CHAPTER THREE

8051 MICROCONTROLLER

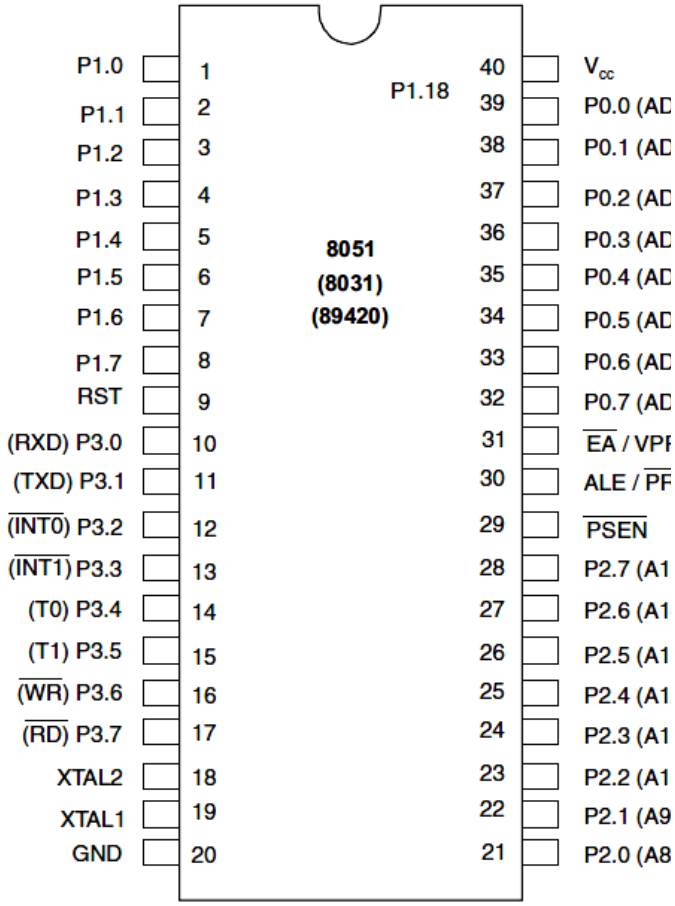
· The 8051 is an 8-bit microcontroller with 8 bit data bus and 16-bit address bus.
· The 16 bit address bus can address a 64K(216) byte code memory space and a separate 64K byte of data memory space.
· The 8051 has 4K on-chip read only code memory and 128 bytes of internal Random Access Memory(RAM)
· Besides internal RAM, the 8051 has various Special Function Registers (SFR) such as the Accumulator(A or AC), the B register, and many other control registers.
· 34 x8-bit general purpose registers in total.
· The ALU performs one 8-bit operation at a time.
· Two 16 bit /Counter timers
· 3 internal interrupts (one serial), 2 external interrupts.
· Four 8-bit I/O ports
· Some 8051 chips come with UART for serial communication and ADC for analog to digital Conversion

(See Figure 2).

3.1 PIN DIAGRAM OF 8051

Pin out Description:

Pins 1-8: Port 1 Each of these pins can be configured as an input or an output.
Pin 9: RST A logic one on this pin disables the microcontroller and clears the contents of most registers. In other words, the positive voltage on this pin resets the microcontroller. By applying logic zero to this pin, the program starts execution from the beginning.
Pins10-17: Port 3 Similar to port 1, each of these pins can serve as general input or output. Besides, all of them have alternative functions:
Pin10: RXD (receive) Serial asynchronous communication input or Serial synchronous communication output.
Pin11: TXD (Transmitting)Serial asynchronous communication output or Serial synchronous communication clock output.
Pin 12: INT0 Interrupt 0 inputs.
Pin 13: INT1 Interrupt 1 input.
Pin 14: T0 Counter 0 clock input.
Pin 15: T1 Counter 1 clock input.
Pin 16: WR Write to external (additional) RAM.



Pin 17: RD Read from external RAM.
Pin 18, 19: XTAL2/XTALI is for oscillator input
Pin 20: GND-Ground.
Pin 21-28: Port 2- If there is no intention to use external memory then these port pins are configured as general inputs/outputs. In case external memory is used, the higher address byte, i.e. addresses A8-A15 will appear on this port. Even though memory with capacity of 64Kb is not used, which means that not all eight port bits are used for its addressing, the rest of them are not available as inputs/outputs.
Pin 29: PSEN'- Program Store Enable. If external ROM is used for storing program, then a logic zero(0) appears on it every time the microcontroller reads a byte from memory.
Pin 30: ALE – Address latch enable 1– Address on AD 0 to AD 7 0 – Data on AD 0 to AD 7
Pin 31: EA' – it indicates the presence of external memory .
Pin 32-39: Port 0 Similar to P2.
Pin 40: VCC → +5V power supply.

