

## EXPERIMENT 10

**AIM:** Interfacing of 8051 Microcontroller with LEDs and Seven segment display.

### THEORY:

#### 8085 Microcontroller:

Digit pattern of a seven segment LED display is simply the different logic combinations of its terminals 'a' to 'h' to display different digits and characters. For example, if you want to display the digit 3 on seven segment then you need to glow the segments a, b, c, d and g, having a binary pattern: 3  $\rightarrow$  1 1 1 1 0 0 1, in hexadecimal converts to 0x79. The table below, demonstrates the Hex decimal values that we need to send to the Display from PORT selected.

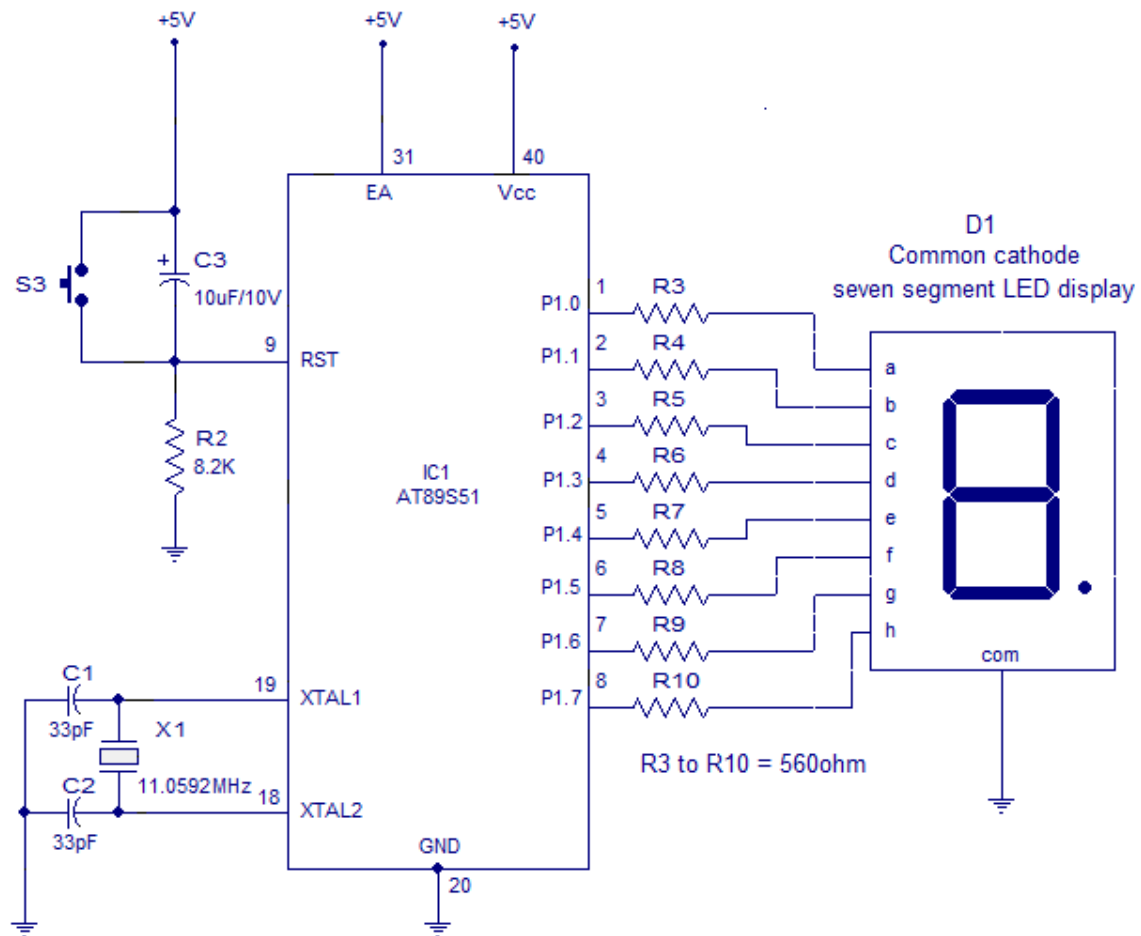
Decimal	ABCDEFG	A	B	C	D	E	F	G
0	0x7E	1	1	1	1	1	1	0
1	0x30	0	1	1	0	0	0	0
2	0x6D	1	1	0	1	1	0	1
3	0x79	1	1	1	1	0	0	1
4	0x33	0	1	1	0	0	1	1
5	0x5B	1	0	1	1	0	1	1
6	0x5F	1	0	1	1	1	1	1
7	0x70	1	1	1	0	0	0	0
8	0x7F	1	1	1	1	1	1	1
9	0x7B	1	1	1	1	0	1	1

