UNIT 3

8051 MICROCONTROLLER INTERFACING:

Interfacing to External Memory, Interfacing with LED, Seven Segment Display and LCD, Sensor interfacing, 8051 Interfacing and applications using Stepper motor, DC motor and Servo motor.

Interfacing-:Interfacing in the microprocessor refers to the process of connecting external devices such as memory, input/output peripherals, and other components to the microprocessor. This connection allows the microcontroller to exchange data with these devices, enabling it to perform various operations, such as reading input from sensors, storing data in memory, or controlling output devices.

The Memory Interfacing in 8051 is **used to access memory quite frequently to read instruction codes and data stored in memory**. This read/write operations are monitored by control signals. The microprocessor activates these signals when it wants to read from and write into memory.

Interfacing of memory means connecting memory IC with microcontroller 8051 has 16 address lines (A0 - A15), hence a maximum of 64 KB of memory locations can be interfaced with it. The memory address space of the 8081 takes values from 0000H to FFFFH.

No of address line (n)	No of memory location (2^n)
8	2^8 =256
9	2^9 = 512
10	2^{10} = 1K
11	2^{11} = 2K
12	2^{12} = 4K
13	2^{13} = 8K
14	2^{14} = 16K
15	2^{15} = 32 K
16	2^{16} = 64K

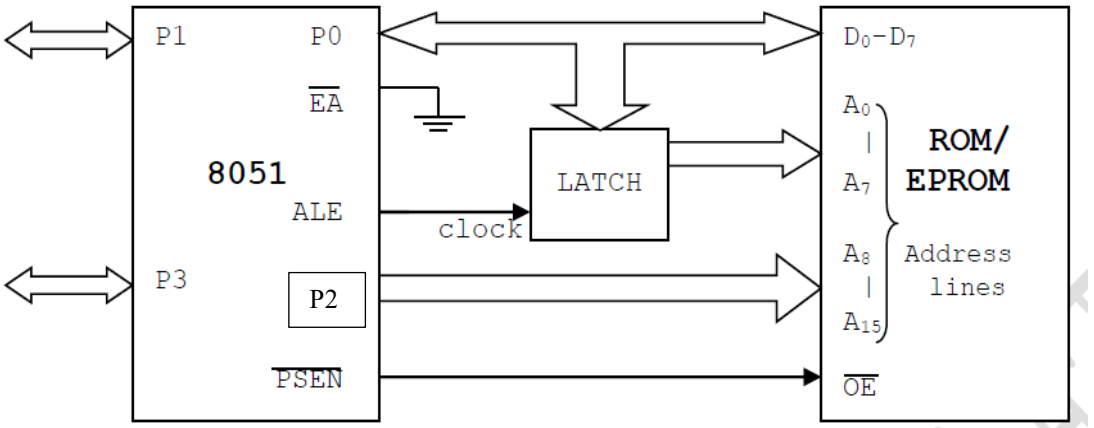
Interfacing to External Memory.

External ROM(Program Memory) Interfacing

=>How to connect or interface external ROM(Program Memory) to 8051.

Steps-:

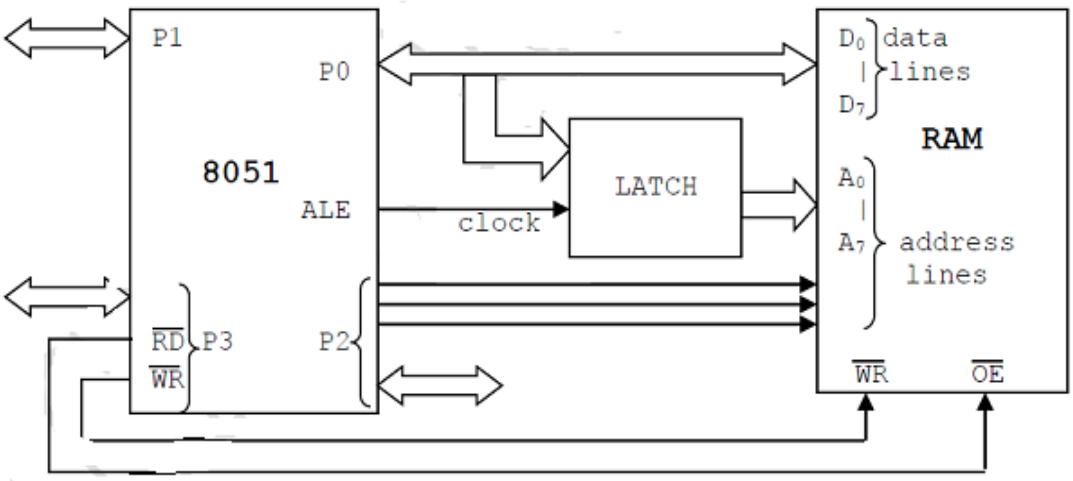
- Port 0 is used as multiplexed data & address lines.It gives lower order 8 bit address(A7-A0) and 8 bit (D7-D0) used as data bus.
- 8 bit address is latched using external latch & ALE signal from 8051.
- Port 2 provides higher order (A15-A8) 8 bit address.
- PSEN is used to activate the output enable signal of external ROM/EPROM.



EXTERNAL RAM (DATA MEMORY) INTERFACING

=> How to connect or interface external RAM (data memory) to 8051.

- Port 0 is used as multiplexed data & address lines.It gives lower order 8 bit address(A7-A0) and 8 bit (D7-D0) used as data bus.
- 8 bit address is latched using external latch & ALE signal from 8051.
- Port 2 provides higher order (A15-A8) 8 bit address.
- RD & WR signals from 8051 selects the memory read & memory write operations respectively.
- For RD & WR signals: generally P3.6 & P3.7 pins of port 3 are used to generate memory read and memory write signals.



Example

Example 1: Design a µController system using 8051 to Interface the external RAM of size 16k x 8.

