UNIT IV - 8051 MICROCONTROLLER

INTRODUCTION:

• 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 AccessMemory(RAM)

• Besides internal RAM, the 8051 has various *Special Function Registers* (SFR) such as the Accumulator, the B register, and many other control registers.

• 34 8-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

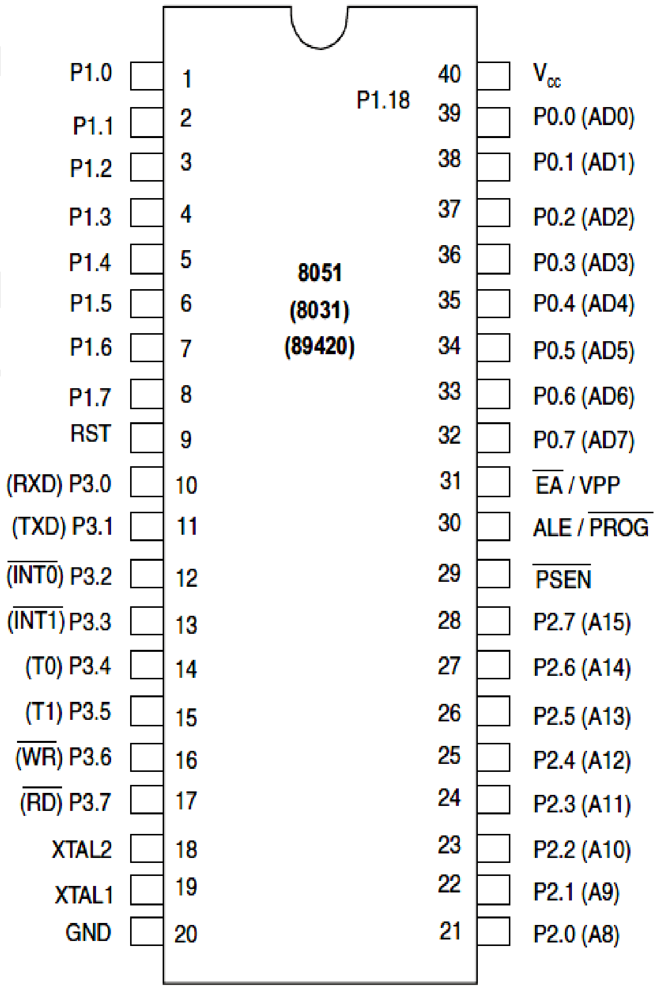
• Some 8051 chips come with UART for serial communication and ADC for analog to digital Conversion

Comparisons of Microcontroller and Microprocessor:

Microcontroller	**Microprocessor**
A microprocessor is a general purpose device which is called a CPU	A microcontroller is a dedicated chip which is also called single chip computer.
A microprocessor do not contain on chip I/O Ports, Timers, Memories etc.	A microcontroller includes RAM, ROM, serial and parallel interface, timers, interrupt circuitry (in addition to CPU) in a single chip.
Microprocessors are most commonly used as the CPU in microcomputer systems.	Microcontrollers are used in small, minimum component designs performing control-oriented applications.

Microprocessor instructions are mainly nibble or byte addressable.	Microcontroller instructions are both bit addressable as well as byte addressable.
Microprocessor based system design is complex and expensive	Microcontroller based system design is rather simple and cost effective
The Instruction set of microprocessor is complex with large number of instructions.	The instruction set of a Microcontroller is very simple with less number of instructions. For, ex: PIC microcontrollers have only 35 instructions**.**
A microprocessor has zero status flag.	A microcontroller has no zero flag.

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 Serial asynchronous communication input or Serial synchronous communication output.

Pin11: TXD 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.

FUNCTIONAL BLOCK DIAGRAM OF 8051 MICROCONTROLLER

- The Intel 8051 contains two separate buses for both program and data. So, it has two distinctive memory spaces of 64K x 8 size for both program and data.
- It is based on an 8 bit central processing unit with an 8 bit accumulator and another 8-bit B register as main processing blocks.
- Other portions of the architecture include few 8 bit and 16b it registers and 8-bit memory

  locations.

- 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.
- 8051 is supported with on-chip peripheral functions like I/O ports, Timers/ Counters, Serial communication port.

