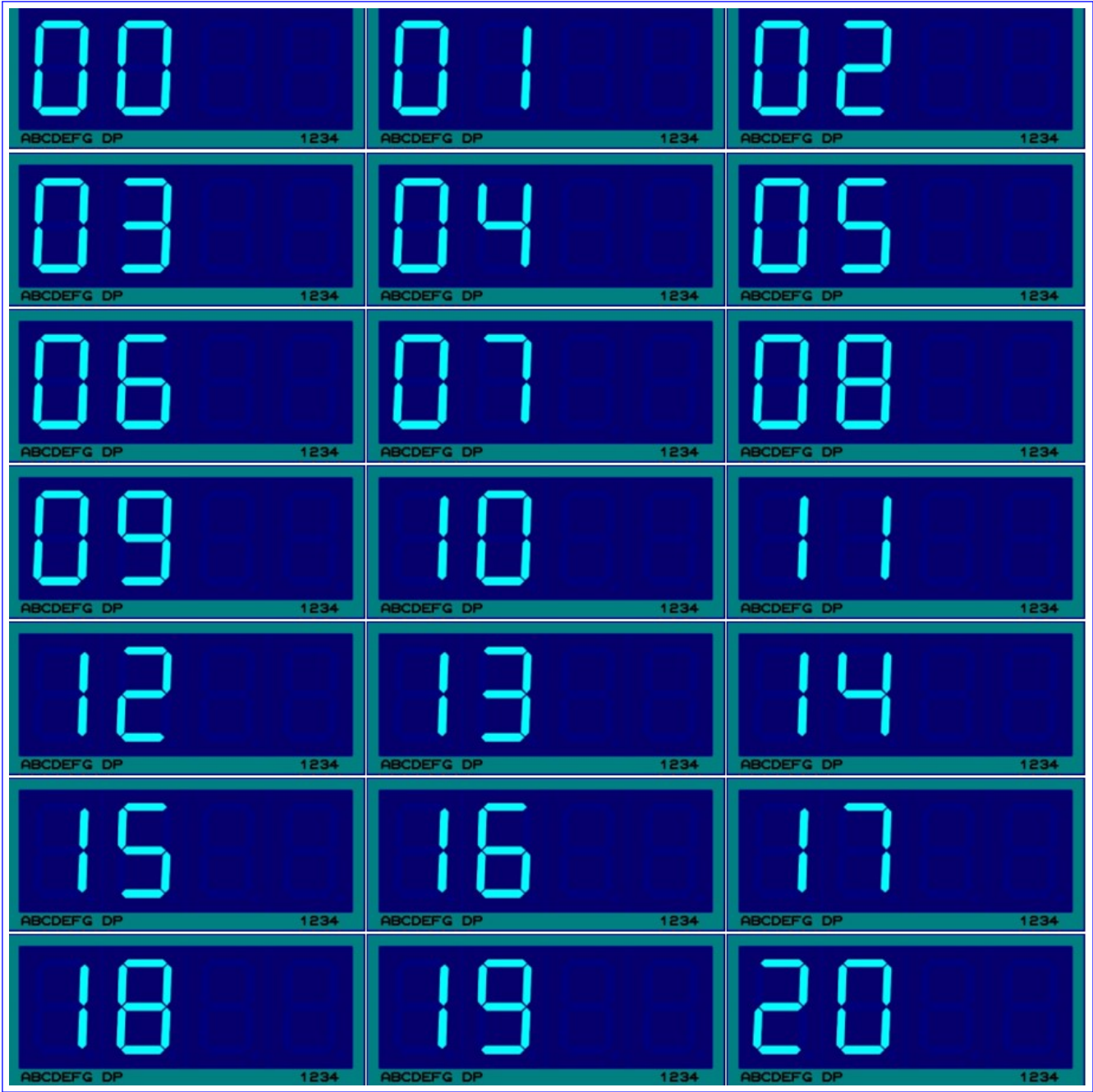


Figure 6: Proteus output for count 00-20

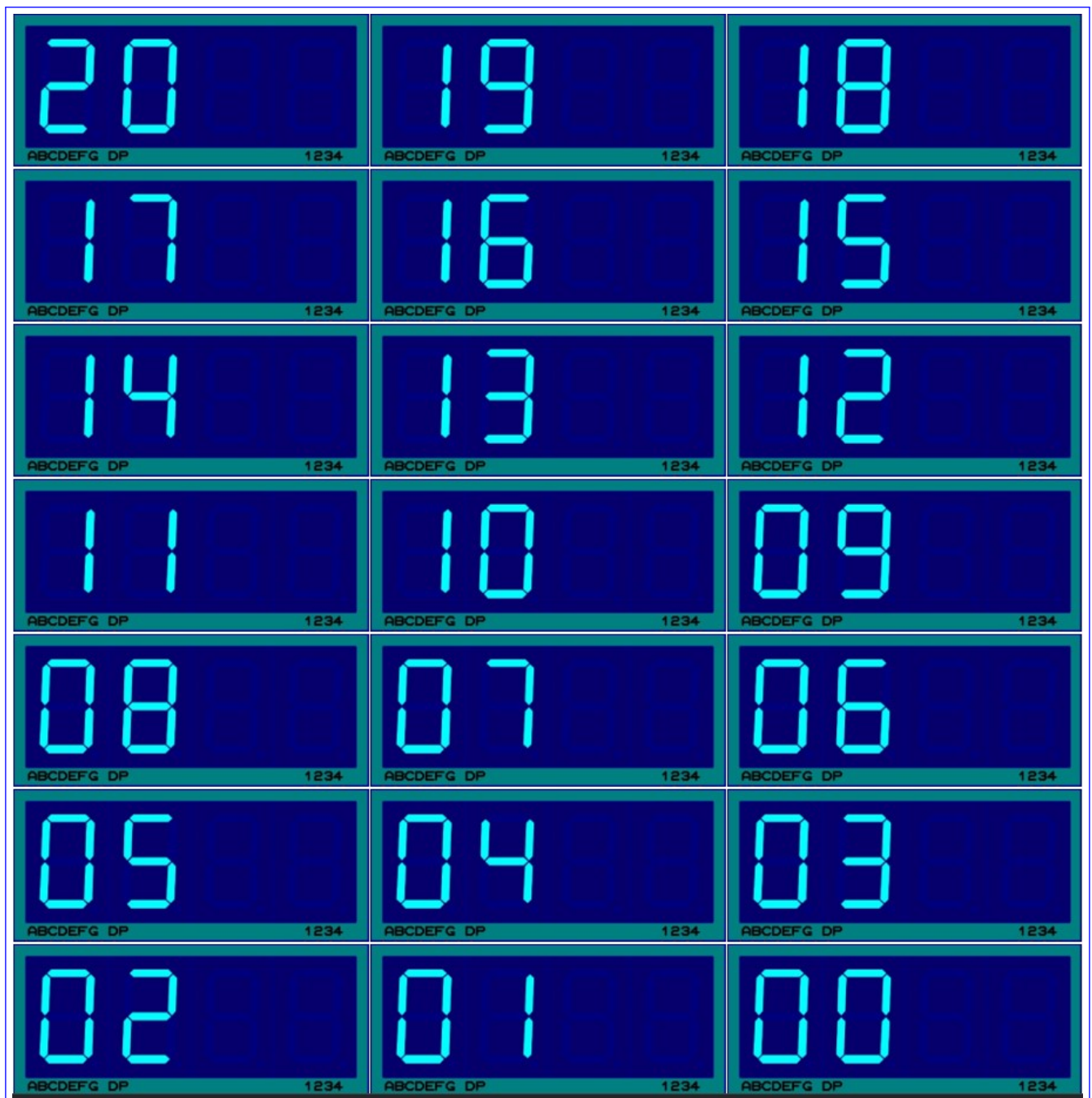


Figure 7: Proteus output for count 20-00

### 5.3 Question -3

Write a code to display the first (N) numbers of the Fibonacci sequence, where the number (N) must be stored in a memory location and can be any integer from 1 to 10. The sequence should repeat indefinitely.

