MODULE-IV (8051 Microcontroller)

INTEL 8051 MICRCONTROLLER :

The 8051 microcontroller is a very popular 8-bit microcontroller introduced by Intel in the year 1981 and it has become almost the academic standard now a days. The 8051 is based on an 8-bit CISC core with Harvard architecture. Its 8-bit architecture is optimized for control applications with extensive Boolean processing. It is available as a 40-pin DIP chip and works at +5 Volts DC. The salient features of 8051 controller are given below.

SALIENT FEATURES : The salient features of 8051 Microcontroller are

i. 4 KB on chip program memory (ROM or EPROM)).

ii. 128 bytes on chip data memory(RAM).

iii. 8-bit data bus

iv. 16-bit address bus

v. 32 general purpose registers each of 8 bits

vi. Two -16 bit timers T_0 and T_1

vii. Five Interrupts (3 internal and 2 external).

ix. Four Parallel ports each of 8-bits (PORT0, PORT1,PORT2,PORT3) with a total of 32 I/O lines.

x. One 16-bit program counter and One 16-bit DPTR (data pointer)

xi. One 8-bit stack pointer

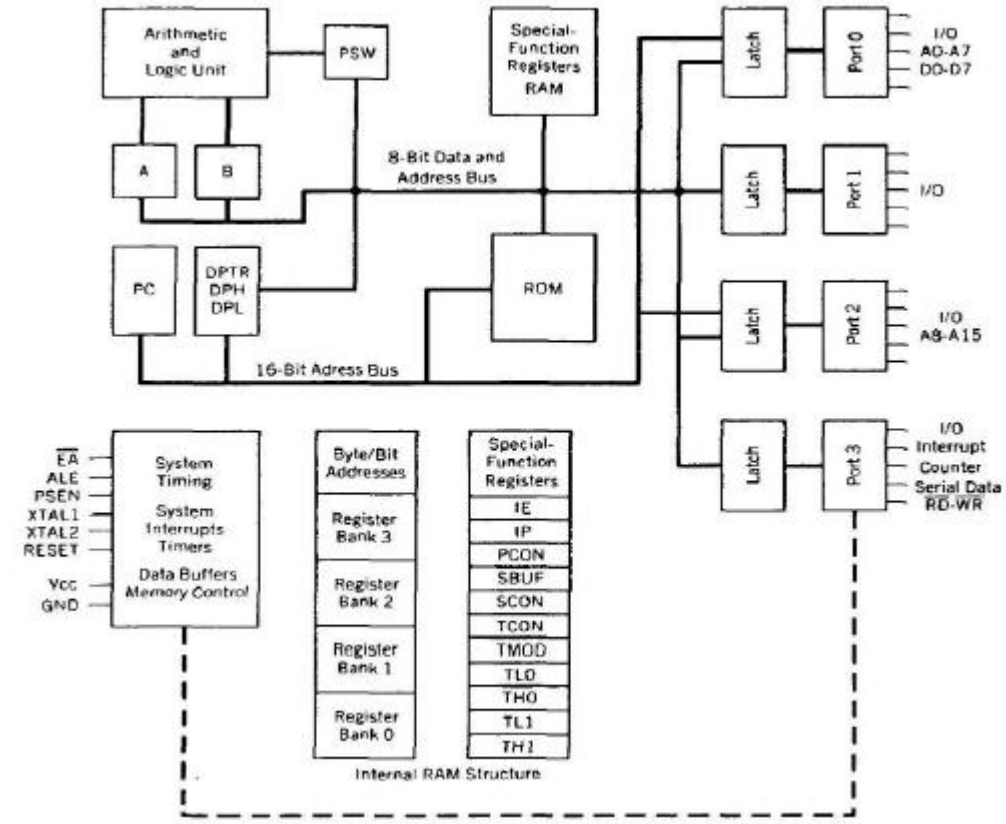
xii. One Microsecond instruction cycle with 12 MHz Crystal.

xiii. One full duplex serial communication port.

8051 Microcontroller Hardware:

The 8051 microcontroller actually includes a whole family of microcontrollers that have numbers ranging from 8031 to 8751 and are available in N-Channel Metal Oxide Silicon (NMOS) and Complementary Metal Oxide Silicon (CMOS) construction housed in a 40-pin DIP.

FIGURE 2.1a 8051 Block Diagram

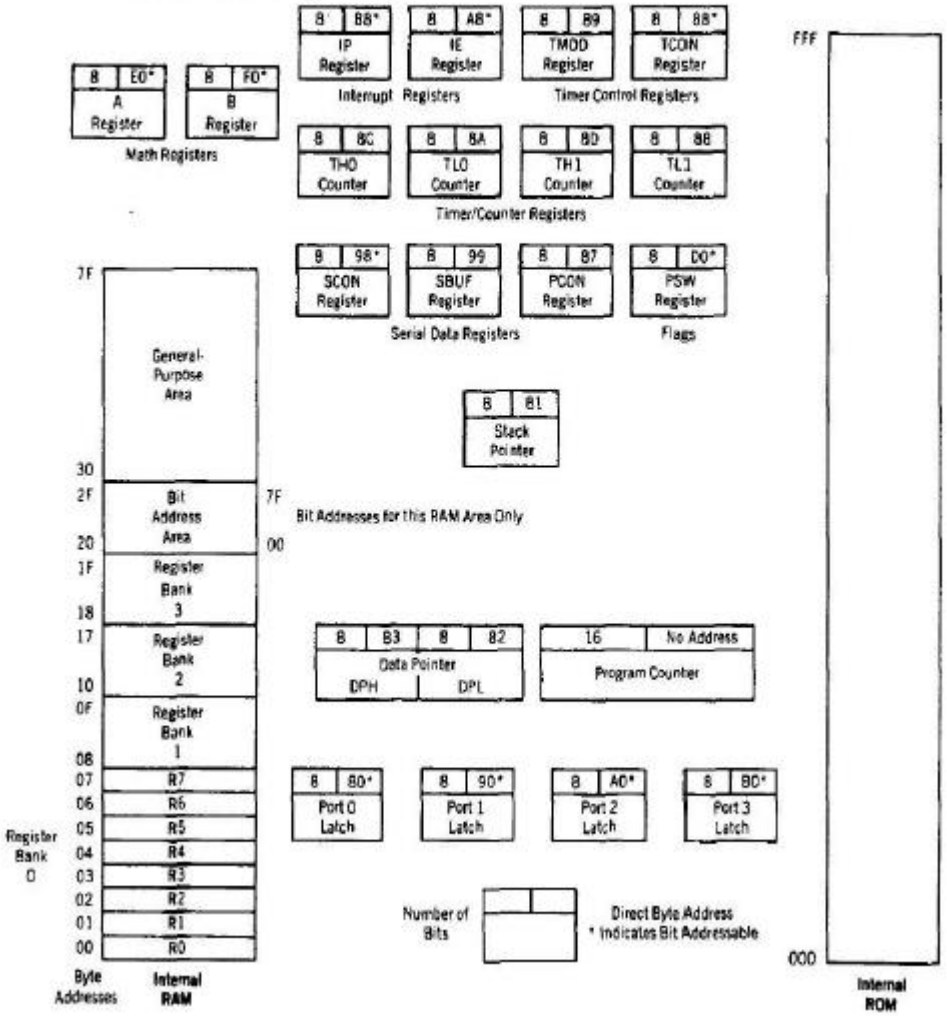


The block diagram in Fig 2.1a shows all of the features unique to microcontrollers:

1. Internal ROM and RAM
2. I/O ports with programmable pins
3. Timers and counters
4. Serial data communication

The figure also shows the usual CPU components: program counter, ALU, working registers, and clock circuits.'