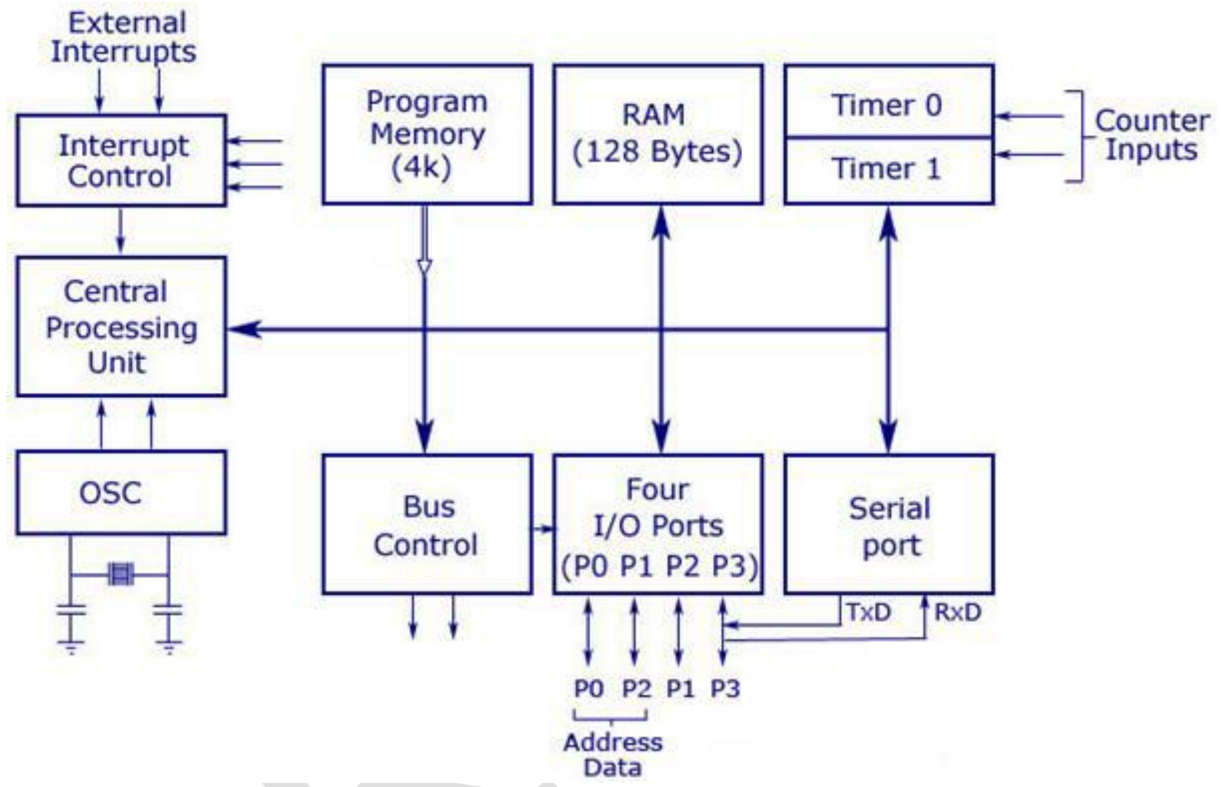


Figure 2: Block Diagram of 8051

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.

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.

ACCUMULATOR

- The accumulator register (ACC 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

