MODULE-5
I/O interfacing with Microcontroller 8051

V.PRAKASH
Asst. Professor(Sr.), SENSE,
VIT Chennai

MODULE-5

I/O interfacing with Microcontroller 8051

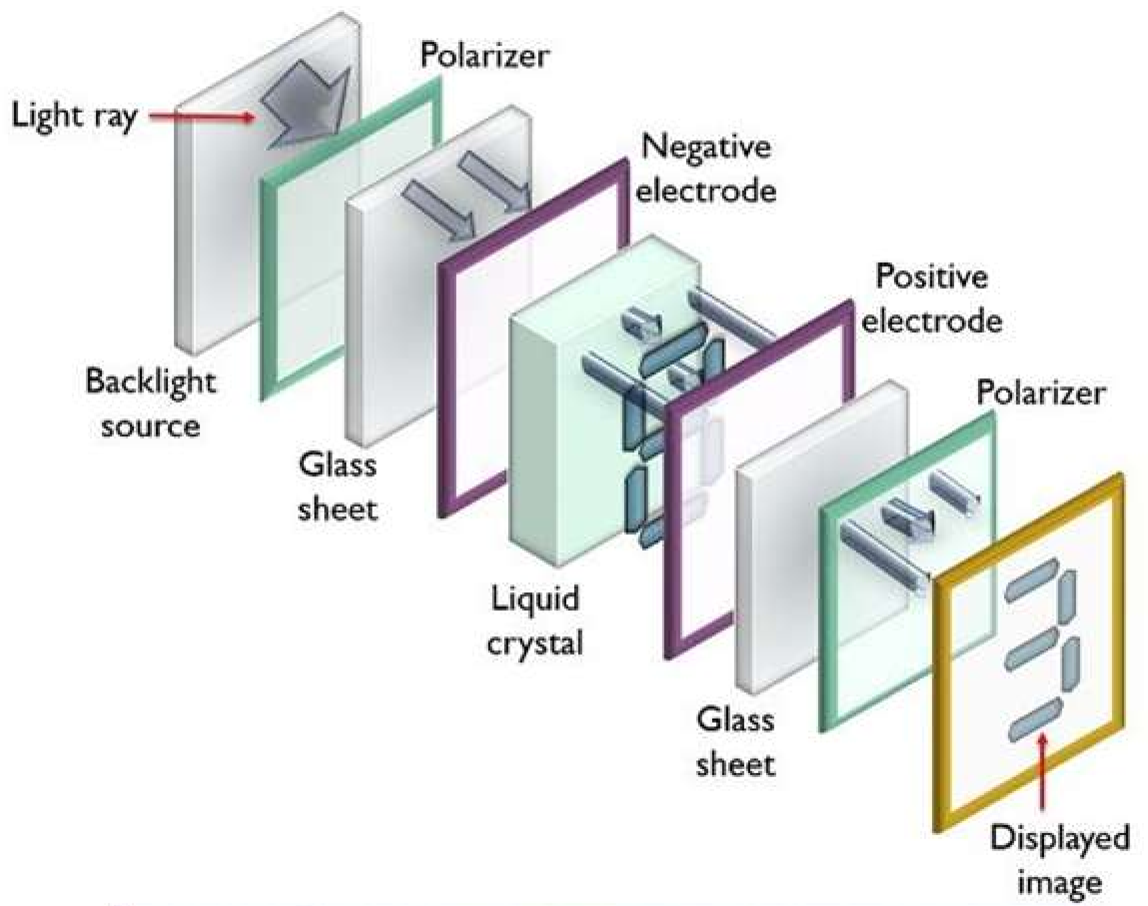
- LCD
- LED
- Keypad
- Analog-to-Digital Convertors,
- Digital-to-Analog Convertors
- Sensor with Signal Conditioning Interface.

LCD

LIQUID CRYSTAL DISPLAY (LCD)

- Display units are the most important output devices in many electronics products and LCD is one of the most used display unit in many applications.
- LCD is composed of liquid crystal particles which do not emit light on their own instead they are illuminated by a backlight hence they need an external light source to work.
- When light from a backlight source is emitted and allowed to fall on the vertical polarizer. Then the unpolarized light by the source gets vertically polarized.
- When initially no external potential is provided between the two electrodes, the molecules of the liquid crystal remain twisted.
- This causes the vertically polarized light to get horizontally polarized due to the orientation of the molecules.

LIQUID CRYSTAL DISPLAY (LCD)



Working structure of LCD

- When an electric current is applied to them, they tend to untwist and causes a change in the light angle passing through them.
- Further this causes a change in the angle of the top polarizing filter with respect to it.
- So little light is allowed to pass through that particular area of LCD.
- Thus that area becomes darker comparing to others.

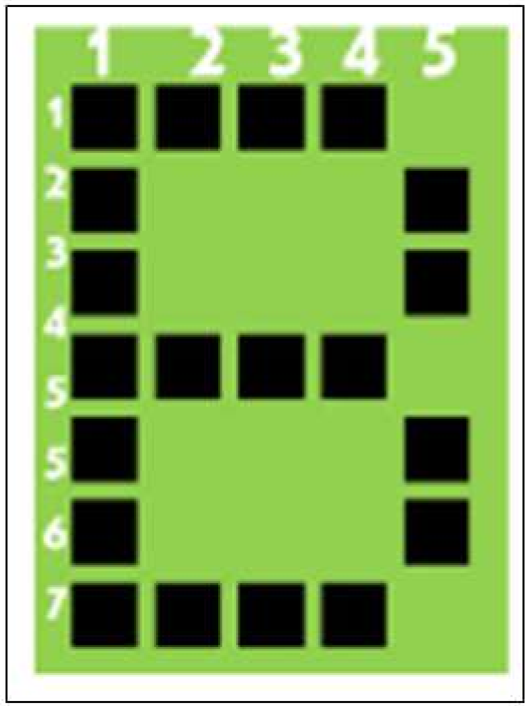
LIQUID CRYSTAL DISPLAY (LCD)

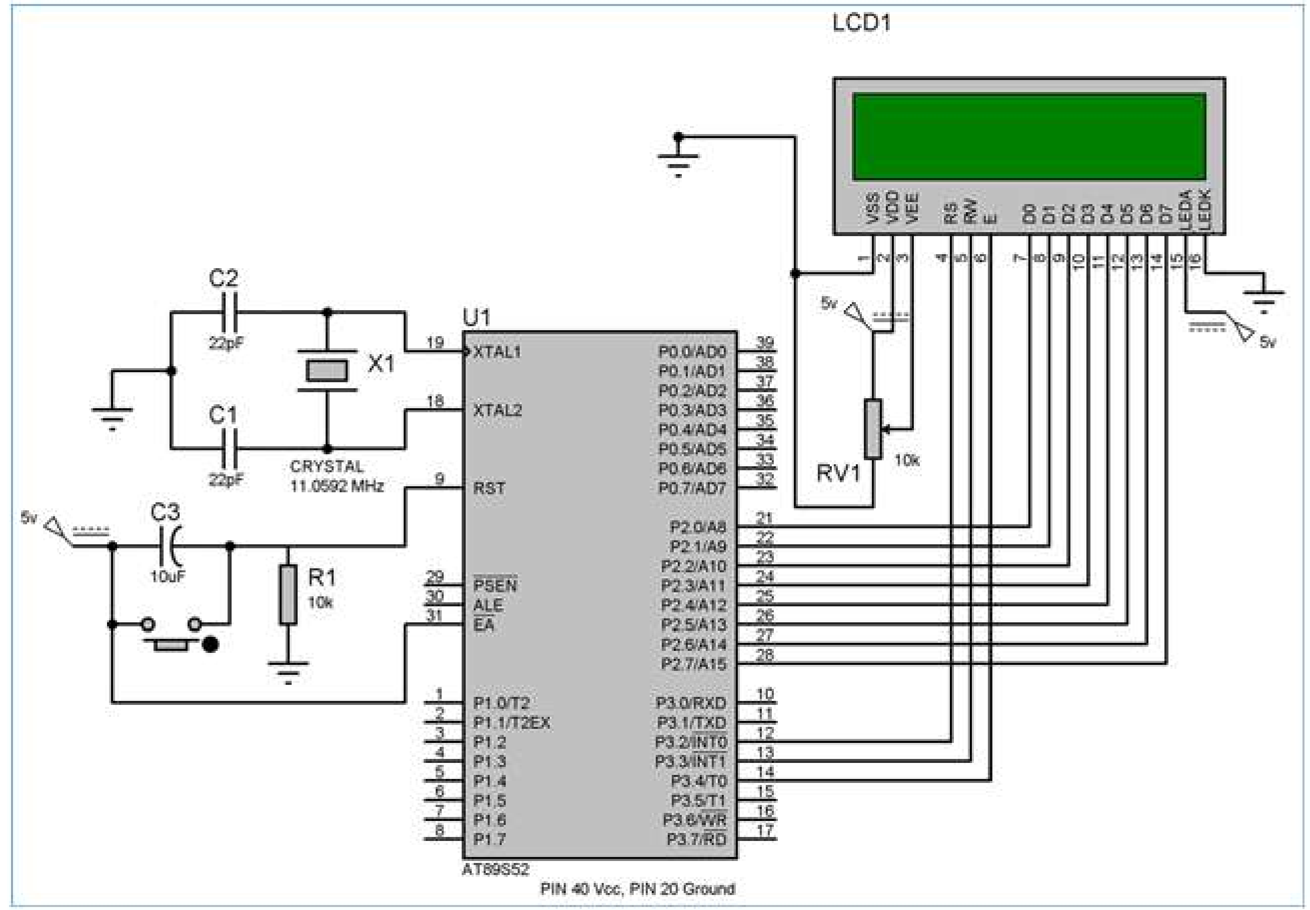
16x2 LCD

PIN NO.	NAME	FUNCTION
1	VSS	Ground pin
2	VCC	Power supply pin of 5V
3	VEE	Used for adjusting the contrast commonly attached to the potentiometer.
4	RS	RS is the register select pin used to write display data to the LCD (characters), this pin has to be high when writing the data to the LCD. During the initializing sequence and other commands this pin should low.
5	R/W	Reading and writing data to the LCD for reading the data R/W pin should be high (R/W=1) to write the data to LCD R/W pin should be low (R/W=0)
6	E	Enable pin is for starting or enabling the module. A high to low pulse of about 450ns pulse is given to this pin.
7	DB0	DB0-DB7 Data pins for giving data (normal data like numbers Characters or command data) which is meant to be displayed
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	LED+	Back light of the LCD which should be connected to Vcc
16	LED-	Back light of LCD which should be connected to ground.

LIQUID CRYSTAL DISPLAY (LCD)

- In 16x2 LCD, 2 represents number of lines and 16 represents number of characters displayed in each line. It supports all the ASCII characters and provides the provision to display the custom characters by creating the pattern. Each character in LCD is displayed in a matrix of 5x7 pixels.





LIQUID CRYSTAL DISPLAY (LCD)

- The 16X2 LCD has two built in registers namely data register and command register.

- Command Register - stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing, clearing the screen, setting the cursor position, controlling display etc.

- Data Register - stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

- For programming LCD follow these steps:
  - STEP1: Initialization of LCD.
  - STEP2: Sending command to LCD.
  - STEP3: Writing the data to LCD.