Solution: Given, Memory size: 16k

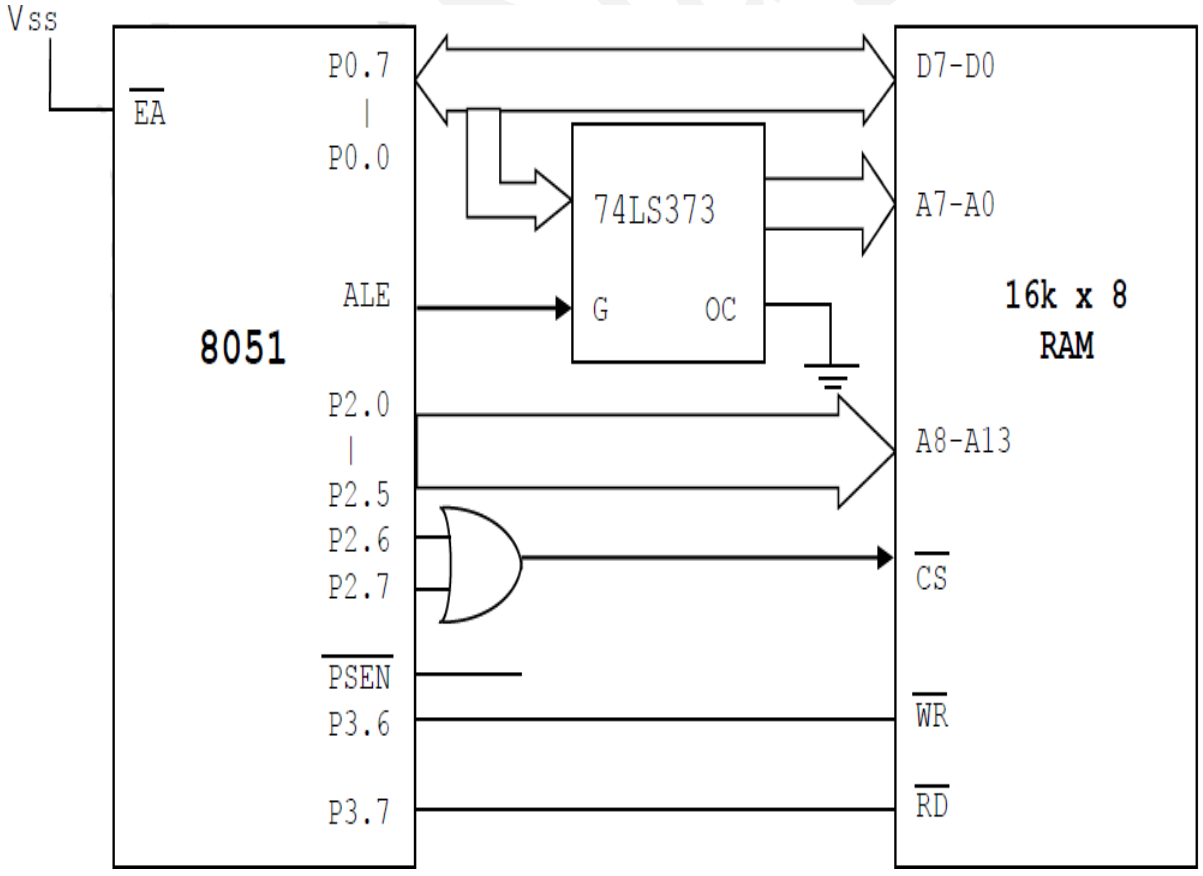
- Which means, we require 2^n =16k (n address lines)
- Here for 16k, n=14 i.e.A0 to A13 address lines are required.
- A14 and A15 are connected through OR gate to CS pin of external RAM.
- When A14 and A15 both are low (logic '0'), external data memory (RAM) is selected.

Address Decoding (Memory Map) For 16k X 8 RAM

Addr	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Hex Addr
Start	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 H
End	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	03FF H

|<----------for CS'-------->|<---------- A0 to A13 of MC to A0 to A13 of RAM ---------->|

Interfacing Diagram of 16k x 8 RAM to 8051.



Example 2: Design a Microcontroller system using 8051 to interface the external ROM of size 4k x 8.

Solution: Given, Memory size: 4k

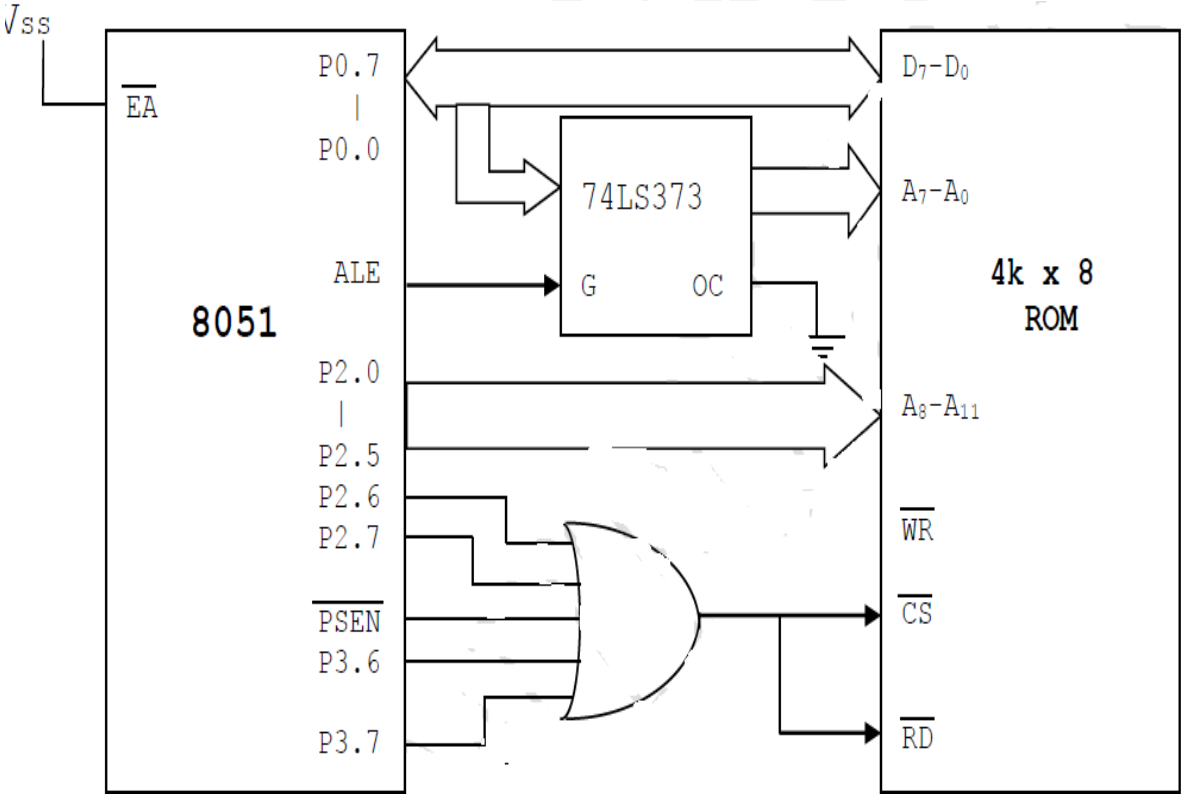
- Which means, we require 2^n =16k (n address lines)
- Here for 4k, n=12 i.e.A0 to A11 address lines are required.
- A12 to A15 and PSEN are connected through OR gate to CS pin abd RD of external ROM.
- When A12 to A15 both are low (logic '0'), external ROM memory is selected

