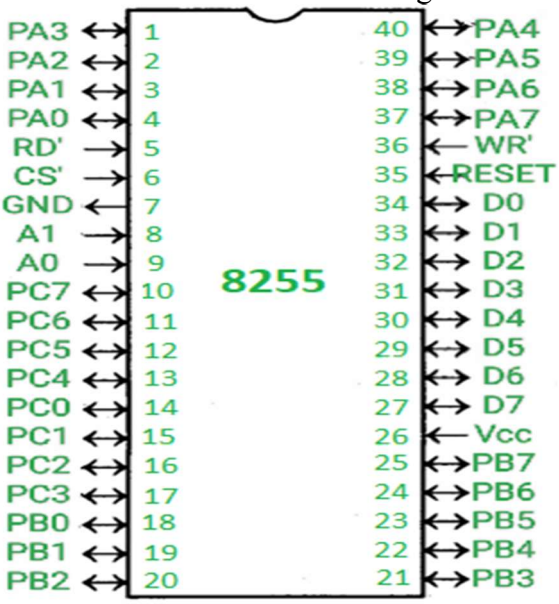
UNIT - V
INTERFACING TECHNIQUES

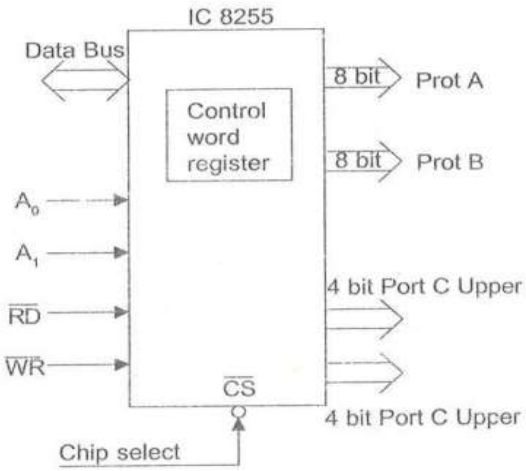
IC 8255 (Programmable Peripheral interface)

8255 Pin details and signal diagram

The Intel 8255 is one such peripheral interface chip. It can be used with the Intel microprocessor or microcontroller. The 8255 is one of the most widely used interface devices for expanding number of input/ output pins.

It is a 40 pin DIP IC. It requires +5V DC power Supply for its operations . The pin out diagram and signal diagram of 8255 A are shown in the fig

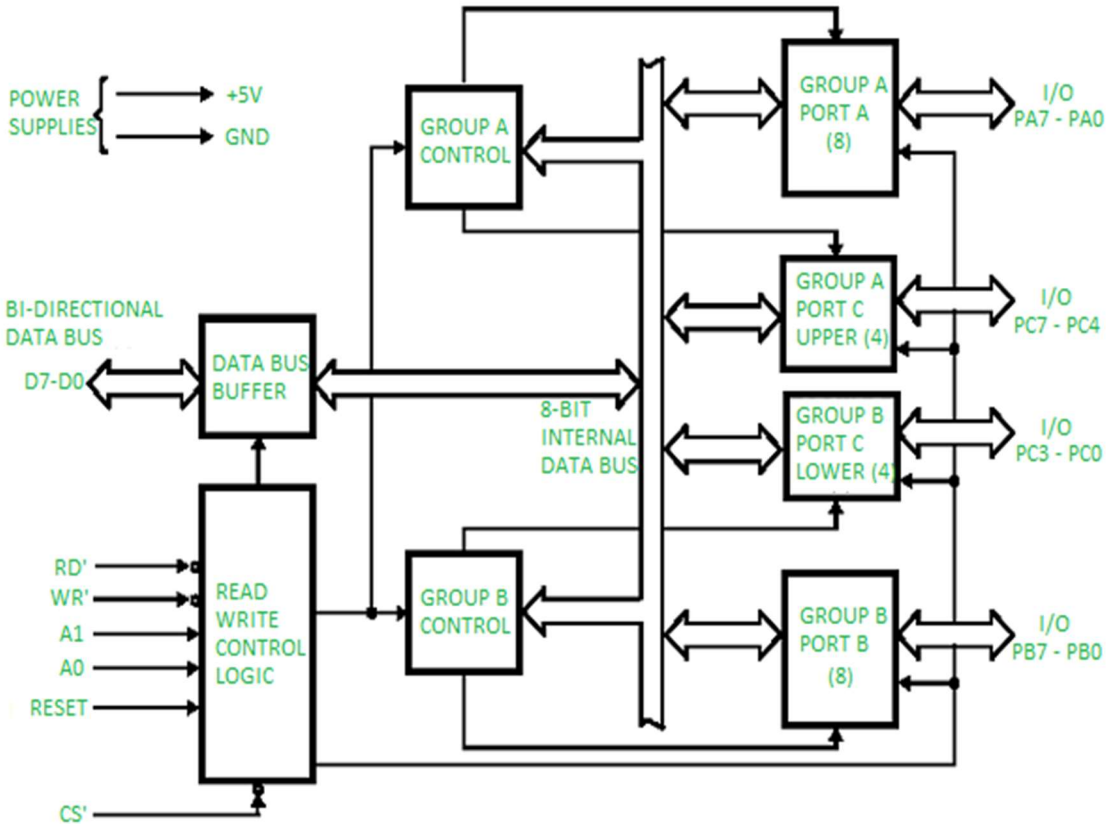




$D_7 - D_0$	Data Bus (Bidirectional)
Reset	Reset Input
CS	Chip select
RD	Read input
WR	Write input
$A_0 - A_1$	Port adddress
$PA_7 - PA_0$	Port A (8 bit)
$PB_7 - PB_0$	Port B (8 bit)
$PC_7 - PC_0$	Port C (8 bit)
V_{CC}	+5 Volts
GND	0 Volts

FUNCTIONAL BLOCK DIAGRAM of 8255

The functional block diagram of 8255 is shown in fig. It contains Data bus buffer, Read/ Write control Logic Unit and I/O Ports.



Input/ Output Ports

The 8255 consists of three numbers of ports namely A,B and C. Each one of them is an 8 bit port. However port C can be divided into two 4 bit ports (port C upper, port C Lower) and used separately.

For the convenience of Programming the ports are grouped as Group A and Group B

Group A = Port A ($PA_7 – PA_0$)
Port C Upper ($PC_7 – PC_4$)
Group B = Port B ($PB_7 – PB_0$)
Port C Lower ($PC_3 – PC_0$)

Data bus Buffer:

It is a bidirectional 8 bit buffer. It is used to interface the 8255 with the system data bus. The data, control world and status information signal in between the microcontroller and 8255 are communicated only through data bus buffer.

Control Logic:

$\overline{RD}$ (Read): When this signal in low, the microcontroller reads data from the selected I/O ports of 8255

$\overline{WR}$ ($\overline{Write}$): When this signal is low, the microcontroller writes data into the selected I/O port or the control register.

Reset: Reset the all: all ports in the input mode. It is an active high signal.

