Module IV

Interfacing of DIP switch'- LED -7 segment displays –alphanumeric LCD – relay interface –Stepper motor –ADC-DAC-interfacing programs using assembly language.

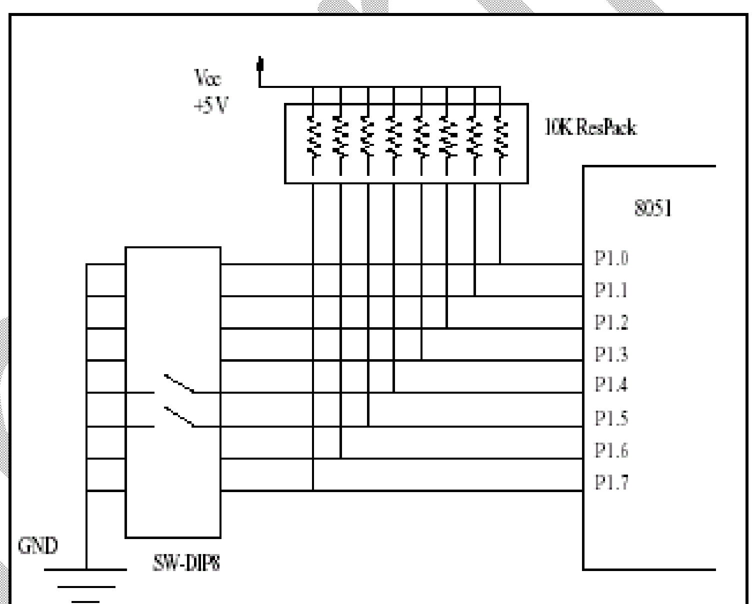
Interfacing of DIP switch

DIP Switches are manual electric switches that are packaged by group into a standard dual in-line package (DIP). This type of switch is designed to be used on a printed circuit board along with other electronic components to customize the behavior of an electronic device in specific situations. DIP switches are also known as toggle switches, which mean they have two possible positions - on or off. (Sometimes instead of on and off, you may see the numbers 1 and 0 on the DIP Switch.)

DIP switches usually have 8 switches.

MOV P1,#FFH ;make P1 as input port.

MOV A,P1 ; Move data from P1 to A.



LED Interfacing

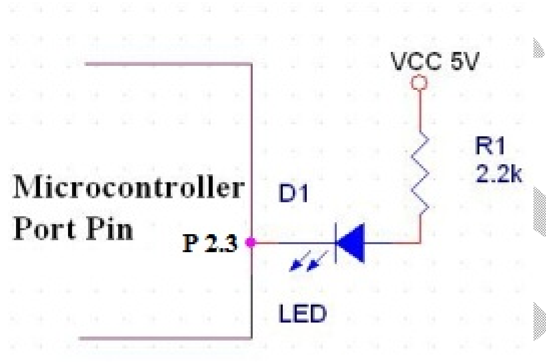
Flashing LED ALGORITHM

1. Start.
2. Turn ON LED.
3. Wait for some time (delay).
4. Turn OFF LED.
5. Wait for some time (delay).

6. Go To 2.

Program

```
        ORG 0000h
loop:
        CLR P2.0
        ACALL DELAY
        SETB P2.0
        ACALL DELAY
        SJMP loop
```



Generating delay

Loop technique

1. Start.
2. Load a number in a RAM location. e.g. R0.
3. Decrement RAM Location.
4. Is RAM = 00? If NO, GO TO 3.
5. STOP.

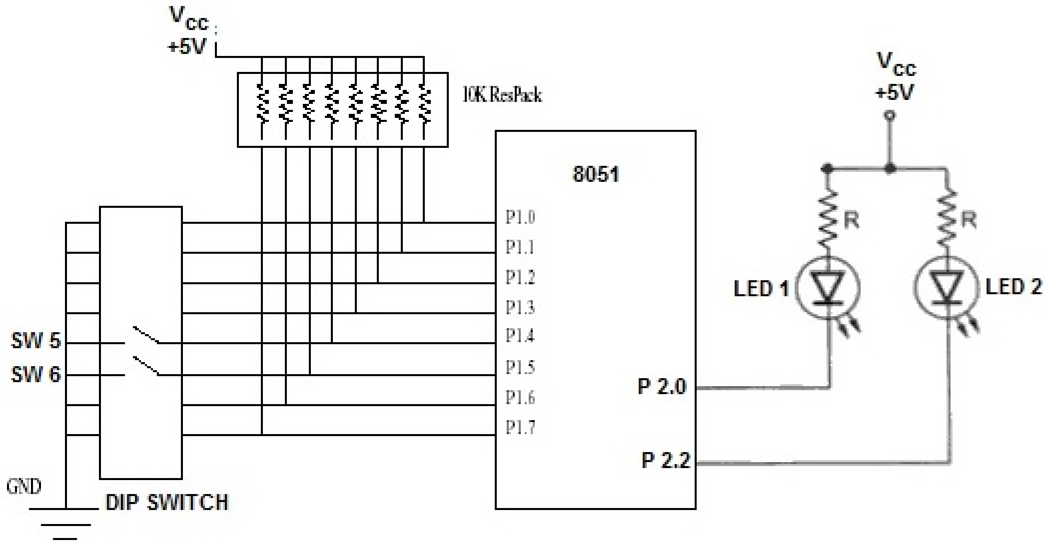
As you can see in the algorithm a number is loaded in a RAM location. It is then decremented & then if the content of the RAM location is not equal to zero a jump is made to the decrementing instruction.

In 8051 a single instruction "**DJNZ**" is specifically designed for this kind of programs. It stands for **Decrement & Jump if Not Zero**. This instruction takes care or STEP 3 & STEP 4 of the above algorithm.

Delay Program

```
Delay:
MOV R7,#100
AGAIN:DJNZ R7,AGAIN
RET
```

Interfacing DIP switch and LEDs with 8051



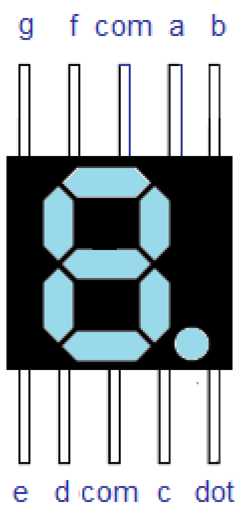
A DIP switch is connected to Port 1 and two LEDs are connected to P2.0 and P2.2 respectively. Write a program to blink the LED 1 if SW5 is ON and blink the LED2 if SW6 is ON. Note that at a time only one LED should be ON.

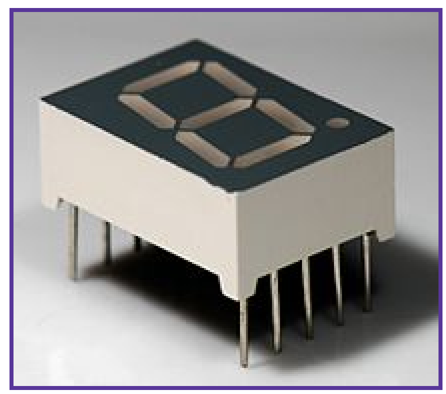
Program

```
        ORG 00H
        MOV P1,#0FFH
AGAIN 1: JB P1.4,LED1
AGAIN2:JB P1.5,LED2
LED1: CLR P2.0
        ACALL DELAY
        SETB P2.0
        ACALL DELAY
        SJMP AGAIN1
LED2: CLR P2.2
        ACALL DELAY
        SETB P2.2
        ACALL DELAY
        SJMP AGAIN2
```

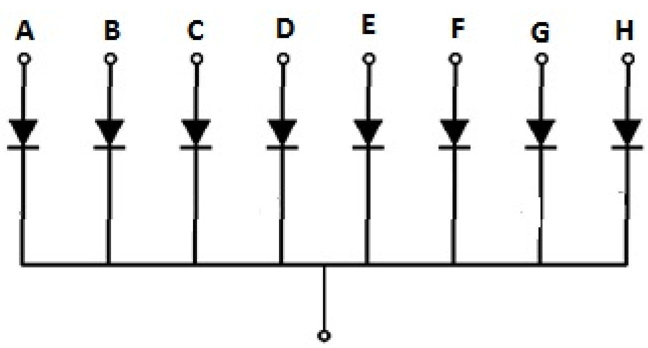
7 Segment Displays

A seven segment display consists of seven LEDs arranged in the form of a squarish **'8'** slightly inclined to the right and a single LED as the dot character. Different characters can be displayed by selectively glowing the required LED segments. Seven segment displays are of two types,***common cathode and common anode.*** In common cathode type , the cathode of all LEDs are tied together to a single terminal which is usually labeled as '**com**' and the anode of all LEDs are left alone as individual pins labeled as a, b, c, d, e, f, g & h (or dot) . In common anode type, the anode of all LEDs are tied together as a single terminal and cathodes are left alone as individual pins. The pin out scheme and picture of a typical 7 segment LED display is shown in the image below.