Command	Function
01	Clear screen
02	Return home
04	Decrement cursor
05	Shift display right
06	Increment cursor
07	Shift display left
08	Display OFF, Cursor OFF
0A	Display OFF, Cursor ON
0C	Display ON, Cursor OFF
0E	Display ON ,Cursor blinking OFF
0F	LCD ON, Cursor ON, Cursor blinking ON
10	Shift cursor position to left
14	Shift cursor position to right
80	Force cursor to the beginning of 1st line
83	Cursor line 1 position 3
C0	Force cursor to the beginning of 2nd line
C1	Jump to second line, position1
38	Use 2 lines and 5×7 matrix

LIQUID CRYSTAL DISPLAY (LCD)

Step1: LCD initialization (common for almost all applications)

1. Send 38H to the 8 bit data line for initialization
2. Send 0FH for making LCD ON, cursor ON and cursor blinking ON.
3. Send 06H for incrementing cursor position.
4. Send 01H for clearing the display and return the cursor.

Step2: Sending command to LCD

1. Send the command data to command register
2. Make R/W low.
3. Make RS=0 if data byte is a command
4. Pulse E from high to low with some delay.
5. Repeat above steps for sending another command.

Step3: Writing the data to LCD

1. Place data byte on the data register.
2. Make R/W low.
3. make RS=1 if the data byte is a data to be displayed.
4. Pulse E from high to low with some delay.
5. Repeat above steps for sending another data.

LIQUID CRYSTAL DISPLAY (LCD)

EXAMPLE-1

Write an 8051 assembly language program to display the message "VIT" on LCD display. Assume following,

- 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

```
ORG 0000H
MOV A, #38H              ; INITIALIZE 2x16 LCD
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
```

LIQUID CRYSTAL DISPLAY (LCD)

```
        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, #'V'         ; display letter N
        ACALL DATAWRT       ; call display subroutine
        ACALL DELAY         ; give LCD some time
        MOV A, #'I'         ; display letter O
        ACALL DATAWRT       ; call display subroutine
        ACALL DELAY         ; give LCD some time
        MOV A, #'T'         ; display letter O
        ACALL DATAWRT       ; call display subroutine
        ACALL DELAY         ; give LCD some time
AGAIN:  SJMP AGAIN          ; stay here
```

LIQUID CRYSTAL DISPLAY (LCD)

```
COMNWRT:  MOV P1, A        ; send command to LCD by coping 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
          ACALL DELAY      ; give LCD some time
          CLR P2.2         ; E=0 for H-to-L pulse
          RET
DATAWRT:  MOV P1, A        ; write data to LCD by coping 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
          ACALL DELAY      ; give LCD some time
          CLR P2.2         ; E=0 for H-to-L pulse
          RET
DELAY:    MOV R3, #50      ; 50 or higher for fast CPUs
HERE2:    MOV R4, #255     ; R4 = 255
HERE:     DJNZ R4, HERE    ; stay until R4 becomes 0
          DJNZ R3, HERE2   ; stay until R3 becomes 0
          RET
```

LIQUID CRYSTAL DISPLAY (LCD)

EXAMPLE-2

Write an 8051 assembly language program to display the message "HELLO" on LCD display using DPTR. Assume ; P1.0-P1.7=D0-D7, P2.0=RS, P2.1=R/W, P2.2=E.

```
        ORG 0000H
        MOV DPTR, #MYCOM
C1:     CLR A
        MOVC A,@A+DPTR
        ACALL COMNWRT
        ACALL DELAY
        INC DPTR
        JZ SEND_DAT
        SJMP C1
```

Exercise: Write an 8051 assembly language program to display "Your Reg. No" on first line of the LCD and "Your name" on the second line of the LCD using DPTR.

LIQUID CRYSTAL DISPLAY (LCD)

```
SEND_DAT:   MOV DPTR, #MYDATA
D1:         CLR A
            MOVC A,@A+DPTR
            ACALL DATAWRT
            ACALL DELAY
            INC DPTR
            JZ AGAIN
            SJMP D1
AGAIN:      SJMP AGAIN
COMNWRT:    MOV P1, A           ; send command to LCD by coping A to P1
            CLR P2.0            ; RS=0 for command
            CLR P2.1            ; R/W=0 for write
            SETB P2.2           ; E=1 for high pulse
            ACALL DELAY         ; give LCD some time
            CLR P2.2            ; E=0 for H-to-L pulse
            RET
```

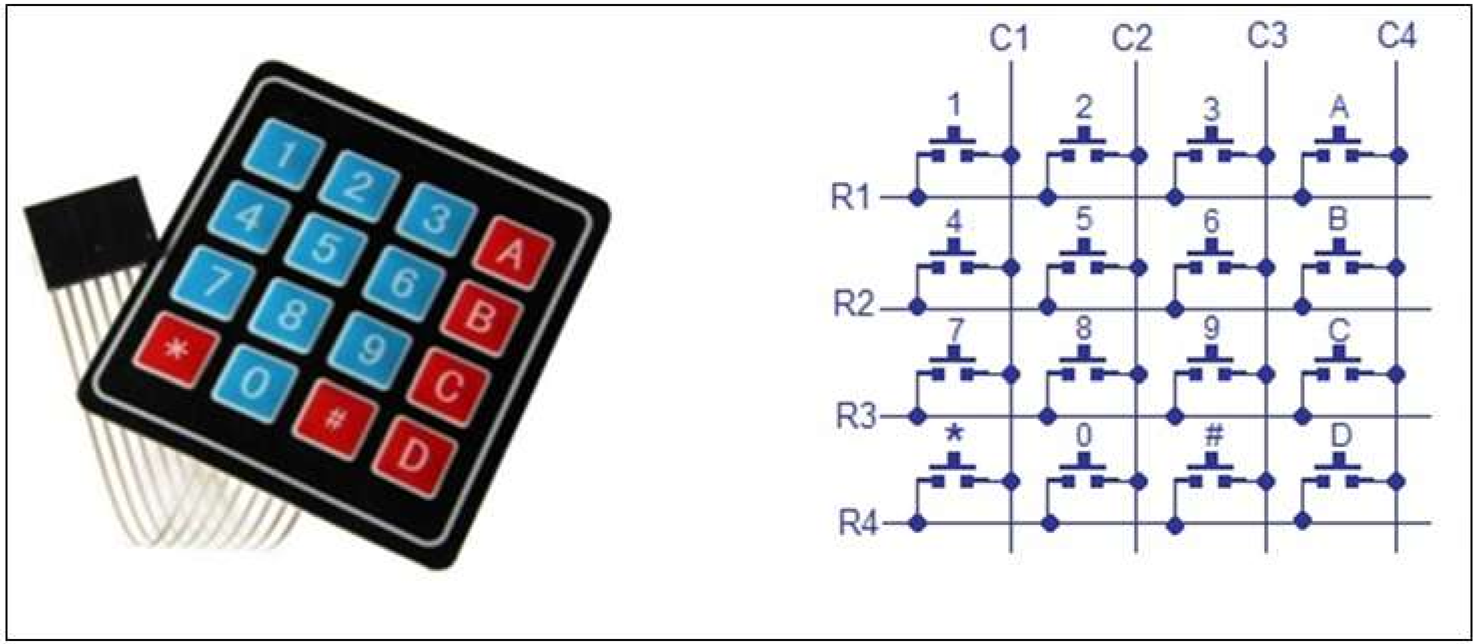
LIQUID CRYSTAL DISPLAY (LCD)

```
DATAWRT:    MOV P1, A            ; write data to LCD by coping A into P1
            SETB P2.0            ; RS=1 for data
            CLR P2.1             ; R/W=0 for write
            SETB P2.2            ; E=1 for high pulse
            ACALL DELAY          ; give LCD some time
            CLR P2.2             ; E=0 for H-to-L pulse
            RET
DELAY:      MOV R3, #250         ; 50 or higher for fast CPUs
HERE2:      MOV R4, #255         ; R4 = 255
HERE:       DJNZ R4, HERE        ; stay until R4 becomes 0
            DJNZ R3, HERE2
            RET
ORG 300H
MYCOM:      DB 38H, 0EH, 01, 06, 84H, 0      ; commands and null
MYDATA:     DB "HELLO", 0
END
```

KEYPAD

KEYPAD

- Keyboards are organized in a matrix of rows and columns
  - A 4x4 matrix connected to two ports - rows are connected to an output port and the columns are connected to an input port
  - When a key is pressed, a row and a column make a contact



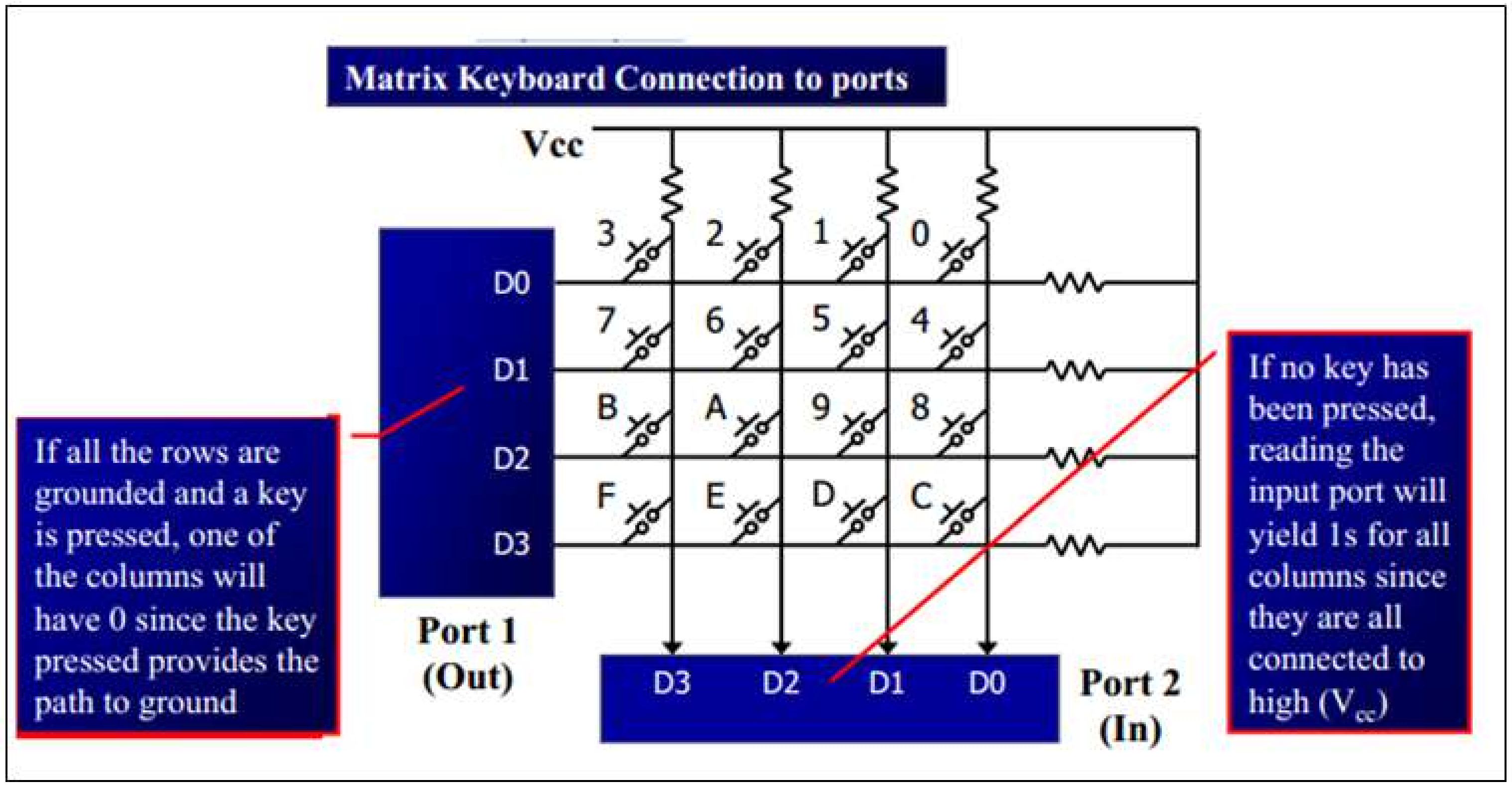
KEYPAD

- It is the function of the microcontroller to scan the keyboard continuously to detect and identify the key pressed
- To detect a pressed key, the microcontroller grounds all rows by providing 0 to the output latch, then it reads the columns
- If the data read from columns is D3 –D0 = 1111, no key has been pressed and the process continues till key press is detected
- If one of the column bits has a zero, this means that a key press has occurred
- It grounds the next row, reads the columns, and checks for any zero, this process continues until the row is identified
- After identification of the row in which the key has been pressed it find out which column the pressed key belongs to.