Address Decoding (Memory Map) For 4k X 8 RAM

Addr	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Hex Addr
Start	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 H
End	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0FFF H

|<--------- for CS'--------->|<----------A0 to A13 of MC to A0 to A13 of RAM - ---------->|

Interfacing Diagram of 4k x 8 ROM to 8051



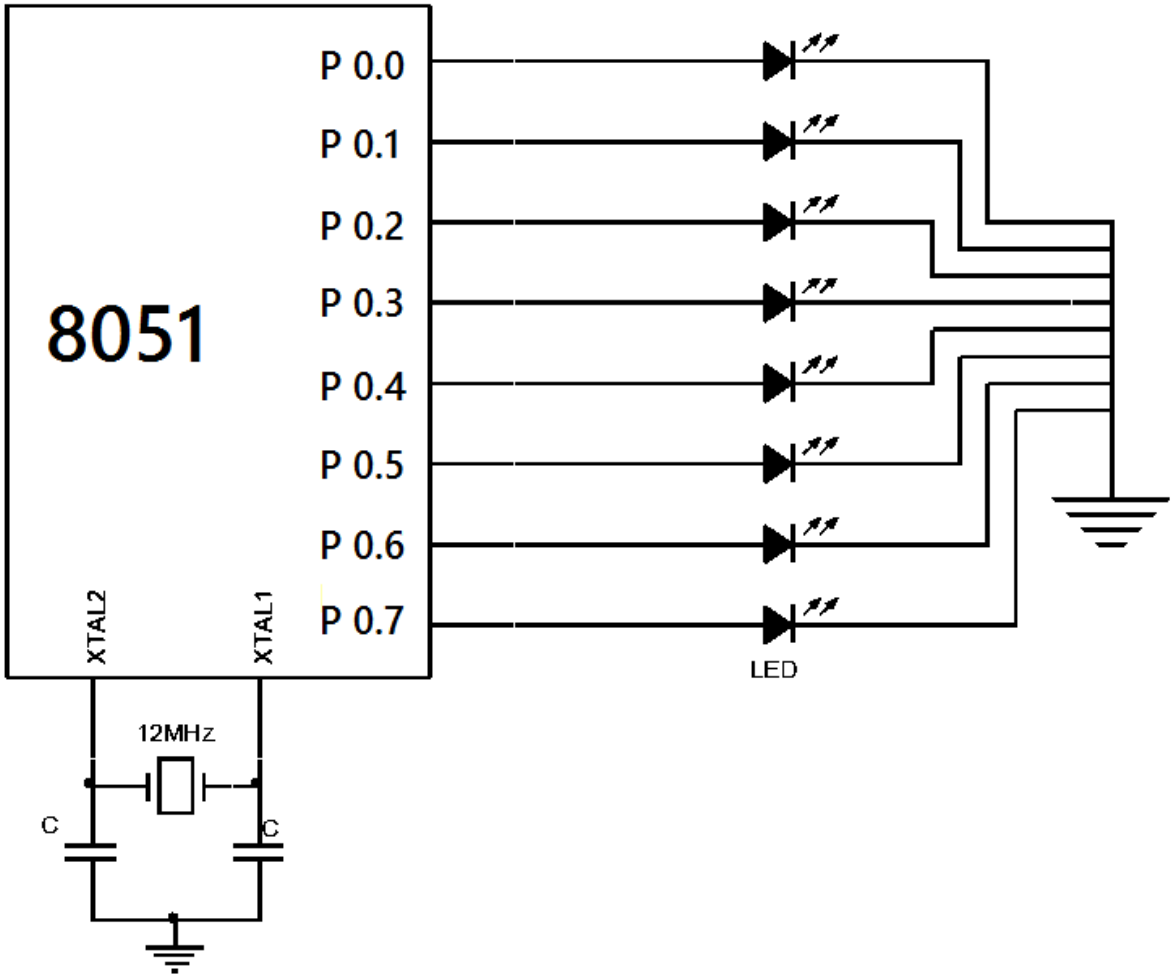
Interfacing with LED

Example 3: Design a µController system using 8051 to Interface LED and develop ALP to toggle the LED alternately

Solution-:

- Anodes of the LEDs are Connected to the port pins and cathodes are connected to common ground connection.
- To turn on particular LED we will need to make value of that pin "High" i.e "1".
- After making a particular pin high or low a small delay is executed to make that LED light visdible.