3.2 FUNCTIONAL BLOCK DIAGRAM OF 8051 MICROCONTROLLER

1. The Intel 8051 contains two separate buses for both program and data.
2. It is based on an 8 bit central processing unit with an 8 bit accumulator (A or AC) and another 8-bit B register as main processing blocks.
3. Other portions of the architecture include few 8 bit and 16b it registers and 8-bit memory locations.
4. It has some amount of data RAM built in the device for internal processing. This area is used for stack operations and temporary storage of data.
5. 8051 is supported with on-chip peripheral functions like I/O ports.
6. Timers/ Counters.
7. Serial communication port.

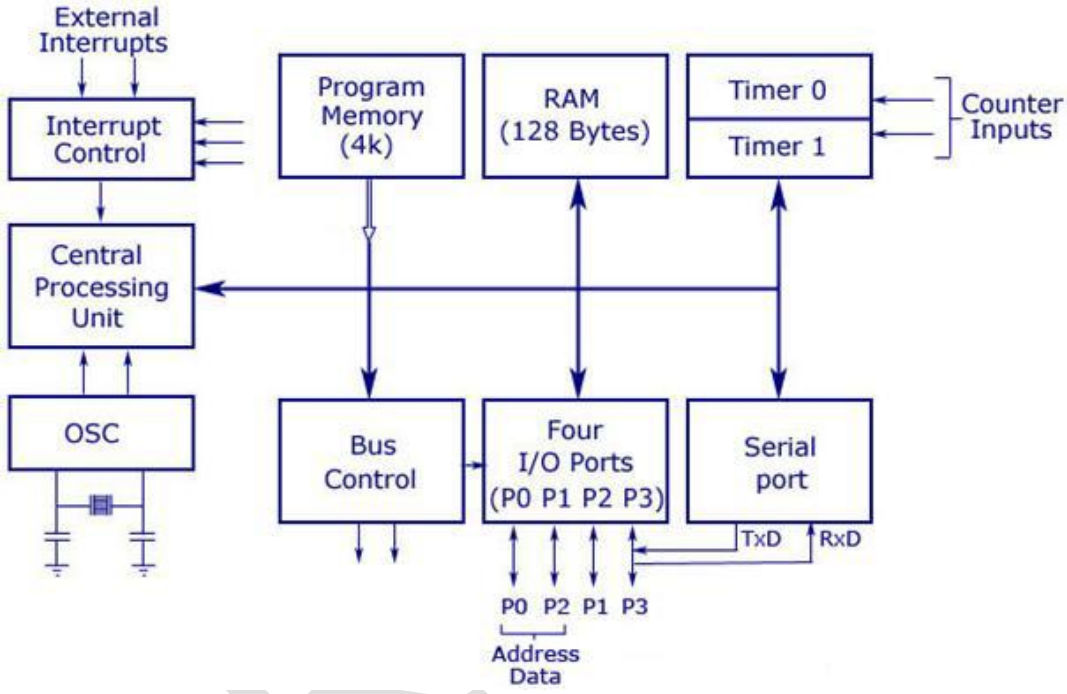


Figure 2: Block Diagram of 8051

3.2.1 CENTRAL PROCESSING UNIT

- The CPU is the brain of the microcontrollers expected task reading user's programs and executing the as per instructions stored there in.
- Its primary elements are an Accumulator (AC), Stack Pointer (SP) Program Counter (PC), Program Status Word (PSW), Data Pointer (DTPR) and few more 8-bit register.

3.2.2 ARITHMETIC LOGIC UNIT (ALU)

- The arithmetic / logic unit performs the computing functions; it includes the accumulator, temporary register, arithmetic and logic circuits.
- The temporary register is used to hold data during an arithmetic / logic operation.
- The result is stored in the accumulator register.