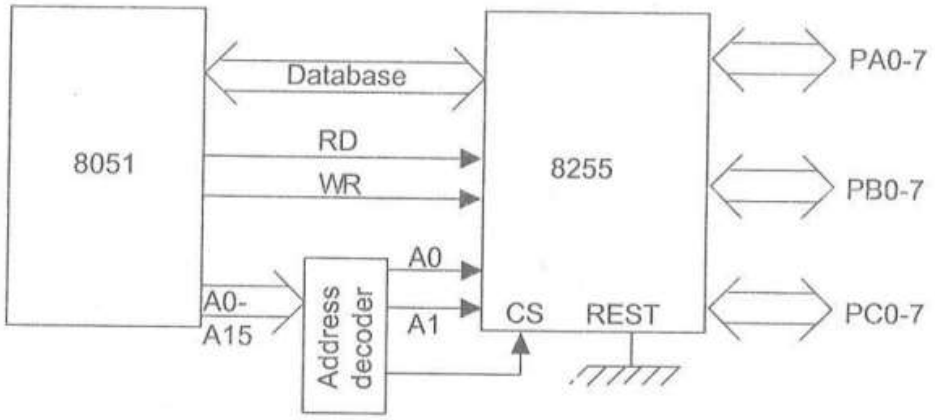
$\overline{CS}$ (Chip select): It enables the Communication between the8255 and microcontroller.

A1-A0: These lines are used to address the three ports and control word Register (CWR)

Port Selection Table

A1	A0	Selected
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Control Word Register

Interfacing 8255 with 8051



Control Word Register:

It is a 8 bit Register. An 8 bit binary word present in the control register is called control word. The control word specifies the function for each I/O Port.

Modes of8255

The operation of the ports can be classified into two broad groups.

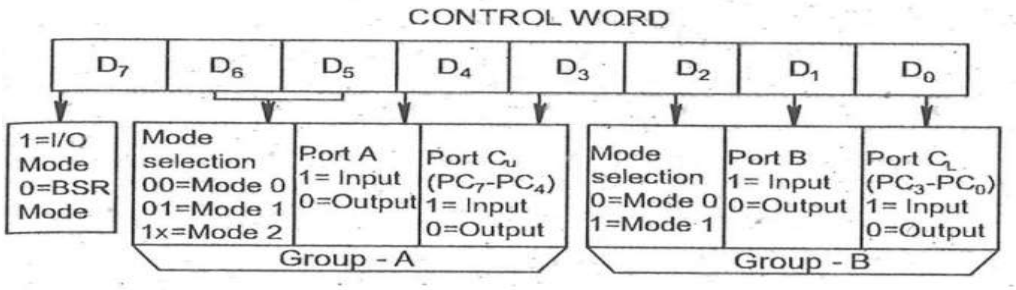
- I/O Mode
- Bit set/ reset mode (BSR)

I/O Mode

I/O mode offers three specific Modes of Operation. There are

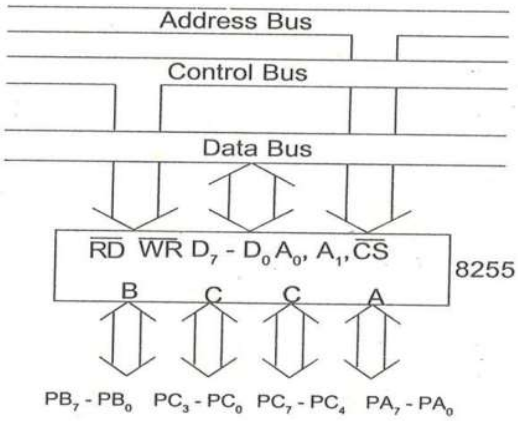
- Mode 0 - Simple I/O Mode
- Mode 1 - Handshake Mode
- Mode 2 - Bidirectional Mode

Control world format for I/O Mode



Mode O – Simple I/O mode

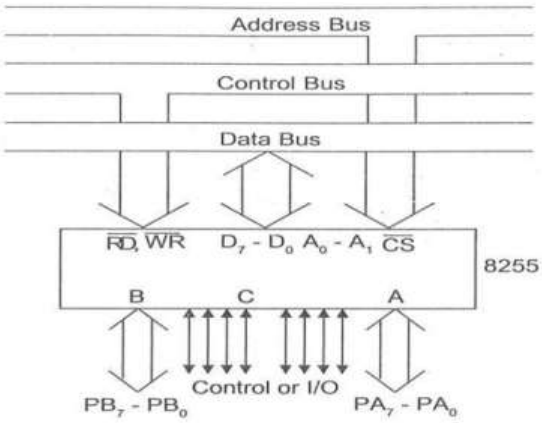
- In this mode all the ports can be programmed either as input or output port.
- Port C can be divided into two 4 bit ports, the C Lower and C Upper each of them can be set independently for input or output operation.



Mode 1: Hand shake Mode

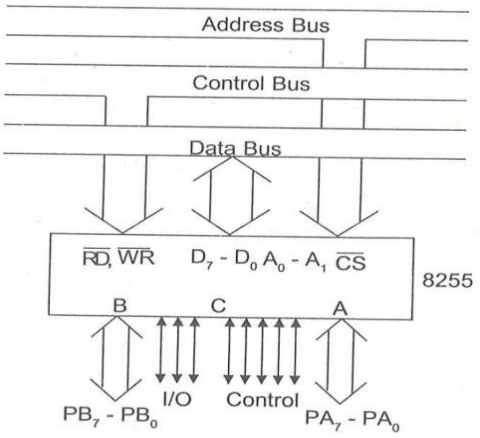
Shown in the fig Mode 1: Hand shake Mode

- In this mode 1 the port A and port B can be set for input or output Operation.
- The Port C are used as control Signals. These Control Signals are used for handshaking.



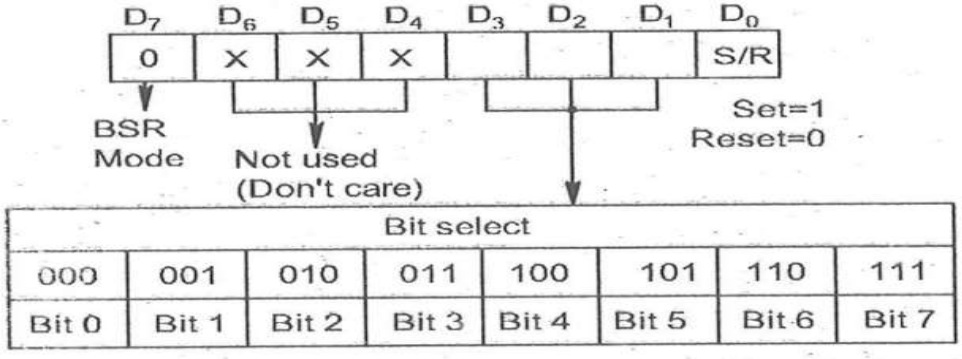
Mode 2 - Bidirectional Mode

- In this mode port A only act as a bidirectional data transfer Port.
- Poet B may be operated as in mode 0 or Mode1
- Port C- 5 bits are used control signals for port A
- Port C – Remaining 3 bits are Used in mode 0 or as handshake signals for port B
- In this mode data transfer is done in both directions between microcontroller and peripheral devices. Shown in the fig mode 2 – bidirectional mode



Bit Set/ Reset Mode (BSR Mode)

- This mode is related to only port C. The bits of port C can be controlled directly by the microcontroller. A control world with bit D7 = 0 in called a BSR Control word.
- Control word format for BSR mode as shown in the fig