FIGURE 2.1b 8051 Programming Model



The programming model of the 8051 in Figure 2.1b shows the 8051 as a collection of 8- and 16-bit registers and 8-bit memory locations. These registers and memory locations can be made to operate using the software instructions that are incorporated as part of the design. The program instructions have to do with the control of the registers and digital data paths that are physically contained inside the 8051, as well as memory locations that are physically located outside the 8051.

The model is complicated by the number of special-purpose registers that must be present to make a microcomputer a microcontroller. A cursory inspection of the model is recommended for the first-time viewer; return to the model as needed while progressing through the remainder of the text.

Most of the registers have a specific function; those that do occupy an individual block with a symbolic name, such as A or THO or PC. Others, which are generally indistinguishable from each other, are grouped in a larger block, such as internal ROM or RAM memory.

Each register, with the exception of the program counter, has an internal 1-byte address assigned to it. Some registers (marked with an asterisk * in Figure 2.1b) are both byte and bit addressable. That is, the entire byte of data at such register addresses may be read or altered, or individual bits may be read or altered. Software instructions are generally able to specify a register by its address, its symbolic name, or both. A pin out of the 8051 packaged in a 40-pin DIP is shown in Figure 2.2 with the full and abbreviated names of the signals for each pin. It is important to note that many of the pins are used for more than one function (the alternate functions are shown in parentheses in Figure 2.2). Not all of the possible 8051 features may be used at the same time.

Programming instructions or physical pin connections determine the use of any multifunction pins. For example, port 3 bit 0 (abbreviated P3.0) may be used as a general purpose I/O pin, or as an input (RXD) to SBUF, the serial data receiver register. The system designer decides which of these two functions is to be used and designs the hardware and software affecting that pin accordingly.

Program Counter and Data Pointer

The 8051 contains two 16-bit registers: the program counter (PC) and the data pointer (DPTR). Each is used to hold the address of a byte in memory.

Program instruction bytes are fetched from locations in memory that are addressed by the PC. Program ROM may be on the chip at addresses OOOOh to OFFFh, external to the chip for addresses that exceed OFFFh, or totally external for all addresses from OOOOh to FFFFh. The PC is automatically incremented after every instruction byte is fetched and may also be altered by certain instructions. The PC is the only register that does not have an internal address.

The DPTR register is made up of two 8-bit registers, named DPH and DPL, that Areused to furnish memory addresses for internal and external code access and external data access. The DPTR is under the control of program instructions and can be specified by its 16-bit name, DPTR, or by each individual byte name, DPH and DPL. DPTR does not have a single internal address; DPH and DPL are each assigned an address.

A and B CPU Registers

The 8051 contains 34 general-purpose, or working, registers. Two of these, registers A and B, comprise the mathematical core of the 8051 central processing unit (CPU). The other 32 are arranged as part of internal RAM in four banks, BO-B3, of eight registers each, named RO to R7.

The A (accumulator) register is the most versatile of the two CPU registers and is used for many operations, including addition, subtraction, integer multiplication and division, and Boolean bit manipulations. The A register is also used for all data transfers between the 8051 and any external memory.

The B register is mainly used for multiplication and division operations along with A register.

MUL AB : DIV AB.

It has no other function other than as a location where data may be stored.

Flags and the Program Status Word (PSW):

Flags are 1 -bit registers provided to store the results of certain program instructions. Other instructions can test the condition of the flags and make decisions based upon the flag states. In order that the flags may be conveniently addressed, they are grouped inside the program status word (PSW) and the power control (PCON) registers.

The 8051 has four math flags that respond automatically to the outcomes of math operations and three general-purpose user flags that can be set to 1 or cleared to 0 by the programmer as desired. The math flags include carry (C), auxiliary carry (AC), overflow (OV), and parity (P). User flags are named FO, GFO, and GF1; they are general-purpose flags that may be used by the programmer to record some event in the program. Note that all of the flags can be set and cleared by the programmer at will. The math flags, however, are also affected by math operations.

The program status word is shown in Figure 2.4. The PSW contains the math flags, user program flag FO, and the register select bits that identify which of the four general purpose register banks is currently in use by the program. The remaining two user flags, GFO and GFl, are stored in PCON, which is shown in Figure 2.13.

PSW Program Status Word Register

7	6	5	4	3	2	1	0
CY	AC	F0	RS1	RS0	OV	—	P

THE PROGRAM STATUS WORD (PSW) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7	CY	Carry flag; used in arithmetic, JUMP, ROTATE, and BOOLEAN instructions
6	AC	Auxilliary carry flag; used for BCD arithmetic
5	F0	User flag 0
4	RS1	Register bank select bit 1
3	RS0	Register bank select bit 0

RS1	RS0	
0	0	Select register bank 0
0	1	Select register bank 1
1	0	Select register bank 2
1	1	Select register bank 3

2	OV	Overflow flag; used in arithmetic instructions
1	—	Reserved for future use
0	P	Parity flag; shows parity of register A: 1 = Odd Parity

Bit addressable as PSW.0 to PSW.7

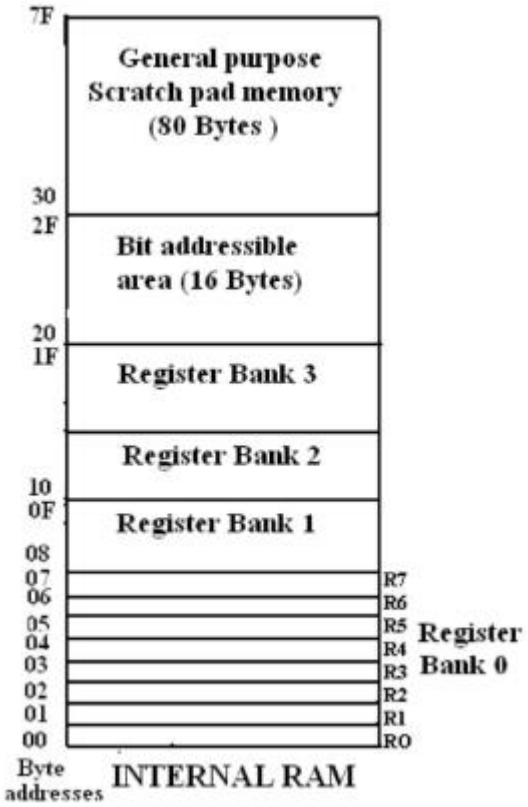
Detailed descriptions of the math flag operations will be discussed in chapters That cover the opcodes that affect the flags.

