4.5 ADDRESSING MODES

CPU access data from memory, register or as immediate value**.** These various way of accessing the data is called addressing mode.

The 8051 micro-controller has five addressing modes.

1. Immediate
2. Register
3. Direct
4. Register indirect
5. Indexed

IMMEDIATE

In this addressing mode, operand is an immediate data revealed in the instruction itself. It must be preceded by "#".

Examples:

```
1. MOV A, #32H              ; Load32h into A register.
2. MOV DPTR, #1234H         ; Load 1234h into DPTR.
```

REGISTER:

In this mode, registers are used to hold data to be manipulated.

Size of destination register and source register are always equal.

The data between to registers Run (n=0 to 7) and Run (n=0 to7) cannot be transferred.

Examples:

```
1. MOV A, R0                ; copy content of R0 into A.
2. MOV R7,A                 ; copy content of A into R7
```

DIRECT:

In this mode, data can be accessed directly from RAM location. We can access 128 Byte RAM directly. Stack operations use this addressing mode.

Examples:

```
1. MOV R0, 30H          ; save content of RAM location 30h into R0.
2. PUSH OEOH            ; Push content of Accumulator into stack
```

REGISTER INDIRECT:

In this mode, registers R0 and R1 are used as pointer to access the data.

- Only register R0 and R1 used in this mode.
- R0 and R1 hold address of RAM location.
- Both registers must be preceded by "@"sign.
- Looping is easily used in this mode.

Examples:

1. MOV A, @R0 ; move content of RAM location whose address is stored in R0.
2. MOV @R1, A ; move content of A into RAM location whose address is stored in R1.

INDEXED:

In this mode, data can be accessed from look up table located in the program memory ROM space of 8051or from external RAM. Register A and DPTR used in this addressing mode.

Examples:

1. MOVC A, @A+DPTR ; Here instruction MOVC is used instead of MOV Because C indicate Code memory (ROM).
2. MOVX @DPTR, A ; Here, X represents external RAM.

4.1 ARCHITECTURE OF 8051

MICROCONTROLLER EVOLUTION

First, microcontrollers were developed in the mid-1970s. These were basically calculator- based processors with small ROM program memories, very limited RAM data memories and a handful of input/output ports.

As silicon technology developed, more powerful, 8-bit microcontrollers were produced. In addition to their improved instruction sets, these microcontrollers included on-chip counter/timers, interrupt facilities, and improved I/O handling. On-chip memory capacity was still small and was not adequate for many applications. One of the most significant developments at this time was the availability of on-chip ultraviolet erasable EPROM memory. This simplified the product development time considerably and for the first time, also allowed the use of microcontrollers in low-volume applications.

The 8051 family was introduced in the early 1980s by Intel. Since its introduction, the 8051 has been one of the most popular microcontrollers a d has been second- sourced by many manufacturers. The 8051 currently has many different versions and some types include on-chip analogue-to-digital converters, a consider ably large size of program and data memories.

INTRODUCTION TO 8051:

The **Intel MCS-51** (commonly referred to as **8051**) is a Harvard architecture, single chip microcontroller (act) series which was developed by Intel in 1980 for use in embedded systems. The 8051 architecture provides many functions (CPU, RAM, ROM, I/O, interrupt logic, timer, etc.) in a single package

Features of 8051:

- 8-bit ALU, Accumulator, 8-bit Registers and 8-bit data bus; hence it is an 8-bit microcontroller
- 16-bit program counter
- 8-bit Processor Status Word(PSW)
- 8-bit Stack Pointer

- Internal RAM of128bytes
- On chip ROM is4KB
- Special Function Registers (SFRs) of 128bytes
- 32 I/O pins arranged as four 8-bit ports (P0 -P3)
- Two 16-bit timer/counters : T0 andT1
- Two external and three internal vectored interrupts
- Full duplex UART (serial port)

BLOCK DIAGRAM

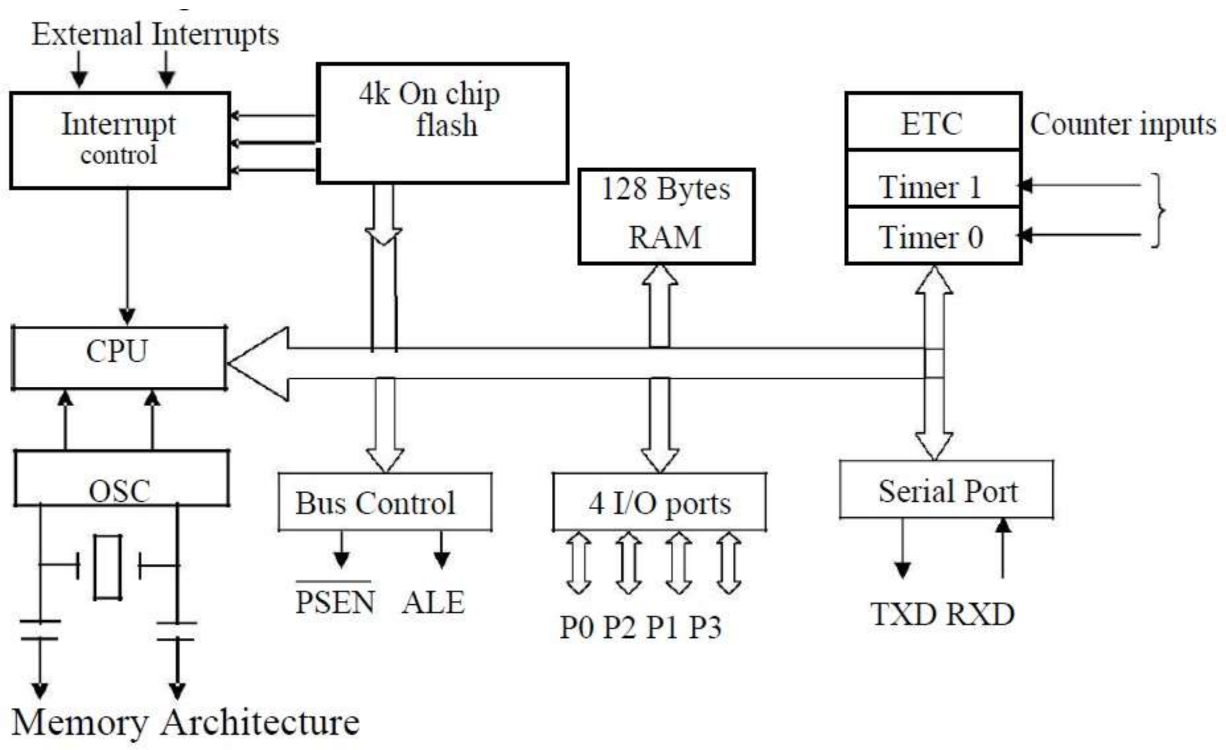


Figure 4.1.1 8051 Microcontroller Block Diagram

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, pg.no.29]

128 BYTES OF INTERNAL RAM STRUCTURE (LOWER ADDRESS SPACE)

30H-7FH	**Scratch Pad**	**80 Bytes**
20H -2FH	**Bit Addressable RAM**	**16 Bytes**
1FH	R7	
	REGISTER BANK 3	
18H	R0	
17H	R7	
	REGISTER BANK 2	**32 Bytes**
10H	R0	
0FH	R7	
	REGISTER BANK 1	
08H	R0	
07H	R7	
	REGISTER BANK 0	
00H	R0	

Figure 4.1.2 RAM Allocation in 8051

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley]

The lower 32 bytes are divided into 4 separate banks. Each register bank has 8 registers of one byte each. A register bank is selected depending upon two bank select bits in the PSW register as shown in Figure 4.1.2.

- Next 16 bytes are bit addressable. In total, 128bits (16X8) are available in addressable area. Each bit can be accessed and modified by suitable instructions. The bit addresses are from 00H (LSB of the first byte in 20H) to 2FH (MSB of the last byte in 2FH).
- Remaining 80bytes of RAM (30H TO 7FH) are available for general purpose.

INTERNAL ARCHITECTURE OF 8051 MICROCONTROLLER

The Internal architecture is shown in Figure 4.1.4 and the various Registers and units are described below.

Accumulator (Act):

•Operand register

• Implicit or specified in the instruction

•Has an address in on chip SFR bank

B Register:

Used to store one of the operands for multiplication and division, otherwise, scratch pad considered as a SFR.

Stack Pointer (SP):

8 bit wide register. Incremented before data is stored on to the stack using PUSH or CALL instructions. Stack defined anywhere on the 128 byte RAM.