KEYPAD

- Identify the row and column of the pressed key for
  - (a) D3 – D0 = 1110 for the row, D3 – D0 = 1011 for the column
  - (b) D3 – D0 = 1101 for the row, D3 – D0 = 0111 for the column

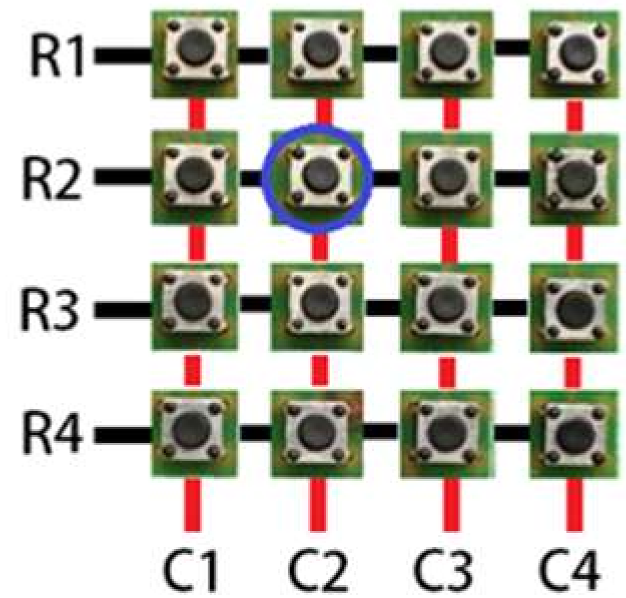
  Answer : (a). 2 (b). 7



KEYPAD

STEPS FOR KEY PRESS IDENTIFICATION

- Initially all switches are assumed to be released. So there is no connection between the rows and columns.
- When any one of the switches are pressed, the corresponding row and column are connected (short circuited). This will drive that column pin (initially high) low.
- Using this logic, the button press can be detected. The colors red and black is for logic high and low respectively.
- Step 1: The first step involved in interfacing the matrix keypad is to write all logic 0's to the rows and all logic 1's to the columns. In the image, black line symbolizes logic 0 and red line symbolizes logic 1.

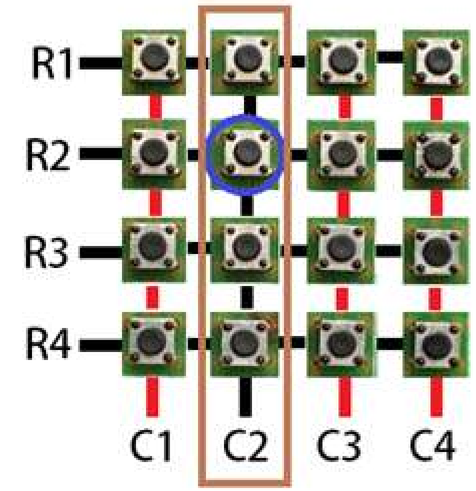


Assume a Key press

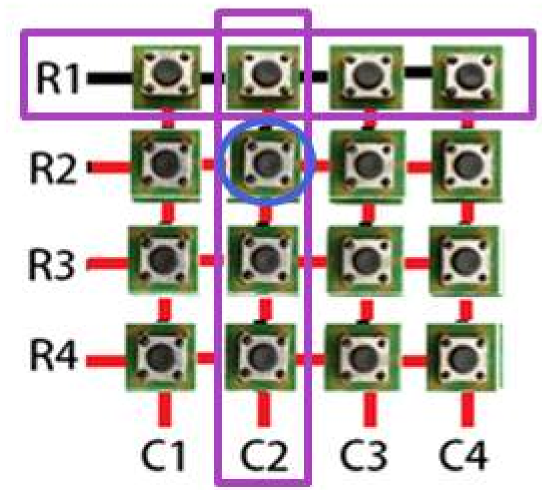
KEYPAD

STEPS FOR KEY PRESS IDENTIFICATION

- Step 2: Now the program has to scan the pins connected to columns of the keypad. If it detects a logic 0 in any one of the columns, then a key press was made in that column. This is because the event of the switch press shorts the C2 line with R2. Hence C2 is driven low.



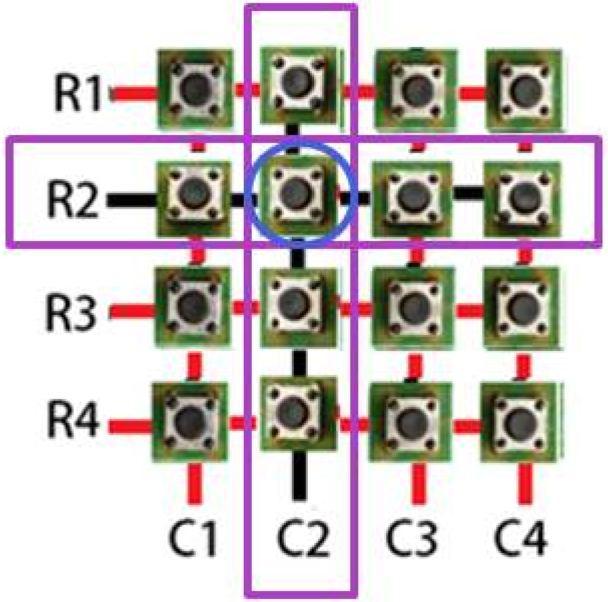
- Step 3: Once the column corresponding to the key pressed is located, start writing logic 0's to the rows sequentially (one after the other) and check if C2 becomes low.



KEYPAD

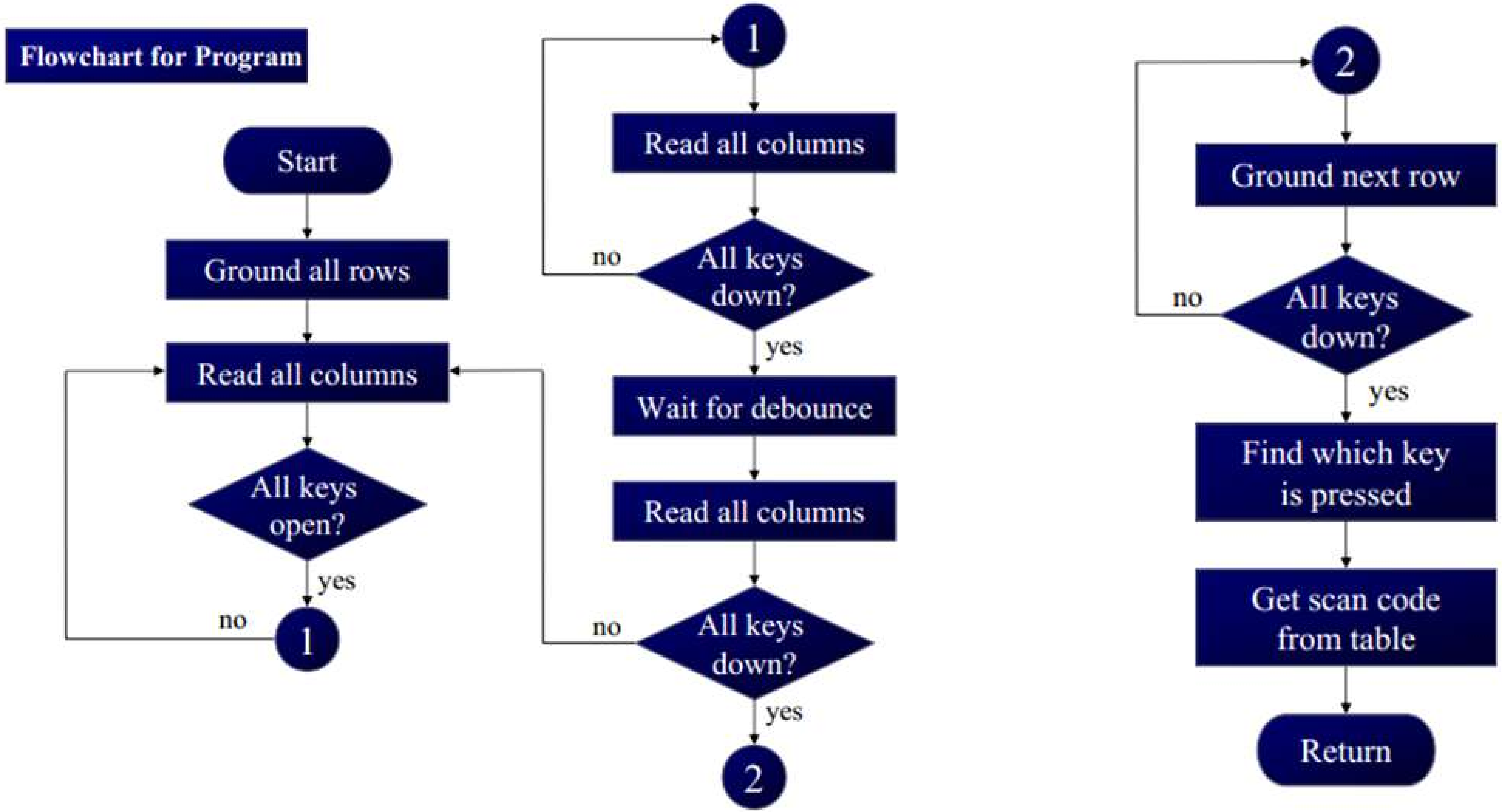
STEPS FOR KEY PRESS IDENTIFICATION

- Step 4: The procedure is followed till C2 goes low when logic low is written to a row. In this case, a logic low to the second row will be reflected in the second column.
- We already know that the key press happened at column 2. Now we have detected that the key is in row 2. So, the position of the key in the matrix is (2,2).
- Once this is detected, its up to us to name it or provide it with a task on the event of the key press.