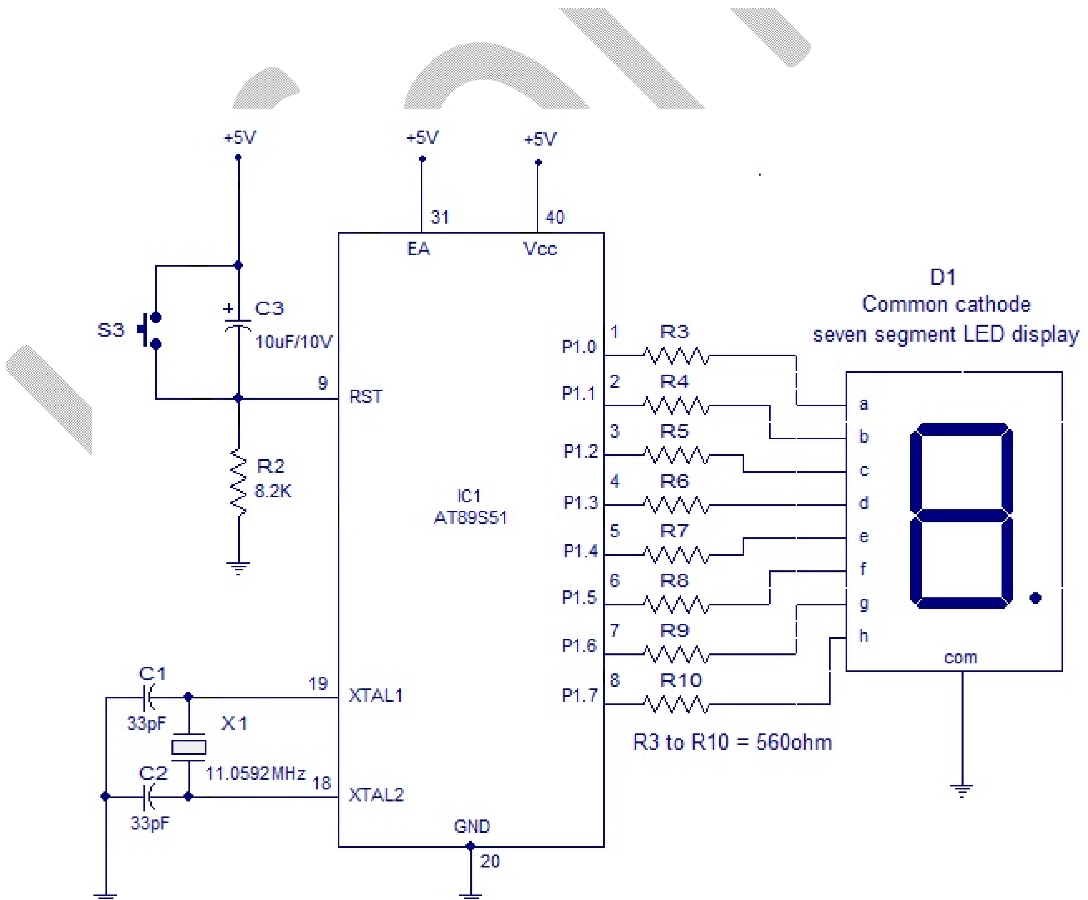
Common Cathode Type 7-Segment display



Steps for interfacing 7 segment display with 8051

1. Check if the seven segment is common anode or common cathode. If it is common anode then connect a VCC to the common anode pin. To switch on any of the respective segment/LED pass 0 to that pin through our 8051 microcontroller.
2. If the Seven Segment is common cathode connect Gnd to the common cathode pin. To switch on any of the respective segments/LED give 1 to that pin through our 8051 microcontroller.

Digit	Dp	g	f	e	d	c	b	a	Hex val
0	0	0	1	1	1	1	1	1	0x3f
1	0	0	0	0	0	1	1	0	0x06
2	0	1	0	1	1	0	1	1	0x5b
3	0	1	0	0	1	1	1	1	0x4f
4	0	1	1	0	0	1	1	0	0x66
5	0	1	1	0	1	1	0	1	0x6d
6	0	1	1	1	1	1	0	1	0x7d
7	0	0	0	0	0	1	1	1	0x07
8	0	1	1	1	1	1	1	1	0x7f
9	0	1	1	0	0	1	1	1	0x67



Interfacing 7 segment display to 8051

Program

```
        ORG 000H //initial starting address
        MOV A,#00H // set a count to point the address of each digit pattern
NEXT: MOV R0,A
        MOV DPTR,#ARRAY
        MOVC A,@A+DPTR // Read the Digit drive pattern
        MOV P1,A // Move to the port for display
        ACALL DELAY // calls the delay
        MOV A,R0 // R0
        INC A
SJMP NEXT
        SJMP $

ARRAY:DB 3FH // digit drive pattern for 0
        DB 06H // digit drive pattern for 1
        DB 5BH // digit drive pattern for 2
        DB 4FH // digit drive pattern for 3
        DB 66H // digit drive pattern for 4
        DB 6DH // digit drive pattern for 5
        DB 7DH // digit drive pattern for 6
        DB 07H // digit drive pattern for 7
        DB 7FH // digit drive pattern for 8
        DB 6FH // digit drive pattern for 9

DELAY: MOV R2,#0FFH // subroutine for delay
        WAIT: DJNZ R2,WAIT
        RET
        END
```

Program Description.

Instruction MOVC A,@A+DPTR is the instruction that produces the required digit drive pattern for the display. Execution of this instruction will add the value in the accumulator A with the content of the data pointer (starting address of the ARRAY) and will move the data present in the resultant address to A. In the program, initial value in A is 00H. Execution of MOVC A,@A+DPTR will add 00H to the content in DPTR . The result will be the address of label DB 3FH and the data

present in this address ie 3FH (digit drive pattern for 0) gets moved into the accumulator. Moving this pattern in the accumulator to Port 1 will display 0 which is the first count.
At the next count, value in A will advance to 01H and after the execution of MOVC A,@+DPTR ,the value in A will be 06H which is the digit drive pattern for 1 and this will display 1 which is the next count and this cycle gets repeated for subsequent counts.
Label DB is known as Define Byte – which defines a byte. This table defines the digit drive patterns for 7 segment display as bytes (in hex format). MOVC operator fetches the byte from this table based on the result of adding DPTR and contents in the accumulator.
Register R0 is used as a temporary storage of the initial value of the accumulator and the subsequent increments made to accumulator to fetch each digit drive pattern one by one from the ARRAY.

Alphanumeric LCD

16×2 LCD module

16×2 LCD module is a very common type of LCD module that is used in 8051 based embedded projects. It consists of 16 rows and 2 columns of 5×7 or 5×8 LCD dot matrices. The module we are talking about here is type number JHD162A which is a very popular one . It is available in a 16 pin package with back light ,contrast adjustment function and each dot matrix has 5×8 dot resolution. The pin numbers, their name and corresponding functions are shown in the table below.

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	
8	DB1	
9	DB2	

10	DB3	
11	DB4	DB0-DB7 Data pins for giving data(normal data like numbers characters or command data) which is meant to be displayed
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.

V_{EE} pin is meant for adjusting the contrast of the LCD display and the contrast can be adjusted by varying the voltage at this pin. This is done by connecting one end of a POT to the Vcc (5V), other end to the Ground and connecting the center terminal (wiper) of the POT to the VEE pin. The JHD162A has two built in registers namely data register and command register. Data register is for placing the data to be displayed , and the command register is to place the commands. The 16×2 LCD module has a set of commands each meant for doing a particular job with the display. High logic at the RS pin will select the data register and Low logic at the RS pin will select the command register. If we make the RS pin high and the put a data in the 8 bit data line (DB0 to DB7), the LCD module will recognize it as a data to be displayed . If we make RS pin low and put a data on the data line, the module will recognize it as a command.

R/W pin is meant for selecting between read and write modes. High level at this pin enables read mode and low level at this pin enables write mode.

E pin is for enabling the module. A high to low transition at this pin will enable the module.

DB0 to DB7 are the data pins. The data to be displayed and the command instructions are placed on these pins.

LED+ is the anode of the back light LED and this pin must be connected to V_{cc} through a suitable series current limiting resistor. LED- is the cathode of the back light LED and this pin must be connected to ground.

The steps involved in interfacing an LCD with 8051

1. Define function for sending command to the LCD using 8051
   a. Make RS pin 0
   b. Make EN pin 1
   c. Write the command to the data/command port of the LCD through our 8051 microcontroller.
   d. Make EN pin 0
   e. Give delay of 300ms
2. Define a function for sending Data to the LCD
   a. Make RS pin 1
   b. Make EN pin 1
   c. Write data to the data port
   d. Make EN pin 0
   e. Give delay of 300ms
3. Initialize the LCD
   a. Send a command of 0x38 to the LCD by our command function. This command enables 8 bit 2 lines 5X7 dots mode.
   b. Send a command of 0x0E to the LCD. This command is used for Display on Cursor on.
   c. Send a command of 0x01 to the LCD This command is used for Clear Display.
4. Call the Data Function to write data onto the LCD.

Scrolling Display

1. For Shifting the Entire display the steps involved are,

   a. First follow the above given steps to display the data.

   b. Take an infinite loop to scroll the data either left or right

i. For left scroll pass a command of 0x18 and then give a delay.
ii. For right scroll pass a command of 0x1C and then give a delay.