Data Pointer (DPTR)*:*

*1*6 bit register contains DPH and DPL Pointer to external RAM address. DPH and DPL allotted separate addresses in SFR bank

Port 0 To 3 Latches & Drivers:

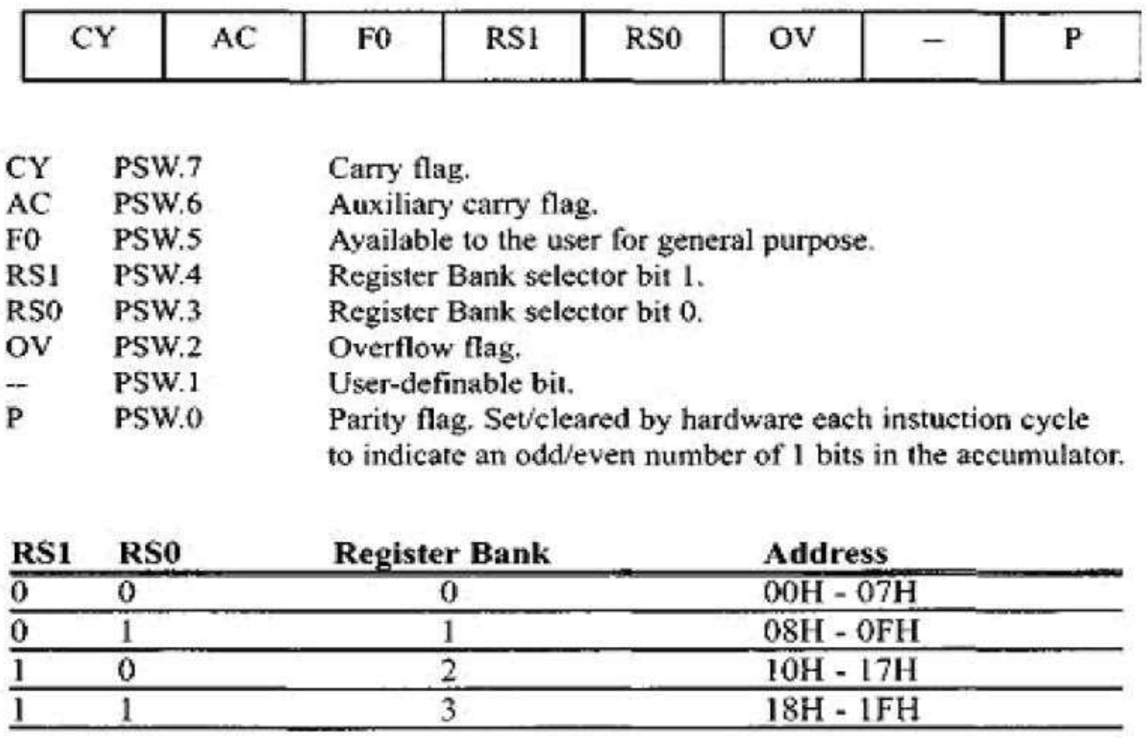
Each I/O port allotted a latch and a driver Latches allotted address in SFR. User can communicate via these ports P0, P1, P2, and P3.

Serial Data Buffer:

Internally had TWO independent registers, TRANSMIT buffer (parallel in serial out – PISO) and RECEIVE buffer (serial in parallel out –SIPO) identified by SBUF and Allotted an address in SFR.

Program Status Word (PSW):

Set of flags contains status information as detailed below in the Figure 4.1.3.



CY	AC	F0	RS1	RS0	OV	–	P

CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary carry flag.
F0	PSW.5	Available to the user for general purpose.
RS1	PSW.4	Register Bank selector bit 1.
RS0	PSW.3	Register Bank selector bit 0.
OV	PSW.2	Overflow flag.
--	PSW.1	User-definable bit.
P	PSW.0	Parity flag. Set/cleared by hardware each instuction cycle to indicate an odd/even number of 1 bits in the accumulator.

RS1	RS0	Register Bank	Address
0	0	0	00H - 07H
0	1	1	08H - 0FH
1	0	2	10H - 17H
1	1	3	18H - 1FH

Figure 4.1.3 Bits of PSW Register

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, pg.no.52]

Timer Registers: for Timer0 (16 bit register – TL0 & TH0) and for Timer1 (16 bit register – TL1 & TH1) four addresses allotted in SFR

Control Registers: Control registers are IP, IE, TMOD, TCON, SCON, and PCON. These registers contain control and status information for interrupts, timers/counters and serial port. Allotted separate address in SFR.

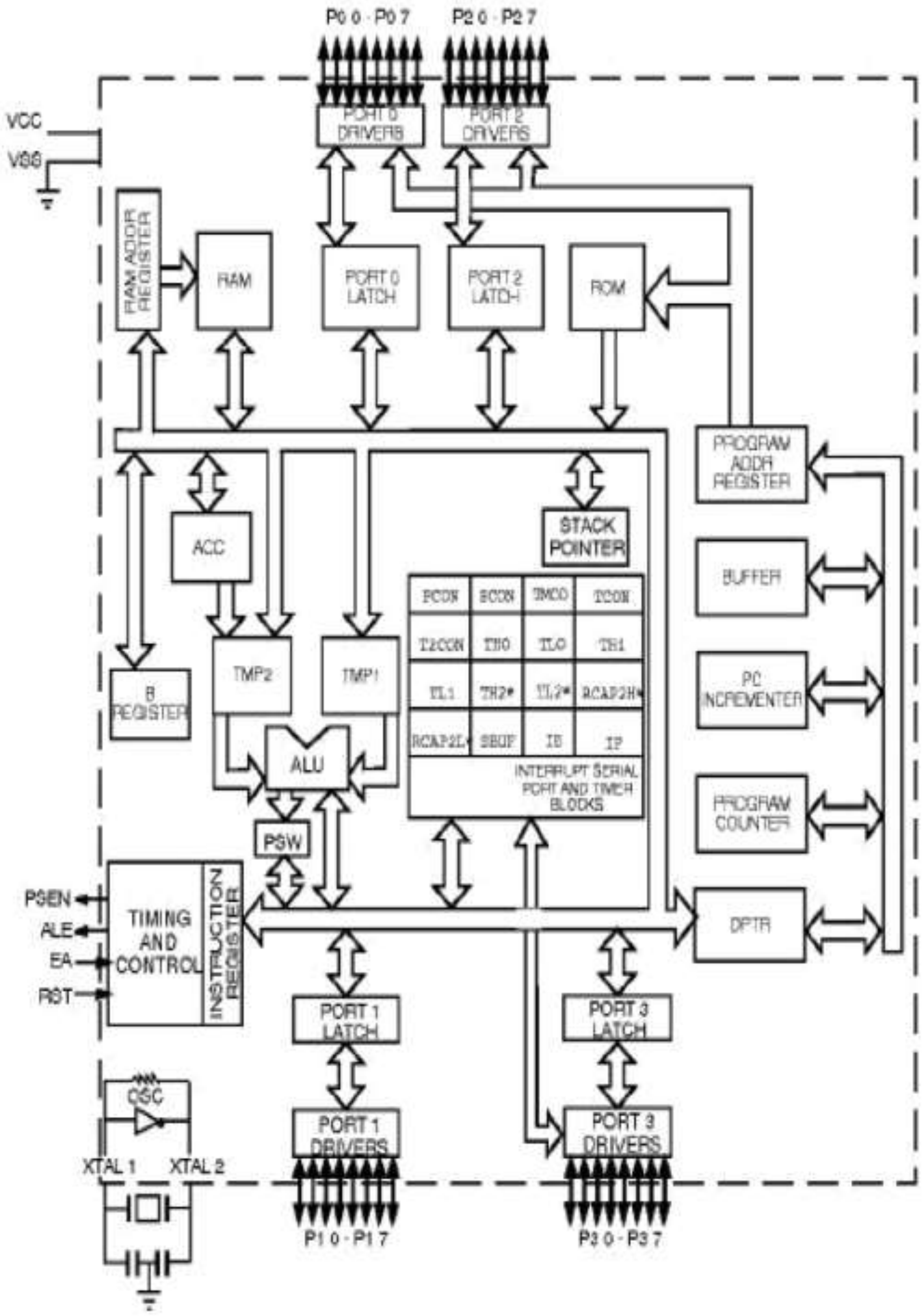
Timing and Control Unit: This unit derives necessary timing and control signals for internal circuit and external system bus.

Oscillator: generates basic timing clock signal using crystal oscillator.

Instruction Register: Decodes the oppose and gives information to timing and control unit.

Figure 4.1.4 8051 Architecture Block diagram

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali



Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.29]

EPROM & program address Register: provide on chip EPROM and mechanism to address it. All versions don't have EPROM.

Ram & Ram Address Register: provide internal 128 bytes RAM and a mechanism to address internally

ALU: Performs 8 bit arithmetic and logical operations over the operands held by TEMP1 and TEMP 2.User cannot access temporary registers.

SFR Register Bank: set of special function registers address range: 80 H to FF H. Interrupt, serial port and timer units control and perform specific functions under the control of timing and control unit.