KEYPAD

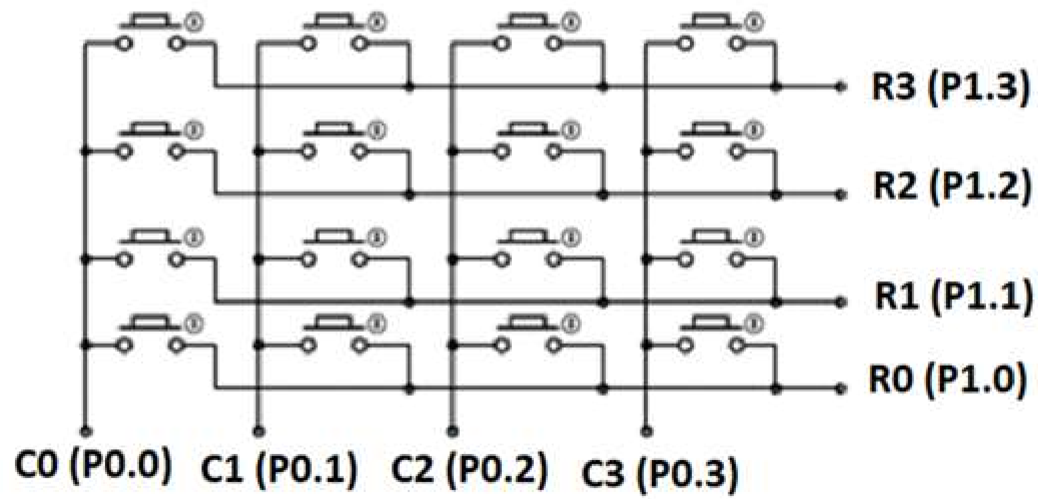
STEPS FOR KEY PRESS IDENTIFICATION



KEYPAD

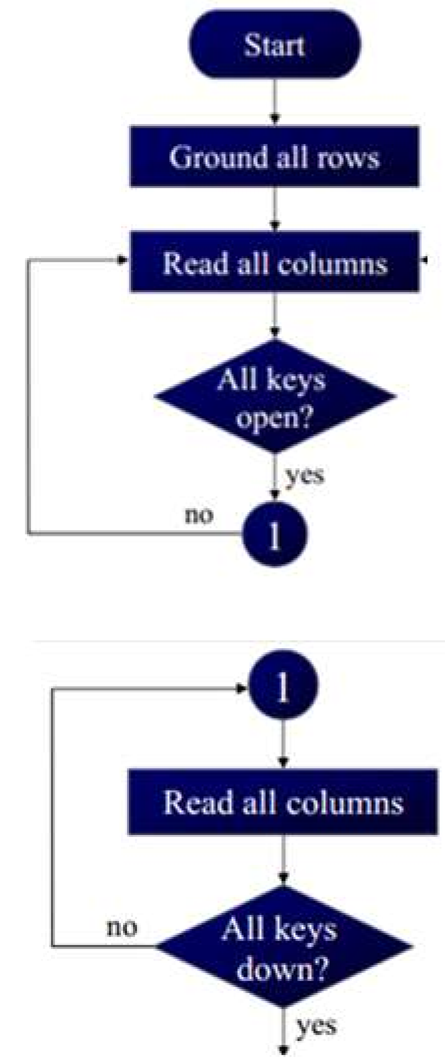
PROGRAM

Write an assembly language program for the 8051 to interface the 4x4 matrix keypad and LCD. Any key pressed on the Keypad must be display in LCD.



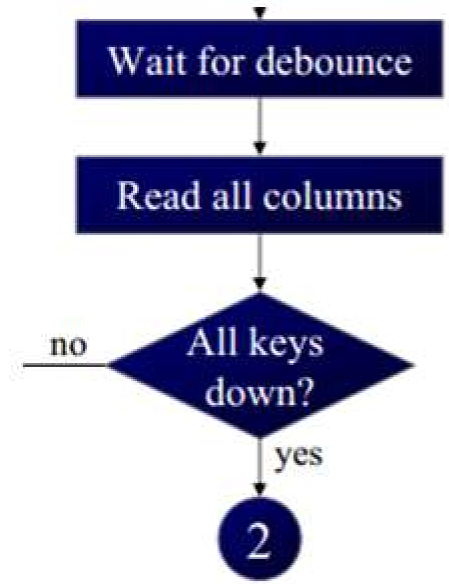
KEYPAD

```
         ORG 000H
         SJMP START

         ORG 0030H
START:   MOV P0,#0FFH              ;MAKE P0 AN INPUT PORT
         ACALL LCD_INITIALIZE

K1:      MOV P1,#0                 ;GROUND ALL ROWS AT ONCE
         MOV A,P0                  ;READ ALL COL.
         ANL A,#00001111B          ;MASKED UNUSED BIT
         CJNE A,#00001111B,K1      ;CHECK  ALL KEYS RELEASED

K2:      ACALL DELAY               ;CALL 20 MS DELAY
         MOV A,P0                  ;SEE IF ANY KEY IS PRESSED
         ANL A,#00001111B          ;MASKED UNUSED BIT
         CJNE A,#00001111B,OVER    ;KEY PRESSED, WAIT
         SJMP K2                   ;CHECK TILL KEY PRESSED
```



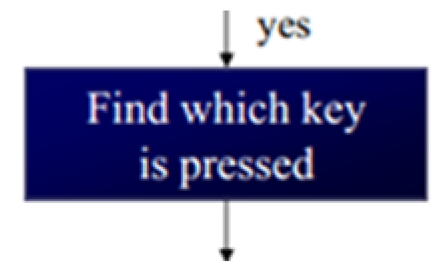
KEYPAD

```
OVER:   ACALL DELAY              ;WAIT 20 ms Key DEBOUNCE TIME
        MOV A,P0                 ;CHECK KEY CLOSURE
        ANL A,#00001111B         ;MASKED UNUSED BIT
        CJNE A,#00001111B,OVER1  ;KEY PRESSED, FIND ROW
        SJMP K2                  ;IF NONE, KEEP POLLING

OVER1:  MOV P1,#1111110B         ;GROUND ROW 0
        MOV A,P0                 ;READ ALL COLUMNS
        ANL A,#00001111B         ;MASKED UNUSED BIT
        CJNE A,#00001111B,ROW_0  ;ROW0, FIND COL
        MOV P1,#11111101B        ;GROUND ROW 1
        MOV A,P0                 ;READ ALL COL.
        ANL A,#00001111B         ;MASKED UNUSED BIT
        CJNE A,#00001111B,ROW_1  ;ROW 1, FIND THE COL
```



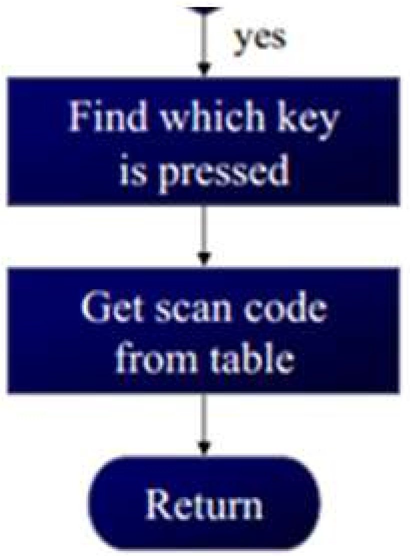
KEYPAD

```
        MOV P1,#11111011B           ;GROUND ROW 2
        MOV A,P0                    ;READ ALL COL.
        ANL A,#00001111B            ; MASKED UNUSED BIT
        CJNE A,#00001111B,ROW_2     ;ROW 2, FIND  COL
        MOV P1,#11110111B           ;GROUND ROW 3
        MOV A,P0                    ;READ ALL COL.
        ANL A,#00001111B            ;MASKED UNUSED BIT
        CJNE A,#00001111B,ROW_3     ;ROW 3, FIND  COL
        LJMP K2                     ;IF NONE, FALSE INPUT, REPEAT
ROW_0:  MOV DPTR, #KCODE0           ;SET DPTR=START OR ROW 0
        SJMP FIND                   ;FIND COLUMN BELONGS TO
ROW_1:  MOV DPTR, #KCODE1           ;SET DPTR=START OR ROW 1
        SJMP FIND                   ;FIND COLUMN BELONGS TO
ROW_2:  MOV DPTR, #KCODE2           ;SET DPTR=START OR ROW 2
        SJMP FIND                   ;FIND COLUMN BELONGS TO
ROW_3:  MOV DPTR, #KCODE3           ;SET DPTR=START OR ROW 3
```



KEYPAD

```
FIND:       RRC A                ;SEE IF ANY CY BIT LOW
            JNC MATCH            ;IF ZERO GET ASCII CODE
            INC DPTR             ;POINT TO NEXT COLUMN
            SJMP FIND            ;KEEP SEARCHING

MATCH:      CLR A
            MOVC A,@A+DPTR
            ACALL LCD_DATA       ;GET CODE FROM LOOK-UP TABLE
            LJMP K1              ;LOOP

DELAY:      MOV R4,#40
REPEAT:     MOV R5,#230
REPEATT:    DJNZ R5,REPEATT
            DJNZ R4,REPEAT
            RET
```



KEYPAD

```
                 ORG 300H
KCODE3:          DB 'C','D','E','F'      ;ROW 3
KCODE2:          DB '8','9','A','B'      ;ROW 2
KCODE1:          DB '4','5','6','7'      ;ROW 1
KCODE0:          DB '0','1','2','3'      ;ROW 0
                 END

LCD_INITIALIZE:  MOV A,#38H
                 ACALL LCD_COMMAND
                 MOV A,#0EH
                 ACALL LCD_COMMAND
                 MOV A,#01H
                 ACALL LCD_COMMAND
                 MOV A,#06H
                 ACALL LCD_COMMAND
                 MOV A,#80H
                 ACALL LCD_COMMAND
                 RET
```

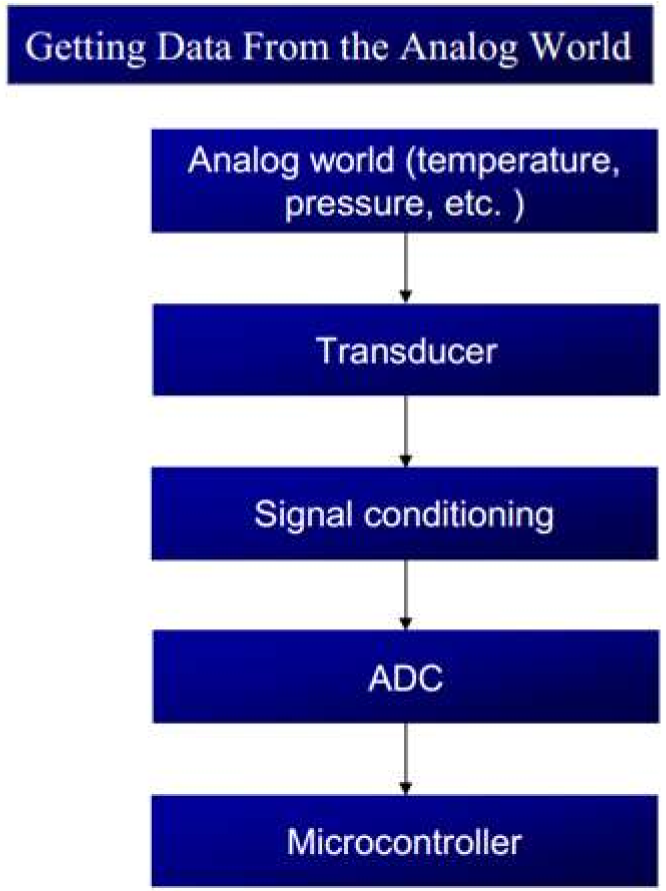
KEYPAD

```
LCD_COMMAND:    MOV P2,A
                CLR P3.7
                CLR P3.6
                SETB P3.5
                ACALL DELAY
                CLR P3.5
                RET

LCD_DATA:       MOV P2,A
                SETB P3.7
                CLR P3.6
                SETB P3.5
                ACALL DELAY
                CLR P3.5
                RET
```

ANALOG TO DIGITAL CONVERTER (ADC)

ANALOG TO DIGITAL CONVERTER (ADC)

