INSTITUTE OF ENGINNERING , 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

List of Figures

List of Codes

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.

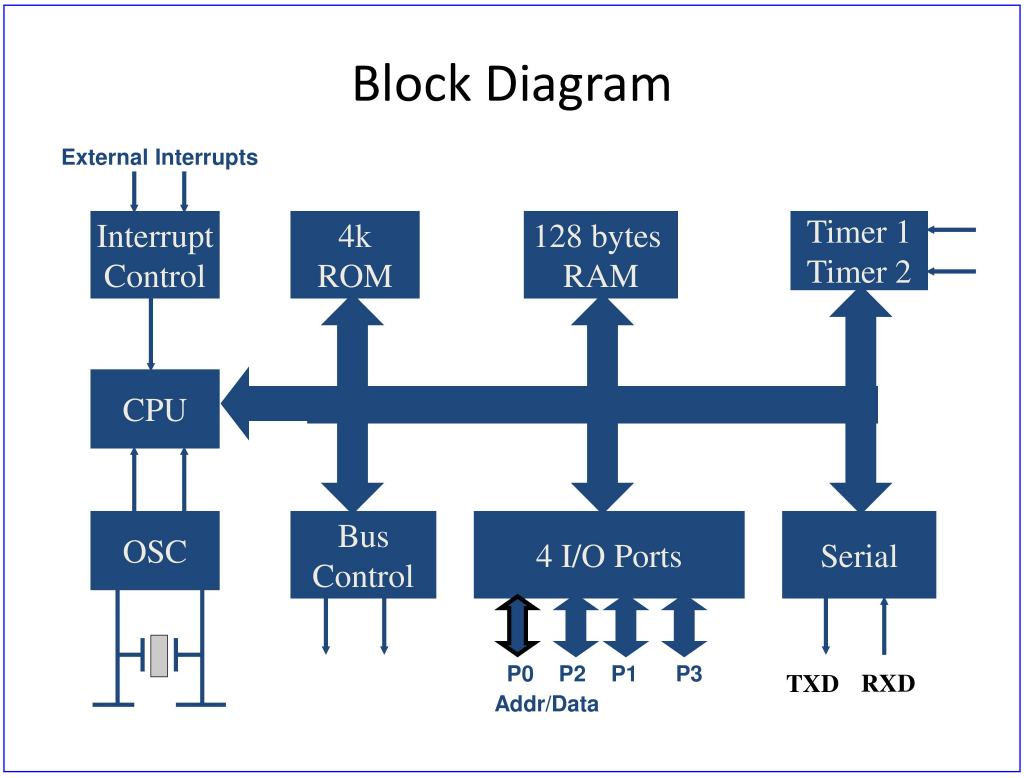

Figure 1: Block diagram of 8051 microcontroller

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 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.