8051 PIN CONFIGURATION

The pin diagram of 8051 microcontroller is shown in Figure 4.1.5 and the pin details are described below.

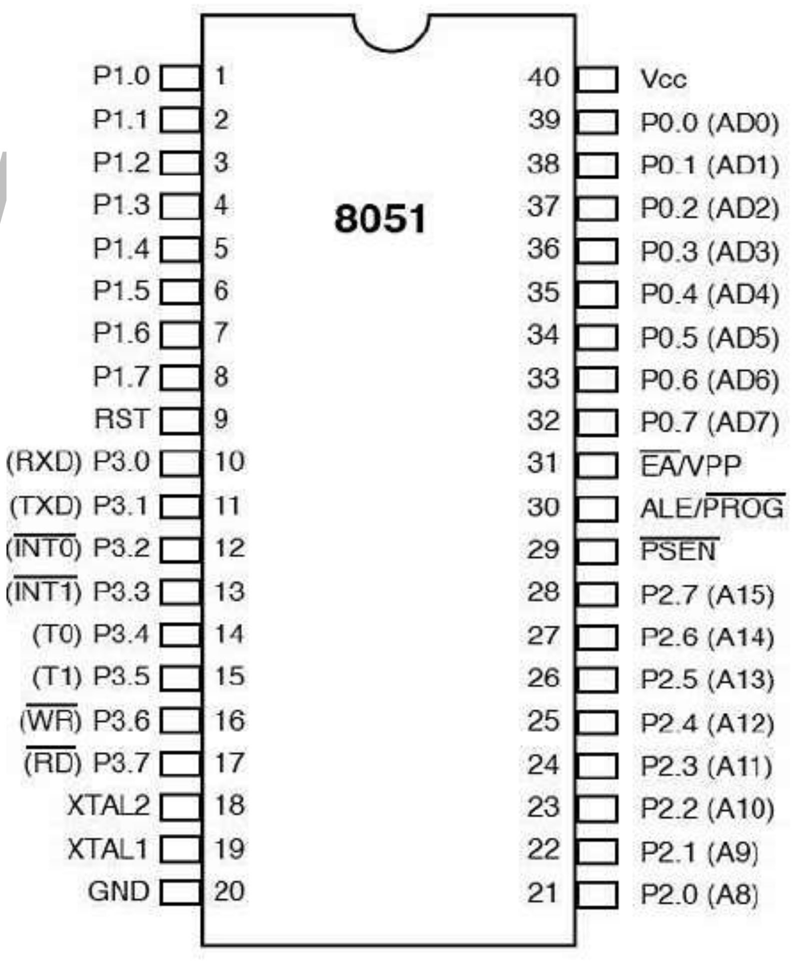


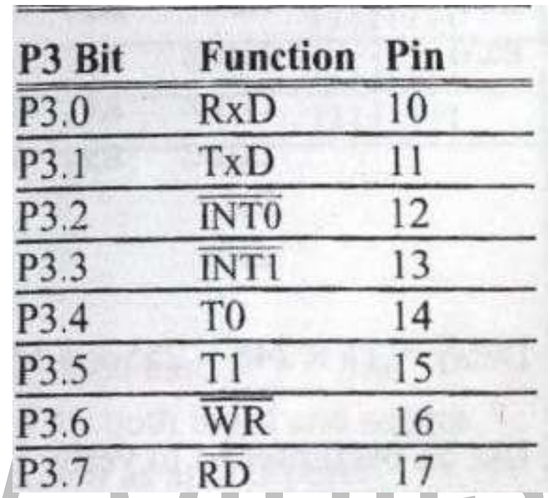
Figure 4.1.5 8051 Pin Configuration

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, pg.no.94]

Pins 1 to 8 − these pins are known as Port 1. This port doesn't serve any other functions. It is internally pulled up, bi-directional I/O port.

Pin 9 − It is a RESET pin, which is used to reset the microcontroller to its initial values.

Pins 10 to 17 − these pins are known as Port 3. This port serves some alternate functions like interrupts, timer input, control signals, serial communication signals Rd. and TX, etc.



P3 Bit	Function	Pin
P3.0	RxD	10
P3.1	TxD	11
P3.2	$\overline{\text{INT0}}$	12
P3.3	$\overline{\text{INT1}}$	13
P3.4	T0	14
P3.5	T1	15
P3.6	$\overline{\text{WR}}$	16
P3.7	$\overline{\text{RD}}$	17

Figure 4.1.6 Port 3 Alternate Functions

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, pg.no.97]

Pins 18 & 19 − these pins are used for interfacing an external crystal to get the system clock.

Pin 20 − this pin provides the power supply to the circuit.

Pins 21 to 28 − these pins are known as Port 2. It serves as I/O port. Higher order address bus signals are also multiplexed using this port.

Pin 29 − this is PSEN pin which stands for Program Store Enable. It is used to read a signal from the external program memory.

Pin 30 − this is EA pin which stands for External Access input. It is used to enable/disable the external memory interfacing.

Pin 31 – this is ALE pin which stands for Address Latch Enable. It is used to DE multiplex the address-data signal of port.

Pins 32 to 39 – these pins are known as Port 0. It serves as I/O port. Lower order address and data bus signals are multiplexed using this port.

Pin 40 – this pin is used to provide power supply to the circuit.

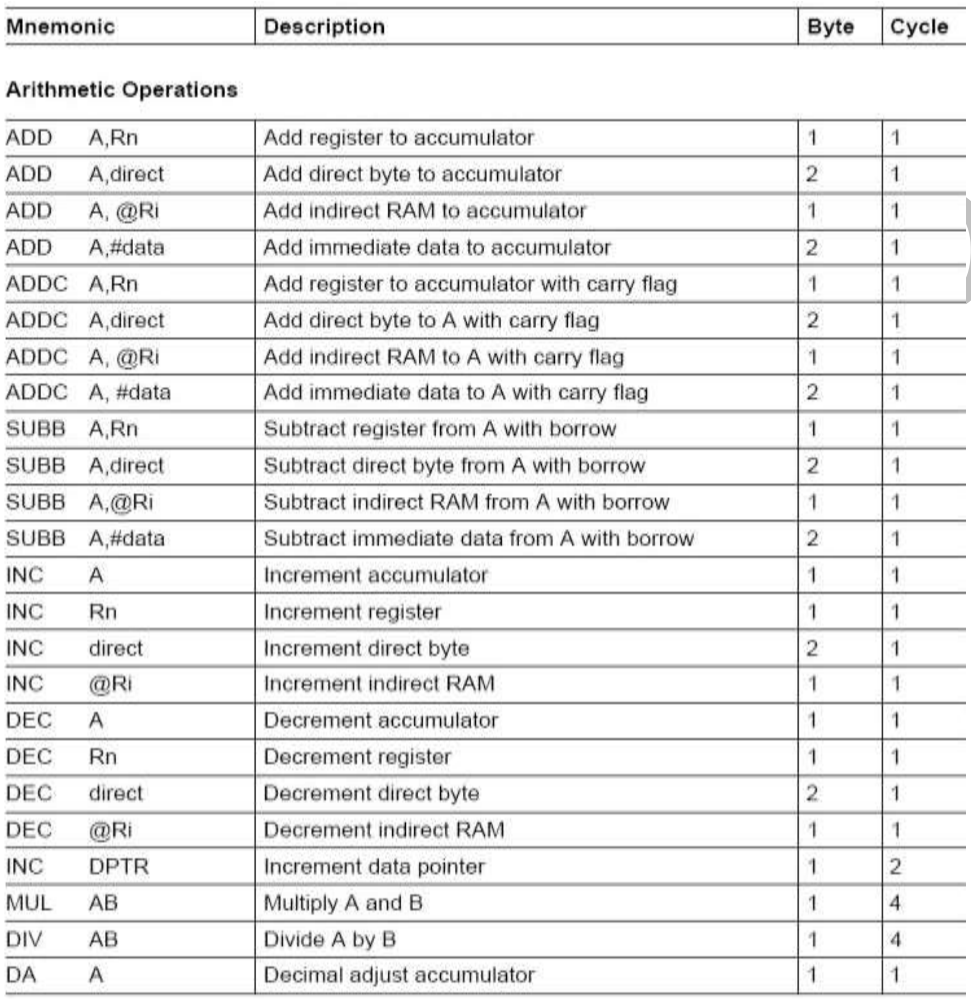
4.4 INSTRUCTION SET

8051 Microcontroller have set of instruction to perform different operations. There are five group of instruction which are listed below.

- Arithmetic Instructions
- Logic Instructions
- Data Transfer Instructions
- Branch Instructions
- Bit-oriented Instructions

1. ARITHMETIC INSTRUCTION:

Arithmetic instructions perform several basic operations such as addition, subtraction, division, multiplication etc. After execution, the result is stored in the first operand.