3.2.3 ACCUMULATOR (AC or A)

- The accumulator register (AC or A) act as an operand register, in case of some instructions.
- This may either be implicit or specified in the instruction.
- The ACC register has been allotted on address in the on-chip special function register bank

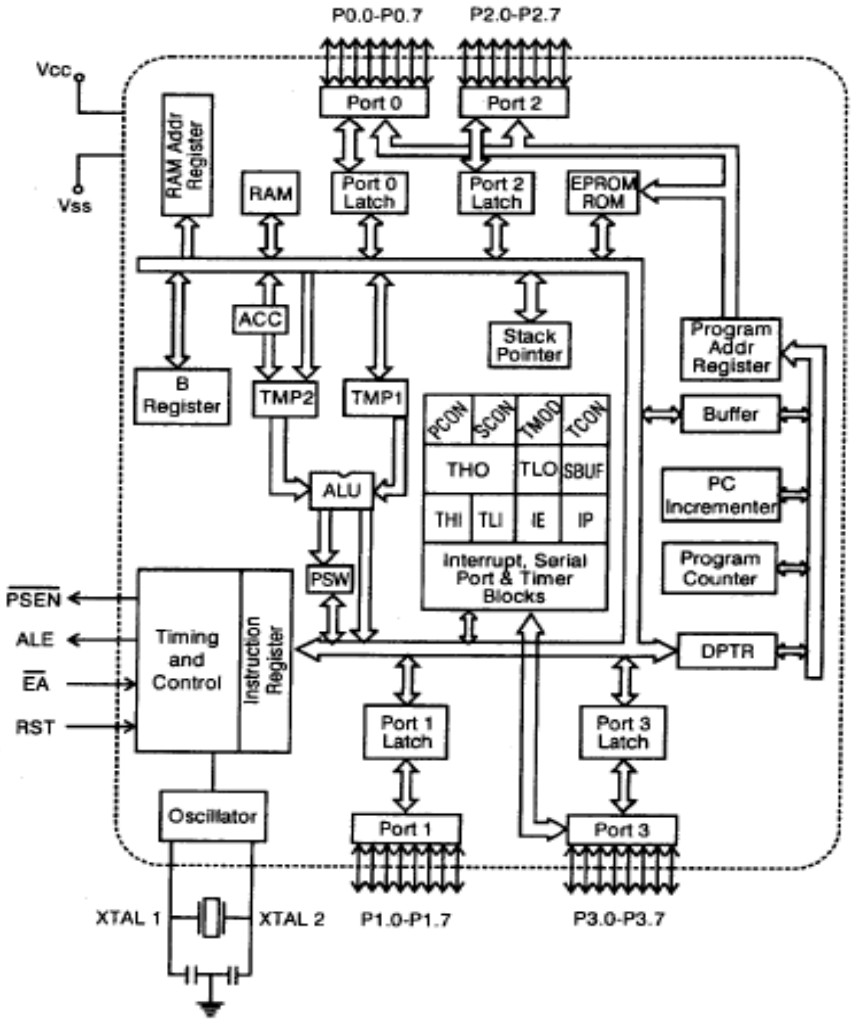


Figure 3: Architecture of 8051

3.2.4 PROGRAM STATUS WORD (PSW)

This set of flags contains the status information and is considered as one of the special function registers.

D7	D6	D5	D4	D3	D2	D1	D0
PSW.7	PSW.6	PSW.5	PSW.4	PSW.3	PSW.2	PSW.1	PSW.0

Figure 3: Program Status Word (PSW)

CY	AC	FO	RS1	RS0	OV	---	P

PSW.0 : CY: Carry Flag
PSW.1 : --- : User Definable Flag
PSW.2 : OV: Overflow Flag
PSW.3 : RS0: Register Bank Select Bit 0
PSW.4 : RS1: Register Bank Select Bit 1
PSW.5 : FO: Flag 0 available for general purpose
PSW.6 : AC: Auxiliary Carry Flag
PSW.7 : CY: Carry Flag

The bits PSW.3 and PSW.4 are denoted as RS0 and RS1. These bits are used to select the bank register of the RAM location. (See Figure 4)

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

Table 1: selection of the register banks and their addresses

3.2.5 RAM and RAM Address Register

These blocks provide internal 128 bytes of RAM and a mechanism to address it internally ,which divided into 32 working registers and they are divided into registers banks(bank0, bank1, bank2, bank3) (as in table 1).

3.2.6 DATA POINTER (DTPR)

· This 16-bit register contains a higher byte (DPH) and the lower byte (DPL) of a 16-bit external data RAM address.
· It is accessed as a 16-bit register or two 8-bit registers as specified above. It has been allotted to two address in the special function register bank, for its two bytes DPH and DPL.

3.2.7 PORT 0 TO 3 LACHES AND DRIVES

· This four latches and driver pairs are allotted to each of the four on-chip I/O ports. · These latches have been allotted address in the especial function register bank.
· Using the allotted addresses, the users can communicate with these ports. These are identified as P0, P1 and P3.