Internal Memory:

A functioning computer must have memory for program code bytes, commonly in ROM, and RAM memory for variable data that can be altered as the program runs. The 8051 has internal RAM and ROM memory for these functions. Additional memory can be added externally using suitable circuits.

Unlike microcontrollers with Von Neumann architectures, which can use a single memory address for either program code or data, but not for both, the 8051 has a Harvard architecture, which uses the same address, in different memories, for code and data. Internal circuitry accesses the correct memory based upon the nature of the operation in progress.
Internal RAM:

The 128-byte internal RAM, which is shown generally in Figure 2.1 and in detail in Figure 2.5, is organized into three distinct areas:



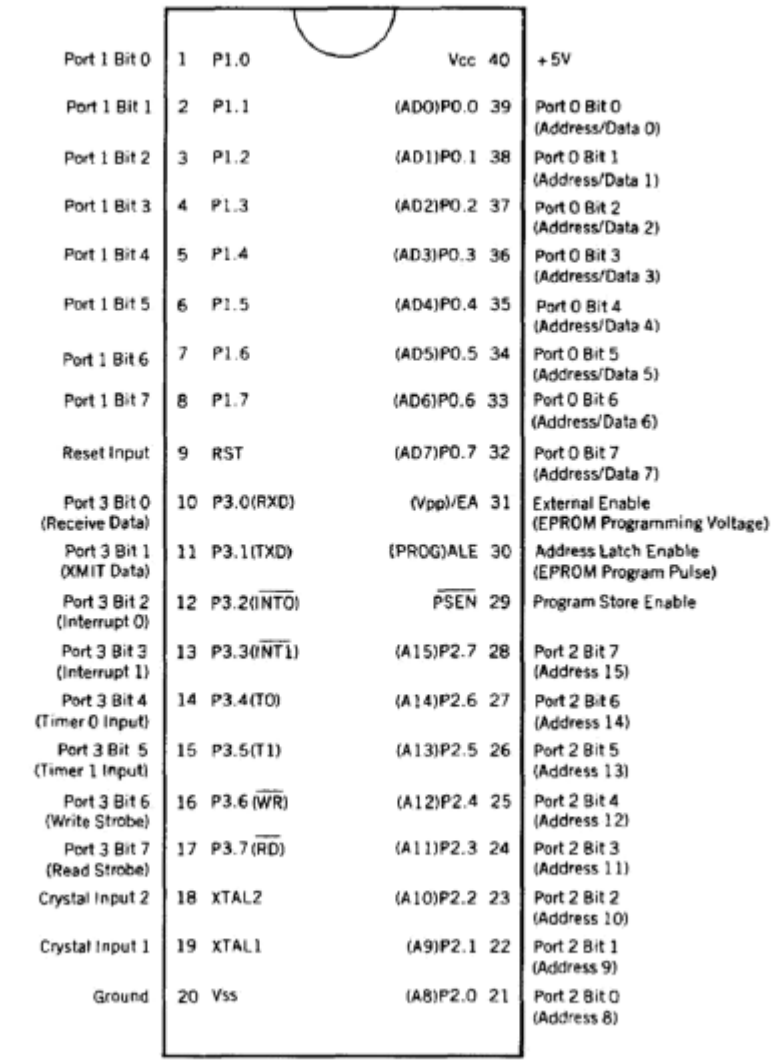
1. Thirty-two bytes from address OOh to I Fh that make up 32 working registers organized as four banks of eight registers each. The four register banks are numbered 0 to 3 and are made up of eight registers named RO to R7. Each registercan be addressed by name (when its bank is selected) or by its RAM address.

Thus RO of bank 3 is RO (if bank 3 is currently selected) or address 18h (whether bank 3 is selected or not). Bits RSO and RSI in the PSW determine which bank of registers is currently in use at any time when the program is running.
Register banks not selected can be used as general-purpose RAM. Bank 0 is selected upon reset.

2. A Wf-addressable area of 16 bytes occupies RAM byte addresses 20h to 2Fh, forming a total of 128 addressable bits. An addressable bit may be specified by its bit address of OOh to 7Fh, or 8 bits may form any byte address from 20h to 2Fh.
Thus, for example, bit address 4Fh is also bit 7 of byte address 29h.
Addressable bits are useful when the program need only remember a binary event (switch on, light off, etc.). Internal RAM is in short supply as it is, so why use a byte when a bit will do?