Interfacing Diagram of 8 LEDs with 8051 microcontroller.



Program-: To toggle the LED alternately

```
        ORG 0000H       ;This sets the program start address to 0000H (the reset vector).
                        ;When the 8051 is powered on or reset, execution starts here.
```

Main Program

```
        MOV P0, 00H     ; Set all pins of Port 0 low(Port 0 initially outputs 0
                        ; (All LEDs OFF if active high).
        MOV A, #55H     ; Load A = 55H (binary: 01010101)
                        ;55H represents an alternating bit pattern (1010101 in LEDs).
```

Main Loop

```
AGAIN:
        MOV P0, A       ; Send pattern to Port 0First,
                        ; 55H goes to Port 0 (LEDs blink in one pattern).
        ACALL DELAY     ; Wait a while
        CPL A           ; Complement A (55H ↔ AAH)
                        ;After a delay, CPL A inverts it to AAH (binary: 10101010),
                        ;flipping which LEDs are ON.
        SJMP AGAIN      ; Repeat forever
                        ;Loop repeats forever, creating a blinking pattern.
```

Delay Subroutine

```
DELAY:                  ;Purpose->Waste CPU cycles to slow down the blinking.
        MOV R3, #0FFH   ; Outer loop counter (255)
L1:     MOV R4, #0FFH   ; Inner loop counter (255)
L2:     DJNZ R4, L2     ; Decrement R4 until zero(Decrement and Jump if Not Zero)
        DJNZ R3, L1     ; Decrement R3, repeat inner loop (Two nested loops
                        (R3 x R4) create a long pause.
        RET             ; Stop execution (infinite loop)
        END             ; End of the program
```

Interfacing with Seven Segment Display

Example 4: Design a µController system using 8051 to interface seven segment display and develop ALP to display alphabetical characters such as A,B,C,D,E,F and numerical characters such as 0,1,2,3,4,5,6,7,8,9.

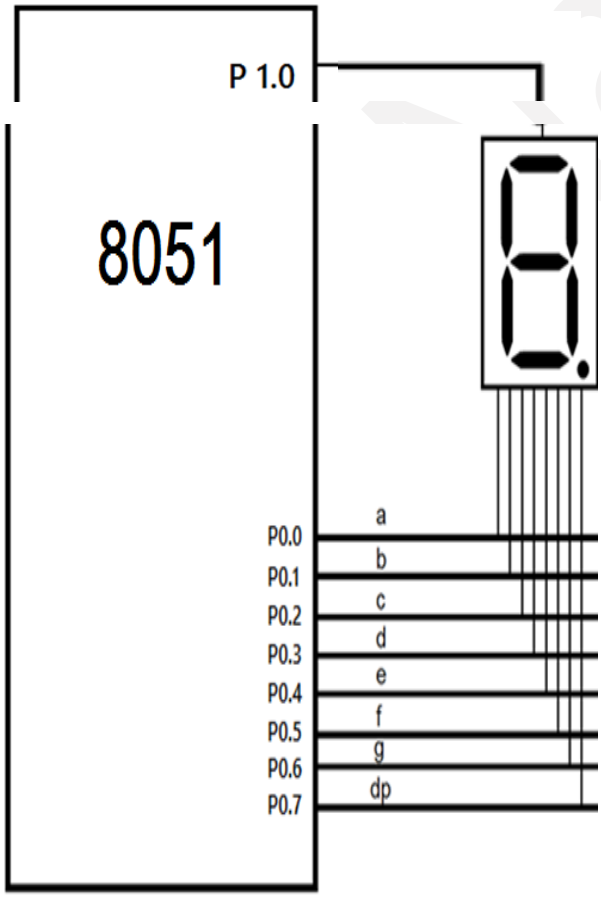
Solution-:

- There are two types of LED 7-segment displays: common cathode

(CC) and common anode (CA).

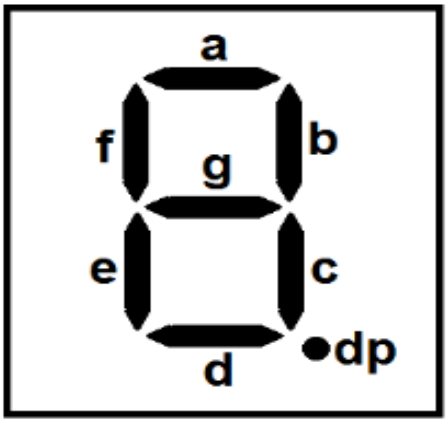
- The difference between the two displays is the common cathode has all the cathodes of the 7-segments connected directly together and the common anode has all the anodes of the 7-segments connected together.
- In this diagram common anode seven segment display is used. So when we want to make any segment glow on, we will just make respective I/O pin low i.e. 0.
- With the help of seven segment display we can display alphabetical characters such as A,B,C,D,E,F and numerical characters such as 0,1,2,3,4,5,6,7,8,9.

Interfacing Diagram of 8051 microcontroller and Seven segment display.