Mnemonic		Description	Byte	Cycle
Arithmetic Operations				
ADD	A,Rn	Add register to accumulator	1	1
ADD	A,direct	Add direct byte to accumulator	2	1
ADD	A, @Ri	Add indirect RAM to accumulator	1	1
ADD	A,#data	Add immediate data to accumulator	2	1
ADDC	A,Rn	Add register to accumulator with carry flag	1	1
ADDC	A,direct	Add direct byte to A with carry flag	2	1
ADDC	A, @Ri	Add indirect RAM to A with carry flag	1	1
ADDC	A, #data	Add immediate data to A with carry flag	2	1
SUBB	A,Rn	Subtract register from A with borrow	1	1
SUBB	A,direct	Subtract direct byte from A with borrow	2	1
SUBB	A,@Ri	Subtract indirect RAM from A with borrow	1	1
SUBB	A,#data	Subtract immediate data from A with borrow	2	1
INC	A	Increment accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
INC	@Ri	Increment indirect RAM	1	1
DEC	A	Decrement accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1
DEC	@Ri	Decrement indirect RAM	1	1
INC	DPTR	Increment data pointer	1	2
MUL	AB	Multiply A and B	1	4
DIV	AB	Divide A by B	1	4
DA	A	Decimal adjust accumulator	1	1

[Source: Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, "The 8051Microcontroller and Embedded Systems: Using Assembly and C", Second Edition, Pearson Education,2011]

1. ADD A,Rn;

Adds the register Rn to the accumulator

Description:

Instruction adds the register Run (R0-R7) to the accumulator. After addition, the result is stored in the accumulator

Before execution:

A=2Eh R4=12h

After execution:

A=40h R4=12h

2. ADD A,@RI-

Adds the indirect RAM to the accumulator.

RI: Register R0 or

R1 Description:

Instruction adds the indirect RAM to the accumulator. Address of indirect RAM is stored in the RI register (R0 or R1). After addition, the result is stored in the accumulator.

Register address:

R0=4Fh

Before execution:

A= 16h SUM= 33h

After execution:

A= 49h

3. ADDA, #DATA

Data: constant within 0-255 (0-FFh)

Description:

Instruction adds data (0-255) to the accumulator. After addition, the result is stored in the accumulator.

ADD A, #33h

Before execution:

A= 16h

After execution:

A= 49h)

4. SUBB A,direct-

Subtracts the direct byte from the accumulator witha borrow

Direct: arbitrary register with address 0-255(0-FFh)

Description:

Instruction subtracts the direct byte from the accumulator with a burr own. If the higher bit is subtracted from the lower bit then the carry flag is set. As it is direct addressing, the direct byte can be any SFRs or general-purpose register with address 0-7Fh. (0-127 Dec.). The result is stored in the accumulator.

SUBB Air

Before execution:

A=C9h, DIF=53h, C=0

After execution:

A=76h, C=0

5. INC A - Increments the accumulator by1

Description:

This instruction increments the value in the accumulator by 1. If the accumulator includes the number 255, the result of the operation will be 0.

Before execution:

A=E4h

After execution:

A=E5h

6. DEC A - Decrements the accumulator by1

Description: Instruction decrements the value in the accumulator by 1. If there is a 0 in the accumulator, the result of the operation is Fifth. (255 dec.)

Syntax: DEC A;

Byte: 1 (instruction code);

STATUS register flags:

No flags are affected;

Before execution:

A=E4h

After execution:

A=E3h

7. DIV AB - Divides the accumulator by the registerB

Description:

Instruction divides the value in the accumulator by the value in the B register. After division the integer part of result is stored in the accumulator while the register contains the remainder. In case of dividing by 1, the flag OV is set and the result of division is unpredictable. The 8-bit quotient is stored in the accumulator and the 8-bit remainder is stored in the B register.

Before execution:

A=Fibs (251dec.) B=12h (18 Dec.)

After execution:

A=0Dh (13dec.) B=11h (17dec.)

13·18 + 17 =251

8. DA A - Decimal adjust accumulator

Description: Instruction adjusts the contents of the accumulator to correspond to a BCD number after two BCD numbers have been added by the ADD and ADDC instructions. The result in form of two 4-digit BCD numbers is stored in the accumulator.

Before execution:

A=56h (01010110) 56BCD

B=67h (01100111)67BCD

After execution:

A=Bodh (10111101)

After BCD conversion:

A=23h (00100011), C=1 (Overflow)

(C+23=123) = 56+67

9. MUL AB - Multiplies A and

Description: Instruction multiplies the value in the accumulator with the value in the B register. The low-order byte of the 16-bit result is stored in the accumulator, while the high byte remains in the B register. If the result is larger than 255, the overflow flag is set. The carry flag is not affected.

Before execution:

A=80 (50h) B=160 (A0h)

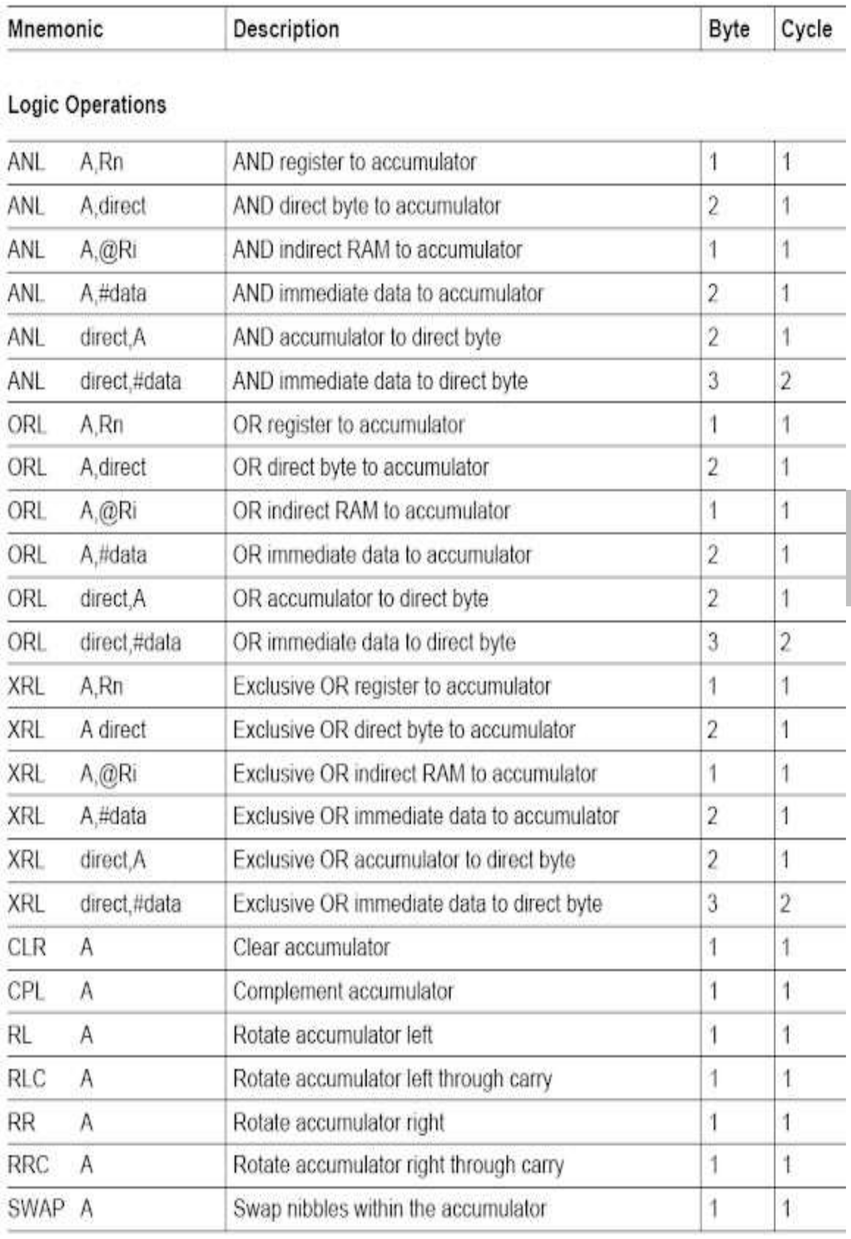
After execution:

A=0 B=32h A·B=80·160=12800

2.LOGICAL INSTRUCTION OF 8051 MICRO-CONTROLLER

Logic instructions perform logic operations upon corresponding bits of two registers. After execution, the result is stored in the first operand.