3. A general-purpose RAM area above the bit area, from 30h to 7Fh, addressable as bytes.

FIGURE 2.2 8051 DIP Pin Assignments



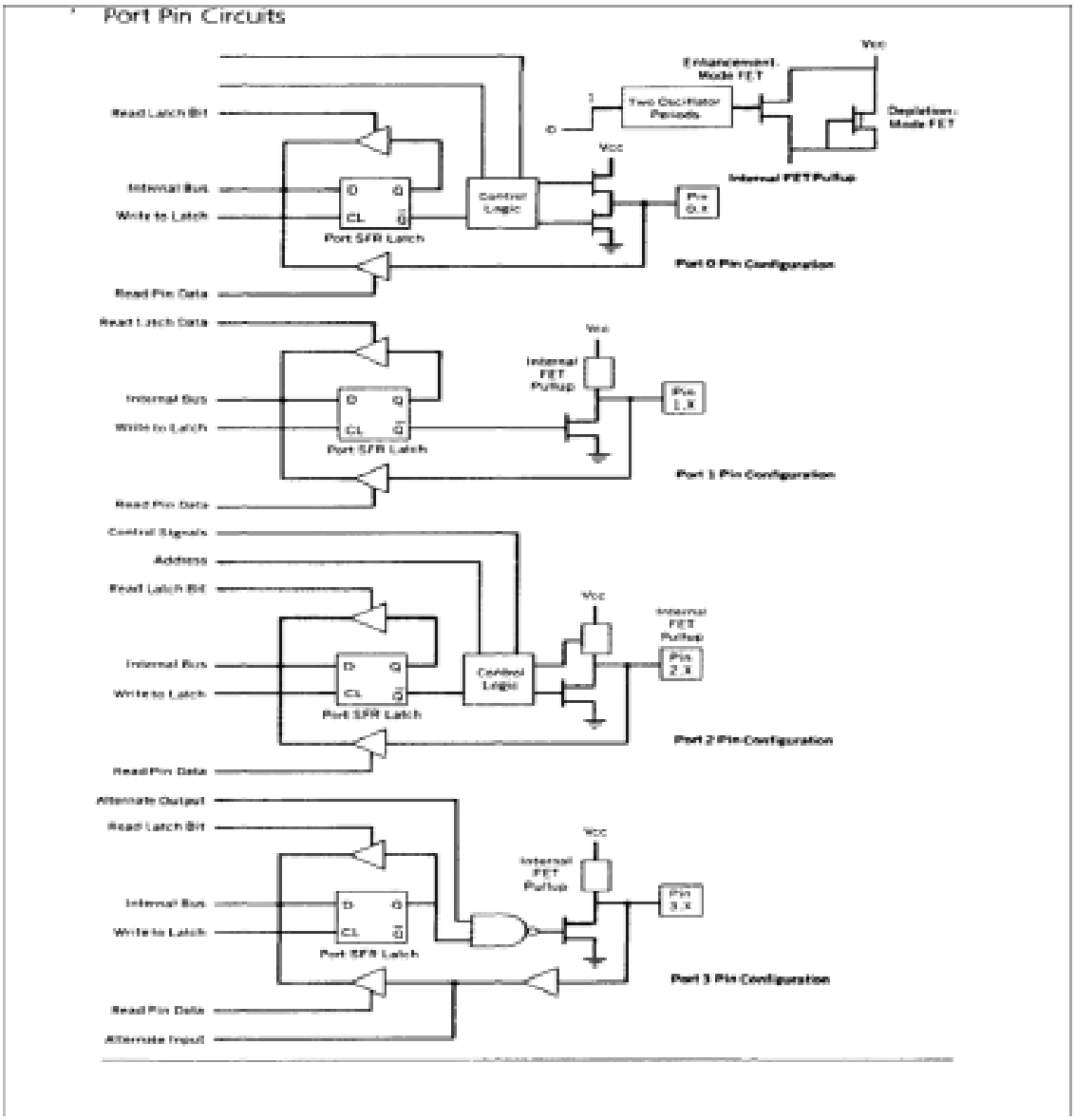
Note: Alternate functions are shown below the port name (in parentheses). Pin numbers and pin names are shown inside the DIP package.

Port 0:

Port 0 pins may serve as inputs, outputs, or, when used together, as a bidirectional Low order address and data bus for external memory. For example, when a pin is to be used as an input, a 1 must be written to the corresponding port 0 latch by the program, thus turning both of the output transistors off, which in turn causes the pin to "float" in a high impedance state, and the pin is essentially connected to the input buffer.

When used as an output, the pin latches that are programmed to a 0 will turn on the lower FET, grounding the pin. All latches that are programmed to a 1 still float; thus, external pull up resistors will be needed to supply a logic high when using port 0 as an output.

When port 0 is used as an address bus to external memory, internal control
Signal switch the address lines to the gates of the Field Effect Transistors (FETs). A logic 1 on an address bit will turn the upper FET on and the lower FET off to provide a logic high at the pin. When the address bit is a zero, the lower FET is on and the upper FET off to provide a logic low at the pin. After the address has been formed and latched into external circuits by the Address Latch Enable (ALE) pulse, the bus is turned around to become a data bus. Port 0 now reads data from the external memory and must be configured as an input, so a logic 1 is automatically written by internal control logic to all port 0 latches.



Port 1

Port 1 pins have no dual functions. Therefore, the output latch is connected directly to the gate of the lower FET, which has an FET circuit labeled "Internal

FET Pull up" as an active pull up load.

Used as an input, a 1 is written to the latch, turning the lower FET off; the pin and the input to the pin buffer are pulled high by the FET load. An external circuit can overcome the high impedance pull up and drive the pin low to input a 0 or leave the input high for a 1.

If used as an output, the latches containing a I can drive the input of an external circuit high through the pull up. If a 0 is written to the latch, the lower FET is on, the pull up is off, and the pin can drive the input of the external circuit low. To aid in speeding up switching times when the pin is used as an output, the internal FET pull up has another FET in parallel with it. The second FET is turned on for two oscillator time periods during a low-to-high transition on the pin, as shown in Figure 2.7.

This arrangement provides a low impedance path to the positive voltage supply to help reduce rise times in charging any parasitic capacitances in the external circuitry.

Port 2

Port 2 may be used as an input/output port similar in operation to port 1. The alternate use of port 2 is to supply a high-order address byte in conjunction with the port 0 low-order byte to address external memory.

Port 2 pins are momentarily changed by the address control signals when supplying the high byte of a 16-bit address. Port 2 latches remain stable when external memory is addressed, as they do not have to be turned around (set to 1) for data input as is the case for port 0.

Port 3

Port 3 is an input/output port similar to port I. The input and output functions can be programmed under the control of the P3 latches or under the control of various other special function registers. The port 3 alternate uses are shown in the following table:-

PIN	ALTERNATE USE	SFR
P3.0–RXD	Serial data input	SBUF
P3.1–TXD	Serial data output	SBUF
P3.2–$\overline{\text{INT0}}$	External interrupt 0	TCON.1
P3.3–$\overline{\text{INT1}}$	External interrupt 1	TCON.3
P3.4–T0	External timer 0 input	TMOD
P3.5–T1	External timer 1 input	TMOD
P3.6–$\overline{\text{WR}}$	External memory write pulse	—
P3.7–$\overline{\text{RD}}$	External memory read pulse	—

Unlike ports 0 and 2, which can have external addressing functions and change All eight port bits when in alternate use, each pin of port 3 may be individually programmed to be used either as I/O or as one of the alternate functions.

External Memory

The system designer is not limited by the amount of internal RAM and ROM available on chip. Two separate external memory spaces are made available by the 16-bit PC and DPTR and by different control pins for enabling external ROM and RAM chips. Internal control circuitry accesses the correct physical memory, depending upon the machine cycle state and the op code being executed.

There are several reasons for adding external memory, particularly program memory, when applying the 8051 in a system. When the project is in the prototype stage, the expense—in time and money—of having a masked internal ROM made for each program "try" is prohibitive.
To alleviate this problem, the manufacturers make available an EPROM version, the 8751, which has 4K of on-chip EPROM that may be programmed and erased as needed as the program is developed. The resulting circuit board layout will be identical to one that uses a factory-programmed 8051. The only drawbacks to the 8751 are the specialized EPROM programmers that must be used to program the non-standard 40-pin part, and the limit of "only" 4096 bytes of program code. The 8751 solution works well if the program will fit into 4K bytes.