16×2 LCD module commands.

16×2 LCD module has a set of preset command instructions. Each command will make the module to do a particular task. The commonly used commands and their function are given in the table below.

Command	Function
0F	For switching on LCD, blinking the cursor.
1	Clearing the screen
2	Return home.
4	Decrement cursor
6	Increment cursor
E	Display on and also cursor on
80	Force cursor to beginning of the first line
C0	Force cursor to beginning of second line
38	Use two lines and 5x7 matrix
83	Cursor line 1 position 3
3C	Activate second line
0C3	Jump to second line position 3
0C1	Jump to second line position1

Programming LCD to 8051:

Coming to the programming, we should follow these steps:

- STEP1: Initialization of LCD.

- STEP2: Sending command to LCD.
- STEP3: Writing the data to LCD.

LCD initialization.

The steps that has to be done for initializing the LCD display is given below and these steps are common for almost all applications.

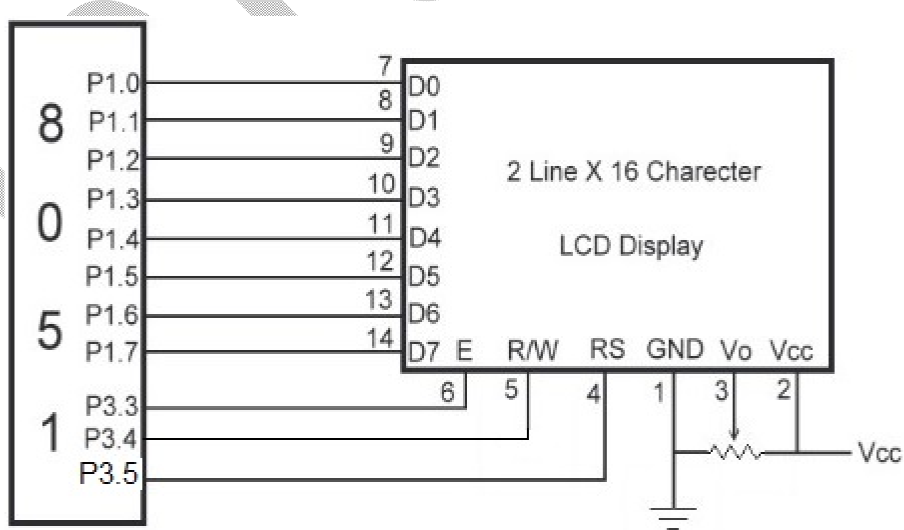
- Send 38H to the 8 bit data line for initialization
- Send 0FH for making LCD ON, cursor ON and cursor blinking ON.
- Send 06H for incrementing cursor position.
- Send 01H for clearing the display and return the cursor.

Sending data to the LCD.

The steps for sending data to the LCD module is given below. As explained, the LCD module has pins namely RS, R/W and E. It is the logic state of these pins that make the module to determine whether a given data input is a command or data to be displayed.

- Make R/W low.
- Make RS=0 if data byte is a command and make RS=1 if the data byte is a data to be displayed.
- Place data byte on the data register.
- Pulse E from high to low.
- Repeat above steps for sending another data.

Circuit diagram



The circuit diagram given above shows how to interface a 16×2 LCD module with 8051 microcontroller. P1.0 to P1.7 pins of the microcontroller is connected to the D_0 to D_7 pins of the module respectively and through this route the data goes to the LCD module. P3.3, P3.4 and P3.5 are connected to the E, R/W, RS pins of the microcontroller and through this route the control signals are transffered to the LCD module. POT is used for adjusting the contrast of the display.

Programming Example:

Write an ALP to initialize the LCD and display message “YES”. Say the command to be given is :38H (2 lines ,5x7 matrix), 0EH (LCD on, cursor on), 01H (clear LCD), 06H (shift cursor right), 86H (cursor: line 1, pos. 6)

Program:

Calls a time delay before sending next data/command ;P1.0-P1.7 are connected to LCD data pins D0-D7 ;P3.5 is connected to RS pin of LCD ;P3.4 is connected to R/W pin of LCD ;P3.3 is connected to E pin of LCD.

```
ORG 0H
MOV A,#38H          ; 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,#86H          ;cursor at line 1, pos. 6
ACALL COMNWRT  ;call command subroutine

ACALL DELAY          ;give LCD some time
MOV A,#’Y’    ;display letter Y
ACALL DATAWRT  ;call display subroutine
ACALL DELAY          ;give LCD some time
MOV A,#’E’          ;display letter E
```

```
ACALL DATAWRT   ;call display subroutine
ACALL DELAY     ;give LCD some time
MOV A,#'S'      ;display letter S
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 P3.5      ;RS=0 for command
        CLR P3.4      ;R/W=0 for write
        SETB P3.3     ;E=1 for high pulse
        ACALL DELAY       ;give LCD some time
        CLR P3.3      ;E=0 for H-to-L pulse
        RET
DATAWRT:  ;write data to LCD
        MOV P1,A      ;copy reg A to port 1
        SETB P3.5     ;RS=1 for data
        CLR P3.4      ;R/W=0 for write
        SETB P3.3     ;E=1 for high pulse
        ACALL DELAY       ;give LCD some time
        CLR P3.3      ;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
        RET
        END
```

Relay Interfacing

A relay is an electromechanical switch, which perform ON and OFF operations without any human interaction. General representation of double contact relay is shown in fig.

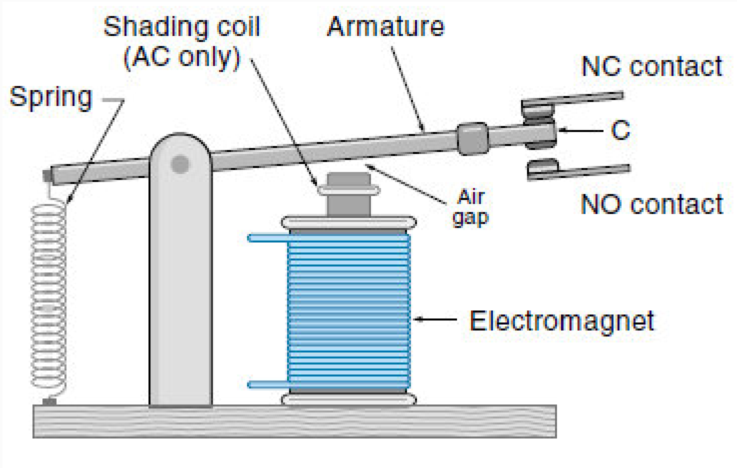


Fig: Constructional Diagram of Relay

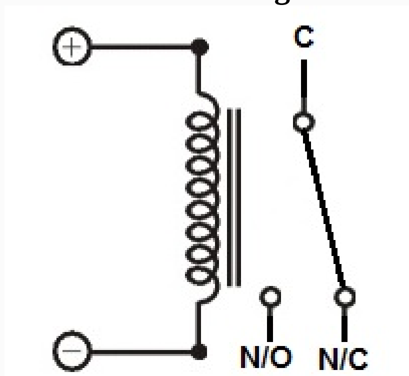


Fig: Schematic Representation of Relay

Generally, the relay consists a inductor coil, a spring, Swing terminal (armature), and two high power contacts named as normally closed (N/C) and normally opened (N/O). Relay uses an Electromagnet to move swing terminal between two contacts (N/O and N/C). When there is no power applied to the inductor coil (Relay is OFF), the spring holds the swing terminal is attached to NC contact.

Whenever required power is applied to the inductor coil, the current flowing through the coil generates a magnetic field which is helpful to move the swing terminal and attached it to the normally open (NO) contact. Again when power is OFF, the spring restores the swing terminal position to NC.

Advantage of relay:

A relay takes small power to turn ON, but it can control high power devices to switch ON and OFF. Consider an example; a relay is used to control the ceiling FAN at our home. The ceiling FAN may runs at 230V AC and draws a current maximum of 4A. Therefore the power required is 4X230 = 920 watts. Off course we can control AC, lights, etc., depend up on the relay ratings.

Applications of Relay

- Relays are used to realize logic functions. They play a very important role in providing safety critical logic.
- Relays are used to provide time delay functions. They are used to time the delay open and delay close of contacts.
- Relays are used to control high voltage circuits with the help of low voltage signals. Similarly they are used to control high current circuits with the help of low current signals.
- They are also used as protective relays. By this function all the faults during transmission and reception can be detected and isolated.