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

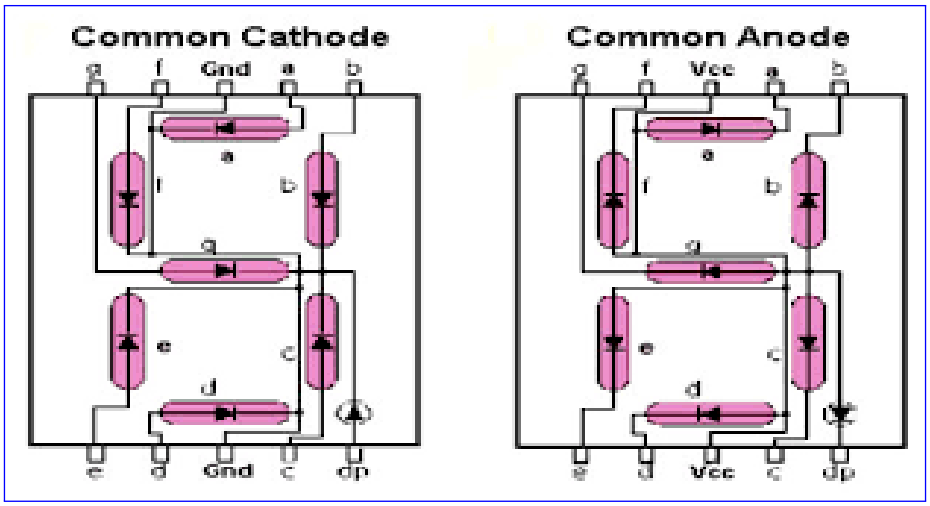

Figure 2: Common cathode vs Common anode 7 segment display

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

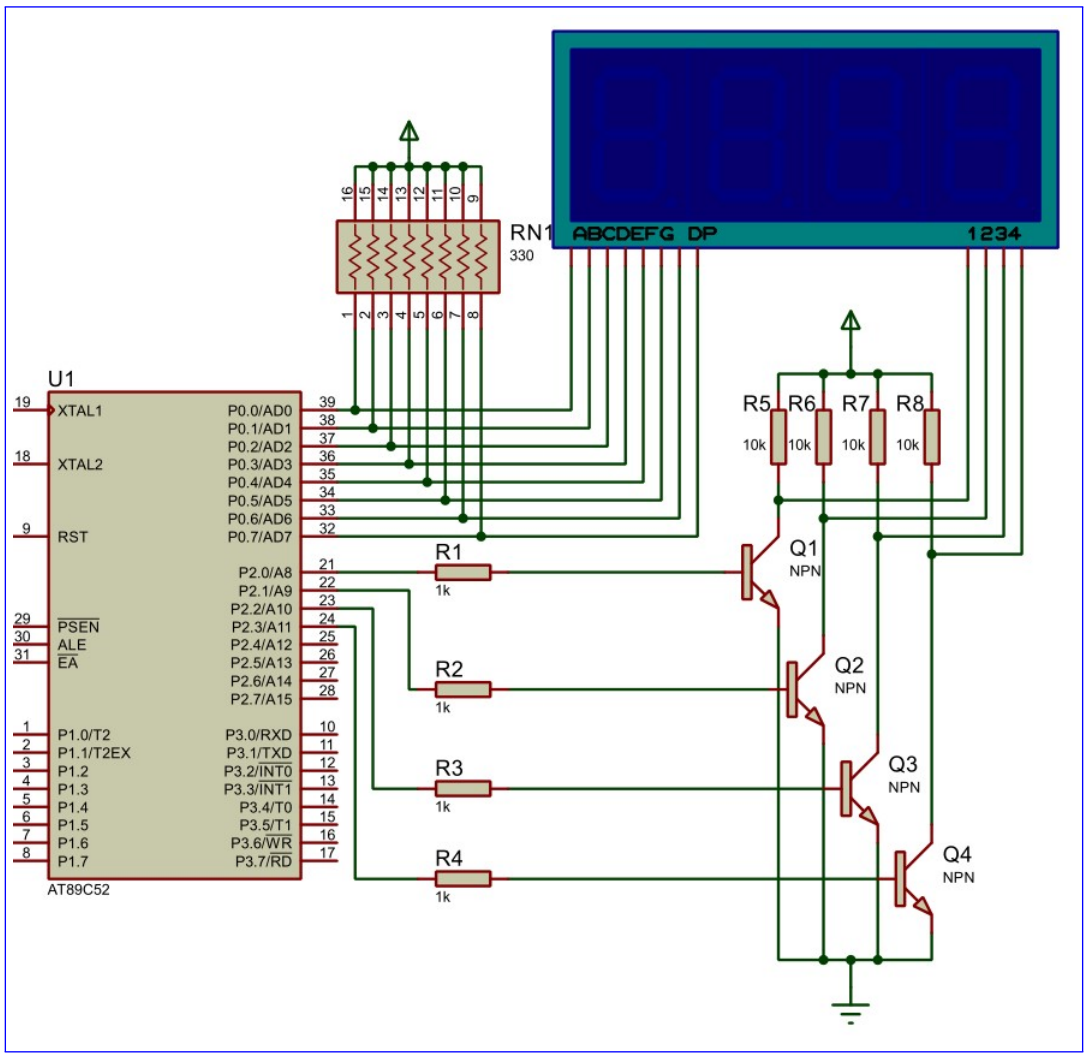

Figure 4: Circuit for Proteus Simulation

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 most 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

```
    ORG 00H
        ;HEX values for digits 0 to 9
        MOV 40H,#3FH
        MOV 41H,#06H
        MOV 42H,#5BH
        MOV 43H,#4FH
        MOV 44H,#66H
        MOV 45H,#6DH
        MOV 46H,#7DH
        MOV 47H,#07H
        MOV 48H,#7FH
        MOV 49H,#6FH

        MOV P2,#01H
AGAIN:  MOV R0,#40H
        MOV R2,#0AH
C_INC:  MOV P0,@R0
        INC R0
        ACALL DELAY
        DJNZ R2,C_INC
        DEC R0 ;display from 8 to 0
        MOV R2,#08H
C_DEC:  DEC R0
        MOV P0,@R0
        ACALL DELAY
        DJNZ R2,C_DEC
        AJMP AGAIN
DELAY:  MOV R3,#5
HERE1:  MOV R4,#255
HERE2:  MOV R5,#255
HERE3:  DJNZ R5,HERE3
        DJNZ R4,HERE2
        DJNZ R3,HERE1
        RET

        END
```

Code 1: Problem no. 1 Assembly

C language

```
#include <reg51.h>
//HEX values for digits 0 to 9
unsigned char led_pattern[10] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f, 0x6f};

void delay(int time)
{
   unsigned int i,j;
   for (i=0;i<time;i++)
      for (j=0;j<125;j++);
}

void display(int i)
{
   P0 = led_pattern[i];
   delay(1000);
}

void main(void)
{
   unsigned int i;
   P2 = 0x01;

   while(1)
   {
      for(i=0; i<10; i++)
         display(i);
      for(i=8; i>0; i--)//display from 8 to 0
         display(i);
   }
}
```