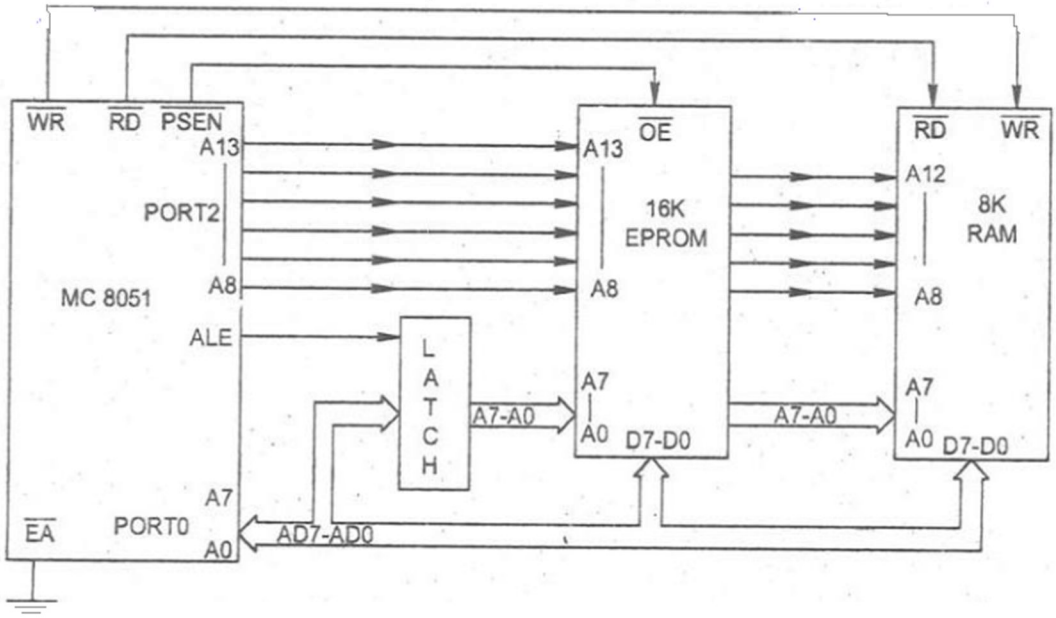
INTERFACINGTECHNIQUES

Designing logic circuits and writing instructions to enable the microcontroller to communicate with peripherals (I/O devices- input devices – keyboard, switches, and A/D converters. Output devices – seven segment LED display, D/A converters, Printers and video monitors) is called interfacing.

INTERFACING EXTERNAL MEMORY TO 8051

- The 8051 has 256 bytes of internal data memory (RAM) for data storage. 4Kbytes of internal ROM is also available to store the Program.
- If the internal Memory is not sufficient for storage, it can able to connect the external memory with8051.
- We can able to connect 64 K bytes of data memory (RAM) and 64Kbytes of Program memory (ROM) externally.

  The interfacing diagram of 16 K bytes of EPROM and 8Kbytes of RAM with Microcontroller 8051 in shown in fig External memory connections with 8051.



When ALE in enabled the port 0 Latch the address signal. Otherwise port 0 has a data.

- If the memory access is from external ROM, the $\overline{PSEN}$ pin will go low
- If the memory access is for a RAM byte
- $\overline{WR}$ pin Low - When data flow from data bus to RAM
- $\overline{RD}$ pin Low - When data flow from RAM to data bus

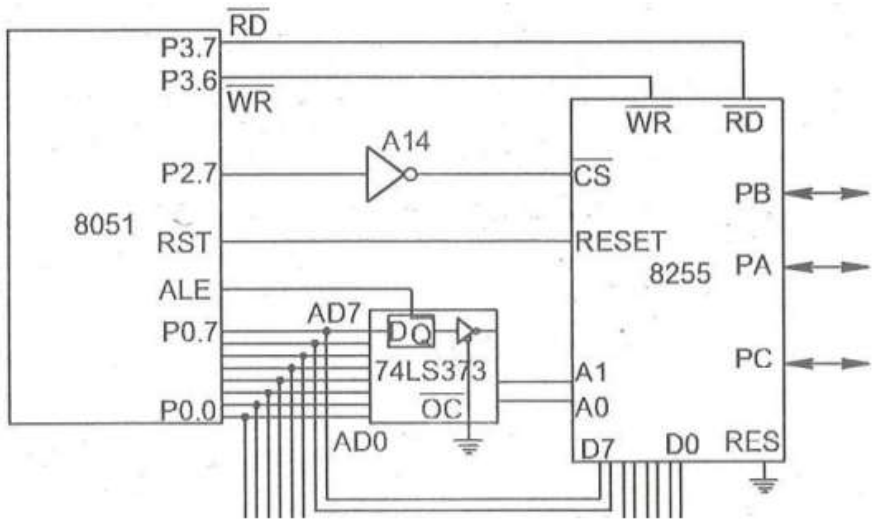
- The $\overline{WR}$ and $\overline{RD}$ signals are alternate Uses for port 3 pins 16 and17
- The Port 0 is used for the lower address bytes and data
- Port 2 is used for upper address bits

The 8051 accesses external RAM whenever MOVX type Instructions are executed.

External ROM accesses whenever $\overline{EA}$pin is connected to ground or when the program counter contains an address in between 1000H and FFFFH.

8051 INTERFACING WITH THE 8255

8051 have four I/O ports (Port 0, Port 1, Port 2, Port 3) for interfacing the peripherals. If not enough these I/O ports we can expend the I/O ports capability of 8051 interfacing with 8255. The interfacing diagram of 8255 with 8051 in shown in Fig 5.10 8051 interfacing with the 8255.



The bidirectional data bus D0- D7 of 8255 in connected to port 0 of 8051.The address latch is used / data bus (AD0- AD7).

The control lines $\overline{WR}$ and $\overline{RD}$ of 8255 in connected to the $\overline{WR}$ and $\overline{RD}$ lines of 8051. By Using the MOVX instruction the microcontroller can access the ports and control word Register of8255.

ASM PROGRAMMING

Assembly Language Programming:

Each family of processor has own set of institution for handling various operation. These set of institution are called "Machine language Institution.

Processors Understand only machine language institution which are strings of 1"s and 0"s.However machine language using computer for software development. So the low level assembly language is designed for a specific family of processors that represents various instructions in symbolic code and more.

Understandable form:

Advantage of Assemble language

1. It requires less memory and execution time.
2. It allows hardware – specific complex jobs in a easier way.
3. It is suitable for time critical jobs.

RELAYS

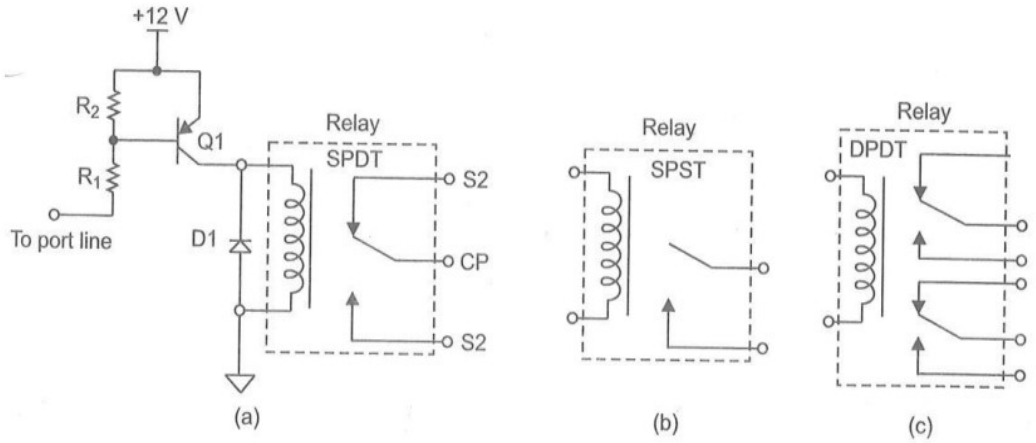
The electro mechanical relays have been used for many years in industry to control high dc or ac voltages and currents. Relays also provide isolation between the controller and the circuit under control. Relays are made up of three basic components:

1. Electromagnet
2. Spring
3. Some Contacts

Relays have two states – Open and close. A contact can be either normally open or normally close. The State of the contact can be changed by passing specified amount of current through the coil of the electromagnet.