Pole and Throw

Relays have the exact working of a switch. So, the same concept is also applied. A relay is said to switch one or more poles. Each pole has contacts that can be thrown in mainly three ways. They are

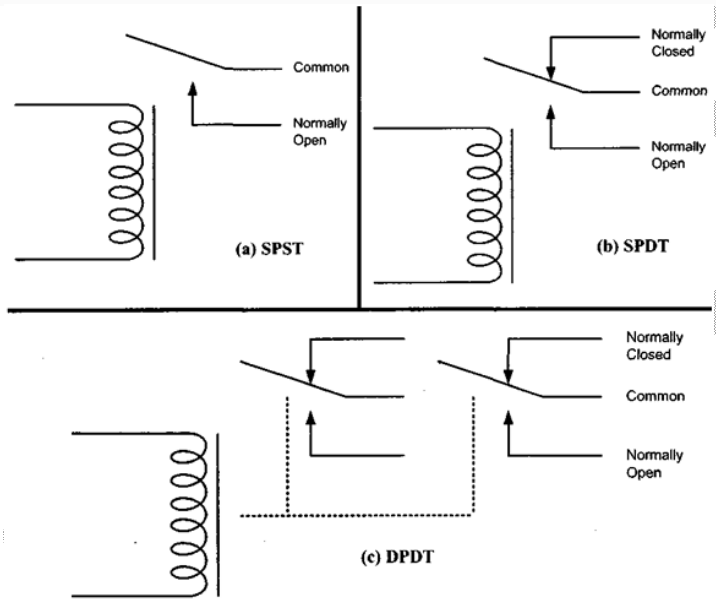
- **Normally Open Contact (NO) –** NO contact is also called a make contact. It closes the circuit when the relay is activated. It disconnects the circuit when the relay is inactive.
- **Normally Closed Contact (NC) –** NC contact is also known as break contact. This is opposite to the NO contact. When the relay is activated, the circuit disconnects. When the relay is deactivated, the circuit connects.
- **Change-over (CO) / Double-throw (DT) Contacts –** This type of contacts are used to control two types of circuits. They are used to control a NO contact and also a NC contact with a common terminal. According to their type they are called by the names **break before make** and **make before break**contacts.

Relays are also named with designations like

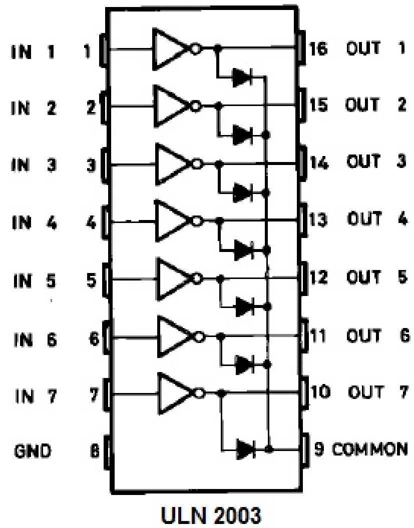
- **Single Pole Single Throw (SPST) –** This type of relay has a total of four terminals. Out of these two terminals can be connected or disconnected. The other two terminals are needed for the coil.
- **Single Pole Double Throw (SPDT) –** This type of a relay has a total of five terminals. Out f these two are the coil terminals. A common terminal is also included which connects to either of two others.

- **Double Pole Single Throw (DPST) –** This relay has a total of six terminals. These terminals are further divided into two pairs. Thus they can act as two SPST's which are actuated by a single coil. Out of the six terminals two of them are coil terminals.

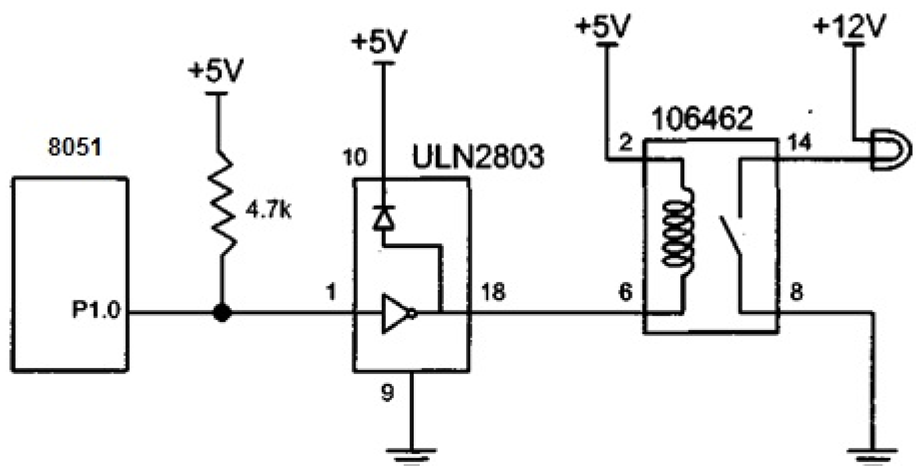
- **Double Pole Double Throw (DPDT) –** This is the biggest of all. It has mainly eight relay terminals. Out of these two rows are designed to be change over terminals. They are designed to act as two SPDT relays which are actuated by a single coil.



We need a large current sinker circuit between the relay and 8051 microcontroller. A transistor which has current sinking capability greater that 70mA is enough to act as a relay driver between the microcontrollers and relay. But the biasing circuit required for the transistor is a bit complex. In some cases like robotic car, the number of relays required will increase. Therefore the number of transistors and its biasing components will increase, PCB size increases, debugging is a bit headache. So the simple way to drive the relay with 8051 microcontroller is by using ULN2003/ULN2803. A ULN driver has 500mA current sinking capability for each output pin. When input pin of ULN driver is at logic high then the corresponding output pin is connected to ground via 500mA fuse. Internally each fuse is designed by using Darlington transistor pair. So interfacing the 8051 microcontroller to relay via ULN driver will not damage the microcontroller port pin. Microcontrollers have internal pull up resistors hence when a port pin is HIGH the output current flows through this internal pull up resistor. ***8051 microcontrollers*** have an internal pull up of 10KΩ. Hence the maximum output current will be 5v/10k = 0.5ma. This current is not sufficient to drive the transistor into saturation and turn ON the relay. Hence an external pull up resistor (4.7K) is used.



Relay Interfacing



The following program turns the lamp on and off shown in Figure. by energizing and de-energizing the relay every second.

```
        ORG  0H
MAIN:
        SETB P1.0
        MOV  R5, #55
        ACALL DELAY
        CLR  P1.0
        MOV  R5, #55
        ACALL DELAY
        SJMP MAIN
DELAY:
H1:     MOV  R4,#100
H2:     MOV  R3,#253
H3:     DJNZ R3, H3
        DJNZ R4, H2
        DJNZ R5, H1
        RET
        END
```

Stepper Motor interfacing

Stepper motor is a widely used device that translates electrical pulses into mechanical movement. Stepper motor is used in applications such as; disk drives, dot matrix printer, robotics etc,. The construction of the motor is as shown in figure 1 below.

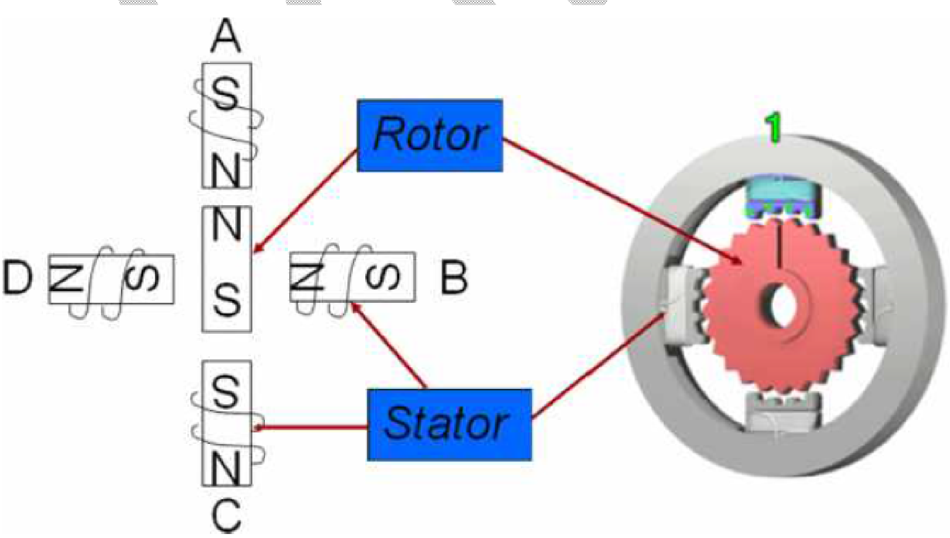


Figure 1: Structure of stepper motor

The stator is a magnet over which the electric coil is wound. One end of the coil are connected commonly either to ground or +5V. The other end is provided with a fixed sequence such that the motor rotates in a particular direction. Stepper motor shaft moves in a fixed repeatable increment, which allows one to move it to a precise position. Direction of the rotation is dictated by the stator poles. Stator poles are determined by the current sent through the wire coils.

Step angle and Steps per revolution:

How much movement is associated with a single step? This depends on the internal construction of the motor, in particular the number of teeth on the stator and the rotor. The *step angle* is the minimum degree of rotation associated with a single step. Various motors have different step angles. Table below shows some step angles for various motors. In Table, notice the term *steps per revolution.* This is the total number of steps needed to rotate one complete rotation or 360 degrees (e.g., 180 steps x 2 degrees = 360).

That means, No of steps per revolution = 360° / step angle. Example: step angle = 2° No of steps per revolution = 180

Steps per second and rpm relation

The relation between rpm (revolutions per minute), steps per revolution, and steps per second is as follows.

$$Steps\ per\ second = \frac{rpm \times Steps\ per\ revolution}{60}$$

Table : Stepper Motor Step Angles

Step Angle	Steps per Revolution
0.72	500
1.8	200
2.0	180
2.5	144
5.0	72
7.5	48
15	24

Switching Sequence of Motor:

As discussed earlier the coils need to be energized for the rotation. This can be done by sending a bits sequence to one end of the coil while the other end is commonly connected. The bit sequence sent can make either one phase ON or two phase ON for a full step sequence or it can be a combination of one and two phase ON for half step sequence. Both are tabulated below.