3.2.8 SERIAL DATA BUFFER

· The serial data buffer internally contains two independent registers. · One of them is a transmit buffer which is necessarily a parallel–in serial–out (PISO)
register. · The other is called receive buffer which is a serial–in parallel–out (SIPO) register.
· Loading a byte to the transmit buffer initiates serial transmission of that byte. · The serial data buffer is identified as SBUF and is one of the special function register.
· If a byte is written to SBUF, it initiates serial transmission and if the SBUF is read, it reads received serial data.

3.2.9 TIMER REGISTER

· These two 16-bit register can be accessed as the lower and upper bytes. · For examples, TL0 represents the lower byte of the timing register 0, while TH0
represents the upper byte of the timing register 0. · Similarly, TL1 and TH1 represent lower and higher byte of the timing register 1.
· All these registers an be accessed using the four addresses allotted to them which lie in the special function register (SFR) address range,i.e.80Hto FF.

3.2.10 CONTROL REGISTER

· The special function register IP, IE, TMOD, TCON, SCON and PCON contain control and status information for interrupts, Timer / Counters and serial port.
· All of the registers have been allotted addresses in the special function register bank of 8051.

3.2.11 TIMING AND CONTROL UNIT

· This unit derives all the necessary timing and control signals recovered for the internal operation of the circuit.
· It also derives control signals recovered for controlling the system bus.

3.2.12 OSCILLATOR

This circuit generates the basic timing clock signal for the operation of the circuit using crystal oscillator

3.2.13 INSTRUCTION REGISTER

This register decodes the Opcode of an instruction to be executed and gives information to the timing and control unit to generate necessary signal for the execution of the instruction.

3.2.14 EPROM and Program Address Register

This block provides an on-chip EPROM and a mechanism to internally address it. (Note that EPROM is not available in all versions of 8051)

3.2.15 STACK POINTER (SP)

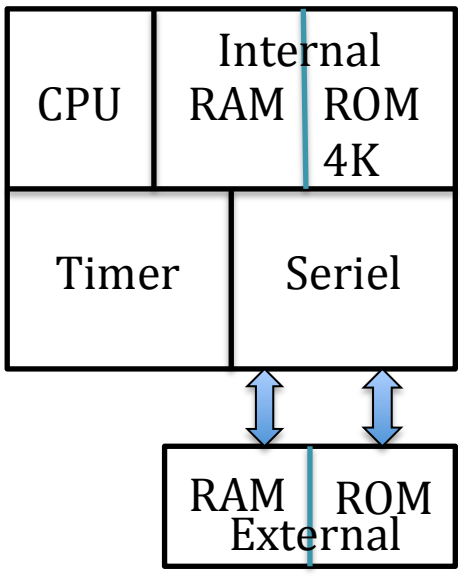
· This 8-bit register is incremented before the data is stored onto the stack using PUSH or CALL instructions.
· This registers contains 8-bit stack top address. The stack may be defined anywhere in the on-chip 128 byte RAM. After reset, the SP register is initialized to 07.
· After each write two stack operation, the 8-bit contents of the operand are stored onto the stack, after incrementing the SP register by 1.
· Thus if SP contains 07H, the forthcoming PUSH operation will store the date at address 08H in the internal RAM.
· The SP content will be incremented to 08.the 8051 stack is not a top-down data structure, like other Intel processors.
· This register has also been allotted on address in the special function register bank.

3.2.16 SFR (Special Function Register) Register Bank

· This is a set of special function registers, which can be addressed using their respective addresses which lie in the range 80H to FFH .
· Finally, the interrupt, serial port and timer units control and perform their specific function under the control of the timing and control unit.

3.3 ADDRESSING MODES OF 8051

The addressing Modes are the ways of accessing data in register or in memory or be provided as an immediate value. The 8051 mnemonics are written with the destination address named first followed by the source address.
The following addressing modes are used to access data:
l. Immediate addressing mode
2. Register addressing mode
3. Direct addressing mode
4. Register indirect addressing mode
5. Indexed addressing mode.