Unfortunately, many times, particularly if the program is written in a high-level language, the program size exceeds 4K bytes, and an external program memory is needed. Again, the manufacturers provide a version for the job, the ROM less 8031. The EA pin is grounded when using the 8031, and all program code is contained in an external EPROM that may be as large as 64K bytes and that can be programmed using standard EPROM programmers.
External RAM, which is accessed by the DPTR, may also be needed when 128 bytes of internal data storage is not sufficient. External RAM, up to 64K bytes,

may also be added to any chip in the 8051 family.

Connecting External Memory

Figure 2.8 shows the connections between an 8031 and an external memory configuration consisting of I6K bytes of EPROM and 8K bytes of static RAM. The 8051 accesses external RAM whenever certain program instructions are executed. External ROM is accessed whenever the EA (external access) pin is connected to ground or when the PC contains an address higher than the last address in the internal 4K bytes ROM (OFFFh). 8051 designs can thus use internal and external ROM automatically; the 8031, having no internal ROM, must have EA grounded.

Figure 2.9 shows the timing associated with an external memory access cycle. During any memory access cycle, port 0 is time multiplexed. That is, it first provides the lower byte of the 16-bit memory address, then acts as a bidirectional data bus to write or read a byte of memory data. Port 2 provides the high byte of the memory address during the entire memory read/write cycle.
The lower address byte from port 0 must be latched into an external register to Savethe byte. Address byte save is accomplished by the ALE clock pulse that provides the correct timing for the '373 type data latch. The port 0 pins then become free to serve as a data bus.

If the memory access is for a byte of program code in the ROM, the PSEN (program store enable) pin will go low to enable the ROM to place a byte of program code on the data bus. If the access is for a RAM byte, the WR (write) or RD (read) pins will go low, enabling data to flow between the RAM and the data bus.

The ROM may be expanded to 64K by using a 27512 type EPROM and connecting the remaining port 2 upper address lines AI4-A15 to the chip.
At this time the largest static RAMs available are 32K in size; RAM can be expanded to 64K by using two 32K RAMs that are connected through address A14 of port 2. The first 32K RAM (OOOOh-7FFFh) can then be enabled when AI5 of port 2 is low, and the second 32K RAM (SOOOh-FFFFh) when A15 is high, by using an inverter.
Note that the WR and RD signals are alternate uses for port 3 pins 16 and 17.
Also,
port 0 is used for the lower address byte and data; port 2 is used for upper address bits. The use of external memory consumes many of the port pins, leaving only port 1 and parts of port 3 for general I/O.

FIGURE 2.8 External Memory Connections

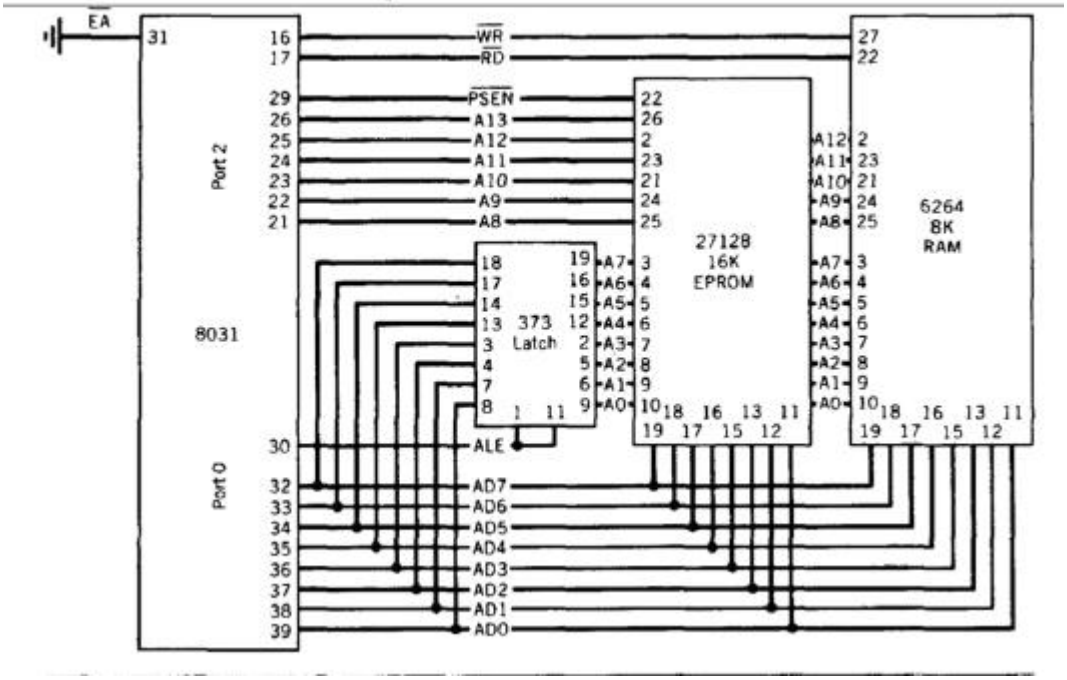
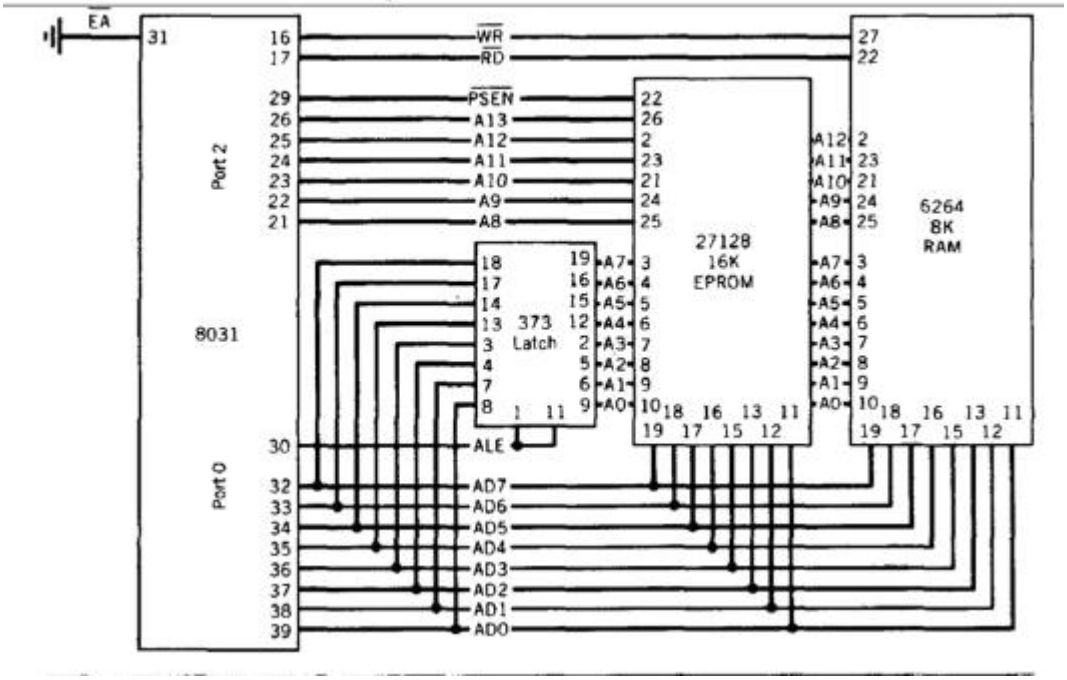