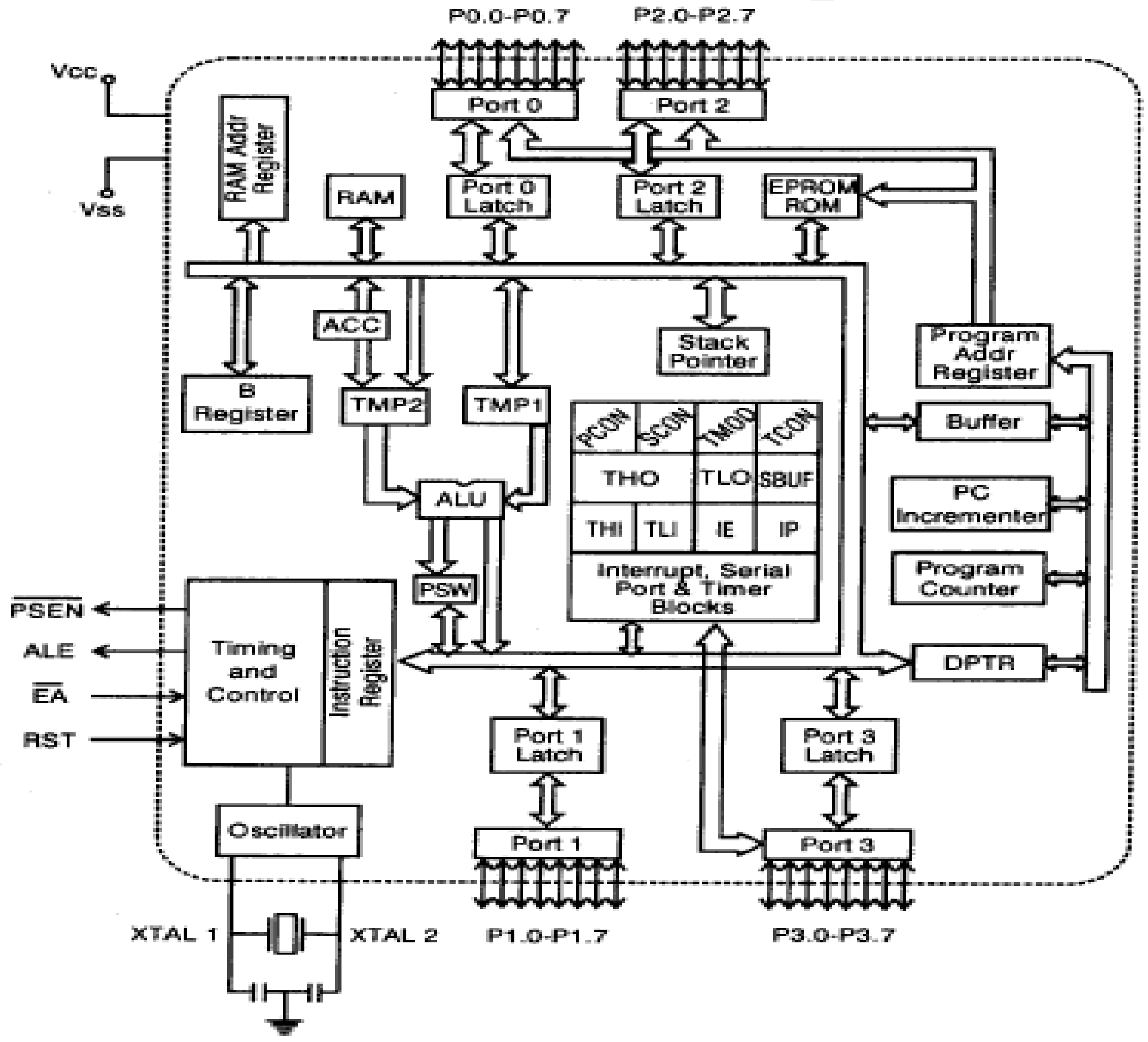


Figure 3: Architecture of 8051

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**
CY	**AC**	**FO**	**RS1**	**RS0**	**OV**	**---**	**P**

Figure 3: Program Status Word (PSW)

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.

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

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 th especial function register bank.

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.

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.

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.

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.

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.

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.

OSCILLATOR

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

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.

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)

RAM and RAM Address Register

Theseblocks provide internal 128 bytes of RAM and a mechanism to address it internally.

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.

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.

1. Immediate Addressing Mode

• When a source operand is a constant rather than a variable, 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 poundsign (#).

• This addressing mode can be used to load information into any of the registers including DPTR register.

Examples:

MOV A, # 18H

$$\boxed{A} \leftarrow 18H$$

MOV B, # 65H

$$\boxed{B} \leftarrow 65H$$

MOV DPTR, #2040H

$$DPL \leftarrow 40H$$

$$DPH \leftarrow 20H$$

2. Register Addressing Mode

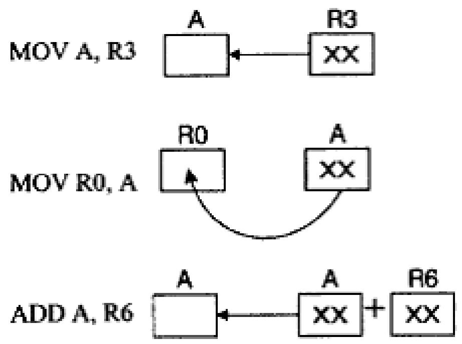
• Register addressing accesses the eight working registers (R0 - R7) of the selected register bank.

• The least significant three bits of the instruction opcode indicate which register is to be used for the operation.