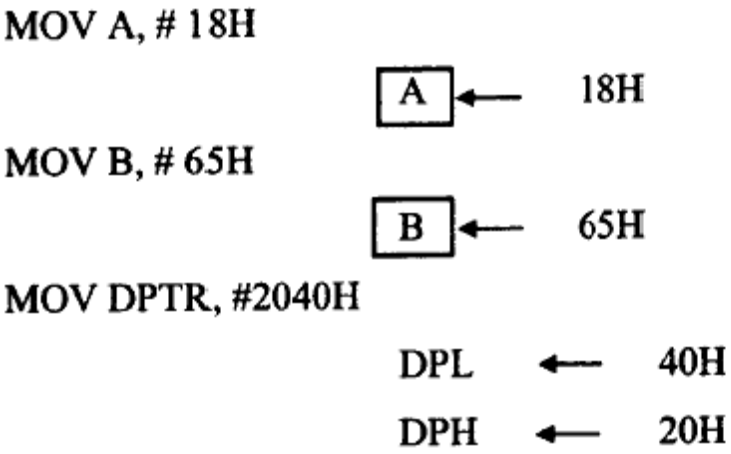


Figur 4 general memory in 8051

1. Immediate Addressing Mode

· When a source operand is a constant rather than a variable, and then the constant can be embedded into the instruction itself.
· This kind of instructions takes two bytes and first one specifies the Opcode and second byte gives the required constant.
· The operand comes immediately after the Opcode. The mnemonic for immediate data is the pound sign (#).
· This addressing mode can be used to load information into any of the registers including DPTR register.

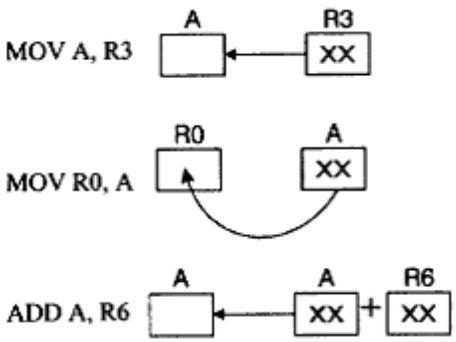
Examples:



2. Register Addressing Mode

· Register addressing accesses the eight working registers (R0 - R7) of the selected register bank to hold the data to be monipolate.
· The least significant three bits of the instruction opcode indicate which register is to be used for the operation.

Examples:



3. Direct Addressing Mode

· In the direct addressing mode, all 128 bytes of internal RAM and the SFRs may be addressed directly using the single-byte addressing need to each RAM location and each SFR.
· Internal RAM uses address from 00H to 7FH to address each byte. The SFR addresses exist from 80H to FFH. (Refer Table below.)