#### Assembly

```

1 ;FIBONACCI SEQUENCE
2     ORG 00H
3
4     MOV P2,#00H
5     MOV DPTR,#LABEL1
6     MOV R0,#50H
7     MOV R7,#8           ;Number of terms
8     (N=8)
9     MOV A,R7
10    MOV R6,A
11    ; Display 1st and 2nd numbers of sequence
12    MOV R1,#00H
13    MOV R2,#01H
14    MOV A,R1
15    MOV @R0,A
16    INC R0
17    DEC R6
18    MOV A,R2
19    MOV @R0,A

```

```

20      INC R0
21      DEC R6
22
23 ;Add consecutive terms to get next term
24 AGAIN: MOV A,R1
25        ADD A,R2
26        MOV @R0,A
27        INC R0
28        MOV B,R2
29        MOV R1,B
30        MOV R2,A
31        DJNZ R6,AGAIN
32
33 ;HEX to DEC conversion and store in
    memory
34      MOV R0,#50H
35      MOV A,R7
36      MOV R6,A
37
38 AGN2:  MOV A,@R0
39      MOV R4,#00H
40      MOV B,#0AH
41      DIV AB
42      MOV R2,A
43      SUBB A,#0AH
44      JC SKIP
45      MOV A,R2
46      MOV R3,B
47      MOV B,#0AH
48      DIV AB
49      MOV R4,A
50      MOV A,B
51      MOV B,R3
52
53 SKIP:  MOV A,R2
54      SWAP A
55      ADD A,B
56      MOV B,R4
57
58      MOV @R0,A
59      INC R0
60      DJNZ R6,AGN2
61
62 REPEAT: MOV R0,#50H
63         MOV A,R7
64         MOV R4,A
65 LOOP1: MOV R6,#255
66 MAIN:  MOV A,@R0
67         MOV B,A
68         ANL A,#0FH
69         MOV P2,#02H
70         ACALL DISPLAY
71         MOV P0,A
72         ACALL DELAY
73
74         MOV A,B
75         ANL A,#0F0H
76         SWAP A
77         MOV P2,#01H
78         ACALL DISPLAY
79         MOV P0,A
80         ACALL DELAY
81
82         DJNZ R6,MAIN
83         INC R0
84         DJNZ R4,LOOP1
85         AJMP REPEAT
86
87 DELAY:  MOV R3,#02H
88 DEL1:   MOV R2,#0FAH
89 DEL2:   DJNZ R2,DEL2
90         DJNZ R3,DEL1
91         RET
92
93
94 DISPLAY: MOV A,@A+DPTR
95         RET
96
97 ;Lookup table
98 LABEL1: DB 3FH
99         DB 06H
100        DB 5BH
101        DB 4FH
102        DB 66H
103        DB 6DH
104        DB 7DH
105        DB 07H
106        DB 7FH
107        DB 6FH
108
109        END

```

Code 5: Problem no. 3 Assembly

### C language

```

1  #include <reg51.h>
2  #define N 8
3  unsigned char led_pattern[10] = { 0x3f, 0x06,
    0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
    0x6f};
4
5  void delay(int time)
6  {
7      unsigned int i,j;
8      for (i=0;i<time;i++)
9          for (j=0;j<125;j++);
10 }
11
12 void display(unsigned int i)
13 {
14     unsigned int j, led1, led2;
15     led1 = i / 10;
16     led2 = i % 10;
17     for(j=0; j<10; j++)//Delay between
        consecutive terms
18     {
19         P2 = 0x1;
20         P0 = led_pattern[led1];
21         delay(40);//selected to avoid flickering
22
23         P2 = 0x2;
24         P0 = led_pattern[led2];
25         delay(40);