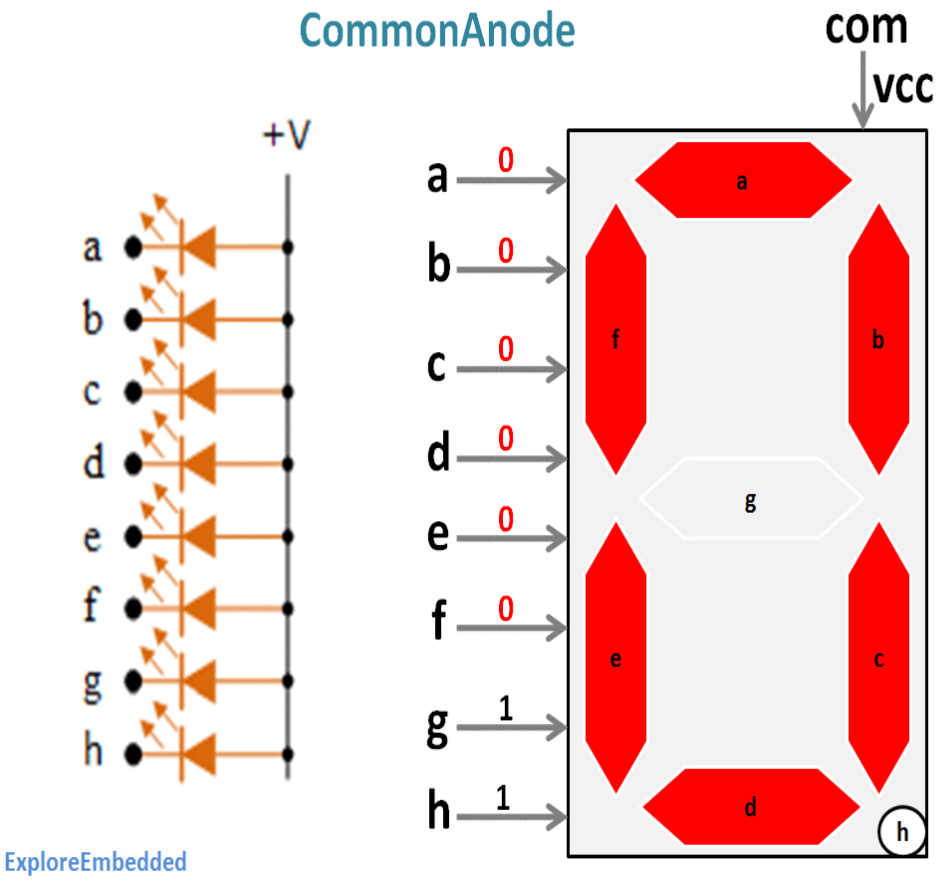


- **Program-: To display alphabetical characters such as A,B,C,D,E,F and numerical characters such as 0,1,2,3,4,5,6,7,8,9.**

Solution-:Fig shows structure of Seven segment display.



The diagram show the common anode seven segment display. So when we want to make any segment glow on, we will just make respective pin low i.e. 0.



Below table shows the binary/hex values for displaying the digits on CommonAnode seven segment display.

Digit	h	g	f	e	d	c	b	a	Hex Value
0	1	1	0	0	0	0	0	0	0xC0
1	1	1	1	1	1	0	0	1	0xF9
2	1	0	1	0	0	1	0	0	0xA4
3	1	0	1	1	0	0	0	0	0xB0
4	1	0	0	1	1	0	0	1	0x99
5	1	0	0	1	0	0	1	0	0x92
6	1	0	0	0	0	0	1	0	0x82
7	1	1	1	1	1	0	0	0	0xF8
8	1	0	0	0	0	0	0	0	0x80
9	1	0	0	1	0	0	0	0	0x90

ORG 0000H ;This sets the program start address to 0000H (the reset vector).

MOV P1, #00000001B ; Bit 0 (P1.0) = 1 → **logic HIGH,** P1 refers to **Port 1** of the 8051 (8-bit I/O port). moves data into a register/port.

ACALL DELAY ;Calls a subroutine named DELAY

MOV P0, #11111111B ;Sets all bits of Port 0 HIGH (logic 1).Often used to turn off all LEDs (if they are active low) or clear the display before writing new data.On a common-cathode 7-segment display with active-low inputs, 11111111B would turn all segments OFF.

START: ; This is label in the program so you can jump back to it later.

MOV P0, #11000000B ; DISPLAY 0 (C0H) , Sends the binary pattern 11000000 to Port 0. If Port 0 is driving a 7-segment display (common cathode), 11000000B lights up digit '0'.

```
ACALL DELAY                ;Calls a subroutine named DELAY
MOV P0, #11111001B         ; DISPLAY 1 (F9H)
ACALL DELAY
MOV P0, #10100100B         ; DISPLAY 2  (A4H)
ACALL DELAY
MOV P0, #10110000B         ; DISPLAY 3  (B0H)
ACALL DELAY
MOV P0, #10011001B         ; DISPLAY 4  (99H)
ACALL DELAY
MOV P0, #10010010B         ; DISPLAY 5  (92H)
ACALL DELAY
MOV P0, #10000010B         ; DISPLAY 6  (82H)
ACALL DELAY
MOV P0, #11111000B         ; DISPLAY 7  (F8H)
ACALL DELAY
MOV P0, #10000000B         ; DISPLAY 8  (80H)
ACALL DELAY
MOV P0, #10010000B         ; DISPLAY 9 (90H)
ACALL DELAY
```

Delay Subroutine

```
DELAY:                     ;Purpose->Waste CPU cycles to slow down the blinking.
        MOV R3, #0FFH      ; Outer loop counter (255)
L1:     MOV R4, #0FFH      ; Inner loop counter (255)
L2:     DJNZ R4, L2        ; Decrement R4 until zero(Decrement and Jump if Not Zero)
        DJNZ R3, L1        ; Decrement R3, repeat inner loop (Two nested loops
                           (R3 x R4) create a long pause.
        RET                ; Stop execution (infinite loop)
        END                ; End of the program
```

Interfacing and applications using Stepper motor

Example 5 : Design a µController system using 8051 to interface Stepper Motor and develop ALP to connect steppper motor to port 1 of 8051 and run in half drive mode.