FIGURE 2.8 External Memory Connections



8051 INSTRUCTION SET

8051 has about 111 instructions. These can be grouped into the following categories

- Arithmetic Instructions
- Logical Instructions
- Data Transfer instructions

 Boolean Variable Instructions
 Program Branching Instructions

The following nomenclatures for register, data, address and variables are used while write instructions
 A: Accumulator
 B: "B" register
 C: Carry bit

- Rn: Register R0 - R7 of the currently selected register bank

- Direct: 8-bit internal direct address for data. The data could be in lower 128bytes of RAM (00 - 7FH) or it could be in the special function register (80 - FFH).

- @Ri: 8-bit external or internal RAM address available in register R0 or R1. This is used for indirect addressing mode.

- #data8: Immediate 8-bit data available in the instruction.

- #data16: Immediate 16-bit data available in the instruction.

- Addr11: 11-bit destination address for short absolute jump. Used by instructions AJMP& ACALL. Jump range is 2 kbyte (one page).

- Addr16: 16-bit destination address for long call or long jump.

- Rel: 2's complement 8-bit offset (one - byte) used for short jump (SJMP) and all conditional jumps.

- bit: Directly addressed bit in internal RAM or SFR

Some Simple Instructions:

```
MOV dest,source ; dest = source
MOV A,#72H ;     A=72H
MOV R4,#62H ;   R4=62H
MOV B,0F9H ;     B=the content of F9’th byte of RAM
MOV DPTR,#7634H
MOV DPL,#34H
```

```
MOV DPH,#76H
MOV P1,A ;      mov A to port 1
```

Note 1:
MOV A,#72H ≠ MOV A,72H
After instruction "MOV A,72H " the content of 72'th byte of RAM will replace in Accumulator.

Note 2:

```
MOV A,R3 ≡ MOV A,3
ADD A, Source ;   A=A+SOURCE
ADD A,#6 ;        A=A+6
ADD A,R6 ;        A=A+R6
ADD A,6 ;         A=A+[6] or A=A+R6
ADD A,0F3H ;      A=A+[0F3H]
SUBB A, Source ;  A=A-SOURCE-C
SUBB A,#6 ;       A=A-6
SUBB A,R6 ;       A=A+R6
```

MUL & Div:

• MUL AB ; B|A = A*B

```
MOV A,#25H
MOV B,#65H
MUL AB ;   25H*65H=0E99
;B=0EH, A=99H
```

• DIV AB ;A = A/B, B = A mod B

```
MOV A,#25
MOV B,#10
DIV AB ;A=2, B=5
SETB bit ; bit=1
CLR bit ; bit=0
SETB C ; CY=1
SETB P0.0 ;bit 0 from port 0 =1
SETB P3.7 ;bit 7 from port 3 =1
SETB ACC.2 ;bit 2 from ACCUMULATOR =1
SETB 05 ;set high D5 of RAM loc. 20h
```

Note:

CLR instruction is as same as
SETB i.e.:

```
CLR C ;CY=0
```

But following instruction is only for CLR:

CLR A ;A=0
DEC byte ;byte=byte-1
INC byte ;byte=byte+1
INC R7
DEC A
DEC 40H ; [40]=[40]-1

RR – RL – RRC – RLC A

EXAMPLE:
RR A

RR:

RRC:

RL:

RLC

ANL - ORL – XRL
Bitwise Logical Operations:
AND, OR, XOR
EXAMPLE:
MOV R5,#89H
ANL R5,#08H
CPL A ;1's complement
Example:
MOV A,#55H ;A=01010101 B
L01: CPL A
MOV P1,A
ACALL DELAY
SJMP L01

8051 Real Time Control:

- Programming Timer Interrupts
- Programming External Hardware
- Interrupts
- Programming the Serial Communication Interrupts
- Programming 8051 Timers and Counters

Interrupts:

1. Enabling and Disabling Interrupts
2. Interrupt Priority
3. Writing the ISR (Interrupt Service Routine)

Interrupt Enable (IE) Register :

• EA : Global enable/disable.
• --- : Undefined.

• ET2 :Enable Timer 2 interrupt.
• ES :Enable Serial port interrupt.
• ET1 :Enable Timer 1 interrupt.
• EX1 :Enable External 1 interrupt.
• ET0 : Enable Timer 0 interrupt.
• EX0 : Enable External 0 interrupt.

EA	—	ET2	ES	ET1	EX1	ET0	EX0

Interrupt Vectors:

Interrupt	Vector Address
System Reset	0000H
External 0	0003H
Timer 0	000BH
External 1	0013H
Timer 1	001BH
Serial Port	0023H
Timer 2	002BH

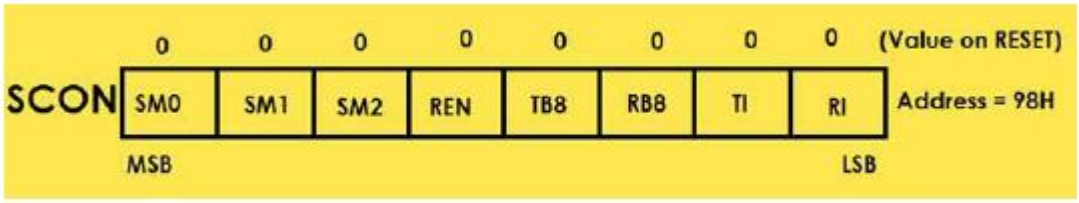
Peripheral Control Registers
PCON (Power Control)

The PCON or Power Control register, as the name suggests is used to control the 8051 Microcontroller's Power Modes and is located at 87H of the SFR Memory Space. Using two bits in the PCON Register, the microcontroller can be set to Idle Mode and Power Down Mode. During Idle Mode, the Microcontroller will stop the Clock Signal to the ALU (CPU) but it is given to other peripherals like Timer, Serial, Interrupts, etc. In order to terminate the Idle Mode, you have to use an Interrupt or Hardware Reset.
In the Power Down Mode, the oscillator will be stopped and the power will be reduced to 2V. To terminate the Power Down Mode, you have to use the Hardware Reset.
Apart from these two, the PCON Register can also be used for few additional purposes. The SMOD Bit in the PCON Register is used to control the Baud Rate of the Serial Port.
There are two general purpose Flag Bits in the PCON Register, which can be used by the programmer during execution.