```



```

26     }
27 }
28
29 void main(void)
30 {
31     unsigned int i, fibo_seq[N] = {0 , 1};
32     for(i=2; i<N; i++)
33         fibo_seq[i] = fibo_seq[i-1] + fibo_seq[i-2];
34     while(1)
35         for(i=0; i<N; i++)
36             display(fibo_seq[i]);
37 }

```

Code 6: Problem no. 3 C language

**OUTPUT:**

Fibonacci sequence of first 8 numbers is shown below:

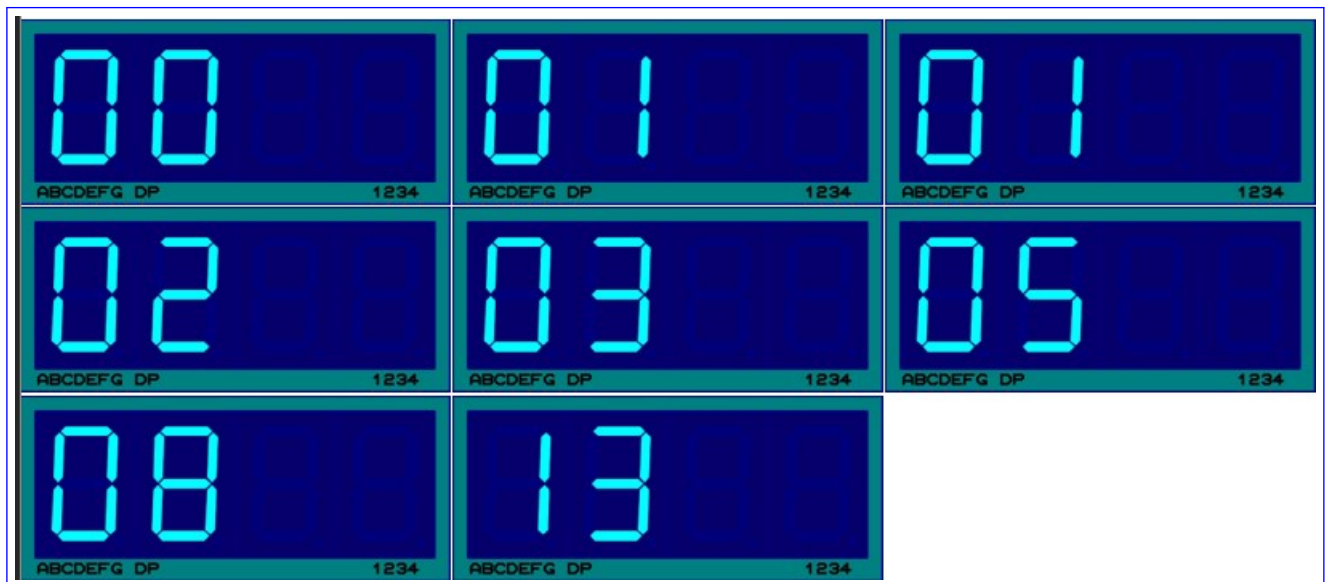


Figure 8: Proteus output Fibonacci sequence of first 8 numbers

**5.4 Question -4**

Write a code to generate the multiplication table of a number (N) stored in a memory location which can be any integer from 1 to 10. Repeat the sequence indefinitely.