Code 2: Problem no. 1 C language

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

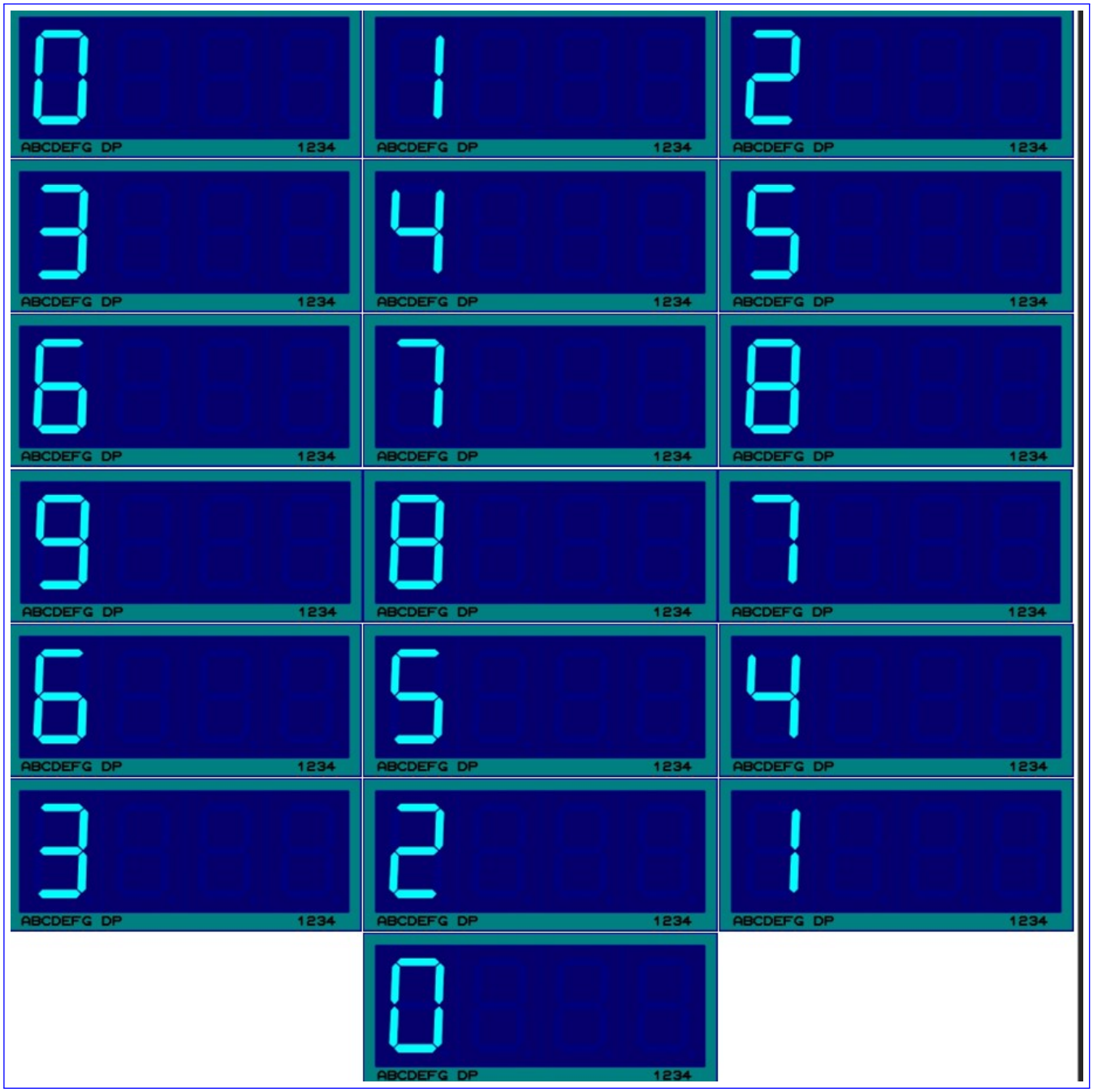

Figure 5: Proteus output for count 0-9 and back to 0

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

```
        ORG 00H

        MOV 40H,#3FH
        MOV 41H,#06H
        MOV 42H,#5BH
        MOV 43H,#4FH
        MOV 44H,#66H
        MOV 45H,#6DH
        MOV 46H,#7DH
        MOV 47H,#07H
        MOV 48H,#7FH
        MOV 49H,#6FH
        MOV 4AH,#3FH

        MOV 50H,40H
        MOV 51H,41H
        MOV 52H,42H

AGAIN:  MOV R1,#50H
```

```
        MOV R6,#02H
LOOP2:  MOV R0,#40H
        MOV R5,#0AH
LOOP1:  MOV R7,#255
MAIN:   MOV A,@R1
        MOV P2,#01H
        MOV P0,A
        ACALL DELAY
        MOV A,@R0
        MOV P2,#02H
        MOV P0,A
        ACALL DELAY
        DJNZ R7,MAIN
        INC R0
        DJNZ R5,LOOP1
        INC R1
        DJNZ R6,LOOP2

        MOV R7,#255
LOP:    MOV A,@R1
        MOV P2,#01H
        MOV P0,A
        ACALL DELAY
        MOV A,@R0
        MOV P2,#02H
        MOV P0,A
        ACALL DELAY
        DJNZ R7,LOP
        DEC R1
```

```

        MOV R6,#02H
LOOP22: MOV R0,#49H
        MOV R5,#0AH
LOOP11: MOV R7,#255
MAIN_D: MOV A,@R1
        MOV P2,#01H
        MOV P0,A
        ACALL DELAY
        MOV A,@R0
        MOV P2,#02H
        MOV P0,A
        ACALL DELAY
        DJNZ R7,MAIN_D
        DEC R0
        DJNZ R5,LOOP11
        DEC R1
        DJNZ R6,LOOP22
        AJMP AGAIN

DELAY:  MOV R3,#02H
DEL1:   MOV R2,#0FAH
DEL2:   DJNZ R2,DEL2
        DJNZ R3,DEL1
        RET

        END
```