- ADCs (analog-to-digital converters) are among the most widely used devices for data acquisition
- A physical quantity, like temperature, pressure, humidity, and velocity, etc., is converted to electrical (voltage, current) signals using a device called a transducer, or sensor
- Analog-to-digital converter needed to translate the analog signals to digital numbers, so microcontroller can read them
- Types of ADC:
    - Parallel - 8 or more pins for the binary data. Ex: 0804, 0808
    - Serial - only one pin for data out. Ex: MAX1112
- ADC Resolution (in bits) : 8,10,12,16 or 24



ANALOG TO DIGITAL CONVERTER (ADC)

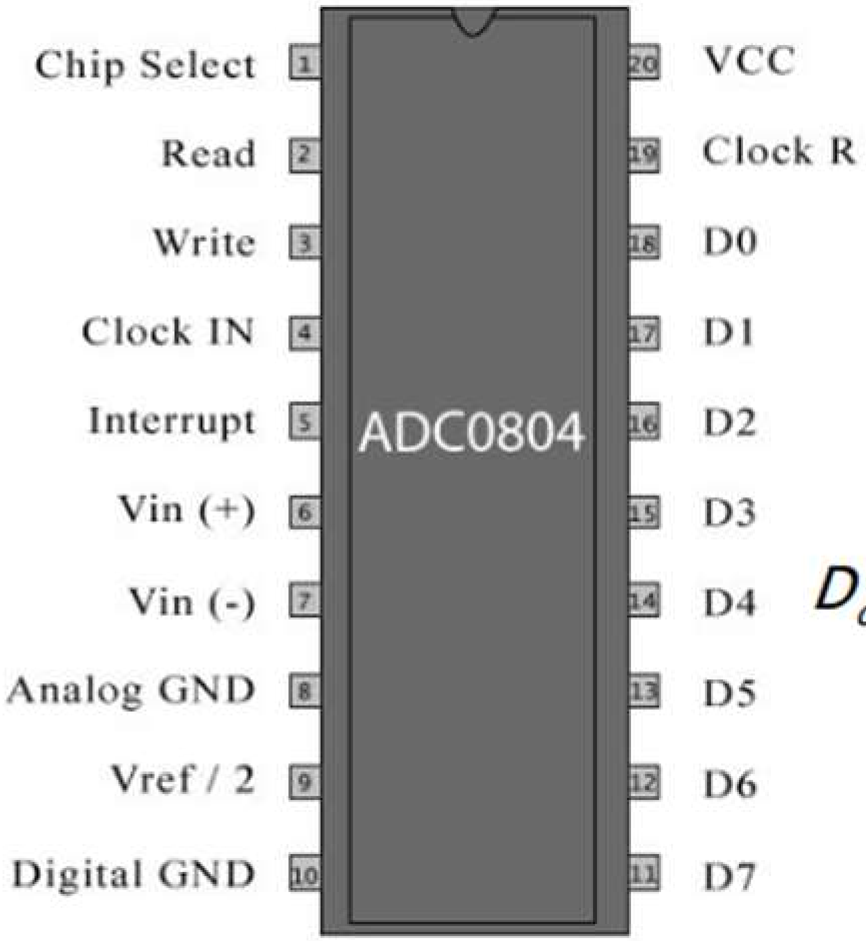
- The higher resolution ADC provides high accuracy by having a smaller step size. Step size is smallest change that can be detected by an ADC.

- Conversion time: Time taken by ADC to convert the analog input to a digital (binary) number.

n-bit	Number of steps	Step Size (mV)
8	2^8 = 256	5/255 = 19.61
10	2^10 = 1024	5/1023 = 4.89
12	2^12 = 4096	5/4095 = 1.22
16	2^16 = 65536	5/65535 = 0.076

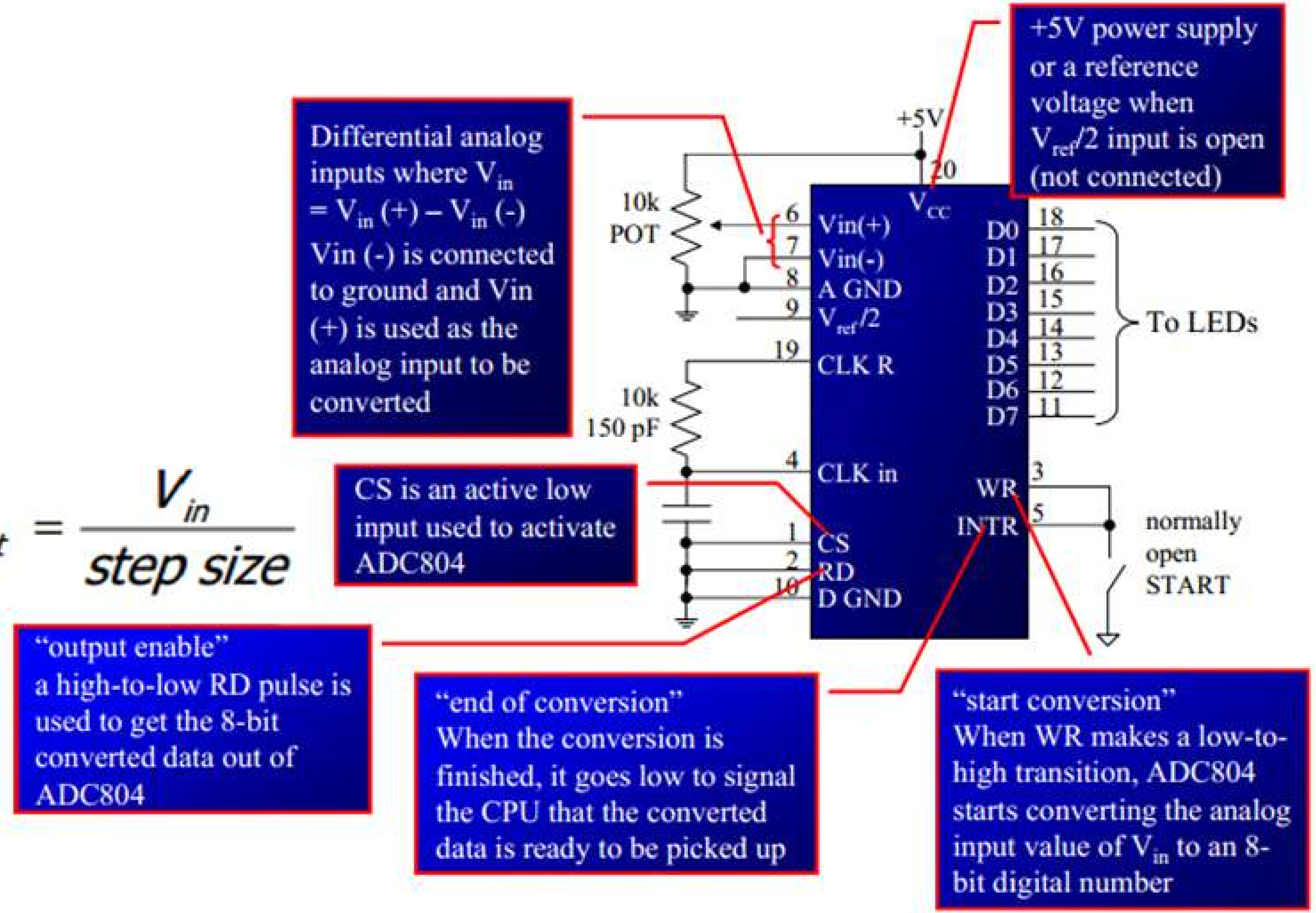
- ADC0804 IC is A 8-bit parallel an analog-to-digital converter
  - Successive approximation ADC
  - It works with +5 volts and has a resolution of 8 bits

ANALOG TO DIGITAL CONVERTER (ADC)



ADC 0804 PIN DIAGRAM

$$D_{out} = \frac{V_{in}}{step\ size}$$



ANALOG TO DIGITAL CONVERTER (ADC)

- **CLK IN and CLK R:**
  - CLK IN is an input pin connected to an external clock source
  - To use the internal clock generator (also called self-clocking), CLK IN and CLK R pins are connected to a capacitor and a resistor, and the clock frequency is determined by

$$f = \frac{1}{1.1\,RC}$$

  - Typical values are R = 10K ohms and C = 150 Pf.
  - We get f= 606 kHz and the conversion time is 110 µs

- **Vref/2:**
  - It is used for the reference voltage
  - If this pin is open (not connected), the analog input voltage is in the range of 0 to 5 volts (the same as the Vcc pin)
  - If the analog input range needs to be 0 to 4 volts, Vref/2 is connected to 2 volt

ANALOG TO DIGITAL CONVERTER (ADC)

- D0-D7:
  - The digital data output pins and these are tri-state buffered
  - The converted data is accessed only when CS =0 and RD is forced low
  - To calculate the output voltage, use the following formula

$$D_{out} = \frac{V_{in}}{step\ size}$$

  Where, Dout= digital data output (in dec.), Vin= analog voltage, and step size (resolution) is the smallest change

- The following steps must be followed for data conversion by the ADC0804 chip
  - Make CS = 0 and send a low-to-high pulse to pin WR to start conversion
  - Keep monitoring the INTR pin
    - If INTR is low, the conversion is finished
    - If the INTR is high, keep polling until it goes low
  - If INTR has become low, make CS = 0 and send a high-to-low pulse to RD pin to get the data out of the ADC0804

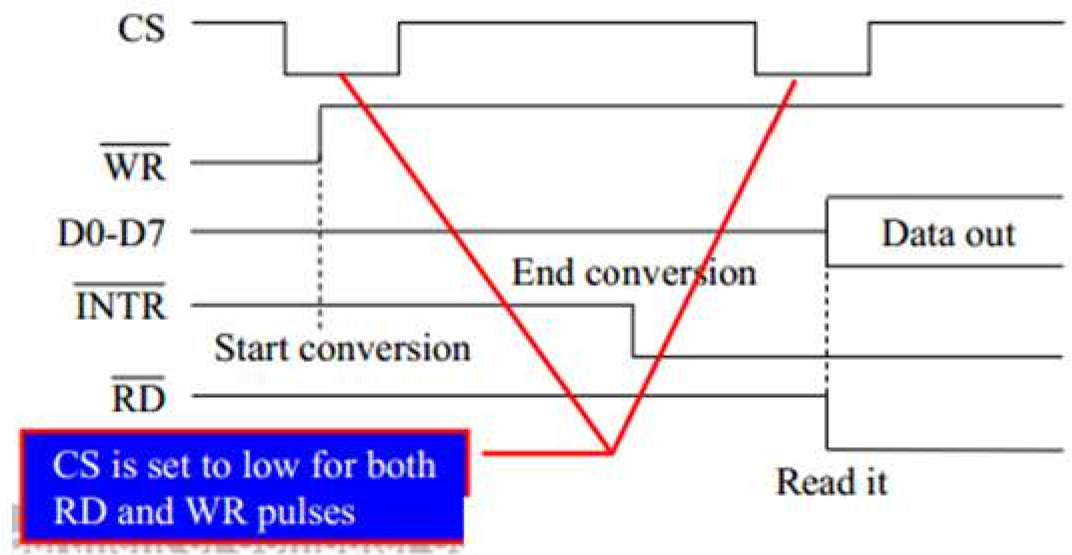
ANALOG TO DIGITAL CONVERTER (ADC)

For 8-bit ADC

$V_{ref}/2$ Relation to V_{in} Range

$V_{ref}/2$(v)	V_{in}(V)	Step Size (mV)
Not connected*	0 to 5	5/256=19.53
2.0	0 to 4	4/256=15.62
1.5	0 to 3	3/256=11.71
1.28	0 to 2.56	2.56/256=10
1.0	0 to 2	2/256=7.81
0.5	0 to 1	1/256=3.90