Full Step:

Two Phase ON

Clockwise ↓ / Counter-clockwise ↑

Step #	A	B	C	D
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1

One Phase ON

Clockwise ↓ / Counter-clockwise ↑

Step #	A	B	C	D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Half Step (8 – sequence):
The sequence is tabulated as below:

Clk ↓ / Anti-Clk ↑

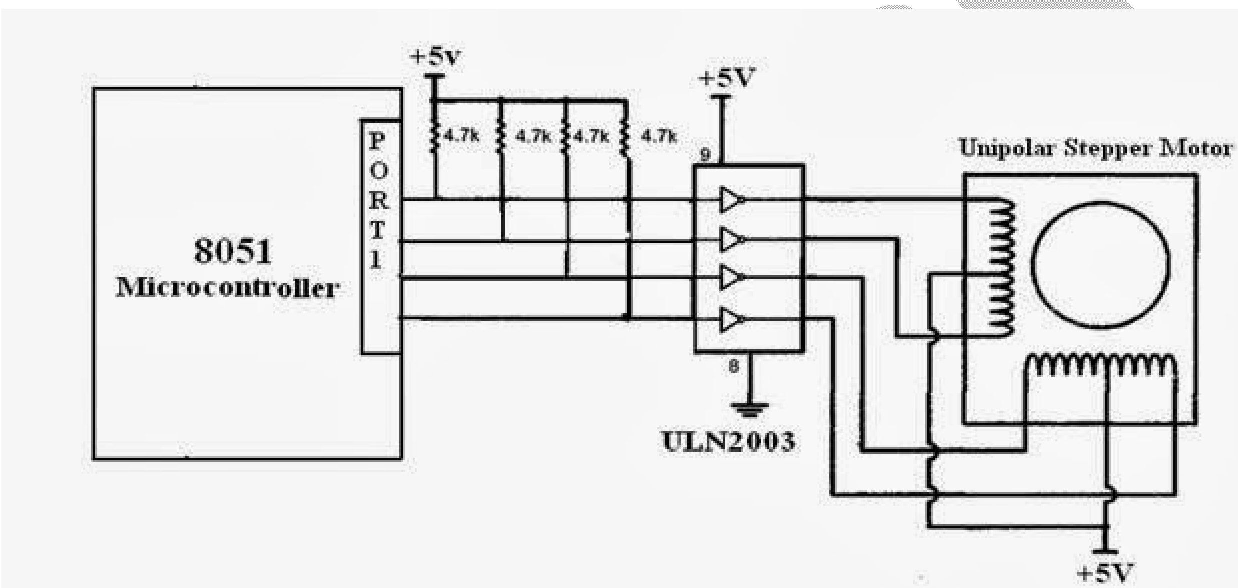
Step #	A	B	C	D
1	1	0	0	1
2	1	0	0	0
3	1	1	0	0
4	0	1	0	0
5	0	1	1	0
6	0	0	1	0
7	0	0	1	1
8	0	0	0	1

The following steps show the 8051 connection to the stepper motor and its programming.

1. Use an ohmmeter to measure the resistance of the leads. This should identify which COM leads are connected to which winding leads.
2. The common wire(s) are connected to the positive side of the motor's power supply. In many motors, +5 V is sufficient.

3. The four leads of the stator winding are controlled by four bits of the 8051 port (Pl.O - P1.3). However, since the 8051 lacks sufficient current to drive the stepper motor windings, we must use a driver such as the ULN2003 to energize the stator. Instead of the ULN2003, we could have used transistors as drivers, as shown in Figure 17-9. However, notice that if transistors are used as drivers, we must also use diodes to take care of inductive current generated when the coil is turned off. One reason that using the ULN2003 is preferable to the use of transistors as drivers is that the ULN2003 has an internal diode to take care of back EMF.

Connection Diagram of 8051 with stepper motor



Note: Change the value of DELAY to set the speed of rotation. We can use the single-bit instructions SETB and CLR instead of RR A to create the sequences.

ULN 2003

In most electronics applications it is sufficient for the controlling circuit to switch a DC output voltage or current “ON” or “OFF” directly as some output devices such as LED’s or displays only require a few milliamps to operate at low DC voltages and can therefore be driven directly by the output of a standard logic gate.

However, sometimes more power is required to operate the output device such as a DC motor than can be supplied by an ordinary logic gate or micro-controller. If the digital logic device cannot supply sufficient current then additional circuitry will be required to drive the device.

The ULN2003 is a monolithic IC consists of seven NPN darlington transistor pairs with high voltage and current capability. These useful chips allow you to drive high current loads like relays and motors which require more power than a microcontroller can supply or sink. A high voltage (5V) on an input pin will turn on the darlington pair transistor pulling down the output pin. The load goes between the output pin and the load supply voltage which can be up to 50V.

(The **Darlington transistor** (often called a **Darlington pair**) is a compound structure consisting of two bipolar transistors (either integrated or separated devices) connected in such a way that the current amplified by the first transistor is amplified further by the second one. This configuration gives a much higher common/emitter current gain than each transistor taken separately.)

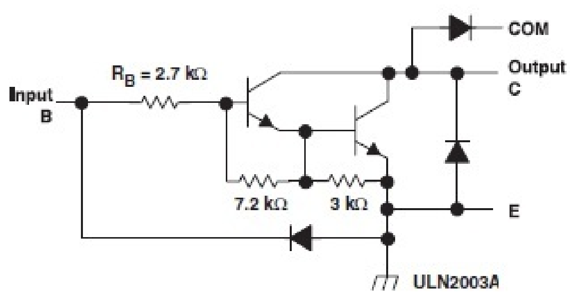
Each channel of the array is rated at 500mA and can withstand peak currents of up to 600mA making it ideal for controlling small motors or lamps or the gates and bases of high power semiconductors. Additional suppression diodes are included for inductive load driving.

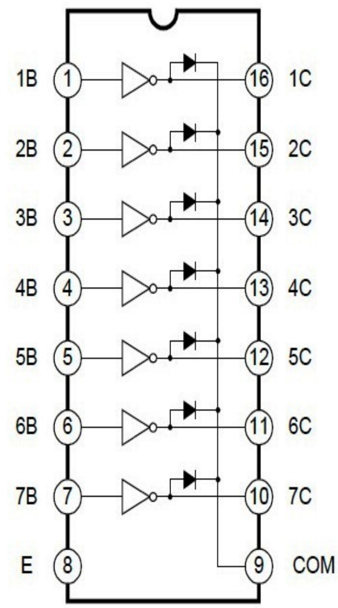
It consists of common cathode clamp diodes for each NPN darlington pair which makes this driver IC useful for switching inductive loads. (The clamp diode provides a path for the inductive discharge current to flow when the driver switch is opened. If not provided, it will generate an arc in the switch—while the arc will not generally damage a switch contact, it will cause contact degradation over time—and it will destroy transistors). The output of the driver is open collector and the collector current rating of each darlington pair is 500mA. Darlington pairs may be paralleled if higher current is required. The driver IC also consists of a 2.7KΩ base resistor for each darlington pair. Thus each darlington pair can be operated directly with TTL or 5V CMOS devices. This driver IC can be used for high voltage applications up to 50V.

Note that the driver provides open collector output, so it can only sink current, cannot source. Thus when a 5V is given to 1B terminal, 1C terminal will be connected to ground via darlington pair and the maximum current that it can handle is 500A. From the logic diagram we can see that cathode of protection diodes are shorted to 9th pin called COM.

When an input (pins 1 to 7) is driven “HIGH” the corresponding output will switch “LOW” sinking current. Likewise, when the input is driven “LOW” the corresponding output switches to a high impedance state. This high impedance “OFF” state blocks load current and reduces leakage current through the device improving efficiency.

Pin 8, (GND) is connected to the loads ground or 0 volts, while pin 9 (Vcc) connects to the loads supply. Then any load needs to be connected between +Vcc and an output pin, pins 10 to 16. For inductive loads such as motors, relays and solenoids, etc, pin 9 should always be connected to Vcc.





Programming Examples:

Example 1: Write an ALP to rotate the stepper motor clockwise / anticlockwise continuously with full step sequence.

Program:

```
        MOV A,#66H
        BACK: MOV P1,A
        RR A
        ACALL DELAY
        SJMP BACK
DELAY: MOV R1,#100
UP1: MOV R2,#50
UP: DJNZ R2,UP
        DJNZ R1,UP1
        RET
```

Note: motor to rotate in anticlockwise use instruction RL A instead of RR A

Example 2: A switch is connected to pin P2.7. Write an ALP to monitor the status of the SW. If SW = 0, motor moves clockwise and if SW = 1, motor moves anticlockwise.