Code 3: Problem no. 2 Assembly

C language

```
#include <reg51.h>
unsigned char led_pattern[10] = { 0x3f, 0x06,
    0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
    0x6f};

void delay(int time)
{
   unsigned int i,j;
   for (i=0;i<time;i++)
      for (j=0;j<125;j++);
}

void display(unsigned int i)
{
   unsigned int j, led[2];
   led[0] = i / 10;
   led[1] = i % 10;
   for(j=0; j<10; j++)  //Delay between two
    consecutive numbers
      for(i=0;i<2;i++)
      {
         P2 = 0x1 * (i + 1);
         P0 = led_pattern[led[i]];
         delay(40);//selected to avoid
    flickering
      }
}

void main(void)
{
   unsigned int i;
   while(1)
   {
      for(i=0; i<20; i++)
         display(i);
      for(i=20; i>0; i--)
         display(i);
   }
}
```

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 them back to 00

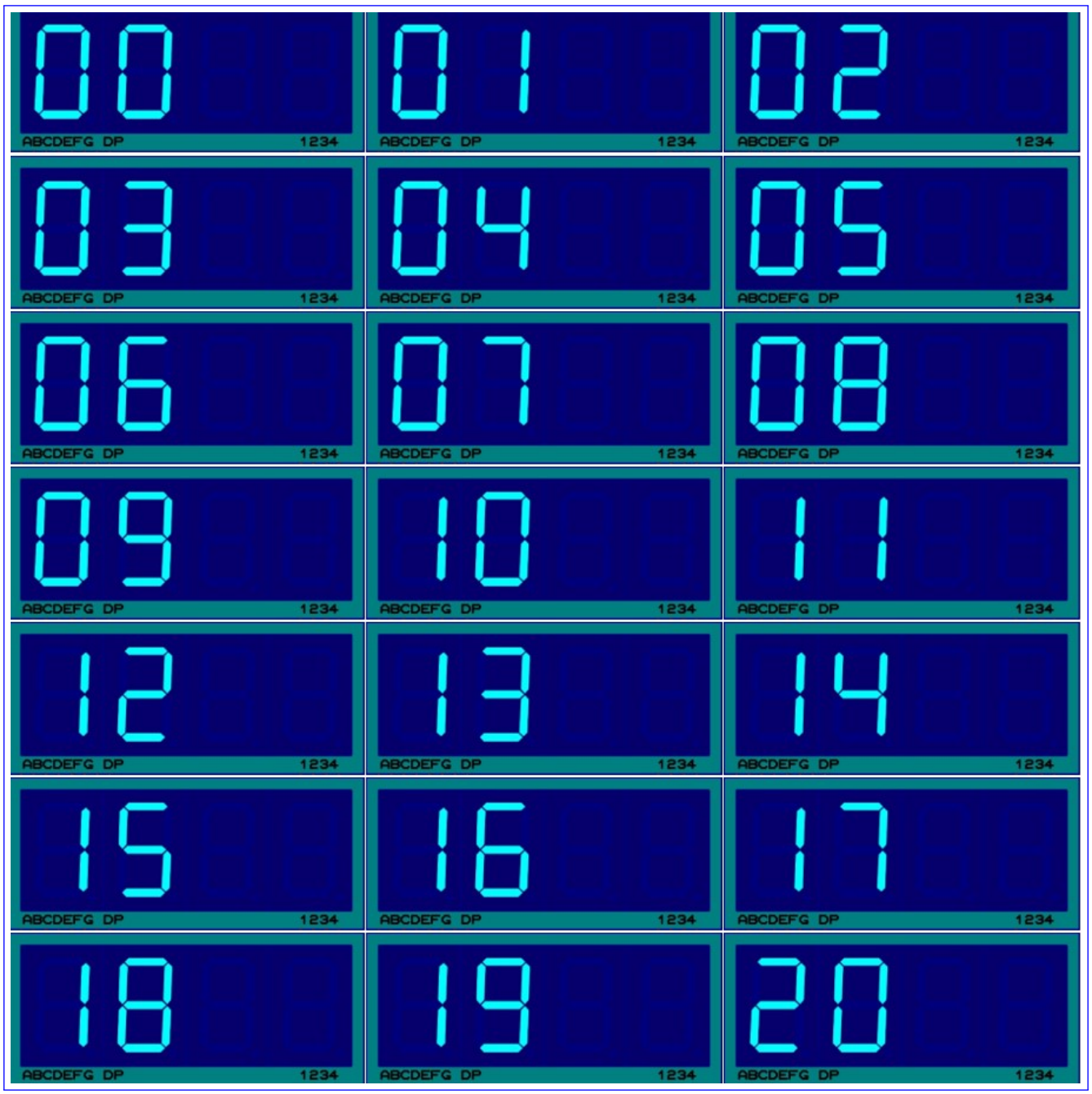

Figure 6: Proteus output for count 00-20

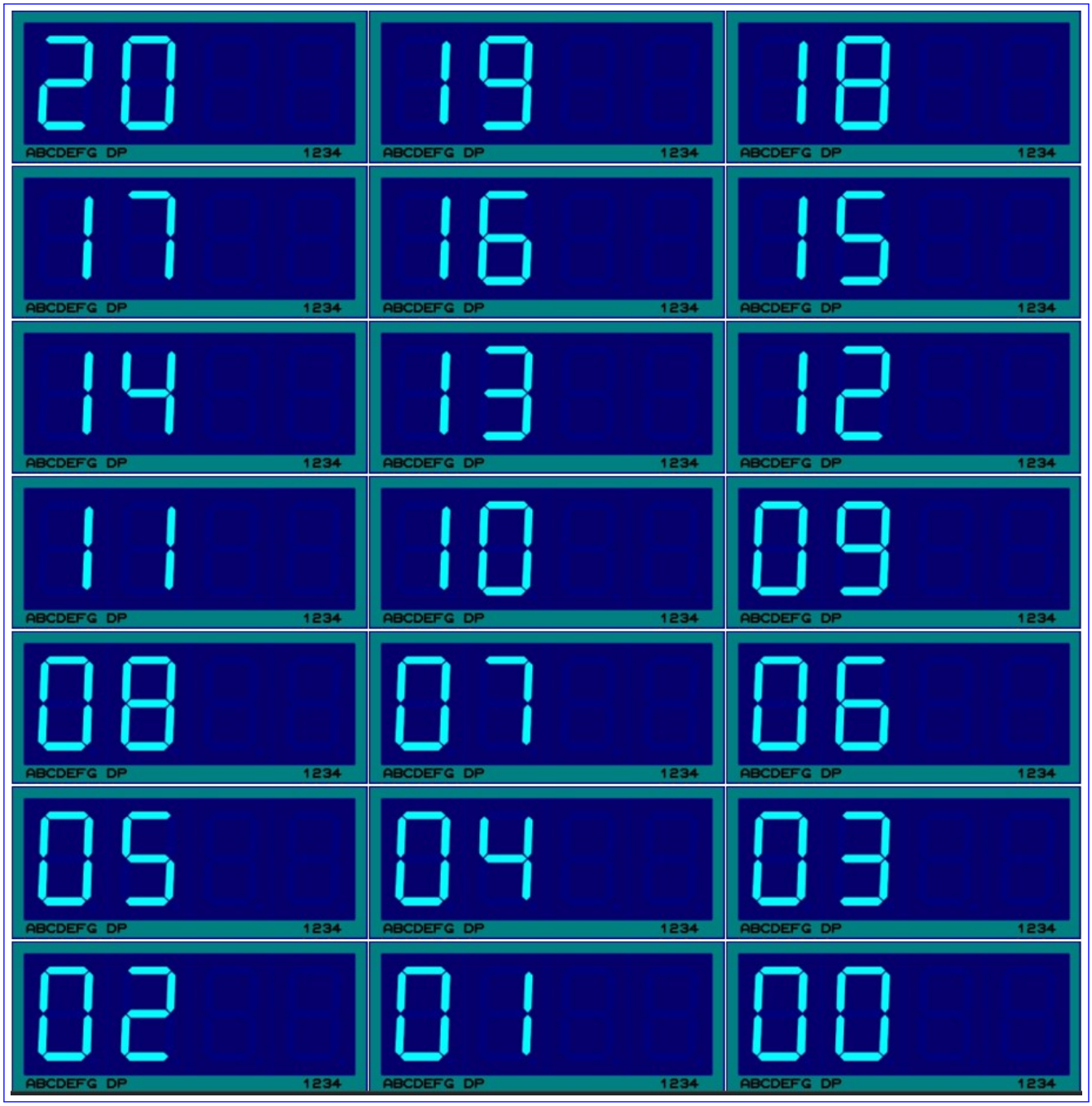

Figure 7: Proteus output for count 20-00

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

```
;FIBONACCI SEQUENCE
        ORG 00H

        MOV P2,#00H
        MOV DPTR,#LABEL1
        MOV R0,#50H
        MOV R7,#8        ;Nummber of terms
    (N=8)
        MOV A,R7
        MOV R6,A

; Display 1st and 2nd numbers of sequence
        MOV R1,#00H
        MOV R2,#01H
        MOV A,R1
        MOV @R0,A
        INC R0
        DEC R6
        MOV A,R2