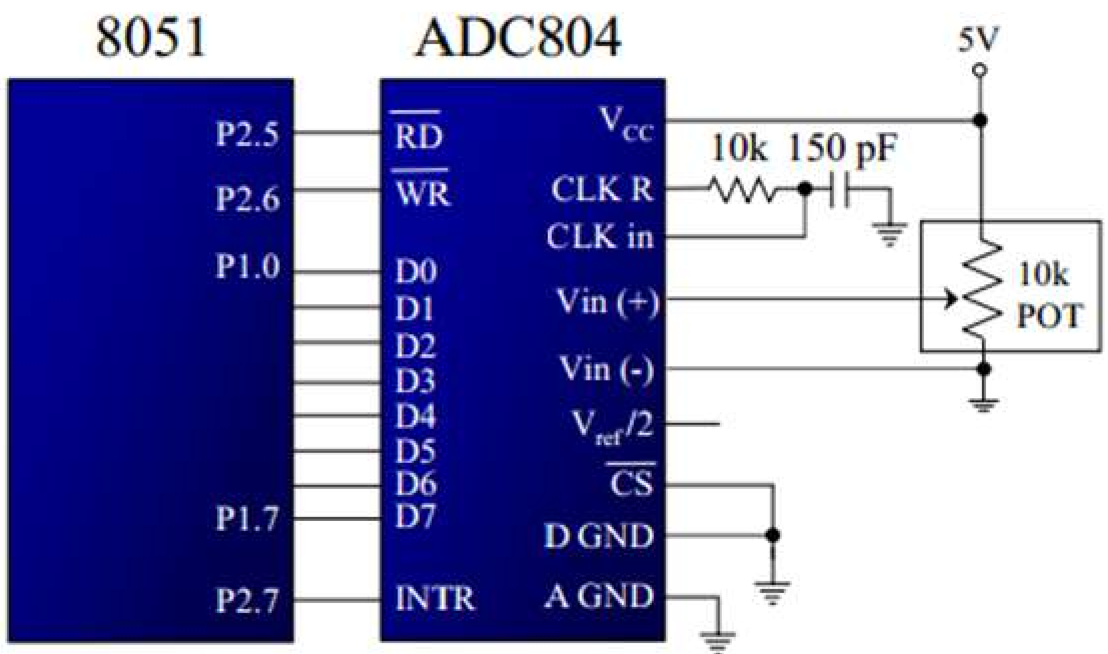
Step size is the smallest change can be discerned by an ADC



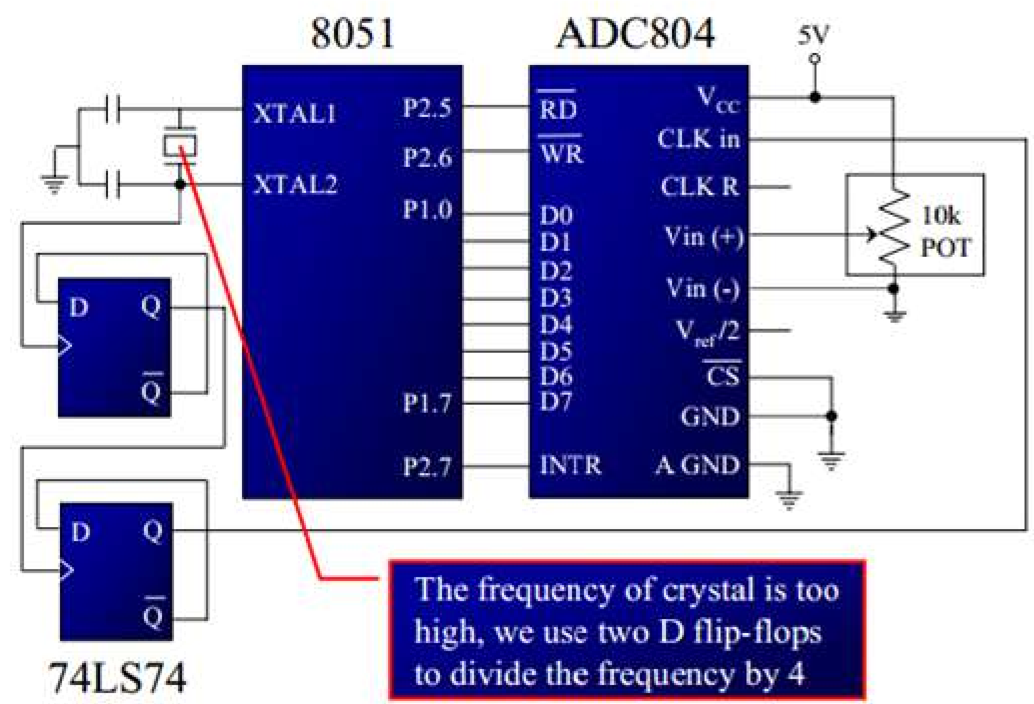
ANALOG TO DIGITAL CONVERTER (ADC)

- Examine the ADC804 connection to the 8051 in Figure. Write a program to monitor the INTR pin and bring an analog input into register A. Do this continuously.

8051 Connection to ADC804 with Self-Clocking



8051 Connection to ADC804 with Clock from XTAL2 of 8051



ANALOG TO DIGITAL CONVERTER (ADC)

```
;p2.6=WR (start conversion needs to L-to-H pulse)
;p2.7 When low, end-of-conversion)
;p2.5=RD (a H-to-L will read the data from ADC chip)
;p1.0 - P1.7= D0 - D7 of the ADC804
;
       MOV   P1,#0FFH      ;make P1 = input
BACK:  CLR   P2.6          ;WR = 0
       SETB  P2.6          ;WR = 1 L-to-H to start conversion
HERE:  JB    P2.7,HERE     ;wait for end of conversion
       CLR   P2.5          ;conversion finished, enable RD
       MOV   A,P1          ;read the data
       ACALL CONVERSION    ;hex-to-ASCII conversion
       ACALL DATA_DISPLAY;display the data
       SETB  p2.5          ;make RD=1 for next round
       SJMP  BACK
```

ANALOG TO DIGITAL CONVERTER (ADC)

INTERFACING LM35 WITH 8051

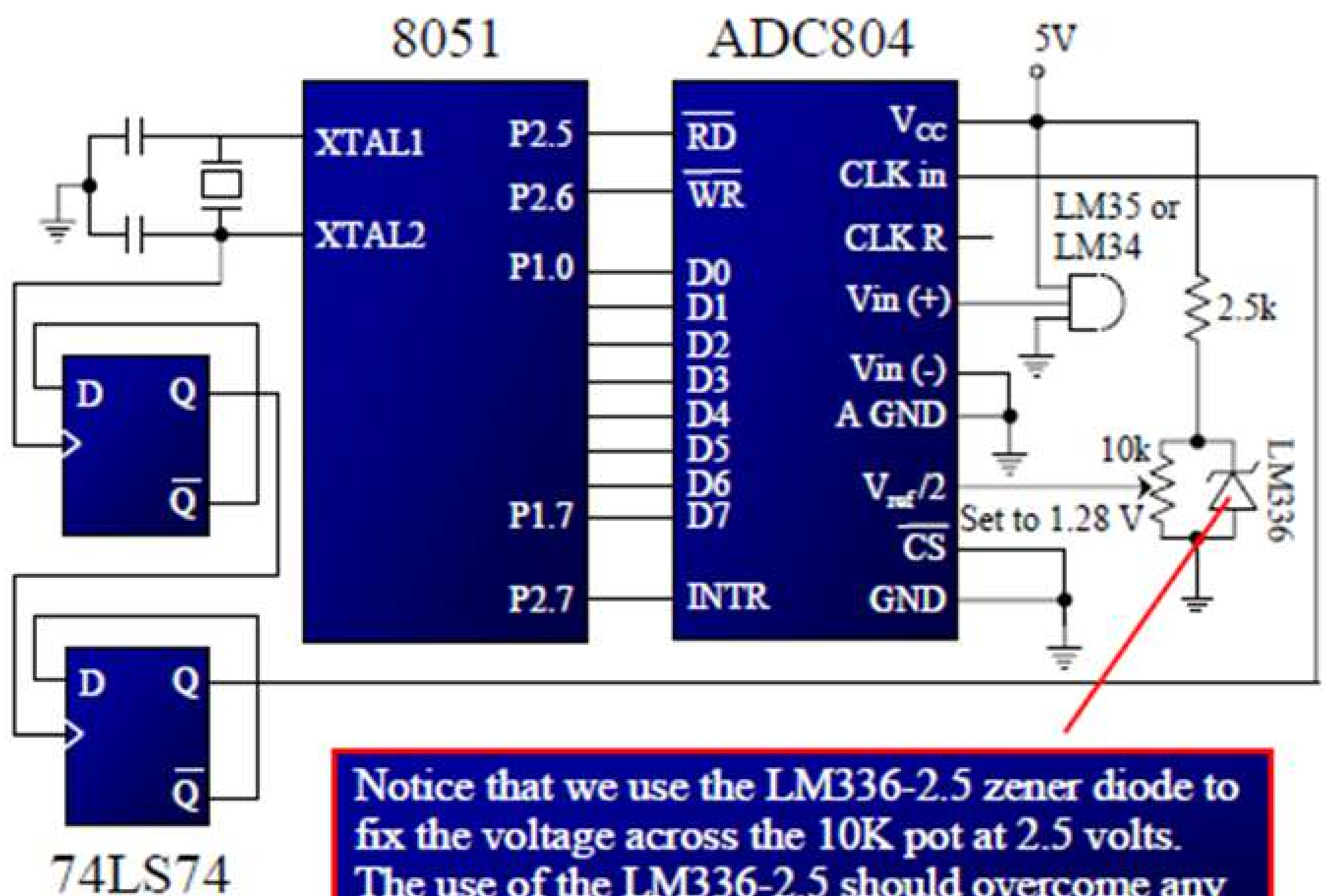
- The ADC804 has 8-bit resolution with a maximum of 256 steps and the LM35 (or LM34) produces 10 mV for every degree of temperature Change
- we can condition Vin of the ADC804 to produce a Vout of 2560 mV full-scale output. Therefore, in order to produce the fullscale Vout of 2.56 V for the ADC804
- We need to set Vref/2 = 1.28. This makes Vout of the ADC0804 correspond directly to the temperature as monitored by the LM35.