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali



Mnemonic		Description	Byte	Cycle
Logic Operations				
ANL	A,Rn	AND register to accumulator	1	1
ANL	A,direct	AND direct byte to accumulator	2	1
ANL	A,@Ri	AND indirect RAM to accumulator	1	1
ANL	A,#data	AND immediate data to accumulator	2	1
ANL	direct,A	AND accumulator to direct byte	2	1
ANL	direct,#data	AND immediate data to direct byte	3	2
ORL	A,Rn	OR register to accumulator	1	1
ORL	A,direct	OR direct byte to accumulator	2	1
ORL	A,@Ri	OR indirect RAM to accumulator	1	1
ORL	A,#data	OR immediate data to accumulator	2	1
ORL	direct,A	OR accumulator to direct byte	2	1
ORL	direct,#data	OR immediate data to direct byte	3	2
XRL	A,Rn	Exclusive OR register to accumulator	1	1
XRL	A direct	Exclusive OR direct byte to accumulator	2	1
XRL	A,@Ri	Exclusive OR indirect RAM to accumulator	1	1
XRL	A,#data	Exclusive OR immediate data to accumulator	2	1
XRL	direct,A	Exclusive OR accumulator to direct byte	2	1
XRL	direct,#data	Exclusive OR immediate data to direct byte	3	2
CLR	A	Clear accumulator	1	1
CPL	A	Complement accumulator	1	1
RL	A	Rotate accumulator left	1	1
RLC	A	Rotate accumulator left through carry	1	1
RR	A	Rotate accumulator right	1	1
RRC	A	Rotate accumulator right through carry	1	1
SWAP	A	Swap nibbles within the accumulator	1	1

ANL A,Rn - AND register to the accumulator A: accumulator Rn: any R register(R0-R7)

Instruction performs logic AND operation between the accumulator and Rn register. The result is stored in the accumulator.

Syntax:

ANL Arm

Before execution:

A= C3h (11000011 Bin.) R5= 55h (01010101 Bin.)

After execution:

A= 41h (01000001 Bin.)

ORL Arm - OR register to the accumulator Rn: any R register (R0-R7)

Instruction performs logic OR operation between the accumulator and Rn register. The result is stored in the accumulator.

Syntax:

Orland

Before execution:

A= C3h (11000011 Bin.) R5= 55h (01010101 Bin.)

After execution:

A= D7h (11010111 Bin.)

XRL Arm - Exclusive OR register to accumulator Rn: any R register (R0-R7)

Instruction performs exclusive OR operation between the accumulator and the Rn register. The result is stored in the accumulator.

Syntax:

XRL Arm

Before execution:

A= C3h (11000011 Bin.) R3= 55h (01010101 Bin.)

After execution**:**

A= 96h (10010110 Bin.)

CLR A - Clears the accumulator A: accumulator

Instruction clears the accumulator.

Syntax:

CLR A

After execution:

A=0

CPL A - Complements the accumulator

Instruction complements all the bits in the accumulator (1==>0, 0==>1).

Syntax:

CPL A

Before execution:

A= (00110110)

After execution:

A= (11001001)

RL A - Rotates the accumulator one bit left a: accumulator

Eight bits in the accumulator are rotated one bit left, so that the bit 7 is rotated into the bit 0 position.

Syntax:

RL A

Before execution:

A= C2h (11000010 Bin.)

After execution:

A=85h (10000101 Bin.)

RR A - Rotates the accumulator one bit right

A: accumulator all eight bits in the accumulator are rotated one bit right so that the bit 0 is rotated into the bit 7 position.

Syntax:

RR A

Before execution:

A= C2h (11000010Bin.)

After execution:

A= 61h (01100001Bin.)

RLC A - Rotates the accumulator one bit left through the carry flag A: accumulator

All eight bits in the accumulator and carry flag are rotated one bit left. After this operation, the bit 7 is rotated into the carry flag position and the carry flag is rotated into the bit 0 position.

Syntax:

RLC A

Before execution:

A= C2h (11000010 Bin.) C=0

After execution:

A= 85h (10000100Bin.) C=1

RRC A - Rotates the accumulator one bit right through the carry flag A: accumulator all eight bits in the accumulator and carry flag are rotated one bit right. After this operation, the carry flag is rotated into the bit 7 position and the bit 0 is rotated into the carry flag position.

Syntax:

RRC A

Before execution:

A= C2h (11000010 Bin.) C=0

After execution:

A= 61h (01100001Bin.) C=0

SWAP A - Swaps nibbles within the accumulator

A: accumulator

A nibble refers to a group of 4 bits within one register (bit0-bit3 and bit4-bit7). This instruction interchanges high and low nibbles of the accumulator.

Syntax:

SWAPA

Before execution:

A=E1h (11100001)bin.

After execution:

A=1Eh (00011110)bin.

3. DATA TRANSFER INSTRUCTION OF 8051

These instructions are used to copy the content of source operand to Destination operand

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley]

Mnemonic		Description	Byte	Cycle
Data Transfer				
MOV	A,Rn	Move register to accumulator	1	1
MOV	A,direct *)	Move direct byte to accumulator	2	1
MOV	A,@Ri	Move indirect RAM to accumulator	1	1
MOV	A,#data	Move immediate data to accumulator	2	1
MOV	Rn,A	Move accumulator to register	1	1
MOV	Rn,direct	Move direct byte to register	2	2
MOV	Rn,#data	Move immediate data to register	2	1
MOV	direct,A	Move accumulator to direct byte	2	1
MOV	direct,Rn	Move register to direct byte	2	2
MOV	direct,direct	Move direct byte to direct byte	3	2
MOV	direct,@Ri	Move indirect RAM to direct byte	2	2
MOV	direct,#data	Move immediate data to direct byte	3	2
MOV	@Ri,A	Move accumulator to indirect RAM	1	1
MOV	@Ri,direct	Move direct byte to indirect RAM	2	2
MOV	@Ri, #data	Move immediate data to indirect RAM	2	1
MOV	DPTR, #data16	Load data pointer with a 16-bit constant	3	2
MOVC	A,@A + DPTR	Move code byte relative to DPTR to accumulator	1	2
MOVC	A,@A + PC	Move code byte relative to PC to accumulator	1	2
MOVX	A,@Ri	Move external RAM (8-bit addr.) to A	1	2
MOVX	A,@DPTR	Move external RAM (16-bit addr.) to A	1	2
MOVX	@Ri,A	Move A to external RAM (8-bit addr.)	1	2
MOVX	@DPTR,A	Move A to external RAM (16-bit addr.)	1	2
PUSH	direct	Push direct byte onto stack	2	2
POP	direct	Pop direct byte from stack	2	2
XCH	A,Rn	Exchange register with accumulator	1	1
XCH	A,direct	Exchange direct byte with accumulator	2	1
XCH	A,@Ri	Exchange indirect RAM with accumulator	1	1
XCHD	A,@Ri	Exchange low-order nibble indir. RAM with A	1	1

*) MOV A,ACC is not a valid instruction

MOV Arm - Moves the Run register to the accumulator

The instruction moves the Run register to the accumulator. The Run register is not affected. Syntax

MOV Arm

Before execution:

R3=58h

After execution:

R3=58h A=58h

MOV A,@RI - Moves the indirect RAM to the accumulator

Instruction moves the indirectly addressed register of RAM to the accumulator. The register address is stored in the RI register (R0 or R1). The result is stored in the accumulator. The register is not affected.

Syntax:

MOV A,@RI

Register Address SUM=F2h R0=F2h

Before execution:

SUM=58h

After execution:

A=58h SUM=58h

MOV A, data - Moves the immediate data to the accumulator

Instruction moves the immediate data to the accumulator.

Syntax:

MOV A, #28

After execution:

A=28h

MOV direct, RI - Moves the indirect RAM to the direct byte

Instruction moves the indirectly addressed register of RAM to the direct byte. The register is not affected.

Syntax:

MOV Rx ,@RI

Register Address SUM=F3

Before execution:

SUM=58h R1=F3

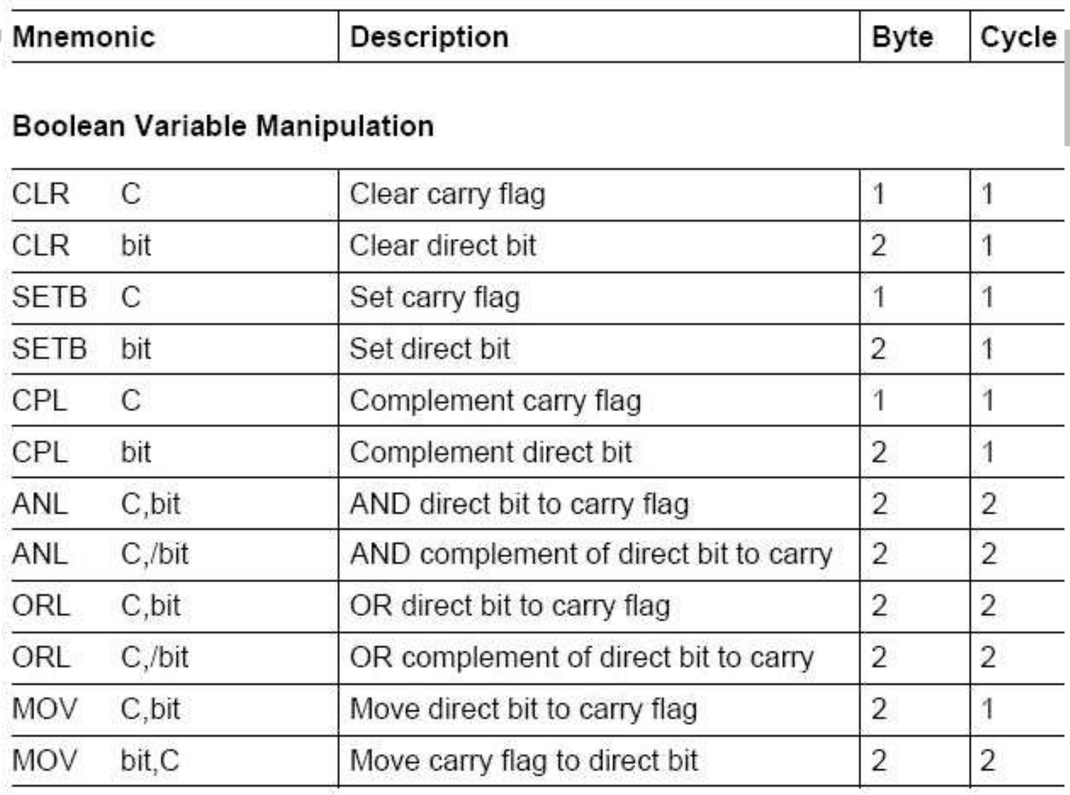
After execution:

SUM=58h TEMP=58h

MOVC A,@A+DPTR - Moves the code byte relative to the DPTR to the accumulator Instruction first adds the 16-bit DPTR register to the accumulator. The result of addition is then used as a memory address from which the 8-bit data is moved to the accumulator.

4. BIT-ORIENTED INSTRUCTIONS

Similar to logic instructions, bit-oriented instructions perform logic operations. The difference is that these are performed upon single bits.



Mnemonic	Description	Byte	Cycle
Boolean Variable Manipulation			
CLR C	Clear carry flag	1	1
CLR bit	Clear direct bit	2	1
SETB C	Set carry flag	1	1
SETB bit	Set direct bit	2	1
CPL C	Complement carry flag	1	1
CPL bit	Complement direct bit	2	1
ANL C,bit	AND direct bit to carry flag	2	2
ANL C,/bit	AND complement of direct bit to carry	2	2
ORL C,bit	OR direct bit to carry flag	2	2
ORL C,/bit	OR complement of direct bit to carry	2	2
MOV C,bit	Move direct bit to carry flag	2	1
MOV bit,C	Move carry flag to direct bit	2	2

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley, pg.no.29]

ANL Cubit - AND direct bit to the carry flag C: Carry

flag Bit: any bit of RAM,

Instruction performs logic AND operation between the direct bit and the carry flag.

Syntax:

ANL Cubit

Before execution:

ACC= 43h (01000011 Bin.)C=1

After execution:

ACC= 43h (01000011 Bin.) C=0

CLR C - clears the carry flag

C: Carry flag, Instruction clears the carry flag. After execution: C=0

CLR bit - clears the direct bit

Bit: any bit of RAM, Instruction clears the specified bit.

Syntax:

CLR P0.3

Before execution:

P0.3=1 (input pin)

after execution:

P0.3=0 (output pin)

CPL bit - Complements the direct bit

Bit: any bit of RAM, Instruction complements the specified bit of RAM (0==>1, 1==>0).

Syntax:

CPL P0.3

Before execution:

P0.3=1 (input pin)

after execution:

P0.3=0 (output pin)

CPL C - Complements the carry flag

C: Carry flag, Instruction complements the carry flag (0==>1, 1==>0).

Syntax:

CPL C

Before execution:

C=1

After execution:

C=0

MOV biotic - Moves the carry flag to the direct bit C: Carry flag, Bit: any bit ofRAM Instruction moves the carry flag to the direct bit. After executing the instruction, the carry flag is not affected.

Syntax:

MOVP1.2, C

After execution:

If C=0 P1.2=0 If C=1 P1.2=1

MOV Cubit - Moves the direct bit to the carry flag C: Carry flag, Bit: any bit of RAM Instruction moves the direct bit to the carry flag. After executing the instruction, the bit is not affected.

Syntax:

MOV C, P1.4

After execution:

If P1.4=0 C=0 If P1.4=1 C=1

SETB C - Sets the carry flag

C: Carry flag, Instruction sets the carry flag.

Syntax:

SETB C

After execution: C=1

SETB bit - Sets the direct

bit: any bit of RAM

Instruction sets the specified bit. The register containing that bit must belong to the group of the so called bit addressable registers.

Syntax:

SETB P0.1

Before execution:

P0.1 = 34h (00110100) pin 1 is configured as an output

after execution:

P0.1 = 35h (00110101) pin 1 is configured as an inputs

5. BRANCH INSTRUCTION OF 8051 CONTROLLER

There are two kinds of branch instructions:

Unconditional jump instructions:

Upon their execution a jump to a new location from where the program continues execution is executed.

Conditional jump instructions:

A jump to a new program location is executed only if a specified condition is met. Otherwise, the program normally proceeds with the next instruction.

Mnemonic		Description	Byte	Cycle
Program and Machine Control				
ACALL	addr11	Absolute subroutine call	2	2
LCALL	addr16	Long subroutine call	3	2
RET		Return from subroutine	1	2
RETI		Return from interrupt	1	2
AJMP	addr11	Absolute jump	2	2
LJMP	addr16	Long iump	3	2
SJMP	rel	Short jump (relative addr.)	2	2
JMP	@A + DPTR	Jump indirect relative to the DPTR	1	2
JZ	rel	Jump if accumulator is zero	2	2
JNZ	rel	Jump if accumulator is not zero	2	2
JC	rel	Jump if carry flag is set	2	2
JNC	rel	Jump if carry flag is not set	2	2
JB	bit,rel	Jump if direct bit is set	3	2
JNB	bit,rel	Jump if direct bit is not set	3	2
JBC	bit,rel	Jump if direct bit is set and clear bit	3	2
CJNE	A,direct,rel	Compare direct byte to A and jump if not equal	3	2
CJNE	A,#data,rel	Compare immediate to A and jump if not equal	3	2
CJNE	Rn,#data rel	Compare immed. to reg. and jump if not equal	3	2
CJNE	@Ri,#data,rel	Compare immed. to ind. and jump if not equal	3	2
DJNZ	Rn,rel	Decrement register and jump if not zero	2	2
DJNZ	direct,rel	Decrement direct byte and jump if not zero	3	2
NOP		No operation	1	1

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley]

I/O PORTS AND CIRCUITS

Each port of 8051 has bidirectional capability. Port 0 is called 'true bidirectional port' as it floats (tristate) when configured as input. Port 1, 2, 3 are called 'quasi bidirectional port'.

Port-0 Pin Structure

Port -0 has 8 pins (P0.0-P0.7) and its structure is shown in Figure 4.3.1.

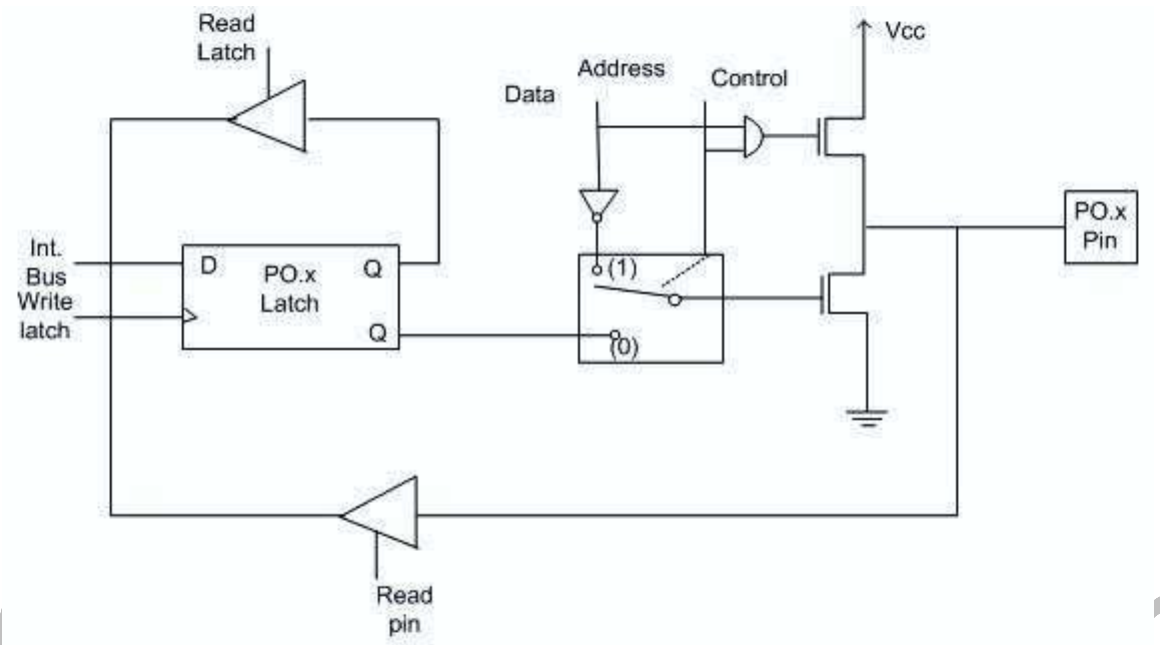


Figure 4.3.1 Port-0 Structure

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.581]