Solution-:

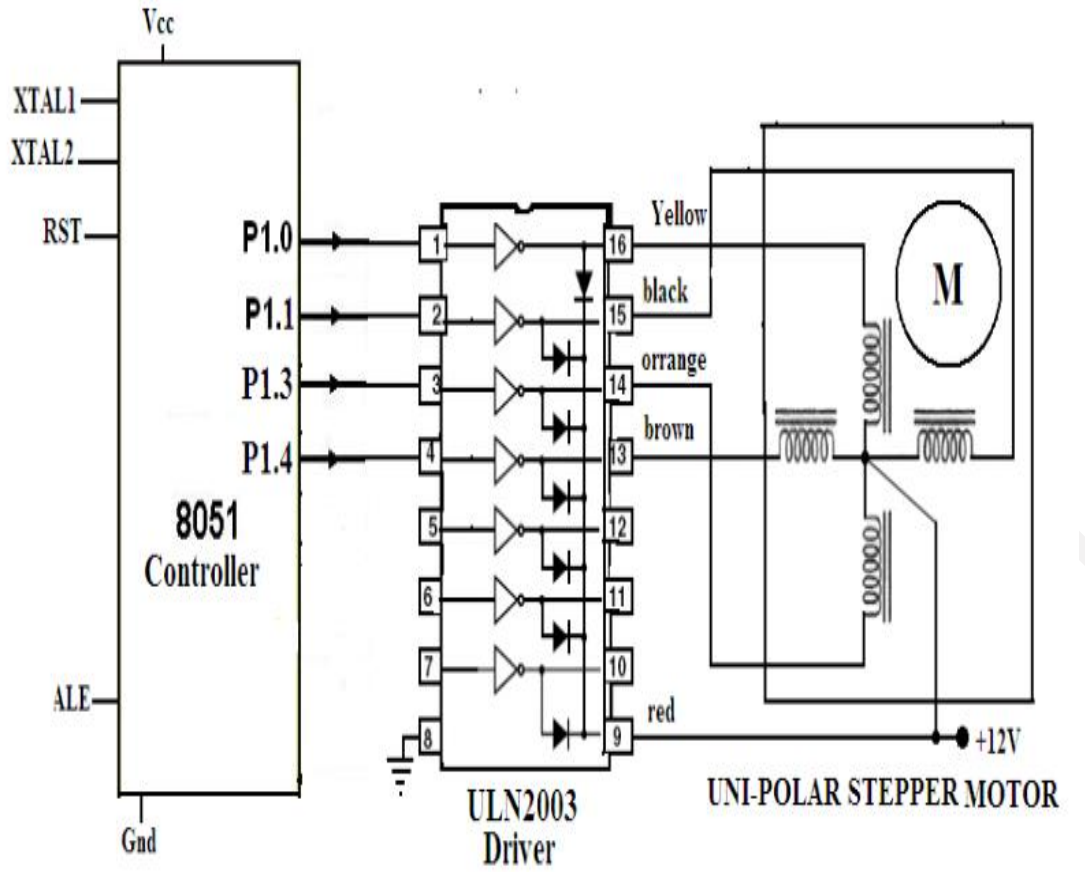
=>A stepper motor is a device that translates electrical pulses into mechanical movementin steps of fixed step angle.

- The stepper motor rotates in steps in responseto the applied signals.
- It is mainly used for position control.
- It is used in disk drives, dot matrix printers, plotters and roboticsand process control circuits

=>Motor Driver Circuit(ULN2003)Stepper motor driver circuits are available readily in the form of ICs.ULN2003 is one such driver IC which is a High-Voltage High-Current Darlington transistor arrayand can give a current of 500mA

=>For example,let us consider the hex code for a stepper motor to rotate in clockwise direction(half drive mode) is CCH , 99H , 33H and 66H.This hex code will be applied to the input terminals of the driver through the assembly language program.To rotate the stepper motor in anti-clockwise direction thesame code is applied in the reverse order.

Interfacing Diagram of 8051 microcontroller and Stepper motor.



Program -Connect steppper motor to port 1 of 8051 and run in half drive mode.

Solution-:

- This program drives a **stepper motor** connected to **Port 1** of the 8051 microcontroller.
- It energizes **two coils at a time** (two-phase excitation) for smoother motion and higher torque.
- The **delay subroutine** controls the speed of rotation.

Bit Patterns for Coils(Let's convert the hex values to binary so you see what's happening:)

Step	Hex	Binary (P1.7 ⋯ P1.0)	Coils Energized
1	0CCH	11001100	Coils 1 & 2 ON
2	99H	10011001	Coils 2 & 4 ON
3	33H	00110011	Coils 3 & 4 ON
4	66H	01100110	Coils 1 & 3 ON

Notes: Each pattern turns **two coils ON** while others remain OFF. This is known as **two-phase-on full-step mode**. The order of these patterns makes the rotor **step forward** (probably clockwise depending on wiring).

Main Program

```
        ORG 0000H           ;This sets the program start address to 0000H (the reset vector).
L1:     MOV A,#0CCH         ;Load the Accumulator with 11001100b.
        MOV P1,A            ; Output pattern to Port 1 → energizes Coils 1 & 2.
        ACALL DELAY         ; Call DELAY so the motor can physically move one step.

        MOV A,#99H          ;Load the Accumulator with 10011001b.
        MOV P1,A            ;Output pattern to Port 1 → energizes Coils 2 & 4.
        ACALL DELAY         ; Call DELAY so the motor can physically move one step
        MOV A,#33H          ;Load the Accumulator with 00110011b.
        MOV P1,A            ;Output pattern to Port 1 → energizes Coils 3 & 4.
        ACALL DELAY         ;Call DELAY so the motor can physically move one step
        MOV A,#66H          ; Load the Accumulator with 11001100b.
        MOV P1,A            ; Output pattern to Port 1 → energizes Coils 1 & 3.
        ACALL DELAY         ;Call DELAY so the motor can physically move one step
        SJMP L1             ;Jump back to first step and repeat indefinitely.
```

Delay Subroutine

```
DELAY:
        MOV R2, #4          ; Outer loop
L3:     MOV R3, #255        ; Inner loop
L2:     DJNZ R3, L2         ; Decrement R3 until zero
        DJNZ R2, L3         ; Repeat outer loop
        RET
```

Applications using Stepper motor

Applications of 8051 Microcontrollers

INTRODUCTION:

A microcontroller is a versatile chip which can be used in various fields starting from simple consumer electronics to high end medical, automobile and defense applications also. So, now a days the microcontrollers are found in every walk of life. In the lab you can use a microcontroller to measure accurately the voltage, current and resistances also . The various fields are listed below.