ANALOG TO DIGITAL CONVERTER (ADC)

Temperature vs. Vout of the ADC804

Temp. (C)	Vin (mV)	Vout (D7 – D0)
0	0	0000 0000
1	10	0000 0001
2	20	0000 0010
3	30	0000 0011
10	100	0000 1010
30	300	0001 1110

8051 Connection to ADC804 and Temperature Sensor



Notice that we use the LM336-2.5 zener diode to fix the voltage across the 10K pot at 2.5 volts. The use of the LM336-2.5 should overcome any fluctuations in the power supply

DIGITAL TO ANALOG CONVERTER (DAC)

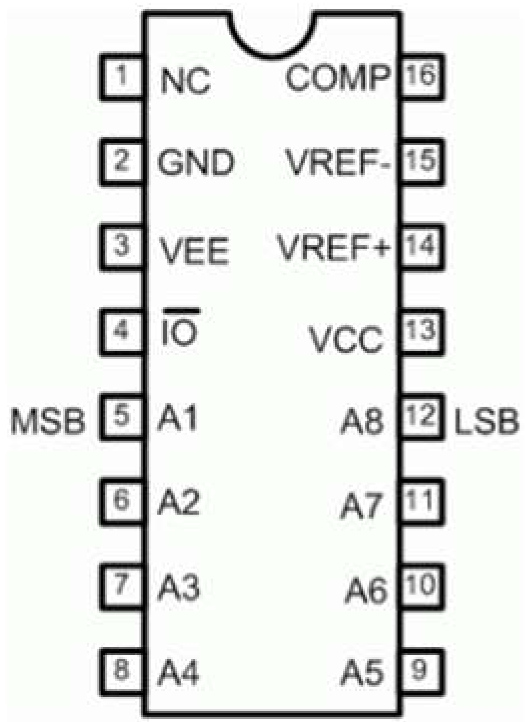
DIGITAL TO ANALOG CONVERTER (DAC)

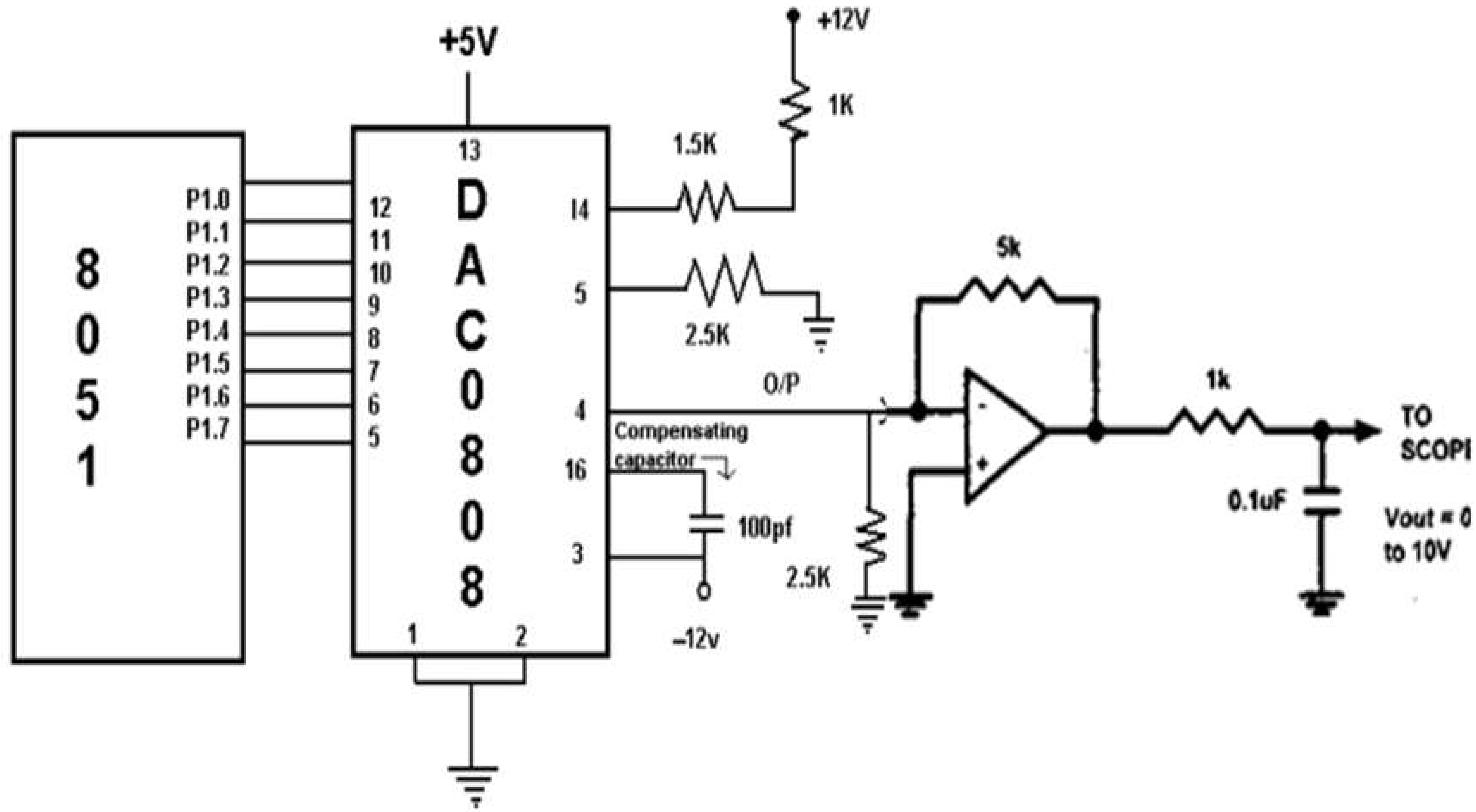
- µC generates output in digital form but the controlling system requires analog signal
- The digital-to-analog converter (DAC) is a device used to converts digital data into equivalent analog voltage/current.
- Most commonly used DAC is R/2R method due to high precision. DAC resolutions (in bits): 8, 10, and 12 bits
- The 8-bit DAC0808 converts digital data into equivalent analog Current hence we require an I to V converter to convert this current into equivalent voltage.
- The total current provided by the Iout pin is as follows:

$$I_{out} = I_{ref}\left(\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256}\right)$$

The Iref current is generally set to 2.0 mA.

DIGITAL TO ANALOG CONVERTER (DAC)





DIGITAL TO ANALOG CONVERTER (DAC)

PROGRAM

- Write a program to send data to the DAC to generate a Sawtooth, triangle and staircase waveforms.

```
                ORG 0000H
SAWTOOTH:       MOV A, #00H
     BACK:      MOV P1,A
                INC A
                CJNE A,#255, BACK
                MOV A,#00
                SJMP SAWTOOTH
                RET
```

DIGITAL TO ANALOG CONVERTER (DAC)

```
TRIANGLE:          MOV A,#00
        INCR:      MOV P1,A
                   INC A
                   CJNE A,#255, INCR
        DECR:      MOV P1,A
                   DEC A
                   CJNE A,#00, DECR
                   SJMP TRIANGLE

STAIRCASE:         MOV A,#00
                   MOV P1,A
        RPT:       ADD A,#51
                   MOV P1,A
                   CJNE A,#255, RPT
                   SJMP STAIRCASE
```

SENSOR WITH SIGNAL CONDITIONING INTERFACE

SENSOR WITH SIGNAL CONDITIONING INTERFACE

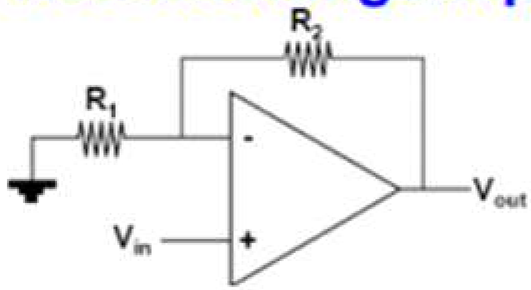
- Signal conditioning circuits are used to process the output signal from sensors to be suitable for the next stage of operation

- The function of the signal conditioning circuits include
  - Signal amplification (op-amp)
  - Filtering (op-amp)
  - Protection (Zener & photo isolation)
  - Linearization
  - Current – voltage change circuits
  - Resistance change circuits (Wheatstone bridge)
  - Error compensation

- Operational amplifiers are the basic element of many signal conditioning modules

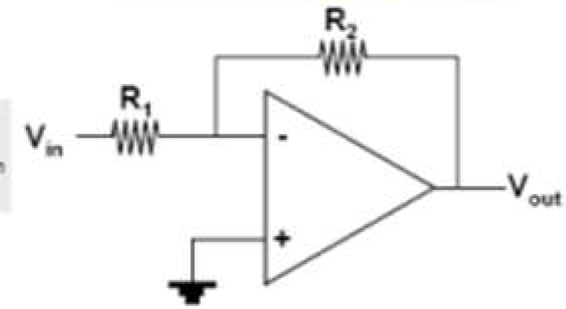
SENSOR WITH SIGNAL CONDITIONING INTERFACE

Non-Inverting Amp



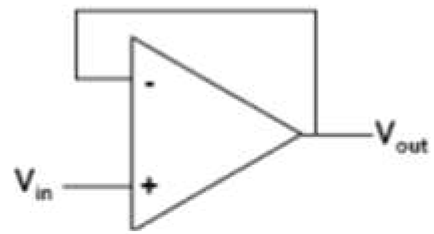
$$V_{out} = \left(1+\frac{R_2}{R_1}\right)V_{in}$$

Inverting Amp



$$V_{out} = -\frac{R_2}{R_1}V_{in}$$

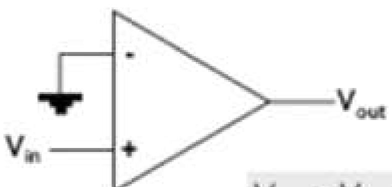
Voltage Follower



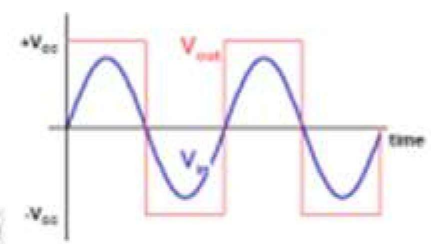
$$V_{out} = V_{in}$$

Voltage Comparator

– digitize input

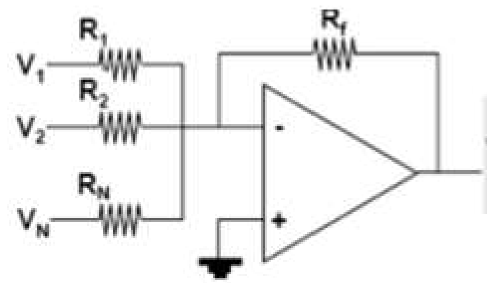


$$V_{out} = V_{CC}\,\mathrm{sign}(V_{in})$$



SENSOR WITH SIGNAL CONDITIONING INTERFACE

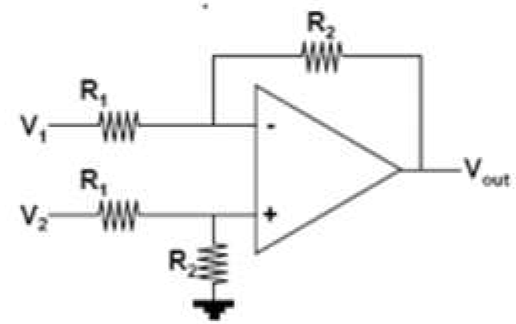
- **Summing Amp**



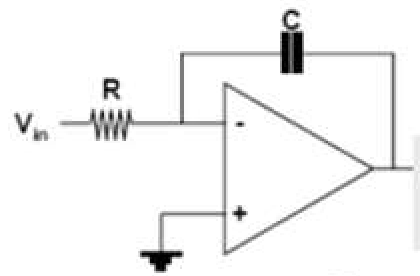
$$V_{out} = -\left(V_1\frac{R_f}{R_1} + V_2\frac{R_f}{R_2} + \cdots + V_N\frac{R_f}{R_N}\right)$$