**Assembly**

```

1      ORG 00H
2
3      MOV R7 ,#7 ;N=7
4      MOV P2 ,#00H
5      MOV DPTR ,#LABEL1
6
7      MOV B ,R7
8      MOV R0 ,#5AH
9      MOV R6 ,#10
10     AGN:  MOV B ,R6
11           MOV A ,R7
12           MUL AB
13           MOV @R0 ,A
14           DEC R0
15           DJNZ R6 ,AGN
16
17     ;HEX TO DEC conversion and store in
18     memory
19     MOV R0 ,#51H
20     MOV R6 ,#10
21     AGN2: MOV A ,@R0
22           MOV R4 ,#00H
23
24           MOV B ,#0AH
25           DIV AB
26           MOV R2 ,A
27           SUBB A ,#0AH
28           JC SKIP
29           MOV A ,R2
30           MOV R3 ,B
31           MOV B ,#0AH
32           DIV AB
33           MOV R4 ,A
34           MOV A ,B
35           MOV R2 ,A
36     SKIP: MOV A ,R2
37           SWAP A
38           ADD A ,B
39           MOV B ,R4
40
41           MOV @R0 ,A
42           INC R0
43           DJNZ R6 ,AGN2
44
45     REPEAT: MOV R0 ,#51H

```

```

46      MOV R4 ,#10
47 LOOP1: MOV R7 ,#255
48 MAIN:  MOV A ,@R0
49        MOV B ,A
50        ANL A ,#0FH
51        MOV P2 ,#02H
52        ACALL DISPLAY
53        MOV P0 ,A
54        ACALL DELAY
55        MOV A ,B
56        ANL A ,#0FOH
57        SWAP A
58        MOV P2 ,#01H
59        ACALL DISPLAY
60        MOV P0 ,A
61        ACALL DELAY
62        DJNZ R7 ,MAIN
63        INC R0
64        DJNZ R4 ,LOOP1
65        AJMP REPEAT
66
67 DELAY:  MOV R3 ,#02H
68 DEL1:   MOV R2 ,#0FAH
69 DEL2:   DJNZ R2 ,DEL2
70         DJNZ R3 ,DEL1
71         RET
72
73 DISPLAY: MOV A ,@A+DPTR
74         RET
75
76 ;Lookup table
77 LABEL1: DB 3FH
78         DB 06H
79         DB 5BH
80         DB 4FH
81         DB 66H
82         DB 6DH
83         DB 7DH
84         DB 07H
85         DB 7FH
86         DB 6FH
87
88         END

```

Code 7: Problem no. 4 Assembly

### C language

```

1  #include <reg51.h>
2  #define N 7 //N=7
3  unsigned char led_pattern[10] = { 0x3f, 0x06,
4                                     0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
5                                     0x6f};
6
7 void delay(int time)
8 {
9     unsigned int i,j;
10    for (i=0;i<time;i++)
11        for (j=0;j<125;j++);
12 }
13 void display(unsigned int i)
14 {
15     unsigned int j;
16     for(j=0; j<15; j++)
17     {
18         P2 = 0x1;
19         P0 = led_pattern[i / 10];
20         delay(40);
21
22         P2 = 0x2;
23         P0 = led_pattern[i % 10];
24         delay(40);
25     }
26 }
27 void main(void)
28 {
29     unsigned int i;
30     while(1)
31         for(i=1; i<=10; i++)
32             display(N*i);
33 }