SCON (Serial Control)

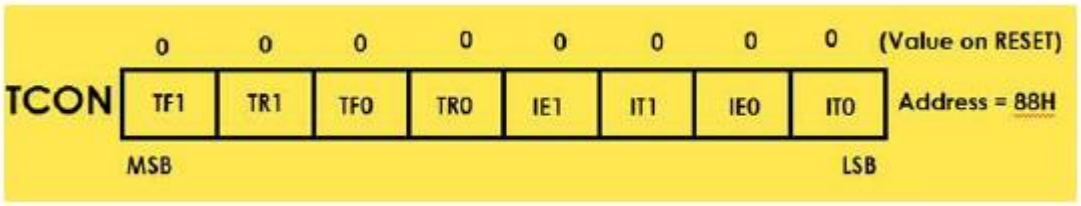
The Serial Control or SCON SFR is used to control the 8051 Microcontroller's Serial Port. It is located as an address of 98H. Using SCON, you can control the Operation Modes of the Serial Port, Baud Rate of the Serial Port and Send or Receive Data using Serial Port.

SCON Register also consists of bits that are automatically SET when a byte of data is transmitted or received.



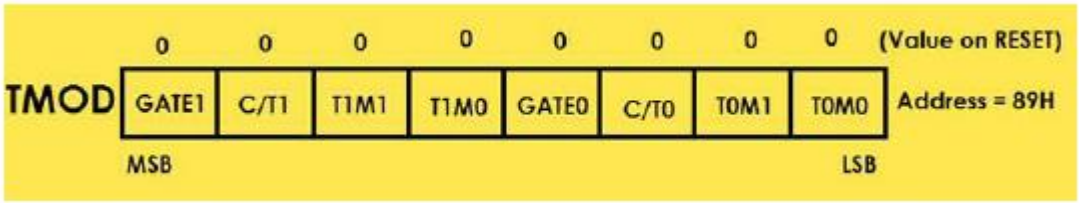
TCON (Timer Control)

Timer Control or TCON Register is used to start or stop the Timers of 8051 Microcontroller. It also contains bits to indicate if the Timers has overflowed. The TCON SFR also consists of Interrupt related bits.



TMOD (Timer Mode)

The TMOD or Timer Mode register or SFR is used to set the Operating Modes of the Timers T0 and T1. The lower four bits are used to configure Timer0 and the higher four bits are used to configure Timer1.



The Gatex bit is used to operate the Timerx with respect to the INTx pin or regardless of the INTx pin.

GATE1 = 1 ==> Timer1 is operated only if INT1 is SET.

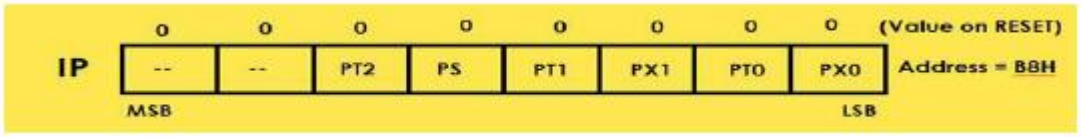
GATE1 = 0 ==> Timer1 is operates irrespective of INT1 pin.

GATE0 = 1 ==> Timer0 is operated only if INT0 is SET.
GATE0 = 0 ==> Timer0 is operates irrespective of INT0 pin.
The C/Tx bit is used selects the source of pulses for the Timer to count.
C/T1 = 1 ==> Timer1 counts pulses from Pin T1 (P3.5) (Counter Mode)
C/T1 = 0 ==> Timer1 counts pulses from internal oscillator (Timer Mode)

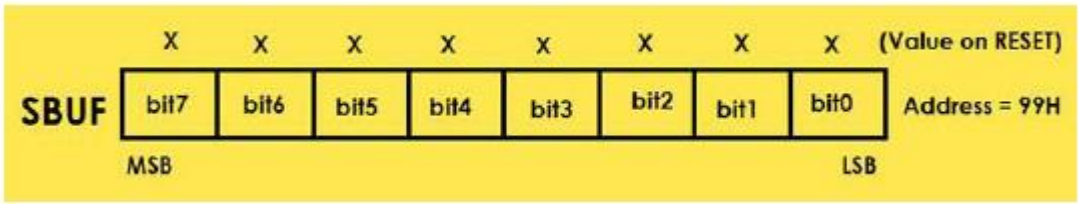
C/T0 = 1 ==> Timer0 counts pulses from Pin T0 (P3.4) (Counter Mode)
C/T0 = 0 ==> Timer0 counts pulses from internal oscillator (Timer Mode)

TxM0	TxM1	Mode	Description
0	0	0	13-bit Timer Mode (THx – 8-bit and TLx – 5-bit)
0	1	1	16-bit Timer Mode
1	0	2	8-bit Auto Reload Timer Mode
1	1	3	Two 8-bit Timer Mode or Split Timer Mode

IP (Interrupt Priority)
The IP or Interrupt Priority Register is used to set the priority of the interrupt as High or Low. If a bit is CLEARED, the corresponding interrupt is assigned low priority and if the bit is SET, the interrupt is assigned high priority.



Peripheral Data Registers
SBUF (Serial Data Buffer)
The Serial Buffer or SBUF register is used to hold the serial data while transmission or reception.



ADDRESSING MODES OF 8051 :

The way in which the data operands are accessed by different instructions is known as the addressing modes. There are various methods of denoting the data operands in the instruction. The 8051 microcontroller supports mainly 5 addressing modes. They are

1.Immediate addressing mode

2.Direct Addressing mode

3.Register addressing mode

4. Register Indirect addressing mode

5.Indexed addressing mode

Immediate addressing mode : The addressing mode in which the data operand is a constant and it is a part of the instruction itself is known as Immediate addressing mode. Normally the data must be preceded by a # sign. This addressing mode can be used to transfer the data into any of the registers including DPTR.

Ex: MOV A , # 27 H : The data (constant) 27 is moved to the accumulator register

ADD R1 ,#45 H : Add the constant 45 to the contents of the accumulator

MOV DPTR ,# 8245H :Move the data 8245 into the data pointer register.

MOV P1,#21 H

Direct addressing mode: The addressing mode in which the data operand is in the RAM location (00 -7FH) and the address of the data operand is given in the instruction is known as Direct addressing mode. The direct addressing mode uses the lower 128 bytes of Internal RAM and the SFRs

MOV R1, 42H : Move the contents of RAM location 42 into R1 register

MOV 49H,A : Move the contents of the accumulator into the RAM location 49.