Relay interfacing with microcontroller 8051

For energize the relay the voltage required for the coil of the electromagnet is normally +5V (or) +12V. On the other hand contact voltage can be 100 or more. Interfacing diagram of relay with 8051 is shown in fig interfacing diagram of relay with 8051.



Micro controller pins could not produce sufficient voltage (or) current to driver the relay. For this reason, we place a driver or a power transistor in between the microcontroller and the relay. Here SPDT (Single pole Double – throw) type relay is Interface with help of driver circuit. Single – refer common point, double - refer to two contact paints. One between CP and S1 is normally close and the contact between CP and S2 is normally Open, When the Electromagnet is energized by passing the desired amount of current through the coil the conditions reverses.

Program

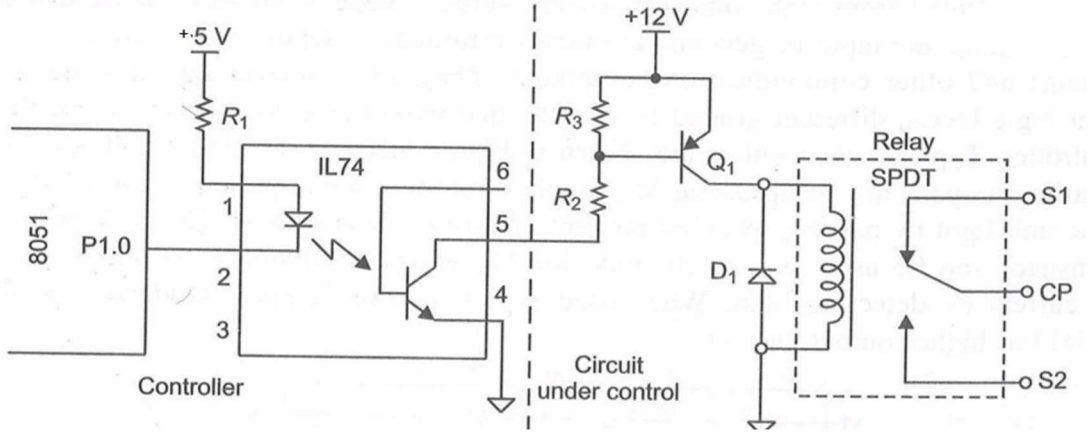
Label	Mnemonics	Comments
	ORG 4100 H	Origin at 4100 h
START	SETBP1.0	Set 1 at port pin p1.0
	ACALL Delay	Call the delay routine
	CLR P1.0	Set the Port Pin P1.0
	A CALL Delay	Call the delay routine
	SJMP START	Jump to the start
	END	

DELAY PROGRAM

Label	Mnemonics
	MOV R0#FF
L1	MOV R1# F0
L2	MOV R2 # FE
L3	DJNER2 L1
	DJNZR1 L2
	DJNZR0 L3
	RET

INTERFACING AND OPTO COUPLER:

An Opto Coupler used to isolate the port line of the microcontroller from the relay circuit as shown in the Fig interfacing and Opto coupler.

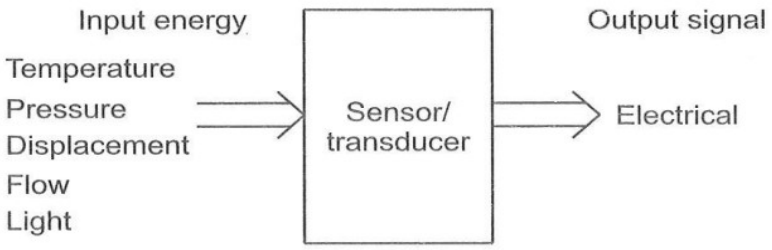


Opto coupler is a one of the solid state Relay. In this relay there is no coil, spring (or) mechanical contact. The entire Relays is made out of semiconductor materials. The switching time of solid state relay is faster than that of Electromechanical relay. So solid state relays are ideal for microcontrollers. They are widely used in controlling pumps, alarms, and other power applications. It is also used in communication equipment.

Label	Mnemonics	Comments
	ORG 2100H	Origin at 4100 h
REPT	MOV R1,#ABH	Load the value ABh at R1 Register
L1	MOV R1,#FFH	Load the value FFh at R1 Register
L2	DJNZ R2,L1	Wait the complete in inner loop
	DJNZ R2,L2	Wait the complete in inner loop
	CPL P1.0	Complement the previous output
	SJMP REPT	Repeat the Loop
	END	

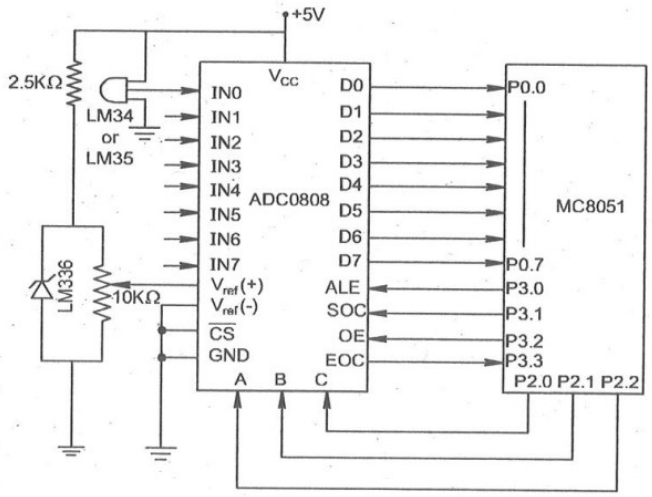
SENSOR:

Sensor is an energy conversion device**.** Which receive physical data such as temperature, pressure, light intensity, Speed or flow and generate electrical signals such as voltage, current, resistance or capacitance depending on the type of sensor used. Sensors also called as Transducers. The function of a Sensor is shown in the fig interfacing sensor with 8051



Temperature is one of the most important parameters that need to be monitored or controlled in a variety of application. Different types of temperature transducers are commercially available.

a) LM34 - Series of transducers are Precision Fahren height temperature Sensors.
b) LM35 - Series of transducers are precision Celsius temperature sensors giving output of 10mv for each degree of Celsius temperature.



The connection diagram for interfacing the temperature sensor LM35 with microcontroller through an ADC is shown in the Fig. The ADC 0808 has 8 bit resolution, with maximum of 256 (2^8) steps. The LM 35 produces 10mV for every degree of temperature change. We can condition V in of the ADC 0808 to produce a V out of 2.550mV (2.55V) for full scale output. Therefore in order to produce the full scale V out of 2.55V for the ADC 0808 correspond directly to the temperature as mentioned by LM35. The zener diode is used to give a steady voltage across the 10 K pot, which overcomes any fluctuations in the power supply.

LM 35 is connected directly to the IN0 input pin of ADC 0808. Port 0 pins of MC 8051 are directly connected to the digital output terminals of ADC 0808.

The port pin P3.0 is connected to ALE, the port pin P3.1 is connected to SOC, the port pin P3.2 is connected to OE and the port pin P3.3 is connected to EOC of 0808 ADC respectively. The port Pins P2.0, P2.1 and P2.2 are connected to the address lines of A, B and C Respectively.