```

Code 8: Problem no. 4 C language

### OUTPUT:

Multiplication table for 7 is shown below



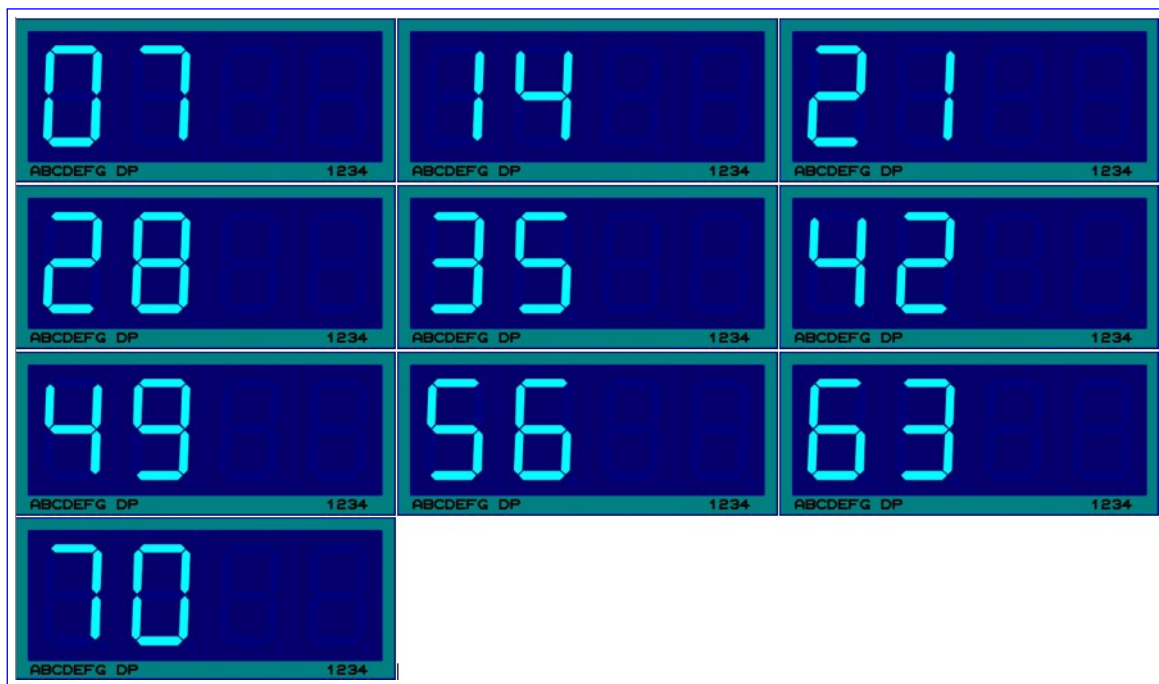


Figure 9: Proteus output for Multiplication table of 7

### 5.5 Question -5

Write a code to display the roll numbers of your lab group members one by one in static format. Each student roll number should be of four characters. Display of student roll numbers should repeat indefinitely.

#### Assembly

```

1      ORG 00H
2 ;Digital drive pattern for 403
3      MOV 40H,#79H
4      MOV 41H,#66H
5      MOV 42H,#3FH
6      MOV 43H,#4fH
7
8 REPEAT: MOV R0,#40H
9         MOV A,@R0
10        SETB P2.0
11        MOV P0,A
12        ACALL DELAY
13        CLR P2.0
14        INC R0
15
16        MOV A,@R0
17        SETB P2.1
18        MOV P0,A
19        ACALL DELAY
20        CLR P2.1
21        INC R0
22
23        MOV A,@R0
24
25        SETB P2.2
26        MOV P0,A
27        ACALL DELAY
28        CLR P2.2
29        INC R0
30
31        MOV A,@R0
32        SETB P2.3
33        MOV P0,A
34        ACALL DELAY
35        CLR P2.3
36
37        AJMP REPEAT
38
39 DELAY: MOV R3,#02H
40 DEL1:  MOV R2,#0FAH
41 DEL2:  DJNZ R2,DEL2
42        DJNZ R3,DEL1
43        RET
44
45        END

```

Code 9: Problem no. 5 Assembly

#### C language

```

1 #include <reg51.h>
2
3 unsigned char led_pattern[4] = {0x79,0x66, 0
   x3f, 0xcf};
4
5
6 void delay(int time)
7 {
8     unsigned int i,j;
9     for (i=0; i<time; i++)
10         for (j=0; j<125; j++);
11 }
12
13 void display()
14 {
15     P2 = 0x1;
16     P0 = led_pattern[0];
17     delay(10); //selected to avoid flickering
               and show as static
18
19     P2 = 0x2;
20     P0 = led_pattern[1];
21     delay(10);
22
23     P2 = 0x4;
24     P0 = led_pattern[2];
25     delay(10);
26
27     P2 = 0x8;
28     P0 = led_pattern[3];
29     delay(10);
30 }
31
32 void main(void)
33 {
34     while(1)
35         display();
36 }

```

Code 10: Problem no. 5 C language

**OUTPUT:**

Here E is used for Electronics and 403 is my class Roll no.