        MOV @R0,A
```

```
        INC R0
        DEC R6

;Add consecutive terms to get next term
AGAIN:  MOV A,R1
        ADD A,R2
        MOV @R0,A
        INC R0
        MOV B,R2
        MOV R1,B
        MOV R2,A
        DJNZ R6,AGAIN

;HEX to DEC conversion and store in
    memory
        MOV R0,#50H
        MOV A,R7
        MOV R6,A

AGN2:   MOV A,@R0
        MOV R4,#00H
        MOV B,#0AH
        DIV AB
        MOV R2,A
        SUBB A,#0AH
        JC SKIP
        MOV A,R2
        MOV R3,B
        MOV B,#0AH
        DIV AB
        MOV R4,A
        MOV A,B
        MOV B,R3

SKIP:   MOV A,R2
        SWAP A
        ADD A,B
        MOV B,R4

        MOV @R0,A
        INC R0
        DJNZ R6,AGN2

REPEAT: MOV R0,#50H
        MOV A,R7
        MOV R4,A
LOOP1:  MOV R6,#255
MAIN:   MOV A,@R0
        MOV B,A
        ANL A,#0FH
        MOV P2,#02H
        ACALL DISPLAY
        MOV P0,A
        ACALL DELAY

        MOV A,B
        ANL A,#0F0H
        SWAP A
        MOV P2,#01H
        ACALL DISPLAY
        MOV P0,A
        ACALL DELAY

        DJNZ R6,MAIN
        INC R0
        DJNZ R4,LOOP1
        AJMP REPEAT

DELAY:  MOV R3,#02H
DEL1:   MOV R2,#0FAH
DEL2:   DJNZ R2,DEL2
        DJNZ R3,DEL1
        RET


DISPLAY:MOVC A,@A+DPTR
        RET

;Lookup table
LABEL1: DB 3FH
        DB 06H
        DB 5BH
        DB 4FH
        DB 66H
        DB 6DH
        DB 7DH
        DB 07H
        DB 7FH
        DB 6FH

        END
```