- **Differential Amp**

$$V_{out} = \frac{R_2}{R_1}(V_2 - V_1)$$

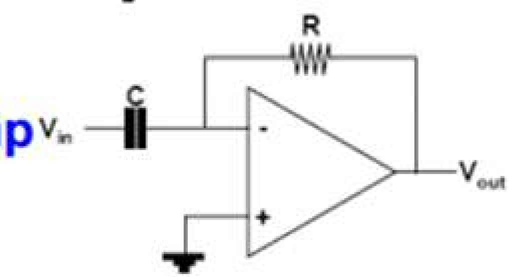


- **Integrating Amp**



$$V_{out} = -\frac{1}{j\omega CR}V_{in} = -\frac{1}{RC}\int V_{in}dt$$

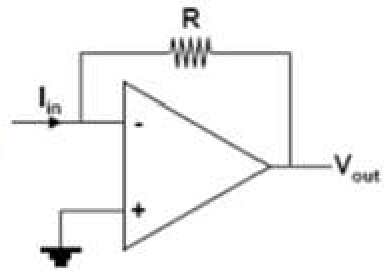
- **Differentiating Amp**



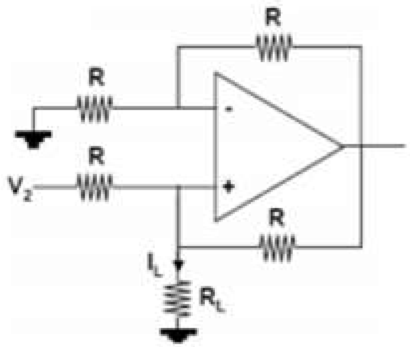
$$V_{out} = -\frac{R}{\frac{1}{j\omega C}}V_{in} = -RC\frac{dV_{in}}{dt}$$

SENSOR WITH SIGNAL CONDITIONING INTERFACE

Current-to-Voltage

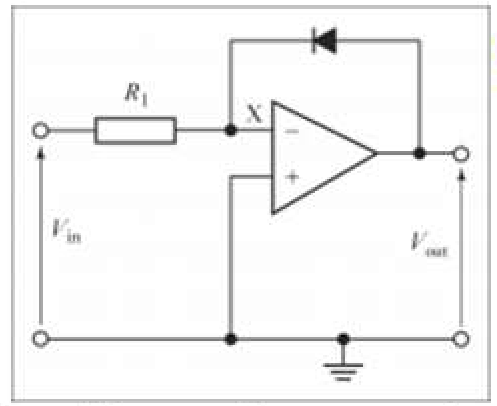


$$V_{out} = -I_{in}R$$



$$I_L = \frac{V_{in}}{R}$$

Voltage-to-Current



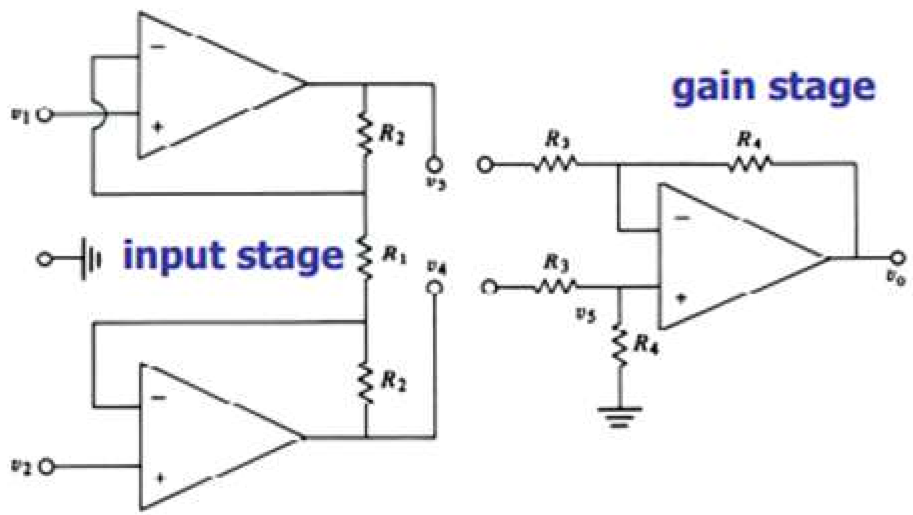
Logarithmic amplifier

$V_{out}= k \ln (V_{in})$

The voltage- current relationship can be approximated by:

$V=C\ln(I) = C\ln(V_{in}/R) = k\ln(V_{in})$; so if $V_{in}=A\exp(at)$ then
$V_{out}= K \ln(A\exp(at)) = k\ln A+at$ which is linear relationship

Instrumentation Amplifier



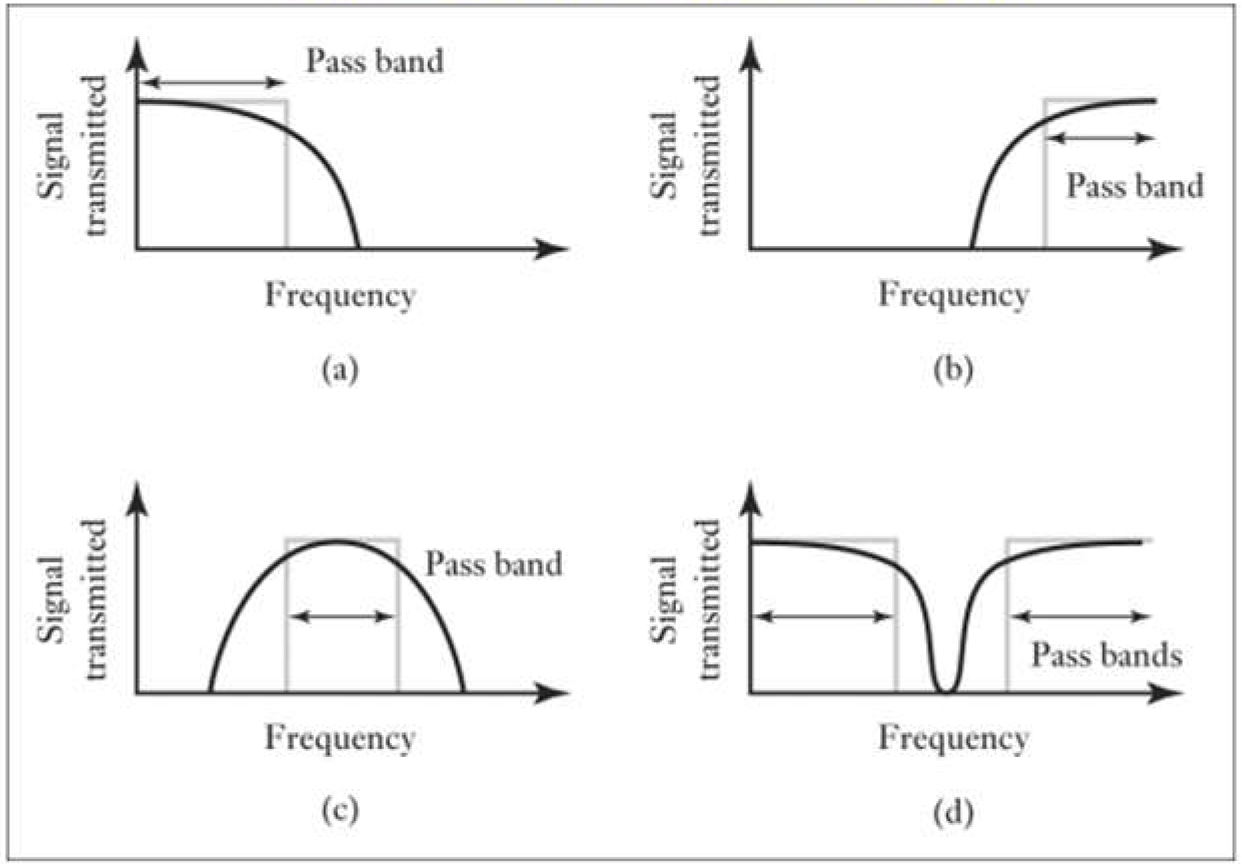
- It is available as single IC is designed to have:
  - -high input impedance (300M ohm)
  - High common mode rejection gain (more than 100 dB)
  - High voltage gain

total differential gain

$$G_d = \frac{2R_2 + R_1}{R_1}\left(\frac{R_4}{R_3}\right)$$

SENSOR WITH SIGNAL CONDITIONING INTERFACE

Signal conditioning: Filtering (2)

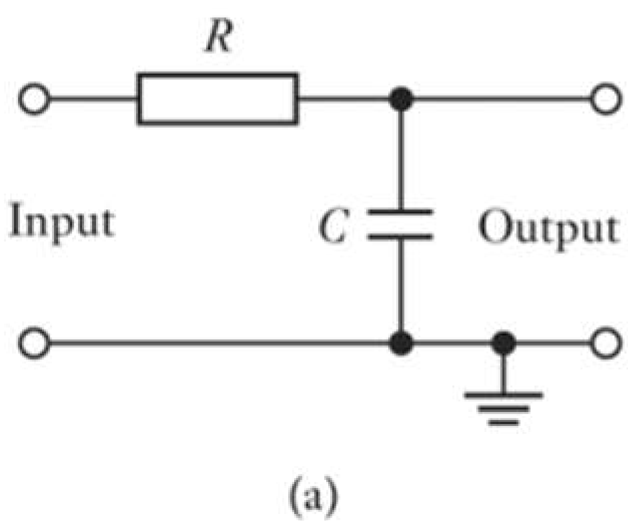


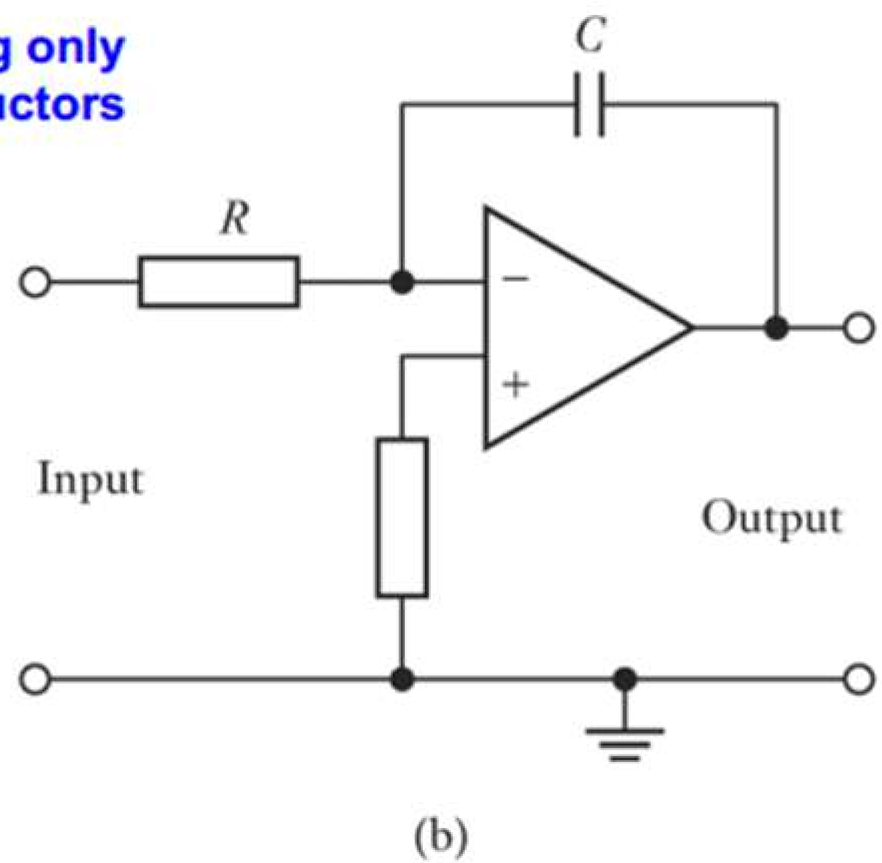
Characteristics of ideal filters: (a) low-pass filter, (b) high-pass filter, (c) band-pass filter, (d) band-stop filter

SENSOR WITH SIGNAL CONDITIONING INTERFACE

Passive Filters made up using only resistors, capacitors and inductors

Active filters involve an operational amplifier





Low-pass filter: (a) passive, (b) active using an operational amplifier

THANK YOU

by
prakash v