### Applications

- Seven segments are widely used in digital clocks to display the time.
- These are used in electronic meters for displaying the numerical information.
- Used in Instrument panels.
- Used in digital readout displays.

### Limitations

- The complexity is increased to display large information.
- It is not possible to display the symbols on seven segment.



**CODE:** To display the last bits of the SAP ID (4190068 → 3FEF74)

```
MOV P0,#79h
MOV P0,#47h
MOV P0,#4Fh
MOV P0,#47h
MOV P0,#70h
MOV P0,#33h
```

The diagram illustrates the 8051 microcontroller with its pin connections, SFR values, and assembly code. The 8051 is shown as a central component with pins labeled P1.0 through P1.7, RST, P3.0 through P3.7, XTAL2, XTAL1, and GND. The pins are connected to various components: P1.0 to Vcc, P1.1 to P0.0, P1.2 to OP0.1, P1.3 to OP0.2, P1.4 to P0.3, P1.5 to P0.4, P1.6 to P0.5, P1.7 to P0.6, RST to OP0.7, P3.0 to OEA, P3.1 to OALE, P3.2 to OPSEN, P3.3 to OP2.7, P3.4 to OP2.6, P3.5 to OP2.5, P3.6 to OP2.4, P3.7 to OP2.3, XTAL2 to OP2.2, XTAL1 to OP2.1, and GND to OP2.0.

The SFR values are listed in a table:

SFR	Value
A	0x00
B	0x00
PSW	0x00
P0	0x79
P1	0x00
P2	0x00
P3	0x00
SP	0x07
DPL	0x00
DPH	0x00
PCON	0x00
TCON	0x00