Program

Label	Mnemonics	Comments
	ORG4100H	Originat4100H
	MOVP0,#FFH	Make port 0 as input
	SETBP3.3	Set EOC as input port pin
	MOVP2#00H	Select channel 0 (INO)
	SETBP3.0	Make ALE as High, enable ALE
	CLR P3.0	Again make ALE as Low
	SETBP3.1	Make SOC as High, initiate start of Conversion
	CLR P3.1	Again make SOC as low
WAIT	JNBP3.3.WAIT	wait for the completion of conversation Process
	SETBP3.2	Enable the output
	MOVA.P0	Read data throughport0 CLR
	CLR P3.2	Clear OE
	ACALL CONVER	Hexadecimal to ASCII Conversion
	ACALL DISPLAY	Go To Display Subroutine Conversion
	MOV B, #0AH	Load The OAH In B Register
	DIV A,B	Divide By using OAH

	MOV RO,B	Store The I's In Ro Register
	DIV A,B	Divide By Using OAH
	MOV R1,B	Store The 10's In R1 Register
	MOV R2,A	Store The 100's In R2 Register
	RET	Return To Main Program

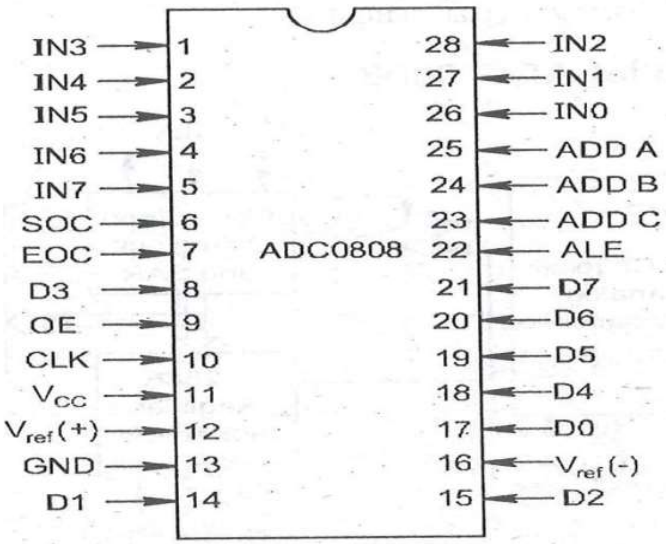
Display

Label	Mnemonics	Comments
	MOV P3,RO	Display 1's
	ACALL DELAY	Call To Delay Subroutine
	MOV P3,R1	Display In 10's
	ACALL DELAY	Call To Delay Subroutine
	MOV P3,R2	Display In 100's
	ACALL DELAY	Call To Delay Subroutine
	RET	

ADC INTERFACING

Digital computers (or) microcontrollers use binary values, but we get electrical signal from sensor (Transducers) are Analog signal. The Transducers convert the physical Quantifies like Temperature, Pressure, speed etc, into electrical signal. So whenever we want binary values the Analog to digital converters widely used.

One of the Analog to digital converters IC is ADC 0808. It is a Parallel analog to digital converter. The pin diagram of ADC 0808 are shown in fig

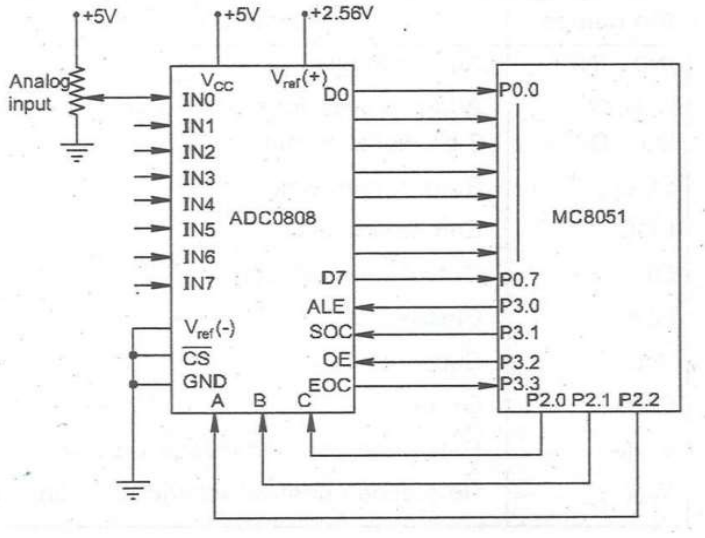


Pin Description are Shown in table

Pin Names	Description
IN0 – IN7	Analog Inputs
A, B, C	Address lines for selecting analog inputs
D0 – D7	8-bit digital output
SOC	Start of Conversion
EOC	End of Conversion
OE	Output of Conversion
CLK	Clock Input
VCC	Supply Voltage
GND	Ground
Vref(+)	Reference positive Voltage (+5V max)
Vref(-)	Reference negative Voltage (0V min)

Interfacing ADC 0808 with micro controller 8051

The interfacing diagram of ADC 0808 with microcontroller 8051 in shown in fig

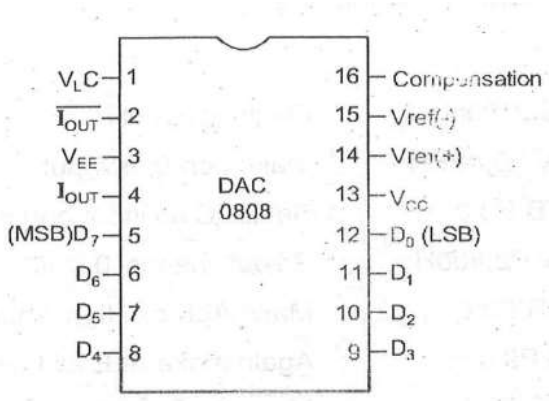


The analog signal is applied to the any input pin INO – IN7. The digital output terminals (D0- D7) of ADC 0808 are directly connected to port 0 in 8051. Port 3 in micro controller are connected different control signals in ADC 0808. P3.0 – ALE, P3.1 – SOC P3.2 – OE, P 3.3 – EOC respectively. Port 2 in micro controllers are connected to the address lines of A, B and C in ADC 0808. The Address lines A, B, C are used to select the particular input Lines

Label	Mnemonics	Comments
	ORG4100H	Origin4100H
	MOV P0# FFH	Set port 0asinput
	SETBP3.3	Set P3.3 as input
	MOV P2#00H	Select channel using Address Line
	SETBP3.0	Enable ALE
	CLRP3.0	disable ALE
	SETBP3.1	Initiate start of conversion
	CLR P3.1 WAIT	Stop the start of Conversion
WAIT	JNBP3.3 WAIT	wait for Conversion
	SETBP3.2	Enable the output
	MOVA,P0	Read data in P0 to ACC
	CLR P3.2	Clear OE
	ACALL CONVER	Hexadecimal to ASCII Conversion
	ACALL DISPLAY	Go To Display Subroutine Conversion
	END	End of Program

DAC INTERFACING:

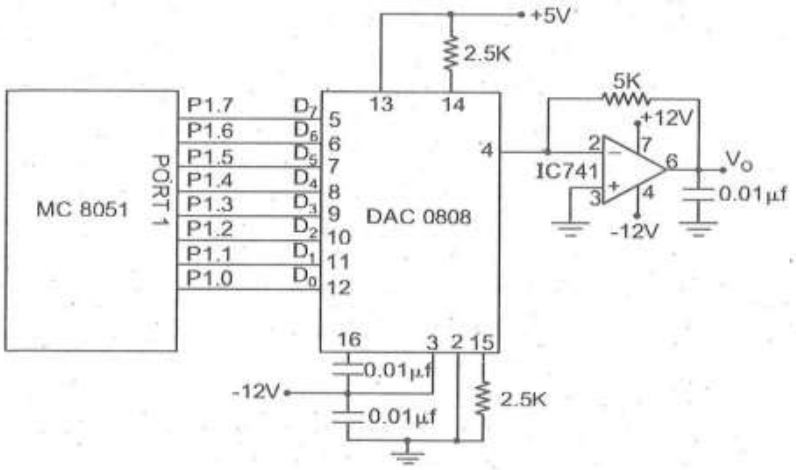
The Microcontroller output is binary values, but application equipment's display, motor; speakers etc. work in analog signal. So we need digital to Analog converters. The IC 0808 is an 8 bit DAC. The pin diagram of DAC 0808 in shown in fig