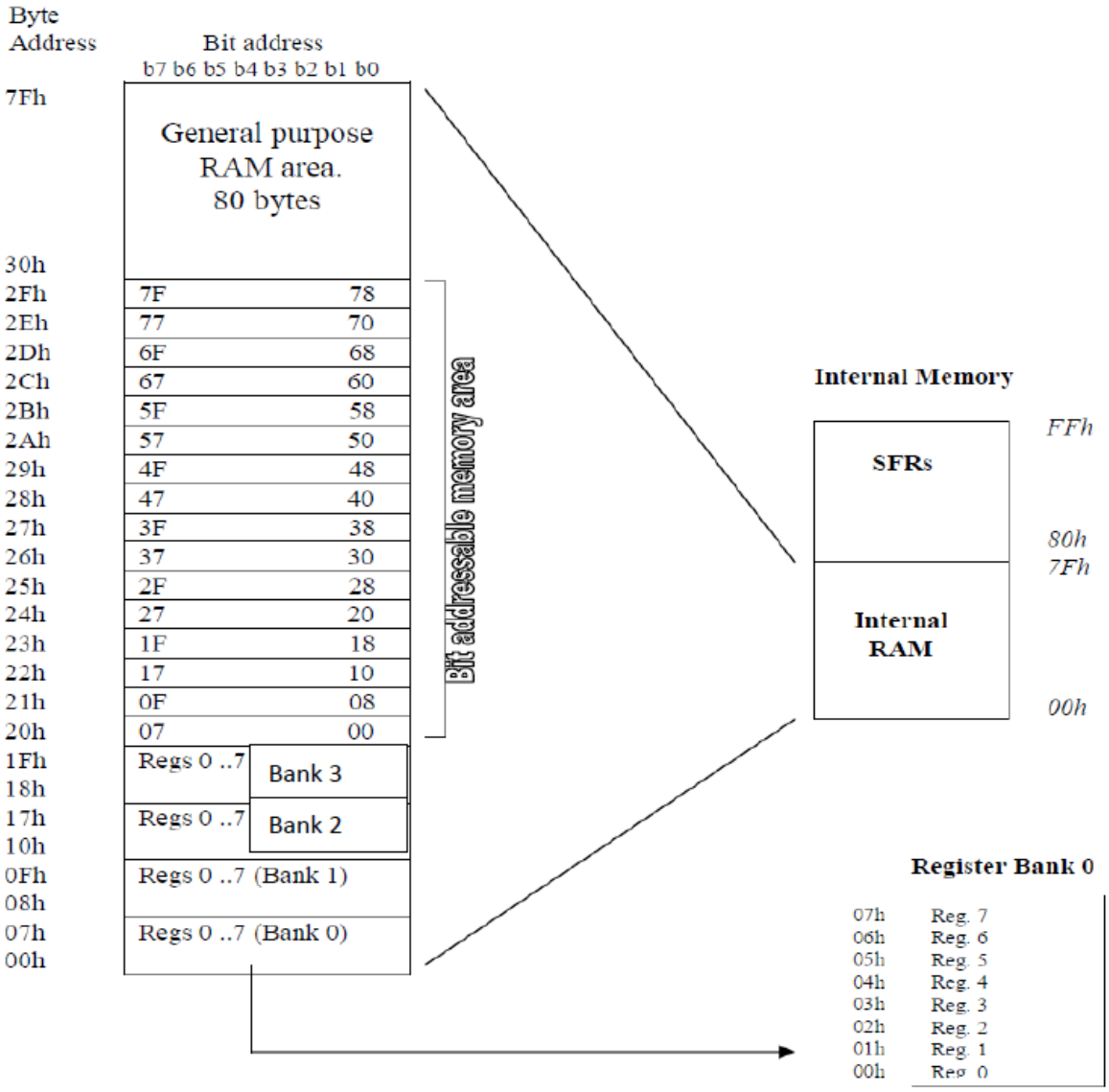
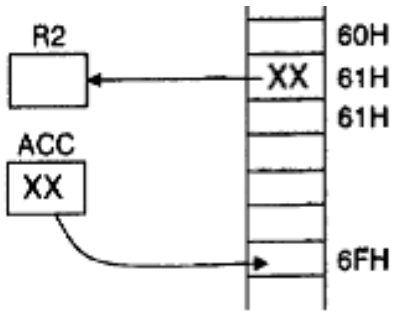


Figure 5 Organization of Internal RAM (IRAM) memory

Examples:

MOV R2, 61H

MOV 6F H, A



4. Register Indirect Addressing Mode

· In this mode a register is used as a pointer to the data. If the data is inside the CPU, only registers R0 and Rl are used for this purpose.
· When R0 and Rl hold the addresses of RAM locations they must be preceded by the "@" sign.

Examples:

MOV @ R1, A: Move contents of A into RAM location whose address is held by R1.
MOV B, @ R0: Move contents of RAM location whose address is held by R0 into B.

5. Indexed Addressing Mode (ROM)

· Only the program memory can be accessed by this mode. This mode is intended for reading lookup tables in the program memory (ROM).
· A 16 bit base registers (DPTR or PC) points to the base of the lookup tables and accumulator carries the constant indicating table entry number (external RAM).
· The address of the exact location of the table is formed by adding the accumulator data to the base pointer.

Example

MOVC A, @A+ DPTR: The contents of A are added to the DPTR to form the 16-bit address of the needed data. 'C' means code in the ROM.

3.4 INTERRUPTS

An interrupt is an internal or external event that interrupts the microcontroller to inform it that a device needs its service. Whenever any device needs its service, the device notifies the microcontroller by sending it as interrupt signal. Upon receiving an interrupt signal, the microcontroller interrupts whatever it is doing and serves the device. The program which is associated with the interrupt is called interrupt Service Routine (ISR). The microcontroller can serve many devices based on the priority assigned to it.

3.4.1 Execution of an Interrupt

In order to use any interrupt, the following steps must be taken.
1. It finishes the instruction it is executing and saves the address of the next instruction (PC) program counter on the stack.
2. It also saves the current status of all the interrupts internally.
3. It jumps to a fixed location in memory called the interrupt vector or table that holds the address of the Interrupt Service Routine (ISR).
4. The microcontroller gets the address of the ISR from the interrupt vector table and jumps to it. It starts to execute the interrupt service subroutine until it reaches the last instruction of the subroutine which is RET 1.
5. Upon executing RET 1 instruction, the microcontroller returns to the place where it was interrupted. First it gets the program counter (PC) address from the stack by popping the top two byes of the stack into the PC. Then it starts to execute from that address.