Code 5: Problem no. 3 Assembly

C language

```
#include <reg51.h>
#define  N  8
unsigned char led_pattern[10] = { 0x3f, 0x06,
    0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
    0x6f};

void delay(int time)
{
   unsigned int i,j;
   for (i=0;i<time;i++)
      for (j=0;j<125;j++);
}

void display(unsigned int i)
{
   unsigned int j, led1, led2;
   led1 = i / 10;
   led2 = i % 10;
   for(j=0; j<10; j++)//Delay between
    consecutive terms
   {
      P2 = 0x1;
      P0 = led_pattern[led1];
      delay(40);//selected to avoid flickering

      P2 = 0x2;
      P0 = led_pattern[led2];
      delay(40);
```

```
    }
}

void main(void)
{
    unsigned int i, fibo_seq[N] = {0 , 1};
    for(i=2; i<N; i++)
        fibo_seq[i] = fibo_seq[i-1] + fibo_seq[i-2];
    while(1)
        for(i=0; i<N; i++)
            display(fibo_seq[i]);
}
```

Code 6: Problem no. 3 C language

OUTPUT:
Fibonacci sequence of first 8 numbers is shown below:

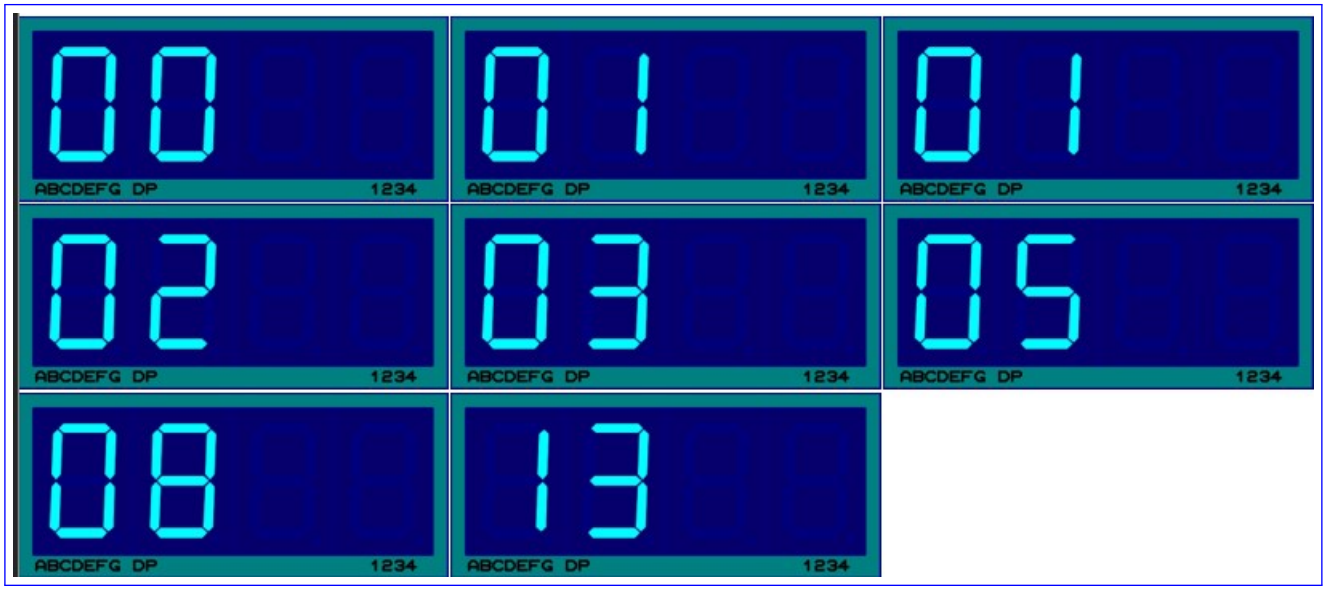

Figure 8: Proteus output Fibonacci sequence of first 8 numbers

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

```
        ORG 00H

        MOV R7,#7 ;N=7
        MOV P2,#00H
        MOV DPTR,#LABEL1

        MOV B,R7
        MOV R0,#5AH
        MOV R6,#10
AGN:    MOV B,R6
        MOV A,R7
        MUL AB
        MOV @R0,A
        DEC R0
        DJNZ R6,AGN

;HEX TO DEC conversion and store in
    memory
        MOV R0,#51H
        MOV R6,#10

AGN2:   MOV A,@R0
        MOV R4,#00H
        MOV B,#0AH
        DIV AB
        MOV R2,A
        SUBB A,#0AH
        JC SKIP
        MOV A,R2
        MOV R3,B
        MOV B,#0AH
        DIV AB
        MOV R4,A
        MOV A,B
        MOV B,R3
        MOV R2,A
SKIP:   MOV A,R2
        SWAP A
        ADD A,B
        MOV B,R4

        MOV @R0,A
        INC R0
        DJNZ R6,AGN2

REPEAT: MOV R0,#51H
```

```
        MOV R4,#10
LOOP1:  MOV R7,#255
MAIN:   MOV A,@R0
        MOV B,A
        ANL A,#0FH
        MOV P2,#02H
        ACALL DISPLAY
        MOV P0,A
        ACALL DELAY
        MOV A,B
        ANL A,#0F0H