Program:

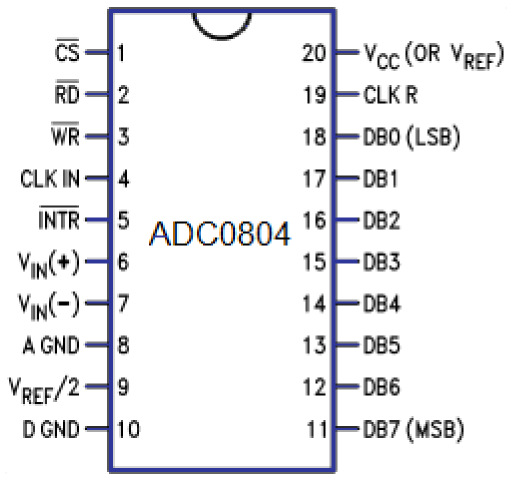
```
        ORG 0000H
        SETB P2.7
        MOV A, #66H
        MOV P1,A
TURN: JNB P2.7, CW
        RL A
        ACALL DELAY
        MOV P1,A
        SJMP TURN
CW: RR A
        ACALL DELAY
        MOV P1,A
SJMP TURN
DELAY: MOV R1,#100
UP1: MOV R2,#50
UP: DJNZ R2,UP
        DJNZ R1,UP1
        RET
```


ADC Interfacing

Analog-to-digital converters are among the most widely used devices for data acquisition. Digital computers or the microcontrollers use binary (discrete) values, but in the physical world everything is analog. Temperature, pressure, humidity and velocity are a few examples of physical quantities that we deal with everyday. A physical quantity can be converted into electrical signals using a device called as transducer. Transducers are also referred to as sensors. Most commonly sensors produce analog signals. We need an analog to digital converter to translate he analog signals to digital numbers so that the microcontroller can read and process them. An ADC has n-bit resolution where n can be 8, 10, 12, 16 or even 24 bits. The highest-resolution ADC is the one where step size is the smallest change that can be measured by an ADC.

ADC 0804

ADC0804 is an 8 bit successive approximation analogue to digital converter from National semiconductors. The features of ADC0804 are differential analogue voltage inputs, 0-5V input voltage range, no zero adjustment, built in clock generator, reference voltage can be externally adjusted to convert smaller analogue voltage span to 8 bit resolution etc. The pin out diagram of ADC0804 is shown in the figure below.



The voltage at Vref/2 (pin9) of ADC0804 can be externally adjusted to convert smaller input voltage spans to full 8 bit resolution. Vref/2 (pin9) left open means input voltage span is 0-5V and step size is 5/255=19.6V. Have a look at the table below for different Vref/2 voltages and corresponding analogue input voltage spans.

Pin Description of ADC804:

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 =150pF. We get f = 606 kHz and the conversion time is 110µs.

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

D0-D7: The digital data output pins. 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.

$$Dout = \frac{Vin}{step\ size}$$

- D_{out} = digital data output (in decimal),
- V_{in} = analog voltage, and
- step size (resolution) is the smallest change.

Analog ground and digital ground: Analog ground is connected to the ground of the analog Vin and digital ground is connected to the ground of the Vcc pin. To isolate the analog Vin signal from transient voltages caused by digital switching of the output D0 – D7. This contributes to the accuracy of the digital data output.

Vin(+) & Vin(-): Differential analog inputs where Vin = Vin (+) – Vin (-). Vin (-) is connected to ground and Vin (+) is used as the analog input to be converted.

RD: Is “output enable” a high-to-low RD pulse is used to get the 8-bit converted data out of ADC804.

INTR: It is “end of conversion” When the conversion is finished, it goes low to signal the CPU that the converted data is ready to be picked up.

WR: It is “start conversion” When WR makes a low-to-high transition, ADC804 starts

converting the analog input value of Vin to an 8- bit digital number.

CS: It is an active low input used to activate ADC804.

Pin No	Pin Functions	Pin Name
1	Make this pin low to active ADC	Chip select (CS)
2	Input pin: High to low pulse brings the data from internal registers to the output pins after conversion	Read (RD)
3	Input pin: Low to high pulse is given to start the conversion	Write (WR)
4	Clock Input pin: connect external clock to this pin.	Clock IN (CLK IN)
5	Output pin: Goes low when conversion is complete	Interrupt (INTR)
6	Analog non-inverting input	Vin(+)
7	Analog inverting Input; normally ground for DC	Vin(-)
8	Ground	Analog Ground (AGND)
9	Input pin: sets the reference voltage for analog input	Vref/2
10	Ground	Digital Ground (DGND)
11	D7 (Output 8 bit Pin)	D7
12	D6	D6
13	D5	D5
14	D4	D4
15	D3	D3
16	D2	D2
17	D1	D1
18	D0	D0
19	Used with Clock IN pin when internal clock source is used	Clock R (CLK R)
20	+ 5V	Vcc

The following steps must be followed for data conversion by the ADC804 chip:

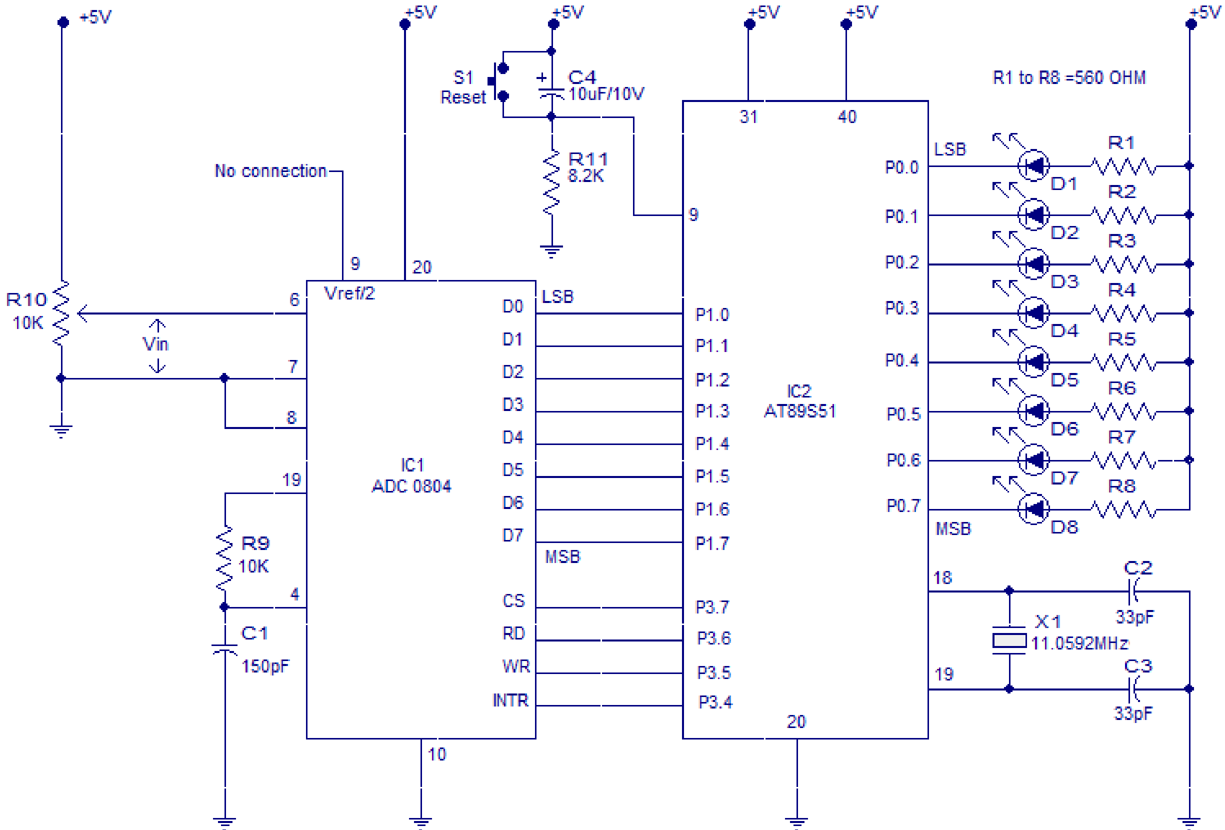
1. Make CS= 0 and send a L-to-H pulse to pin WR to start conversion.

2. Monitor the INTR pin, if high keep polling but if low, conversion is complete, go to next step.
3. Make CS= 0 and send a H-to-L pulse to pin RD to get the data out.

Table: $V_{ref}/2$ relation to V_{in} Range (ADC 0804)

Vref/2 (pin9) (volts)	Input voltage span (volts)	Step size (mV)
Left open	0 – 5	5/255 = 19.6
2	0 – 4	4/255 = 15.69
1.5	0 – 3	3/255 = 11.76
1.28	0 – 2.56	2.56/255 = 10.04
1.0	0 – 2	2/255 = 7.84
0.5	0 – 1	1/255 = 3.92

Circuit Diagram



The figure above shows the schematic for interfacing ADC0804 to 8051. The circuit initiates the ADC to convert a given analogue input , then accepts the corresponding digital data and displays it on the LED array connected at P0. For example, if the analogue input voltage Vin is 5V then all LEDs will glow indicating 11111111 in binary which is the equivalent of 255 in decimal. AT89s51 is the microcontroller used here. Data out pins (D0 to D7) of the ADC0804 are connected to the port pins P1.0 to P1.7 respectively. LEDs D1 to D8 are connected to the port pins P0.0 to P0.7 respectively. Resistors R1 to R8 are current limiting resistors. In simple words P1 of the microcontroller is the input port and P0 is the output port. Control signals for the ADC (INTR, WR, RD and CS) are available at port pins P3.4 to P3.7 respectively. Resistor R9 and capacitor C1 are associated with the internal clock circuitry of the ADC. Preset resistor R10 forms a voltage divider which can be used to apply a particular input analogue voltage to the ADC. Push button S1, resistor R11 and capacitor C4 forms a debouncing reset mechanism. Crystal X1 and capacitors C2,C3 are associated with the clock circuitry of the microcontroller.