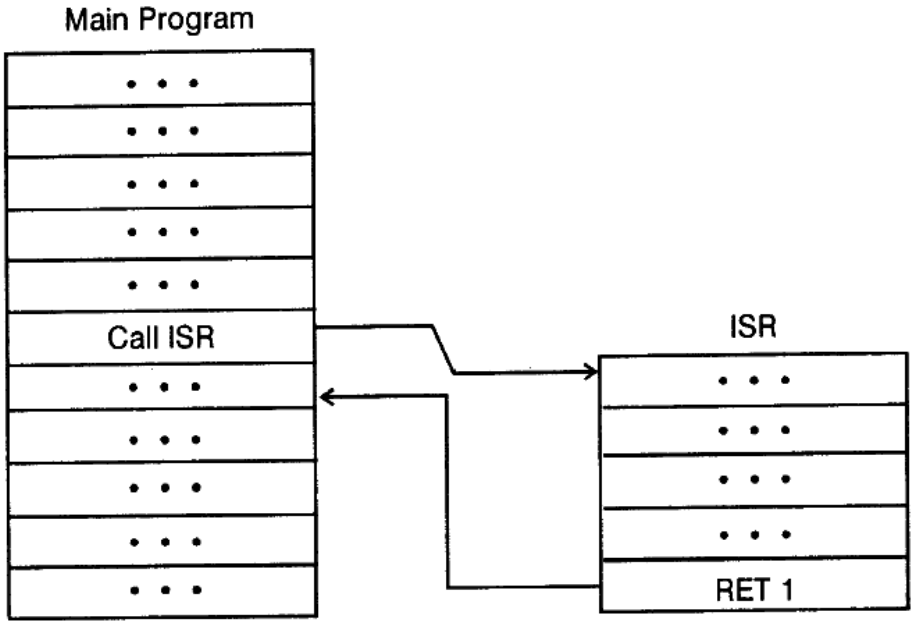


Figure 6:Interrupt Structure

3.4.2 Interrupts in 8051

· Five interrupts are provided in the 805l.
· Three of these a regenerated by internal operations: Timer Flag 1 & 0, and the serial port interrupt (RI or TI).
· Two interrupts are triggered by external signals provided by circuitry that is connected to pin
INT0' and INT1' (port pins P3.2 and P3.3)

Types		**Interrupt**	**Vector Address**
Internal	TF0	Timer flag 0 interrupt	$000B_H$
	TF1	Timer flag 1 interrupt	$001B_H$
	RI/TI	Serial port interrupt	0023_H
External	$\overline{\text{INT0}}$	External interrupt 0	0003_H
	$\overline{\text{INT1}}$	External interrupt 1	0013_H

Table 3: Interrupt Vector

a) Timer flag interrupts

Table 3: Interrupt Vector

- When a timer / counter overflows, the corresponding timer flag TF0 or TFI (location: 000B H or 001B H) is set to l.
- The flag is cleared to 0 when the resulting interrupt generates a program call to the appropriate timer subroutine in memory.

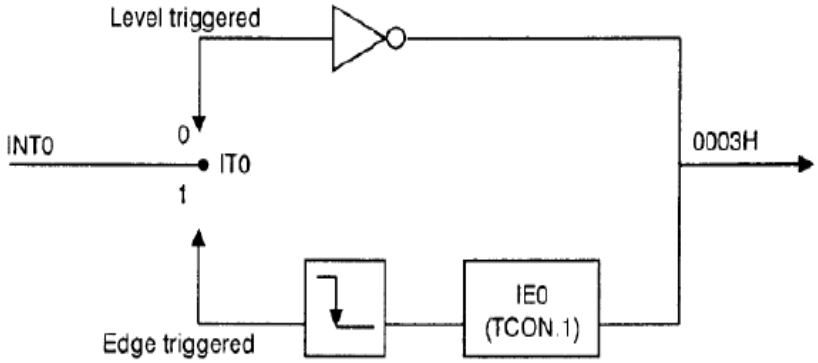


Fig. 7 Activation of INT0

b) External interrupts

- The external hardware interrupts INT0 and INT1 are located on pins P3.2 and P3.3.
- Inputs on these pins can set the interrupt flags IE0 and IE1 in the TCON register to 1 by level triggering or edge-triggering.
- Fig. 5. Shows the activation of INT0 and Fig.6. shows the activation of INT1