        SWAP A
        MOV P2,#01H
        ACALL DISPLAY
        MOV P0,A
        ACALL DELAY
        DJNZ R7,MAIN
        INC R0
        DJNZ R4,LOOP1
        AJMP REPEAT

DELAY:  MOV R3,#02H
DEL1:   MOV R2,#0FAH
DEL2:   DJNZ R2,DEL2
        DJNZ R3,DEL1
        RET

DISPLAY:MOVC A,@A+DPTR
        RET

;Lookup table
LABEL1: DB 3FH
        DB 06H
        DB 5BH
        DB 4FH
        DB 66H
        DB 6DH
        DB 7DH
        DB 07H
        DB 7FH
        DB 6FH

        END
```

Code 7: Problem no. 4 Assembly

C language

```
#include <reg51.h>
#define N   7  //N=7
unsigned char led_pattern[10] = { 0x3f, 0x06,
    0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f,
    0x6f};

void delay(int time)
{
   unsigned int i,j;
   for (i=0;i<time;i++)
      for (j=0;j<125;j++);
}

void display(unsigned int i)
{
   unsigned int j;
   for(j=0; j<15; j++)
   {
      P2 = 0x1;
      P0 = led_pattern[i / 10];
      delay(40);

      P2 = 0x2;
      P0 = led_pattern[i % 10];
      delay(40);
   }
}

void main(void)
{
   unsigned int i;
   while(1)
      for(i=1; i<=10; i++)
         display(N*i);
}
```

Code 8: Problem no. 4 C language

OUTPUT:
Multiplication table for 7 is shown below

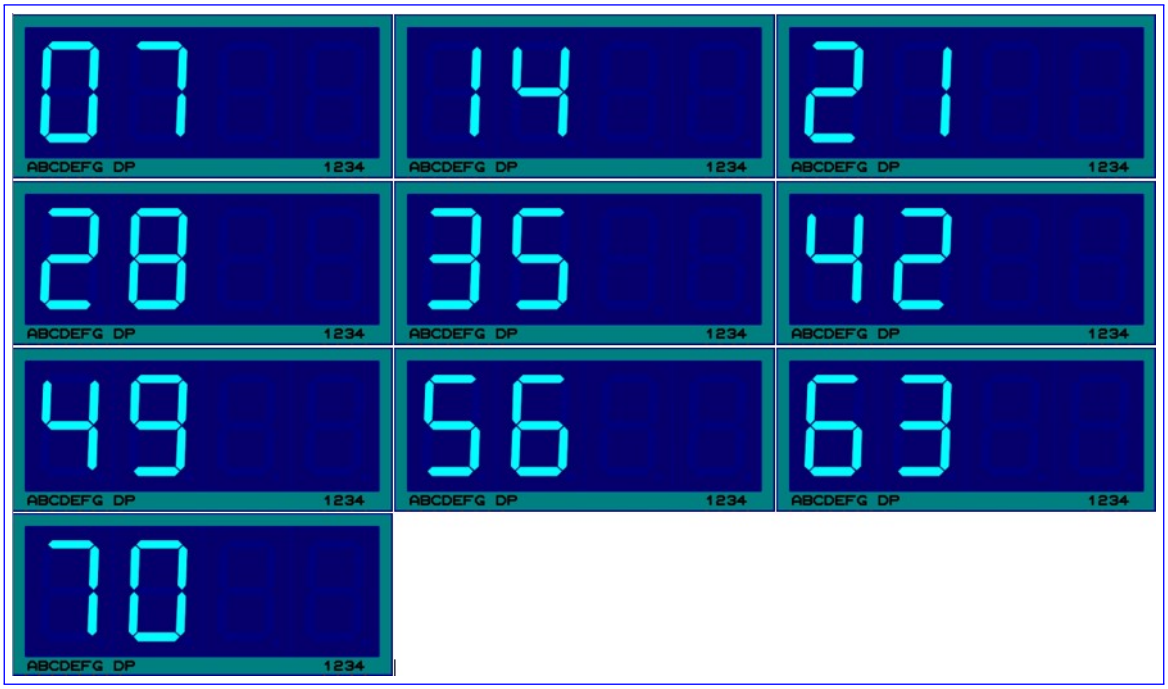

Figure 9: Proteus output for Multiplication table of 7

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

```
        ORG 00H
;Digital drive pattern for 403
        MOV 40H,#79H
        MOV 41H,#66H
        MOV 42H,#3FH
        MOV 43H,#4fH

REPEAT: MOV R0,#40H
        MOV A,@R0
        SETB P2.0
        MOV P0,A
        ACALL DELAY
        CLR P2.0
        INC R0

        MOV A,@R0
        SETB P2.1
        MOV P0,A
        ACALL DELAY
        CLR P2.1
        INC R0

        MOV A,@R0
        SETB P2.2
        MOV P0,A
        ACALL DELAY
        CLR P2.2
        INC R0

        MOV A,@R0
        SETB P2.3
        MOV P0,A
        ACALL DELAY
        CLR P2.3

        AJMP REPEAT

DELAY:  MOV R3,#02H
DEL1:   MOV R2,#0FAH
DEL2:   DJNZ R2,DEL2
        DJNZ R3,DEL1
        RET

        END
```