In DAC 0808 the digital inputs are converted to current (Iout). The current Iout is converted into voltage using Op- amp The Current value in Iout pin is depends upon the binary numbers at the D0 – D7 inputs of the DAD 0808 and the reference current (I ref)

$$I\,out = I\,ref\left\{\frac{D_7}{2^1} + \frac{D_6}{2^2} + \frac{D}{2^3} + \frac{D_4}{2^4} + \frac{D_3}{2^5} + \frac{D_2}{2^6} + \frac{D_1}{2^7} + \frac{D_0}{2^8}\right.$$

The interfacing diagram of DAC 0808 with microcontroller 8051 in show in interfacing diagram of DAC with 8051.



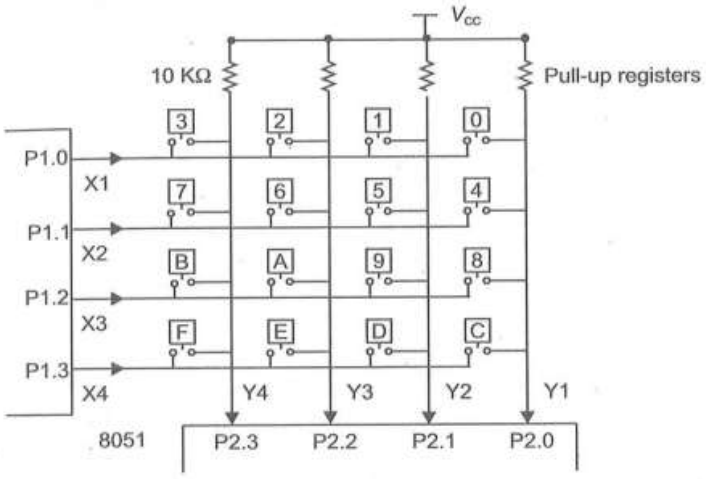
Label	Mnemonics	Comments
	ORG4200H	Origin 4200
	MOVX DPTR, #4300H	Load 4300 in DPTR
	MOVXA,@DPTR	Get data in
	ACCMOVP1,A	Send it port1
	A Call Delay	Make some delay
Delay	MOV RO,#F2H	Load the value at F2h
L1	MOV R1,#FFH	Load the value at FFh
L2	DJNZ R1,L1	Repeat if R2 not = 0
	DJNZ R0,L2	Repeat if R1 not = 0
	End	End of Program

KEY BOARDINTERFACING

A keyboard is a collection of push button type switches. Which is commonly used as input devices to a microcontroller based system. When a key is pressed, the microcontroller

identifies the pressed key by using either a software based or hardware based technique and then performs the assigned operation.

The key board is interface with micro controller the keys are arrange in a two dimensional matrix form as shown in fig



Here a row number and column number together uniquely identify a key. when a key is pressed, it is necessary to identify the row and column numbers of that key. The most commonly used approach for this purpose is known as row scanning technique While interfacing a keyboard the following issues are taken into consideration

- Key –bounce
- Multiple key press (rollover)
- Key Pressed and held

Key bounce:

The problem of key – bounce arises when a push- button key is pressed or released because of Mechanical Spring action, the key vibrates for a small duration of time say 10 to 20 ms making and breaking the contact several times before finally closing the contact or Opening the contact. So a single key closure may appear to be multiple key closures to the interface circuit. The Microcontroller Should be able to distinguish between a bounce and a genuine key press. Otherwise each time a switch is pressed; the microcontroller will detect several fast key presses.

Key bounce can be overcome by generating a delay of 10-20ms after sensing a key actuation and then re- sensing the key closure. If the key is still closed, the key closure is accepted as a valid key closure.

Multiple Key Press:

Multiple – key press problem is commonly known as rollover, which arises when two or more key are pressed simultaneously. This problem should be overcome so that the microcontroller does not perform an operation corresponding to a wrong key . There are three approaches to resolve the problem of rollover.

1. Two – Key rollover

This provides protection against the simultaneous closure of two keys. To ignore the keyboard until a single key press is detected, and the last key to remain pressed is accepted by the microcontroller. Performing the assigned operation is the best approach to overcome this issue.

2. N- Key rollover

This approach gives protected against N- Keys pressed simultaneously. This issues are resolve to store all the key closures in some internal buffer and perform the respective operations in sequence.

3. K. Key Lockout

In this method, a single key closures is recognized and additional key closures are ignored until the first one is released.

4. Key pressed and Held:

The Pressed key is accepted by the microcontroller after the debounce delay. No additional key press in accepted until all the keys are seen open certain period of time.

Row Scanning Technique:

A 4x4 keyboard matrix interfaced with 8051 microcontroller. There are four rows connected to the four port lines P1.0- P1.3 of port 1 and four column lines connected to the four port lines P2.0-P2.3 of port 2.

Closure of any one of the 16 keys in identified by row- scanning technique by searching one row at a time, is a time division multiplexed manner. In row scanning technique the following steps are performed.

Step 1: In this step the row lines (P1.0 - P1.3) are configured as output lines. While the column lines (P2.0 - P2.3) are configured as input lines.

Step 2: In this step check whether all the keys are open or not. This step in performed by generating 0‟s on all the (P1.0 - P1.3) row lines and then reading the (P2.0 - P2.3) column lines. The controller keeps on looping till all the (P2 .0 – P2.3) Column lines are 1, indicating that no key is closed.

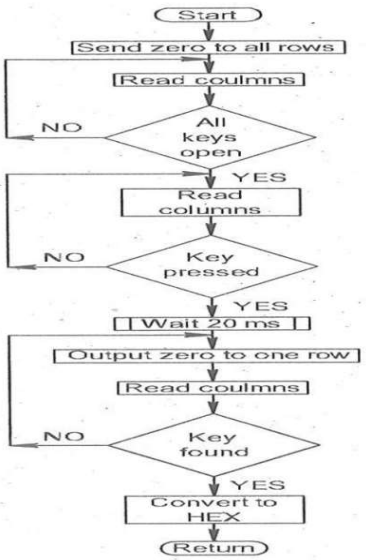
Step 3: In this step check whether any new key closure has taken place or not.Thisisdonebyoutputting0‟sonallthe (P1.0-P1.3) rows and reading the P2.0- P2. 3 line values. The controller keeps on looping until one of the column lineis„0; which indicates that a key closure has taken place.

Step 4: In this step a delay of 20ms is generated to overcome key bounce problems.

Step 5: In this step, actual identification of the key pressed is performed by scanning one row at a time. This is performed by outputting „0‟ on the row under scan and „1‟ on the remaining lines and reading P2.0 – P 2.3 Lines A „0‟ on any of the P2.0 – P 2.3 lines indicates that a key on that column in the row under scan has been pressed.

Step 6: In this step, first the decoded codes corresponding to the row and column numbers are obtained. Then the decoded row and column codes are used. Obtain the key code of the key pressed.