Port-0 can be configured as a normal bidirectional I/O port or it can be used for address/data interfacing for accessing external memory. When control is '1', the port is used for address/data interfacing. When the control is '0', the port can be used as a normal bidirectional I/O port.Let us assume that control is '0'. When the port is used as an input port, '1' is writtento the latch. In this situation, both the output MOSFETs are 'off'. Hence the output pin floats. This high impedance pin can be pulled up or low by an external source. When the port is used as an output port, the output pin to float. An external pull output a '1'. But when '0' is written to the latch, the pin is pulled down by the lower MOSFET. Hencethe output becomes zero.

When the control is '1', address/data bus controls the output driver MOSFETs. If the address/data bus (internal) is '0', the upper MOSFET is 'off' and the lower MOSFET is 'on'. The output becomes '0'. If the address/data bus is '1', the upper transistor is 'on' and the lower transistor is 'off'. Hence the output is '1'. Hence for normal address/data interfacing

Port-1 Pin Structure

Port-1 has 8 pins (P1.1-P1.7). The structure of a port-1 pin is shown in Figure 4.3.2 below.

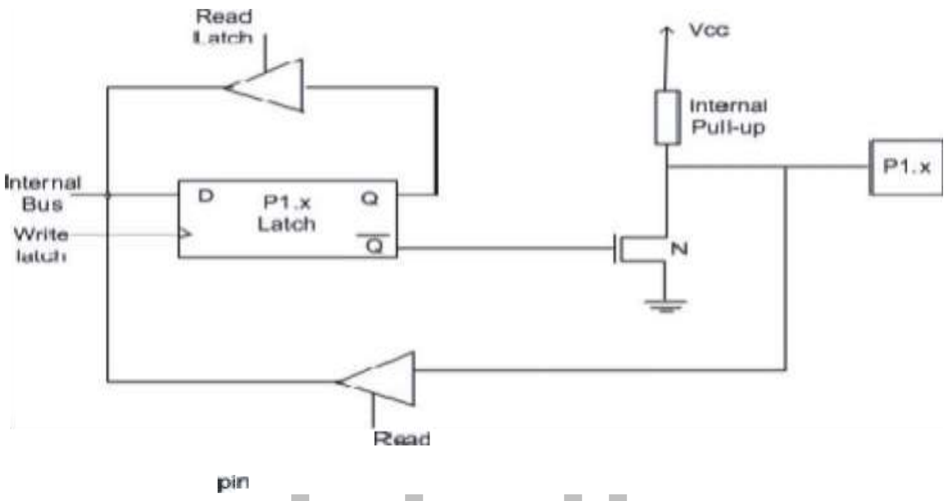


Figure 4.3.2 Port-1 Structure

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.577]

Port-1 does not have any alternate function i.e. it is dedicated solely for I/O interfacing. When used as output port, the pin is pulled up or down through internal pull-up. To use port-1 as input port, '1' has to be written to the latch. In this input mode when '1' is written to the pin by the external device then it read fine. But when '0' is written tothe pin by the external device then the external source must sink current due to internalpull-up. If the external device is not able to sink the current the pin voltage may rise,leading to a possible wrong reading.

PORT 2 Pin Structure

Port-2 has 8-pins (P2.0-P2.7). The structure of a port-2 pin is shownin Figure

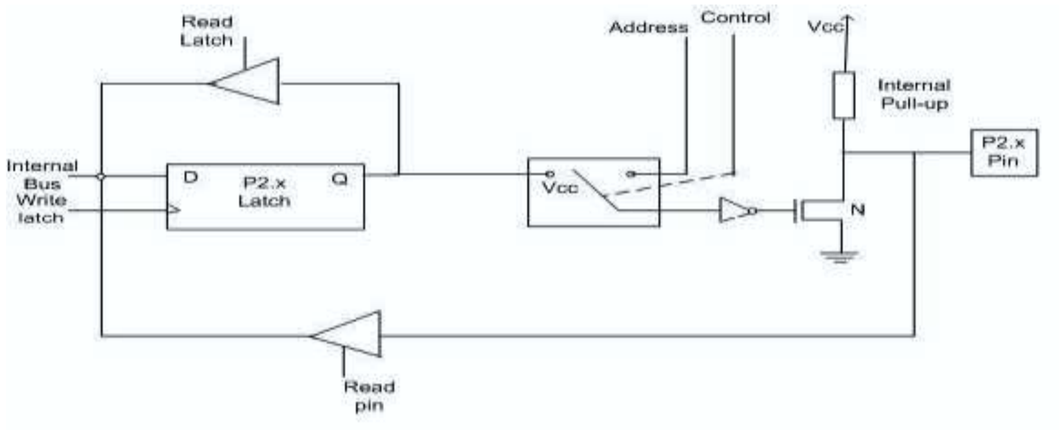


Figure 4.3.3 Port-2 Structure

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.577]

Port-2 is used for higher external address byte or a normal input/output port. The I/O operation is similar to Port-1. Port-2 latch remains stable when Port-2 pin are used for external memory access. Here again due to internal pull-up there is limited current driving capability.

PORT 3 Pin Structure

Port-3 has 8 pin (P3.0-P3.7). Port-3 pins have alternate functions. The structure of a port-3 pin is shown in Figure 4.3.4.

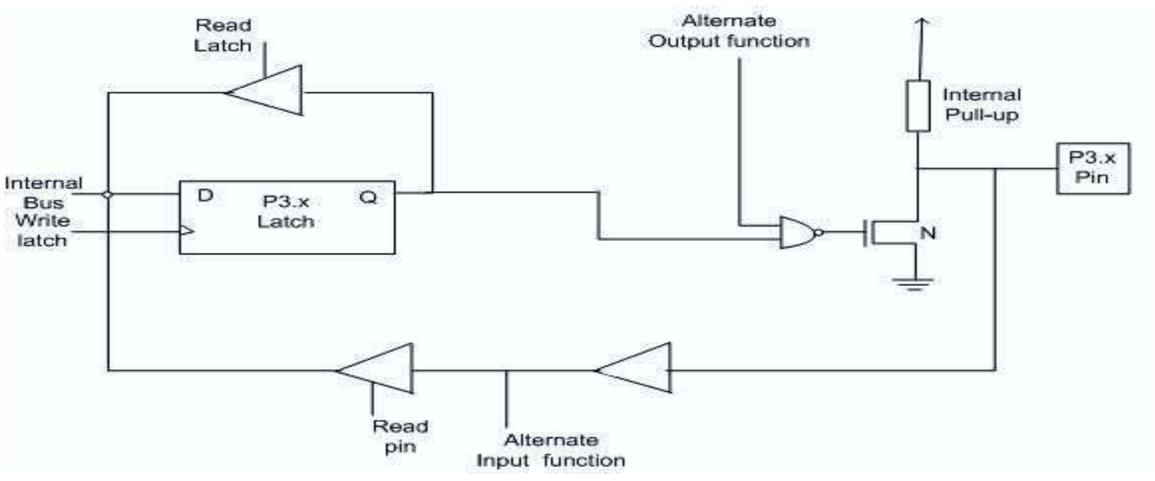


Figure 4.3.4 Port-3 Structure

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Mazidi, Janice Gillispie Mazidi, Rolin McKinlay , pg.no.577]

Each pin of Port-3 can be individually programmed for I/O operation or for alternate function. The alternate function can be activated only if the corresponding latch has been written to '1'. To use the port as input port, '1' should be written to the latch. This port also has internal pull-up and limited current driving capability.

4.2 SPECIAL FUNCTION REGISTERS (SFRs)

8051 MICROCONTROLLER SPECIAL FUNCTION REGISTERS

SFR registers exist in the address range of 80h through Fifth. Each SFR has an address (80h through Fifth) and a name. The following Table 4.2.1 provides an 8051's SFRs, their names and their address. Although the address range of 80H through FFH offers 128 possible addresses, there are only 21 SFRs in a standard 8051.