Figure 10: Proteus output for Roll no. Display

**5.6 Question -6**

Write a code to display the roll numbers of your lab group members in scrolling format, separated by using decimal point. Roll numbers should be scrolled towards the left and is repeated indefinitely.

**Assembly**

```

1      ORG 00H
2      MOV 40H,#79H
3      MOV 41H,#66H
4      MOV 42H,#3FH
5      MOV 43H,#4fH
6      MOV 44H,#79H
7      MOV 45H,#66H
8      MOV 46H,#3FH
9
10 REPEAT: MOV R0,#40H
11         MOV R4,#04H
12 LOOP1: MOV R7,#255
13 MAIN:  MOV A,@R0
14         SETB P2.0
15         MOV P0,A
16         ACALL DELAY
17         CLR P2.0
18
19         INC R0
20
21         MOV A,@R0
22         SETB P2.1
23         MOV P0,A
24         ACALL DELAY
25         CLR P2.1
26         INC R0
27
28         MOV A,@R0
29         SETB P2.2
30         MOV P0,A
31         ACALL DELAY
32         CLR P2.2
33         INC R0
34
35         MOV A,@R0

```

```

35      SETB P2.3
36      MOV P0,A
37      ACALL DELAY
38      CLR P2.3
39      ;Scrolling happens here
40      DEC R0
41      DEC R0
42      DEC R0
43
44      DJNZ R7,MAIN
45
46      INC R0
47
48      DJNZ R4,LOOP1
49      AJMP REPEAT
50
51 DELAY: MOV R3,#02H
52 DEL1:  MOV R2,#0FAH
53 DEL2:  DJNZ R2,DEL2
54        DJNZ R3,DEL1
55        RET
56
57      END

```

Code 11: Problem no. 6 Assembly

## C language

```

1  #include <reg51.h>
2
3  unsigned char scroll_pattern[8] = { 0x79,0x66,
4                                     0x3f, 0xcf,
5                                     0x79,0x66, 0x3f, 0
6                                     xcf};
7
8 void delay(int time)
9 {
10     unsigned int i,j;
11     for (i=0; i<time; i++)
12         for (j=0; j<125; j++);
13 }
14
15 void display(unsigned int i)
16 {
17     unsigned int j;
18     for(j=0; j<150; j++)
19     {
20         P2 = 0x1;
21         P0 = scroll_pattern[i-4];
22         delay(10); //selected to avoid flickering
23
24         P2 = 0x2;
25         P0 = scroll_pattern[i-3];
26         delay(10);
27
28         P2 = 0x4;
29         P0 = scroll_pattern[i-2];
30         delay(10);
31
32         P2 = 0x8;
33         P0 = scroll_pattern[i-1];
34         delay(10);
35     }
36 }
37
38 void main(void)
39 {
40     unsigned int i;
41     while(1)
42     {
43         for(i=4; i<8; i++)
44             display(i); //scrolling happens here
45     }
46 }

```

Code 12: Problem no. 6 C language

## OUTPUT:

My Roll no. E403. is Shown in scrolling Format and scrolling toward left.



Figure 11: Proteus output for Scrolling roll no.

## 6 Discussion and Conclusion

In this Experiment , we familiarize ourself with seven segment display and its operation through 8051 microcontroller. We performed single digit count, double digit count, generate Multiplication Table, Fibonacci sequence and scrolling effect of certain number. we used both Assembly and C language approach to achieve the above task.