Program.

```
ORG 00H
MOV P1,#0FFH // initiates P1 as the input port
SETB P3.4
MAIN: CLR P3.7 // makes CS=0
    SETB P3.6 // makes RD high
    CLR P3.5 // makes WR low
    SETB P3.5 // low to high pulse to WR for starting conversion
WAIT: JB P3.4,WAIT // polls until INTR=0
    CLR P3.7 // ensures CS=0
    CLR P3.6 // high to low pulse to RD for reading the data from ADC
    MOV A,P1 // moves the digital data to accumulator
    CPL A // complements the digital data (*see the notes)
    MOV P0,A // outputs the data to P0 for the LEDs
    SJMP MAIN // jumps back to the MAIN program
    END
```

Notes.

- The entire circuit can be powered from 5V DC.
- ADC 0804 has active low outputs and the instruction CPL A complements it t0 have a straight forward display. For example, if input is 5V then the output will be 11111111 and if CPL A was not used it would have been 00000000 which is rather awkward to see.

Interfacing ADC0808/0809 with 8051 controller

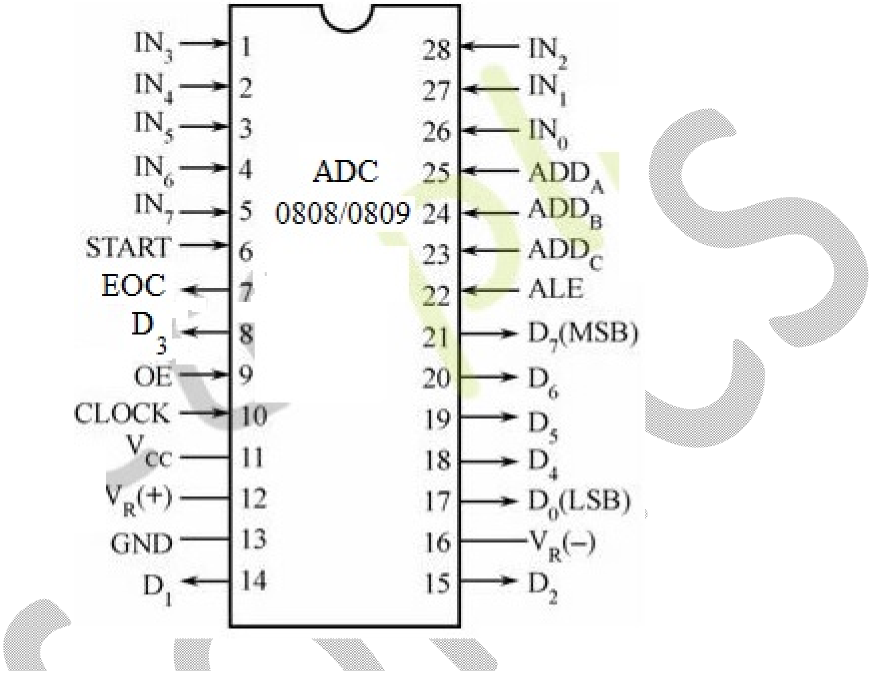
ADC0808/0809 chip:

ADC808 has 8 analog inputs. It allows us to monitor up to 8 different transducers using only single chip. The chip has 8-bit data output just like the ADC804. The 8 analog input channels are multiplexed and selected according to the values given to the three address pins, A, B, and C. that is; if CBA=000, CH0 is selected; CBA=011, CH3 is selected and so on. The pin details of ADC0808 are as shown in the figure 11 below. (Explanation can be done as is with ADC0804).

Features of ADC 0808:

1. Inbuilt 8 analog channels with multiplexer
2. Zero or full scale adjustment is not required
3. 0 to 5V input voltage range with a single polarity 5V supply
4. Output is TTL compatible
5. High speed

6. Low conversion time (100micro second)
7. High accuracy
8. 8-bit resolution
9. Low power consumption (less than 15mW)
10. Compatible with microcontroller



Pin Description

ADDRESS LINE A, B, C - The device contains 8-channels. A particular channel is selected by using the address decoder line. The TABLE 1 shows the input states for address lines to select any channel.

Address Latch Enable (ALE) - The address is latched on the Low – High transition of ALE.

START - The ADC's Successive Approximation Register (SAR) is reset on the positive edge i.e. Low- High of the Start Conversion pulse. Whereas the conversion is begun on the falling edge i.e. High – Low of the pulse.

Output Enable (OE) - Whenever data has to be read from the ADC, Output Enable pin has to be pulled high thus enabling the TRI-STATE outputs, allowing data to be read from the data pins D0-D7.

End of Conversion (EOC) - This pin becomes High when the conversion has ended, so the controller comes to know that the data can now be read from the data pins.

Clock - External clock pulses are to be given to the ADC; this can be given either from LM 555 in astable mode or the controller can also be used to give the pulses.

D0 – D7 – D0- D7 are the digital data output pins since ADC0808 is a parallel ADC chip. These are tri state buffered and the converted data is accessed only when CS=0 and RD is forced low. To calculate the output voltage, we can use following formula

$D_{out} = V_{in}$ / step size

Where D_{out} = digital data output, V_{in} = analog input voltage, and step size is the smallest input change.

MULTIPLEXER

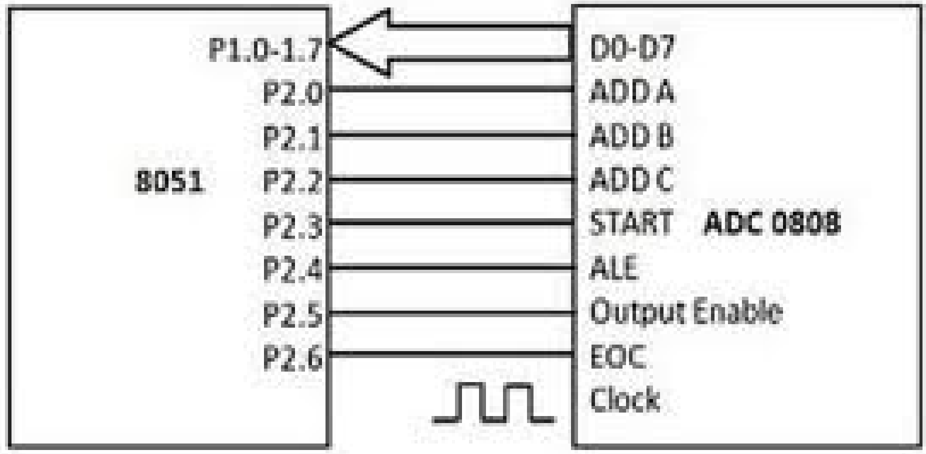
The device contains an 8-channel single-ended analog signal multiplexer. A particular input channel is selected by using the address decoder. Table 1 shows the input states for the address lines to select any channel. The address is latched into the decoder on the low-to-high transition of the address latch enable signal.

Table : Analog Channel Selection

SELECTED ANALOG CHANNEL	ADDRESS LINE		
	C	B	A
IN0	L	L	L
IN1	L	L	H
IN2	L	H	L
IN3	L	H	H
IN4	H	L	L
IN5	H	L	H
IN6	H	H	L
IN7	H	H	H

Steps to Program ADC0808/0809

1. Select an analog channel by providing bits to A, B, and C addresses.
2. Activate the ALE pin. It needs an L-to-H pulse to latch in the address.
3. Activate SC (start conversion) by an L-to-H pulse to initiate conversion.
4. Monitor EOC (end of conversion) to see whether conversion is finished. H –to-L indicates that the data is converted.
5. Activate OE (output enable) to read data out of the ADC chip. An L-to-H pulse to the OE pin will bring digital data out of the chip.



Programming ADC0808/0809 in assembly

```
        ORG 0000H
        MOV P1, #0FFH
        SETB P2.6 //EOC=1
        CLR P2.4  //ALE=00
        CLR P2.3 //START=0
        CLR P2.5 //OE=0
BACK: CLR P2.0  //A=0
        CLR P2.1  //B=0
        SETB P2.2  //C=1
        ACALL DELAY
        SETB P2.4
        ACALL DELAY
        SETB P2.3
        ACALL DELAY
        CLR P2.4
        CLR P2.3
HERE: JB P2.6, HERE
        SETB P2.5
        ACALL DELAY
```

```
MOV A, P1
CLR P2.5
ACALL CONVERSION
ACALL DATA DISPLAY
SJMP BACK
```

DAC interfacing with 8051

Digital-to-analog (DAC) converter

The digital-to-analog converter (DAC) is a device widely used to convert digital pulses to analog signals. In this section we discuss the basics of interfacing a DAC to the 8051. There are two methods of creating a DAC: binary weighted and R/2R ladder. The vast majority of integrated circuit DACs, including the MC1408 (DAC0808) used in this section, use the R/2R method since it can achieve a much higher degree of precision. The first criterion for judging a DAC is its resolution, which is a function of the number of binary inputs. The common ones are 8, 10, and 12 bits. The number of data bit inputs decides the resolution of the DAC since the number of analog output levels is equal to 2″, where *n* is the number of data bit inputs. Therefore, an 8-input DAC such as the DAC0808 provides 256 discrete voltage (or current) levels of output.