ADD A, 56H : Add the contents of the RAM location 56 to the accumulator

Register addressing mode :The addressing mode in which the data operand to be manipulated lies in one of the registers is known as register addressing mode.

MOV A,R0 : Move the contents of the register R0 to the accumulator

ADD A,R6 :Add the contents of R6 register to the accumulator

MOV P1, R2 : Move the contents of the R2 register into port 1

MOV R5, R2 : This is invalid .The data transfer between the registers is not allowed.

Register Indirect addressing mode : The addressing mode in which a register is used as a pointer to the data memory block is known as Register indirect addressing mode.

MOV A,@ R0 :Move the contents of RAM location whose address is in R0 into **A** (accumulator)

MOV @ R1 , B : Move the contents of B into RAM location whose address is held by R1

When R0 and R1 are used as pointers, they must be preceded by @ sign

One of the advantages of register indirect addressing mode is that it makes accessing the data more dynamic than static as in the case of direct addressing mode.

Indexed addressing mode : This addressing mode is used in accessing the data elements of lookup table entries located in program ROM space of 8051.

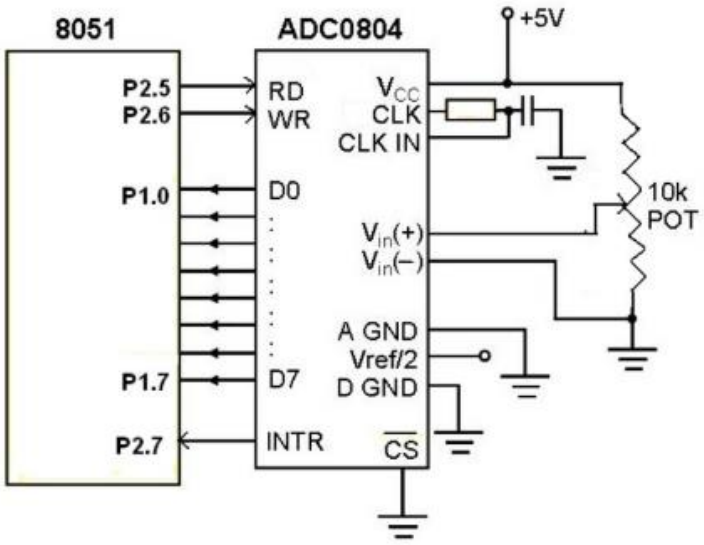
Ex : MOVC A,@ A+DPTR

The 16-bit register DPTR and register A are used to form the address of the data element stored in on-chip ROM. Here C denotes code .In this instruction the contents of A are added to the 16-bit DPTR register to form the 16-bit address of the data operand.

Interfacing of ADC 0804 to 8051 Microcontroller :

ADC 0804 is a single channel analog to digital converter i.e., it can take only one analog signal. ADC 0804 has 8 bit resolution. The higher resolution ADC gives smaller step size. Step size is smallest change that can be measured by an ADC. For an ADC with resolution of 8 bits, the step size is 19.53mV (5V/255). The time taken by the ADC to convert analog data into digital form depends on the frequency of clock source. The conversion time of ADC 0804 is around 110us. To use the internal clock a capacitor and resistor are used as shown in the circuit. The input to the ADC is given from a regulated power supply and a 10K potentiometer

The 8051 Microcontroller is used to provide the control signals to the ADC. CS(chip select) pin of ADC is directly connected to ground. The pin P1.1, P1.0 and P1.2 are connected to the pin WR, RD and INTR of the ADC respectively. When the input voltage from the preset is varied the output of ADC varies also varies.



From the circuit it is clear that the ADC interfaced directly to the microcontroller. The Port1 is used as an input port which receives the digital data from the ADC.Port pins P2.5 and P2.6 are used for SOC and EOC operation.When the conversion is over the ADC will send an interrupt signal to the microcontroller through the pin P2.7 .Now the Microcontroller receives digital data through the Port1.This data after conversion to decimal data is displayed on the LCD module .

The assembly language program for ADC is given below .

```
        MOV P1 , 0FF H  ; Make the port1 high and configure port1 as Input port
BACK:   CLR P2.6        ; Generation of SOC pulse
        SETB P2.5       ;
LOOP    JB P2.7 , LOOP  ; Wait for conversion, Is conversion over?
        CLR  P2.5       ; Enable Read the digital data

MOV A ,P1       ; Read digital data through Port1
SETB P2.5       ; Disable read after read operation
CALL DISPLAY    ; Display the data on LCD module
SJMP BACK       ; Continue the conversion process
```

INTERFACING DAC -8051 MICROCONTROLLER

The DAC 0800 is a simple monolithic 8-bit D/A converter. It has fast settling time of 100ns. It can be directly interfaced to TTL, CMOS, PMOS and others. It operates at 4.5V to +18V supply. The number of data bit inputs decides the resolution of the DAC since the number of analog output levels is equal to 2^n, 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.

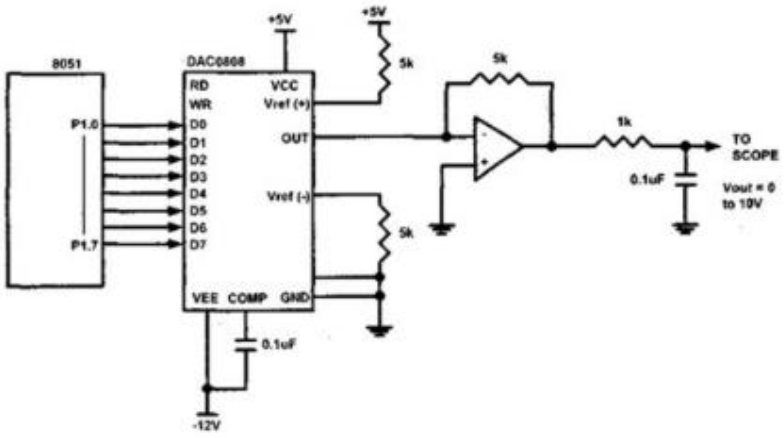
The interfacing circuit is shown below. port 1(8 bits of the microcontroller is connected to the input data lines of DAC-08.The reference current is determined by the resistor R_1 and the reference voltage V_{ref}. The resistor R_2 is generally equal to R1 to match the input impedance of

reference source. The output (taken from pin number 4 is observed either on a digital multimeter or on a cathode ray oscilloscope.

The output current Io is calculated as follows:

$$I_o = V_{ref}/R_1[A_o/2 + A_1/4 + A_2/8 + \ldots + A7/256]$$

The output voltage Vo is obtained as follows: $V_o = I_o * R_1$



Assembly Language Program

```
        MOV     A, #DATA*   ; (A) = #Data
START : MOV     90H, A      ; (port -1) = (A)
        INC A
        LJMP    START       ; Repeat
```