Code 9: Problem no. 5 Assembly

C language

```c
#include <reg51.h>

unsigned char led_pattern[4] = {0x79,0x66, 0
    x3f, 0xcf};


void delay(int time)
{
   unsigned int i,j;
   for (i=0; i<time; i++)
      for (j=0; j<125; j++);
}

void display()
{
      P2 = 0x1;
      P0 = led_pattern[0];
      delay(10);//selected to avoid flickering
     and show as static
      P2 = 0x2;
      P0 = led_pattern[1];
      delay(10);

      P2 = 0x4;
      P0 = led_pattern[2];
      delay(10);

      P2 = 0x8;
      P0 = led_pattern[3];
      delay(10);
      }

void main(void)
{
   while(1)
         display();
}
```

Code 10: Problem no. 5 C language

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

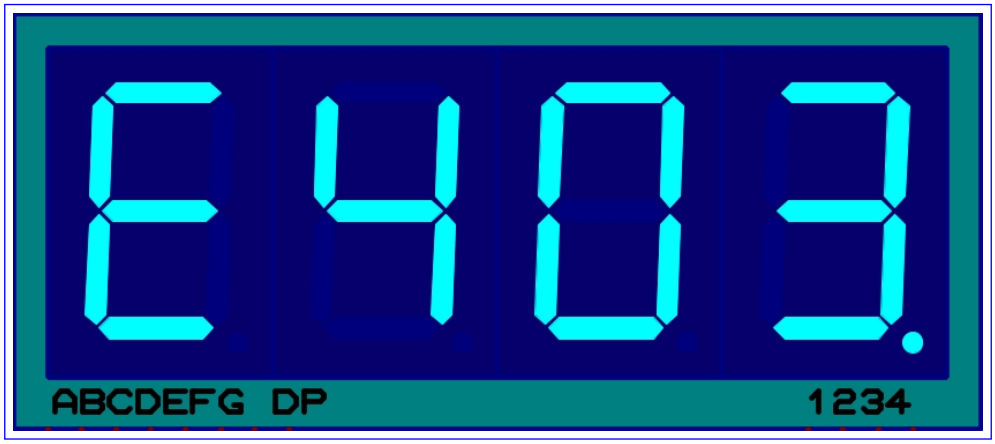

Figure 10: Proteus output for Roll no. Display

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

```
        ORG 00H
        MOV 40H,#79H
        MOV 41H,#66H
        MOV 42H,#3FH
        MOV 43H,#4fH
        MOV 44H,#79H
        MOV 45H,#66H
        MOV 46H,#3FH

REPEAT: MOV R0,#40H
        MOV R4,#04H
LOOP1:  MOV R7,#255
MAIN:   MOV A,@R0
        SETB P2.0
        MOV P0,A
        ACALL DELAY
        CLR P2.0
        INC R0

        MOV A,@R0
        SETB P2.1
        MOV P0,A
        ACALL DELAY
        CLR P2.1
        INC R0

        MOV A,@R0
        SETB P2.2
        MOV P0,A
        ACALL DELAY
        CLR P2.2
        INC R0

        MOV A,@R0
```

```
        SETB P2.3
        MOV P0,A
        ACALL DELAY
        CLR P2.3
        ;Scrolling happens here
        DEC R0
        DEC R0
        DEC R0

        DJNZ R7,MAIN

        INC R0
        DJNZ R4,LOOP1
        AJMP REPEAT

DELAY:  MOV R3,#02H
DEL1:   MOV R2,#0FAH
DEL2:   DJNZ R2,DEL2
        DJNZ R3,DEL1
        RET

        END
```

Code 11: Problem no. 6 Assembly

C language

```
#include <reg51.h>

unsigned char scroll_pattern[8] = { 0x79,0x66, 0x3f, 0xcf,
                                    0x79,0x66, 0x3f, 0xcf};

void delay(int time)
{
   unsigned int i,j;
   for (i=0; i<time; i++)
      for (j=0; j<125; j++);
}

void display(unsigned int i)
{
   unsigned int j;
   for(j=0; j<150; j++)
   {
      P2 = 0x1;
      P0 = scroll_pattern[i-4];
      delay(10);//selected to avoid flickering

      P2 = 0x2;
      P0 = scroll_pattern[i-3];
      delay(10);

      P2 = 0x4;
      P0 = scroll_pattern[i-2];
      delay(10);

      P2 = 0x8;
      P0 = scroll_pattern[i-1];
      delay(10);
   }
}

void main(void)
{
   unsigned int i;
   while(1)
      for(i=4; i<8; i++)
         display(i);//scrolling happens here
}
```

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.