The Row Scanning technique is shown in the form of a flowchart as shown in fig



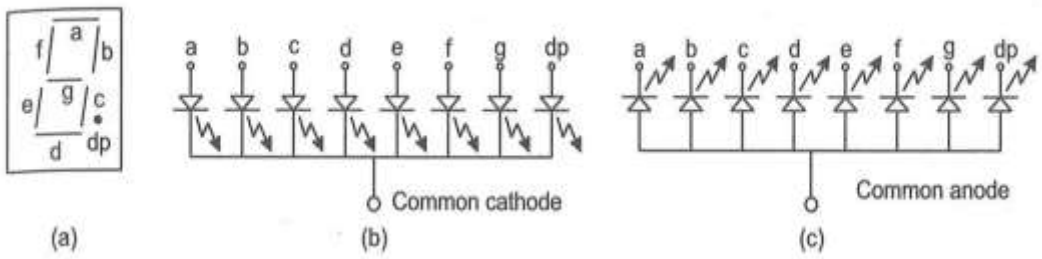
Label	Mnemonics	Comments
	MOV P3,#OFFh	Make p2 an input port
K1	MOV P1,#00h	Ground all rows at once
	MOV A,P3	Read all Columns. Ensure all keys open.
	ANL A,# OOOO1111B	Mask unused bits
	CJNE A, #OOOO1111Row _0	key row 0,find the col
	MOV P1,#111111O1B	ground ROW 1
	MOV A,P3	Read all Column
	ANL A,# OOOO1111B	Mask unused bits
	CJNE A, #OOOO1111Row _1	key row 1,find the col
	MOV P1,#11111O11B	ground ROW 2
	MOV A,P3	Read all Column
	ANL A,# OOOO1111B	Mask unused bits
	CJNE A, #OOOO1111Row _3	key row 3,find the col
	LJMP K2	if none , false input ,repeat
ROW_0	MOV DPTR,#KCODE0	Set DPTR = Start of row 0
	SJMP FIND	find the col key belong to
ROW_1	MOV DPTR,#KCODE0	Set DPTR = Start of row 1
	SJMP FIND	find the col key belong to
ROW_2	MOV DPTR,#KCODE0	Set DPTR = Start of row 2
	SJMP FIND	find the col key belong to
ROW_3	MOV DPTR,#KCODE0	Set DPTR = Start of row 3
FIND	RRC A	See if any cy bit is low
	JNC MATCH	If Zero ,Get the ASCII CODE
	INC DPTR	Point to next col. Address
	SJMP FIND	Keep Searching
MATCH	CLR A	Set A=0 (Match is Found)
	MOV A@A+DPTR	Get ASCII Code from Table
	LJMP K1	

ASCII LOOK TABLE FOR EACH ROW

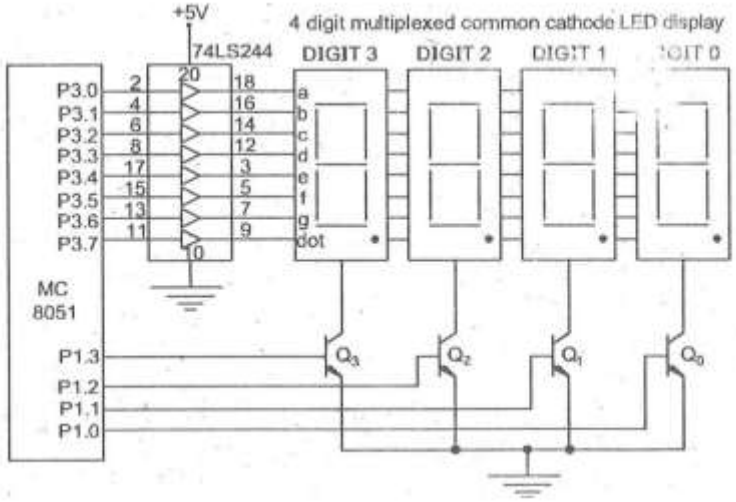
```
        ORG 0300H
KCODE: DB 0, 1, 2, 3: ROW 0
KCODE: DB 4, 5, 6, 7: ROW 1
KCODE: DB 8, 9, A, B: ROW 2
KCODE: DB C, D, E, F: ROW 3
```

SEVEN SEGMENT LED DISPLAYINTERFACING

There are many applications where you have to display numbers. The most popular display device used for displaying number is seven segment LED displays. In each module seven (Eight including the decimal point) LED segments are fabricated in a pattern as shown in Fig 5.20 (a). The seven segments are numbered as a, b, c, d, e, f, g and the decimal point is dp . To reduce the number of pin counts, either all the cathodes are connected together inside the module providing common cathode type display (1-ON, 0 - OFF) as shown fig 5.20 (b) on all the anodes are connected together providing common anode type display (0 – ON, 1 - OFF) as shown is Figure 5.20 (c). If the display units are multiplexed together, we can display more than one character at a time.



Interfacing diagram of 4-digit multiplexed LED display with microcontroller 8051 is shown fig. In this four 7 segment LEDS are multiplexed together. The port 3 pins are used to drive the segments of the LED through driver IC. Similarly the port pins P1.0 through P1. 3 are used to drive the digits through driver transistors.



For displaying a Number 1,2,3,4 in the display unit the hexa code may be formed as follows.

Number to be displayed	G	F	E	D	C	B	A	Equivalent Hex Code
0	0	1	1	1	1	1	1	3F
1	0	0	0	0	1	1	0	06
2	1	0	1	1	0	1	1	5B
3	1	0	0	1	1	1	1	4F
4	1	1	0	0	1	1	0	66
5	1	1	0	1	1	0	1	6D
6	1	1	1	1	1	0	1	7D
7	0	0	0	0	1	1	1	07
8	1	1	1	1	1	1	1	7F
9	1	0	0	1	1	1	1	4F

The Number 1 may be displayed at digit 3 (P 1.3), the Number 2 Displayed at digit 2 (P1.2), the number 3 displayed at digit 1 (P 1.1) and the 4 may be displayed at digit 0 (P1.0)

If more than one display is to be used, the they can be time multiplexed. The human eye cannot detect the blinking if each display is relit every 10 ms or so. A segment will be lit only if the segment line is brought high and the common cathode is brought low. Transistor must be used to handle the currents required by the LEDS, typically l0mA for each segment and 7mA for each cathode.

Label	Mnemonics	Comments
	ORG4100H	Origin at4100H
REPEAT	MOVR0,88H	Load the digit drive codeinR0 MOV
	DPTR, #TABLE	Load the starting address of table in DPTR
	MOVR1, #04H	Load number of charactersinR1 FIRST
	MOVX A,@DPTR	Get the HEXA code in A
	MOVP3,A	Move the HEXA code through Port 3
	MOVP1, R0	Move the digit drive code through Port1
	ACALLDELAY	Call delay subroutine
	INCDPTR	Get the address of next HEXA code for displaying next character
	MOVA,R0	Get the previous digit code in A
	RR A	Rotate right to get next digit code
	MOVR0, A	Place the digital codeinR0 DJNZR1
	SJMPREPEAT	Repeat the Process
	END	End of Program
DELAY	MOV R3,# 25H	Load the outer loop value
WAIT2	MOV R4,# FFH	Load the inner loop value
WAIT 1	DJNZ R4,WAIT1	Wait for delay
	DJNZR3,WAIT 2	Wait for delay
	RET	Return to main program

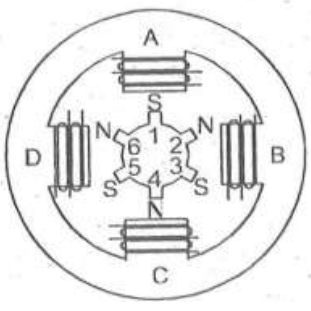
TABLE: 03H, 53H, 4FH 66H

STEPPER MOTORINTERFACING

Stepper motor is a device used for getting accurate position control of rotating shafts. It converts electrical pulses into mechanical movements. A stepper motor employs rotation of its shaft, in terms of steps rather than continues rotation.