MC1408 DAC (or DAC0808)

In the MC1408 (DAC0808), the digital inputs are converted to current (I_{out}), and by connecting a resistor to the I_{out} pin, we convert the result to voltage.

The total current provided by the I_{out} pin is a function of the binary numbers at the DO – D7 inputs of the DAC0808 and the reference current (I_{re}f), and is as follows:

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

where D_O is the LSB, D_7 is the MSB for the inputs, and I_{re}f is the input current that must be applied to pin 14. The I_{re}f current is generally set to 2.0 mA. Figure shows the generation of current reference (setting I_{re}f = 2 mA) by using the standard 5-V power supply and IK and 1.5K-ohm standard resistors. Some DACs also use the zener diode (LM336), which overcomes any fluctuation associated.

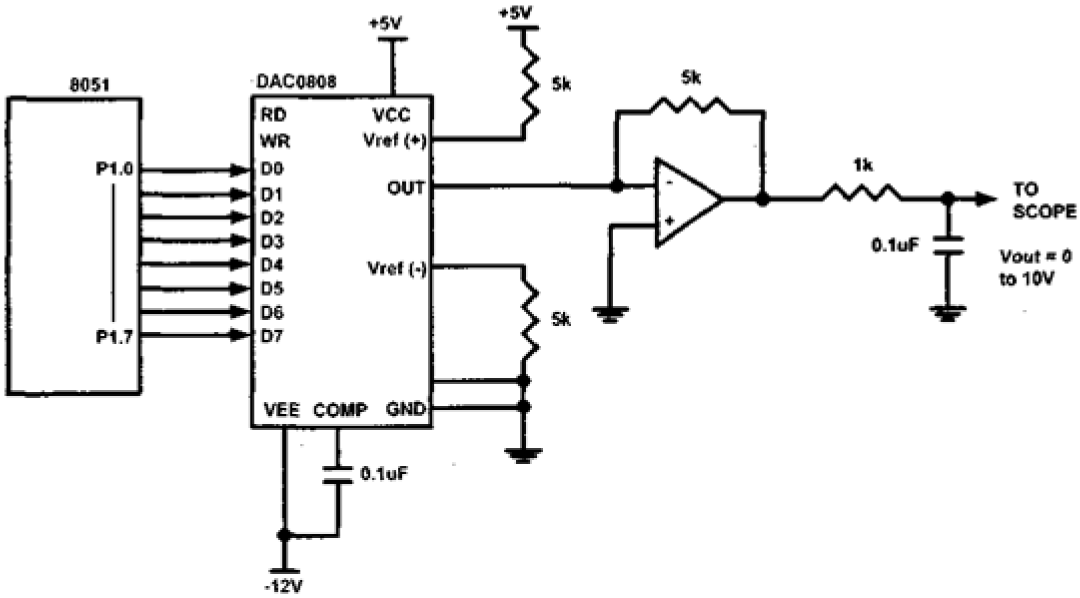


Figure. 8051 Connection to DAC808

Converting I_{out} to voltage in DAC0808

Ideally we connect the output pin I_{out} to a resistor, convert this current to voltage, and monitor the output on the scope. In real life, however, this can cause inaccuracy since the input resistance of the load where it is connected will also affect the output voltage. For this reason, the I_{ref} current output is isolated by connecting it to an op-amp such as the 741 with R_f = 5K ohms for the feedback resistor. Assuming that R = 5K ohms, by changing the binary input, the output voltage changes as shown in following example.

ProgrammingExample

In order to generate a stair-step ramp, set up the circuit in Figure 13-18 and connect the output to an oscilloscope. Then write a program to send data to the DAC to generate a stair-step ramp.

```
        CLR   A
AGAIN:  MOV   P1,A          ;send data to DAC
        INC   A             ;count from 0 to FFH
        ACALL DELAY         ;let DAC recover
        SJMP  AGAIN
```

Generating a sine wave

To generate a sine wave, we first need a table whose values represent the magnitude of the sine of angles between 0 and 360 degrees. The values for the sine function vary from -1.0 to +1.0 for 0- to 360-degree angles. Therefore, the table values are integer numbers representing the voltage magnitude for the sine of theta. This method ensures that only integer numbers are output to the DAC by the 8051 microcontroller. Table 13-7 shows the angles, the sine values, the voltage magnitudes, and the integer values representing the voltage magnitude for each angle (with 30-degree increments). To generate Table 13-7, we assumed the full-scale voltage of 10 V for DAC output (as designed in Figure 13-18). Full-scale output of the DAC is achieved when all the data inputs of the DAC are high. Therefore, to achieve the full-scale 10 V output, we use the following equation.

$$V_{out} = 5\text{ V} + (5 \times \sin\theta)$$

V_{out} of DAC for various angles is calculated and shown in the following Table.

Table : Angle vs. Voltage Magnitude for Sine Wave

Angle 9 (degrees) Sin 0	V_{out} (Voltage Magnitude) Values Sent to DAC (decimal) 5 V + (5 V X sin 6)	(Voltage Mag. X 25.6)
0 0	5	128
30 0.5	7.5	192
60 0.866	9.33	238
90 1.0	10	255
120 0.866	9.33	238
150 0.5	7.5	192
180 0	5	128
210 -0.5	2.5	64
240 -0.866	0.669	17
270 -1.0	0	0
300 -0.866	0.669	17
330 -0.5	2.5	64
360 0	5	128

Verify the values given for the following angles: (a) 30° (b) 60°.

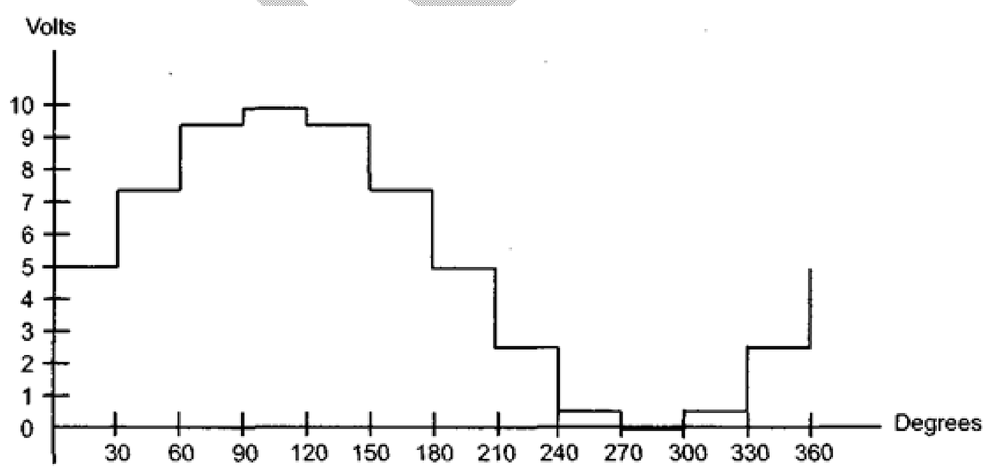
Solution:

(a) $V_{out} = 5\text{ V} + (5\text{ V} \times \sin\theta) = 5\text{ V} + 5 \times \sin 30° = 5\text{ V} + 5 \times 0.5 = 7.5\text{ V}$
DAC input values = $7.5\text{ V} \times 25.6 = 192$ (decimal)

(b) $V_{out} = 5\text{ V} + (5\text{ V} \times \sin\theta) = 5\text{ V} + 5 \times \sin 60° = 5\text{ V} + 5 \times 0.866 = 9.33\text{ V}$
DAC input values = $9.33\text{ V} \times 25.6 = 238$ (decimal)

To find the value sent to the DAC for various angles, we simply multiply the V_{out} voltage by 25.60 because there are 256 steps and full-scale V_{out} is 10 volts. Therefore, 256 steps /10 V = 25.6 steps per volt. To further clarify this, look at the following code. This program sends the values to the DAC continuously (in an infinite loop) to produce a crude sine wave. See Figure shown below.

```
AGAIN:      MOV  DPTR,#TABLE
            MOV  R2,#COUNT
BACK:       CLR  A
            MOVC A,@A+DPTR
            MOV  P1,A
            INC  DPTR
            DJNZ R2,BACK
            SJMP AGAIN
            ORG  300
TABLE:      DB  128,192,238,255,238,192  ;see Table 13-7
            DB  128,64,17,0,17,64,128
```



Example : Write an ALP to generate a triangular waveform.

Program:

MOV A, #00H

INCR: MOV P1,

A INC A

CJNE A, #255, INCR

DECR: MOV P1, A

DEC A

CJNE A, #00, DECR

SJMP INCR

END