- Automobile
- Aeronautics
- Space
- Rail Transport
- Mobile communications
- Industrial processing
- Remote sensing , Radio and Networking
- Robotics
- Consumer electronics , music players, Computer applications
- Security (e-commerce, smart cards)
- Medical electronics (hospital equipment, and mobile monitoring) and
- Defense application

Interfacing with LCD

Example 6 : Design a µController system using 8051 to interface LCD and develop ALP to display character on LCD.

Solution-:

LCD (Liquid Crystal Display) interface LCDs can display numbers, characters, and graphics. To produce a proper display, the information has to be periodically refreshed. This can be done by the CPU or internally by the LCD device itself.

Pin description of LCD

- Vss and VDD provide +5v and ground, V0 is used for controlling LCD contrast.
- If RS=0, the instruction command register is selected, allowing the user to send a command such as clear display, cursor at home, etc.
- If RS=1 the data register is selected, allowing the user to send data to be displayed on the LCD.
- R/W input allows the user to Read/ Write the information tothe LCD.
- The enable pin is used by the LCD to latch information presented to its data pins.
- The 8-bit data pins are used to send information to LCD.

LCD COMMAND CODES

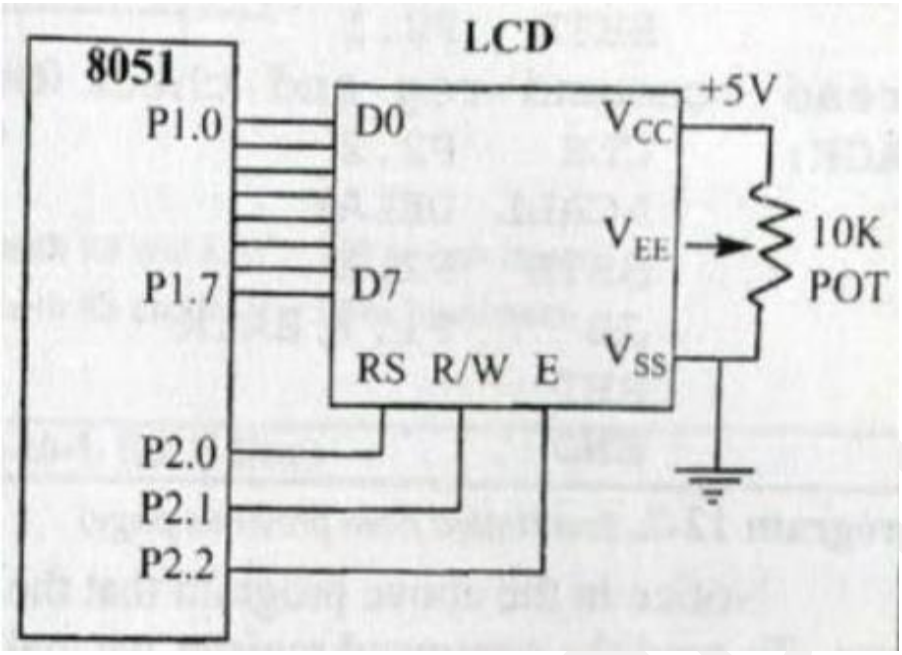
The LCD's internal controller can accept several commands and modify the display accordingly.

These commands would be things like:

- Clear screen
- Return home
- Decrement/Increment cursor

After writing to the LCD, it takes some time for it to complete its internal operations. During this time, it will not accept any new commands or data

Interfacing Diagram of 8051 microcontroller and LCD



Program-: To dsiplay character on LCD

Solution-:

To send any of the commands to the LCD, make pin RS=0. For data, make RS=1. Then send a high-to-low pulse to the E pin to enable the internal latch of the LCD. This is shown in the code below.

```
;Calls a time delay before sending next data/command
;P1.0-P1.7 are connected to LCD data pins D0-D7
 ;P2.0 is connected to RS pin of LCD
;P2.1 is connected to R/W pin of LCD
;P2.2 is connected to E pin of LCD

    MOV A,#38H          ;INIT. LCD 2 LINES, 5X7 MATRIX
    ACALL COMNWRT       ;call command subroutine
    ACALL DELAY         ;give LCD some time
    MOV A,#0EH          ;display on, cursor on
    ACALL COMNWRT       ;call command subroutine
```

```
ACALL DELAY          ;give LCD some time
MOV A,#01            ;clear LCD
ACALL COMNWRT        ;call command subroutine
ACALL DELAY          ;give LCD some time
MOV A,#06H           ;shift cursor right
ACALL COMNWRT        ;call command subroutine
ACALL DELAY          ; give LCD some time
MOV A,#84H           ;cursor at line 1, pos. 4
ACALL COMNWRT        ;call command subroutine
ACALL DELAY          ;give LCD some time
MOV A,#'N'           ;display letter N
ACALL DATAWRT        ;call display subroutine
ACALL DELAY          ;give LCD some time
MOV A,#'O'           ;display letter O
ACALL DATAWRT        ;call display subroutine
AGAIN: SJMP AGAIN    ;stay here

COMNWRT:             ;send command to LCD
MOV P1,A             ;copy reg A to port 1
CLR P2.0             ;RS=0 for command
CLR P2.1             ;R/W=0 for write
SETB P2.2            ;E=1 for high pulse
CLR P2.2             ;E=0 for H-to-L pulse
RET

DATAWRT              : write data to LCD
MOV P1,A             ;copy reg A to port 1
SETB P2.0            ;RS=1 for DATA
CLR P2.1             ; R/W=0 for write
```

```
SETB P2.2                ;E=1 for high pulse
CLR P2.2                 ;E=0 for H-to-L pulse
RET

DELAY   : MOV R3,#50     ;50 or higher for fast CPUs
HERE 2  : MOV R4,#255    ;R4 = 255
HERE    : DJNZ R4,HERE   ;stay until R4 becomes 0
          DJNZ R3, HERE 2
          RET
```