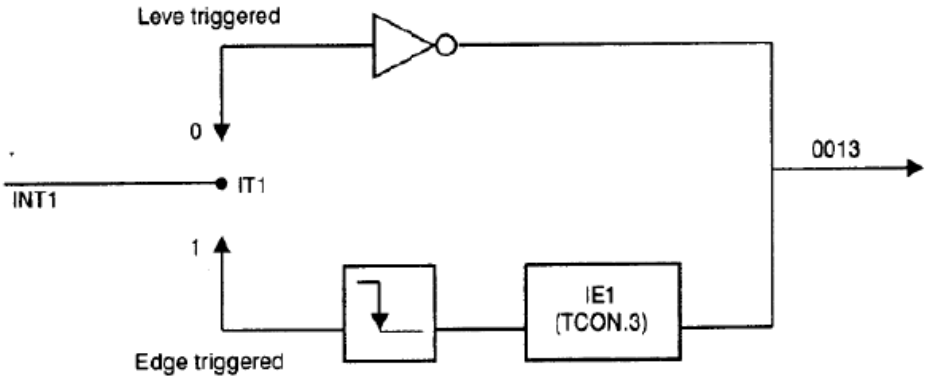


Fig. 8 Activation of INT1

c) Serial Port Interrupt

- In SCON, if RI = l, a data byte is received If TI = 1, a data byte has been transmitted.
- These are ORed together to provide a single interrupt to the processor.
- The interrupt bit in the IE register is used to both send and receive data.
- If IE.4 [ES- Enable serial port interrupt] is enabled, when RI or TI is raised and 8051gets interrupted and jumps to memory address location 0023H to execute the ISR.
- The Fig.7.Showsthe serial interrupt is invoke by TI or RI flags.

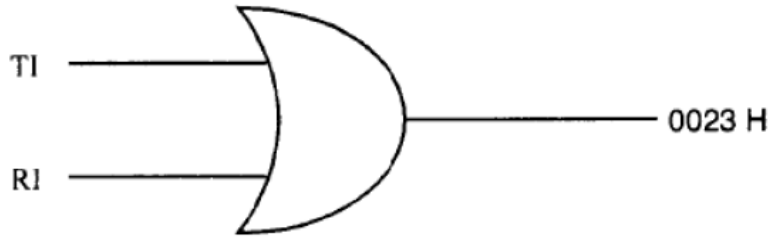


Fig.9. serial port interrupt

3.4.3 **INTERRUPT CONTROL**

All interrupt functions are under the control of the program. The programmer is able to alter control bits in the:

1. Interrupt Enable Register (IE)
2. Interrupt Priority Register (IP) and
3. Timer Control Register (TCON).

1.Interrupt Enable Register (IE)

- The IE register holds the programmable bits that can enable or disable all the interrupts.
- Bit D7 of the IE register (EA) must be set high to allow the rest of the register to take effect.
- If EA = l, interrupts are enabled and will be responded to if their corresponding bits in IE are high.
- If EA = 0, no interrupt will be responded to, even if the associated bit in the EI register is high.

D7	D6	D5	D4	D3	D2	D1	D0
EA	-	-	ES	ET1	EX1	ET0	EX0

Figure 10: IE Register

EA: Enable interrupts bits.

- Set to 1 to permit individual interrupts to be enabled by their enable bits.
- Cleared to 0 by program to disable all interrupts.

ES: Enable serial port interrupt.

- Set to 1 to enable by program.
- Cleared to 0 to disable serial port interrupt.

ET1: Enable/ disable the Timer 1 overflow interrupt.

EX1: Enable external interrupt 1.

- Set to 1 by program to enable **INT1'**interrupt.
- Cleared to 0 to disable **INT1**'interrupt.

ET0: Enable / disable the Timer 0 overflow interrupt.
EX0: Enable/ disable the external interrupt 0.

- Set to 1 by program to enable **INT0'** interrupt.
- Cleared to 0 to disable**INT0**' interrupt.

2. Interrupt Priority Register (IP)

- Interrupt priority (IP) register determines the interrupt priority.
- Bits in IP registers set to 1 give the accompanying interrupt a high priority; a 0 assigns a low priority.
- Interrupts with a high priority can interrupt another interrupt with a lower priority and the lower priority continues after the higher is finished.
- If two interrupts with the same priority occur at the same time, then they have the following ranking:

1. IE0
2. TF0
3. IE1
4. TF1
5. RI/TI

- The bit addressable IP register is shown in Fig.9. If the bit is 0, the corresponding interrupts has a lower priority, otherwise higher priority.