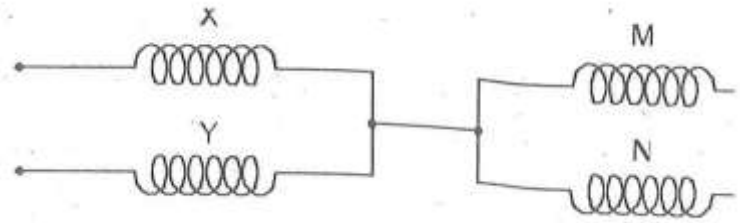
To rotate the shaft of the stepper motor a sequence of pulses is needed for applying to the windings of the stepper motor. The number of pulses required for one complete rotation of the shaft of the stepper motor is equal to its number of teeth on its rotor. The stator teeth and rotor teeth lock with each other to fix a position of the shaft. The schematic diagram of a stepper motor in shown in fig

Schematic diagram of Stepper motor



Stepper motor is a permanent magnet type stepper motor. It contains a four pole stator and a rotor with six permanent poles. The stator is made up of laminated Soft iron. The stator windings are energized by the application of pulses. Each pole of stator has two coils wound in an opposite sense. So that each pole can be made either a north pole or a south pole as described by applying appropriate pulse to one of the coil. This type of winding in known as filler pole Winding.

bi - filler pole windings



The structure of bi-filler pole windings is shown in the fig . The X and Y are two coils wound on the same pole. Similarly M and N are the two coils wound on the same pole, which are situated at diametrically opposite position.

The excitation sequence of a stepper motor is shown in the table below.

Step	Winding D	Winding C	Winding B	Winding A	Hexa	Direction
1	1	0	0	1	09H	FORWARD ↓ / REVERSE ↑
2	0	0	1	1	03H	
3	0	1	1	0	06H	
4	1	1	0	0	0CH	

The connection diagram of stepper motor with Microcontroller 8051 shown in the Fig The four coils of stepper motor are connected to the port 0 pins through drivers. The port pin P2.0 is connected to the coil A, the Port pin P2.1 is connected to coil B, the Port pin P2.2 it's connected to the coil C and the Port pin P23 is connected to the Coil D.

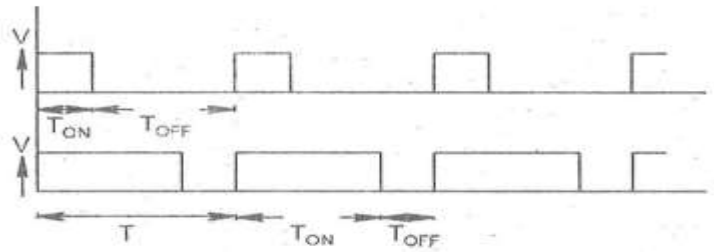
Label	Mnemonics	Comments
	ORG4000H	Origin4000H
	MOV A,#09H	Place first data in Acc.
REPEAT	MOVP2,A	Apply the data to Port1
	LCALLDELAY	Call delay subroutine
	RLA	Repeat the next data by using rotation
	SJMPREPEAT	Repeat the above process
	END	End of the program

DELAY SUBROUTINE

Label	Mnemonics	Comments
DELAY	MOV R0, # XXH	Load R0 register
FIRST	MOV R1, # XXH	Load R1 register
NEXT	DJNZ R1, NEXT	Decrement inner loop up to 00H
	DJNZ R0, FIRST	Decrement outer loop up to 00H
	RET	Return to the main program

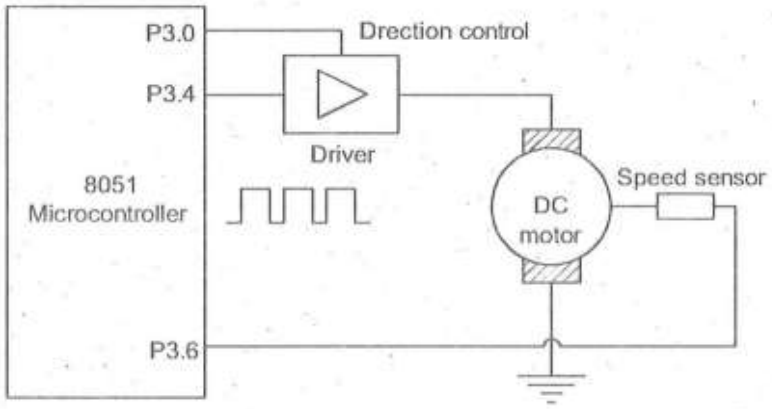
DC MOTOR INTERFACING USING PWM

DC Motor rotates continuously at constant speed of DC Motor depends upon the average DC Voltage applied across its armature. If we want to change the speed of the motor to vary the applied DC Voltage, If the polarity of the applied voltage is changed, the motor can run in reverse direction.



In modern days, the speed of the DC Motor is Controlled by using Pulse width modulation (PWM) method. By using the PWM signal, the average voltage of the DC motor can be varied, there by the speed of the DC motor is controlled.

Two different pulse modulated signals are shown in the fig. In the second waveform, ON period is high and OFF period is low. Therefore its average voltage is high. In the first waveform ON period is low and the OFF period is high. Therefore its average voltage is low. In pulse width modulation, the period of the pulse is kept constant. Pulse ON period and OFF period times may be varied. The interfacing diagram of DC motor with 8051 microcontroller is shown in the fig



The speed of the DC motor is increased, when the duty cycle of the signal is high. Similarly, the speed of the DC motor is decreased, when the duty cycle of the signal is low. By using the program, we can be able to change the duty cycle of the PWM signal and also the motor speed may be varied.

Normally the current produced from the microcontroller is not sufficient to drive the DC motor effectively. Hence drivers may be connected in between the motor and microcontroller.

In some applications, we keep the speed of the motor as constant. For doing this, a feedback circuit may be connected to it. The feedback circuit contains a speed sensor for getting the present speed of the motor. According to the sensor output, the microcontroller adjusts the speed of the motor by changing the duty cycle of the signal applied to it. Hence constant speed can be maintained.

Label	Mnemonics	Comments
	ORG4000H	Origin at 4000H
	SETBP3.0	Set in forwarded direction
REPEAT	SETBP3.4	Make high level output
	ACALLONDLY	Call ON period delay
	CLRP3.4	Make low level output
	ACALLOFFDLY	OFF period delay
	SJMPREPEAT	Repeat the process
	END	End of Program

On delay program

Label	Mnemonics	Comments
ONDLY	MOV R1,# A0	ON time delay
WAIT2	MOV R2, # FFH	Load R2 with FFH
WAIT1	DJNZ R2,WAIT1	Wait for the completion On Time
	DJNZ R1,WAIT2	Wait for the completion On Time
	RET	Return to man program

Off delay program

Label	Mnemonics	Comments
OFFDLY	MOVR1, #5F	OFF time delay
	MOV R2, # FFH	Load R2 with FFH
NEXT 2	DJNZ R2,NEXT1	Wait for the completion OFF Time
NEXT 1	DJNZ R1,NEXT 2	Wait for the completion OFF Time
	RET	Return to man program