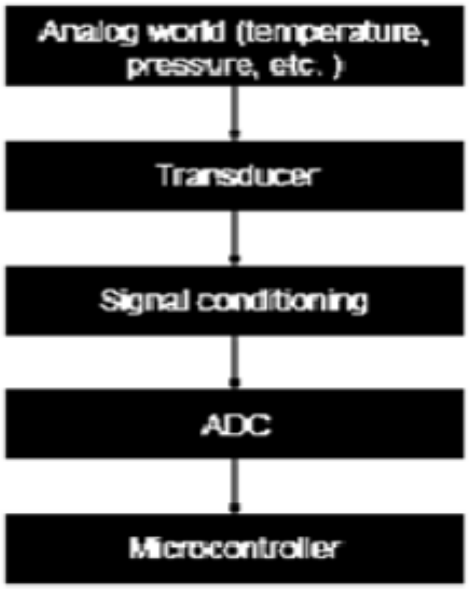

Sensor Interfacing

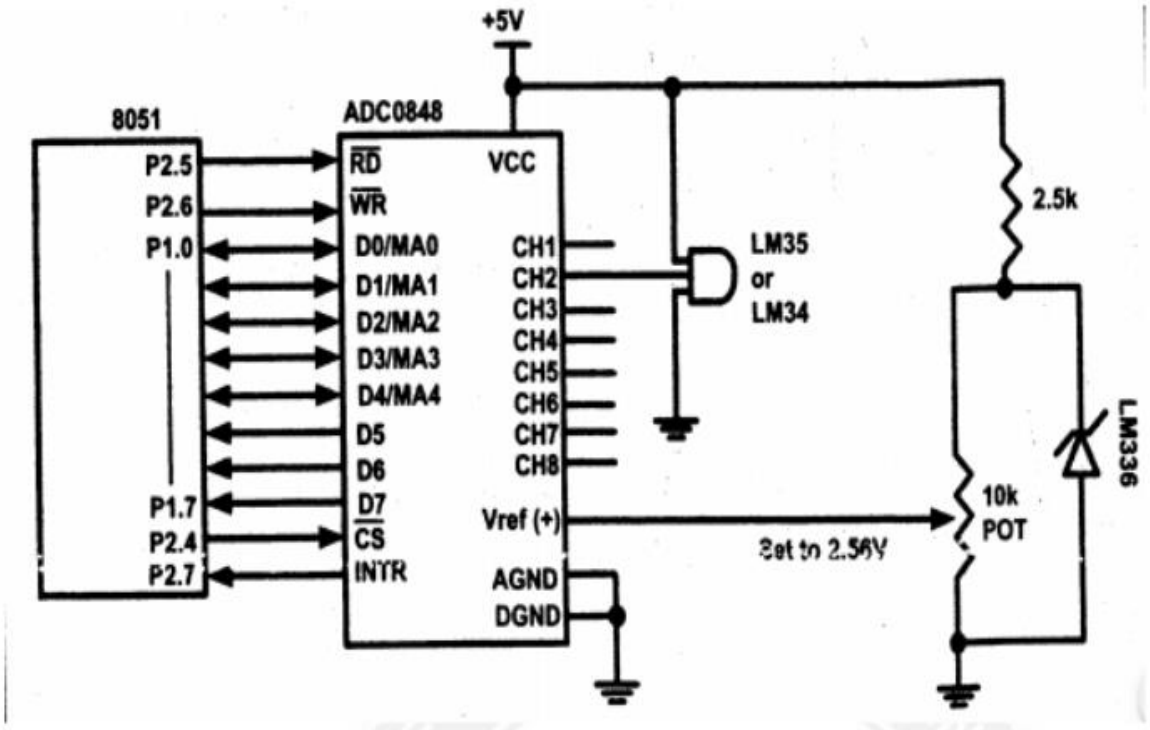
Example 7 : Design a µController system using 8051 to interface Temparature sensor

Solution-:

LM35 Temperature Sensors;:

- The LM35 series sensors are precision integrated-circuit temperature sensors whose output voltage is linearly proportional to the celsius (centigrade) temperature.
- The LM35 requires no external calibration since it is internally calibrated.
- It outputs 10mV for each degree of centigrade temperature.





Program-

```
RD BIT P2.5                ;RD
 WR BIT P2.6               ;WR
INTR BIT P2.7              ; END OF CONVERSION
MYDATA EQU P1              ; P1.0-P1.7 = D0-D7 OF THE ADC0848
MOV P1,#0FFH               ;make P1 = input
SETB INTR
BACK: CLR WR               ;WR = 0
 SETB WR                   ;WR = 1 L-to-H to start conversion
 HERE: JB INTR, HERE       ;wait for end of conversion
 CLR RD                    ;conversion finished, enable RD
 MOV A,MYDATA              ;read the data
 ACALL CONVERSION          ;hex-to-ASCII conversion
 ACALL DATA_DISPLAY        ;display the data
 SETB RD                   ;make RD=1 for next round
 SJMP BACK

 CONVERSION:
 MOV B,#10
 DIV AB
 MOV R7,B
 MOV B,#10
 DIV AB
 MOV R6,B
 MOV R5,A
 RET
 DATA_DISPLAY:
 MOV P0,R7
```

```
ACALL DELAY
MOV P0,R6
ACALL DELAY
MOV P0,R5
ACALL DELAY
RET
```

Interfacing DC motor

Example 8 : Design a µController system using 8051 to interface DC motor

DC Motor interfacing

 DC motor is a device that translates electrical pulses into mechanical movement

 The DC motor has + and – leads

 Connecting them to a DC voltage source moves the motor in one direction and by reversing the polarity, the DC motor will move in opposite direction

Motor Control Using L293