Table 4.2.1 8051 Special Function Registers (SFRs) Addresses

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali

Symbol	Name	Address
ACC*	Accumulator	0E0H
B*	B register	0F0H
PSW*	Program status word	0D0H
SP	Stack pointer	81H
DPTR	Data pointer 2 bytes	
DPL	Low byte	82H
DPH	High byte	83H
P0*	Port 0	80H
P1*	Port 1	90H
P2*	Port 2	0A0H
P3*	Port 3	0B0H
IP*	Interrupt priority control	0B8H
IE*	Interrupt enable control	0A8H
TMOD	Timer/counter mode control	89H
TCON*	Timer/counter control	88H
T2CON*	Timer/counter 2 control	0C8H
T2MOD	Timer/counter mode control	0C9H
TH0	Timer/counter 0 high byte	8CH
TL0	Timer/counter 0 low byte	8AH
TH1	Timer/counter 1 high byte	8DH
TL1	Timer/counter 1 low byte	8BH
TH2	Timer/counter 2 high byte	0CDH
TL2	Timer/counter 2 low byte	0CCH
RCAP2H	T/C 2 capture register high byte	0CBH
RCAP2L	T/C 2 capture register low byte	0CAH
SCON*	Serial control	98H
SBUF	Serial data buffer	99H
PCON	Power control	87H

* Bit-addressable

,

The 21 Special Function Registers of 8051 Microcontroller are categorized in to seven groups. They are:

- ***Math or CPU Registers***: A and B
- ***Status Register***: PSW (Program Status Word)
- ***Pointer Registers***: DPTR (Data Pointer - DPL, DPH) and SP (Stack Pointer)
- ***I/O Port Latches***: P0 (Port 0), P1 (Port 1), P2 (Port 2) and P3 (Port3)
- ***Peripheral Control Registers***: PCON, SCON, TCON, TMOD, IE and IP
- ***Peripheral Data Registers***: TL0, TH0, TL1, TH1 and SBUF

CPU OR MATH REGISTERS

A OR ACCUMULATOR (ACC)

The Accumulator or Register A is the most important and most used 8051 Microcontroller SFRs. The Register A is located at the address E0H in the SFR memory space. The Accumulator is used to hold the data for almost all the ALU Operations.

B (REGISTER B)

The B Register is used along with the ACC in Multiplication and Division operations. These two operations are performed on data that are stored only in Registers A and B. During Multiplication Operation, one of the operand (multiplier or multiplicand) is stored in B Register and also the higher byte of the result.

In case of Division Operation, the B Register holds the divisor and also the remainder of the result. It can also be used as a General Purpose Register for normal operations and is often used as an Auxiliary Register by Programmers to store temporary results. Register B is located at the address F0H of the SFR Address Space.

PROGRAM STATUS WORD (PSW)

The PSW or Program Status Word Register is also called as Flag Register and is one of the important SFRs. The PSW Register consists of Flag Bits, which help the programmer in checking the condition of the result and also make decisions

POINTER REGISTERS

DATA POINTER (DPTR - DPL AND DPH)

The Data Pointer is a 16-bit Register and is physically the combination of DPL (Data Pointer Low) and DPH (Data Pointer High) SFRs. The Data Pointer can be used as a single 16-bit register (as DPTR) or two 8-bit registers (as DPL and DPH).

DPTR doesn't have a physical Memory Address but the DPL (Lower Byte of DPTR) and DPH (Higher Byte of DPTR) have separate addresses in the SFR Memory Space. DPL = 82H and DPH = 83H. The DPTR Register is used by the programmer addressing external memory (Program - ROM or Data - RAM).

SP or Stack Pointer points out to the top of the Stack and it indicates the next data to be accessed. Stack Pointer can be accessed using PUSH, POP, and CALL and RET Instructions. The Stack Pointer is an 8- bit register and upon reset, the Stack Pointer is initialized with 07H.

I/O Port Registers (P0, P1, P2 and P3)

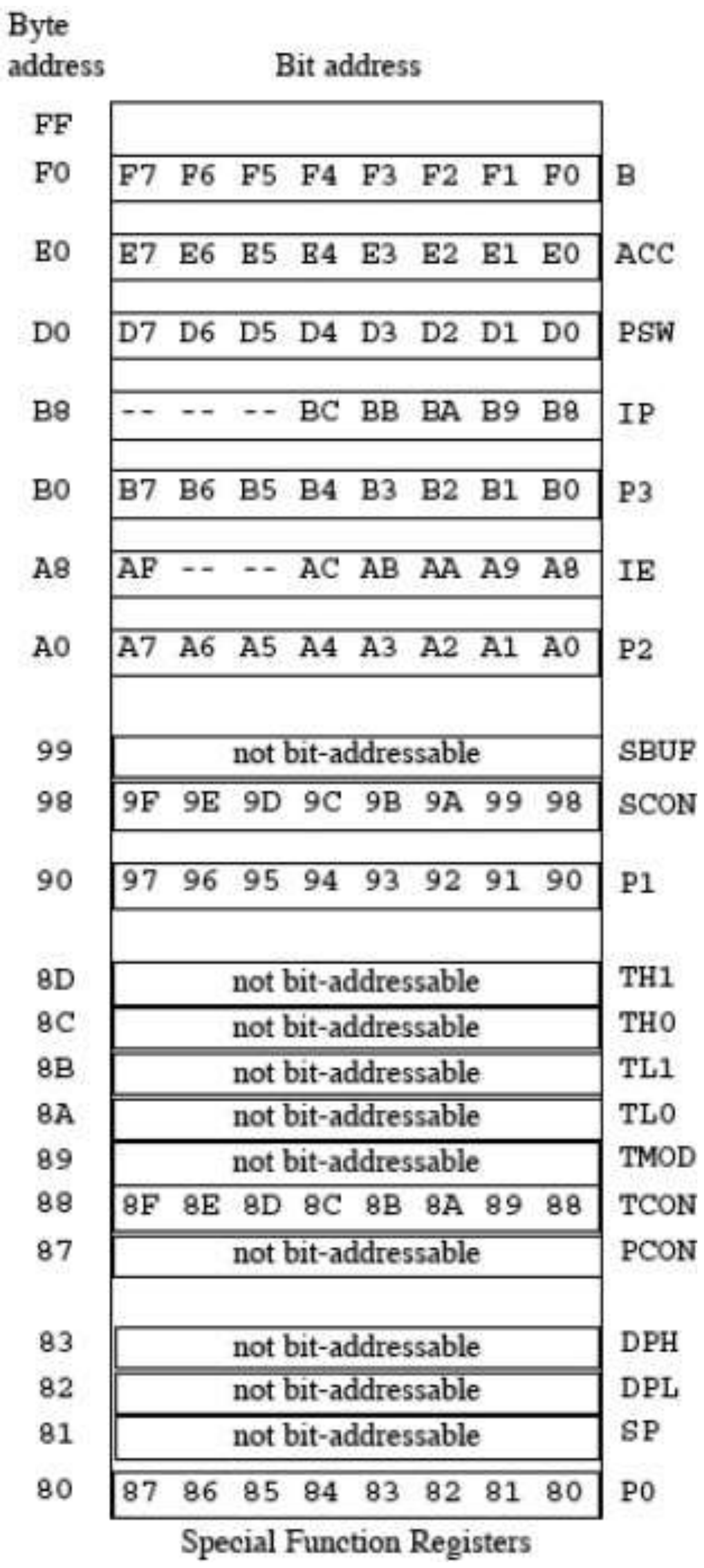
The 8051 Microcontroller four Ports which can be used as Input and/or Output. These four ports are P0, P1, P2 and P3. Each Port has a corresponding register with same names (the Port Registers are also P0, P1, P2 and P3. The addresses of the Port Registers are as follows: P0 - 80H, P1 - 90H, P2 - A0H and P2 -B0H.

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 or Power down Mode.

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, we can control the Operation Modes of the Serial Port, Baud Rate of the Serial Port and Send or Receive Data using Serial Port.



Byte address	Bit address								
FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
B8	--	--	--	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	--	--	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
99	not bit-addressable								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	not bit-addressable								TH1
8C	not bit-addressable								TH0
8B	not bit-addressable								TL1
8A	not bit-addressable								TL0
89	not bit-addressable								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	not bit-addressable								PCON
83	not bit-addressable								DPH
82	not bit-addressable								DPL
81	not bit-addressable								SP
80	87	86	85	84	83	82	81	80	P0

Special Function Registers

Figure 4.2.1 8051 SFR RAM Address (Byte and Bit)

[Source: "The 8051Microcontroller and Embedded Systems: Using Assembly and C" by Mohamed Ali Maida, Janice Gillespie Maida, Rollin McKinley]

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.

IE (Interrupt Enable)

The IE or Interrupt Enable Register is used to enable or disable individual interrupts. If a bit is SET, the corresponding interrupt is enabled and if the bit is cleared, the interrupt is disabled. The Bit7 of the IE register i.e. EA bit is used to enable or disable all the interrupts

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.