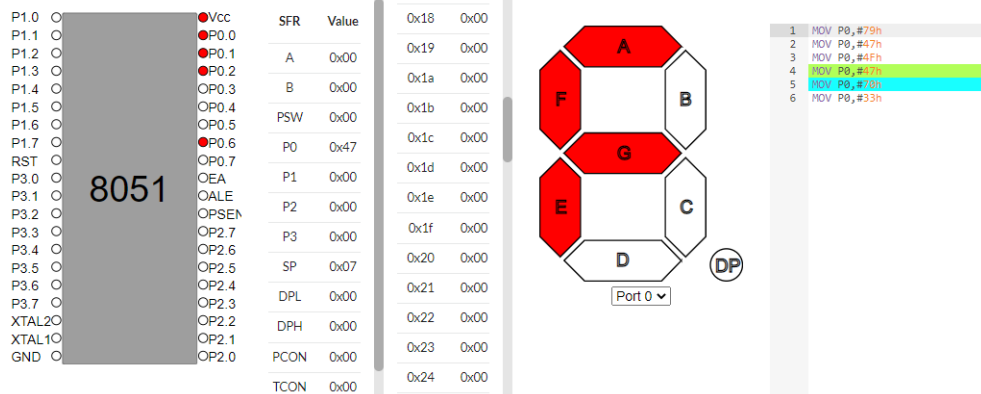
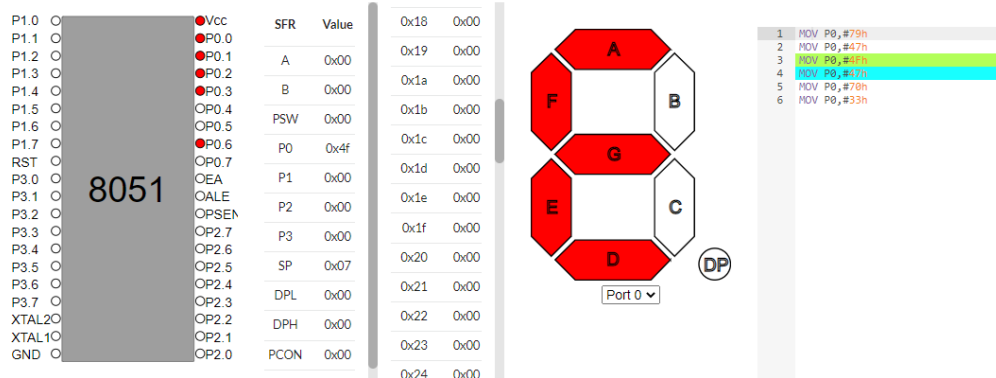
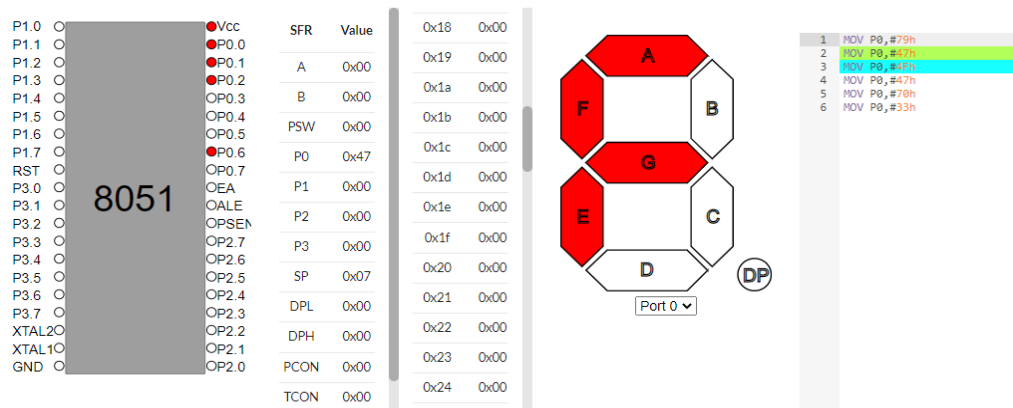
The assembly code is shown in a table:

Port 0	Value
0x18	0x00
0x19	0x00
0x1a	0x00
0x1b	0x00
0x1c	0x00
0x1d	0x00
0x1e	0x00
0x1f	0x00
0x20	0x00
0x21	0x00
0x22	0x00
0x23	0x00
0x24	0x00

The 8051 is shown with its internal components: P0, P1, P2, P3, and DP. The DP is labeled as Port 0.

The assembly code is shown in a table:

Port 0	Value
1	MOV P0,#79h
2	MOV P0,#47h
3	MOV P0,#4Fh
4	MOV P0,#47h
5	MOV P0,#70h
6	MOV P0,#33h



The screenshot displays the Proteus simulation environment. On the left, the 8051 microcontroller's pin configuration is shown, with P0 pins connected to the 7-segment display. The central panel shows the SFR (Special Function Register) values, with A=0x00, B=0x00, PSW=0x00, P0=0x70, P1=0x00, P2=0x00, P3=0x00, SP=0x07, DPL=0x00, DPH=0x00, and PCON=0x00. The right panel shows the 7-segment display with segments A, B, C, D, E, F, and G, and a DP pin. The code in the editor shows a sequence of MOV instructions loading values 79h, 47h, 4Fh, 47h, 79h, and 33h into P0. The 7-segment display shows the hexadecimal digit 3.

## CONCLUSION:

In this experiment, I learnt interfacing 7 segment display with 8051 microcontroller. The 7 segment display have 7 segments a, b, c, d, e, f, g and the respective segments need to be lit in order to display a digit. Each digit has it's own pattern for example, 3 → 1 1 1 1 0 0 1, in hexadecimal converts to 0x79. Thus, in the microcontroller we feed 79H to the port and the number is displayed. In this experiment I displayed the hex value of the last bits of my SAP ID (4190068 → 3FEF74)