• One of the four banks of registers is to be predefined in the PSW before using register addressing instruction.

• ACC, B and DPTR can also be addressed in this mode.

Examples:



3. Direct Addressing Mode

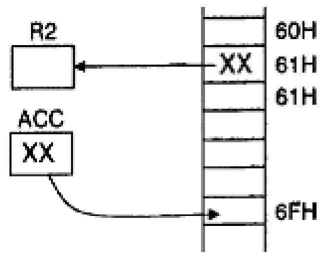
• In the direct addressing mode, all 128 bytes of internal RAM and the SFRs may be addressed directly using the single-byte addressing need to each RAM location and each SFR.

• Internal RAM uses address from 00H to 7FH to address each byte. The SFR addresses exist from 80H to FFH. (Refer Table 2.)

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 content sof A into RAM location whose address is held by R1.

MOV B, @ R0: Move contents of RAM location whose address is held by R0into B.

5. Indexed Addressing Mode

• Only the program memory can be accessed by this mode. This mode is intended for reading lookup tables in the program memory.

• 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.

• 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.

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.

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) 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. Thenit startsto execute from that address.

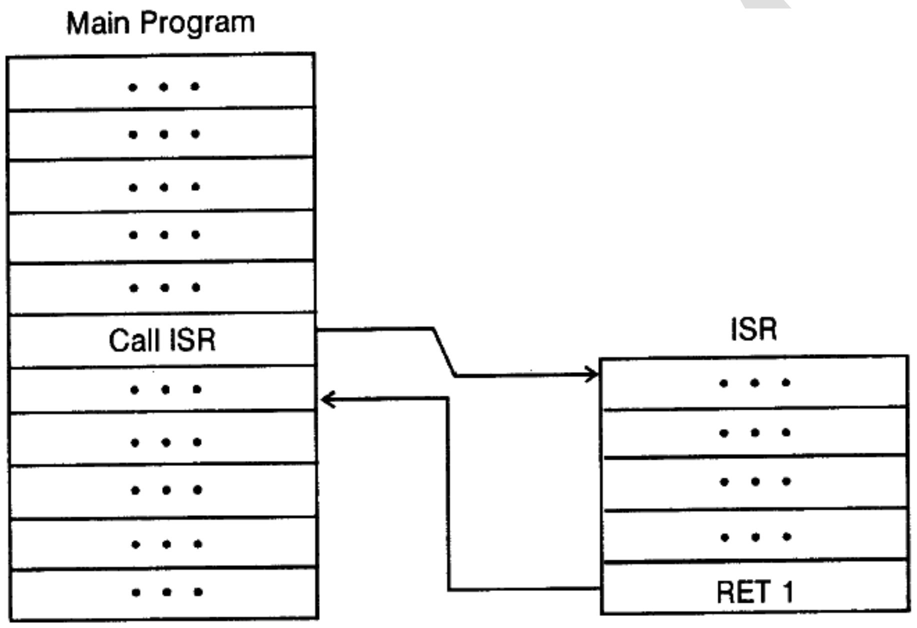


Figure 4:Interrupt Structure

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

• 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.

b) External interrupts

• The external hardware interrupts INT0 andINT1 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.

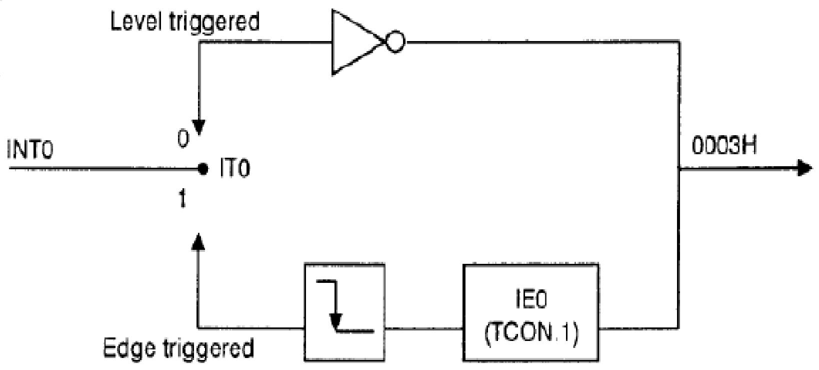


Figure 5: Activation of INT0

• Fig. 4. Shows the activation of INT0 and Fig.5. shows the activation of INT1

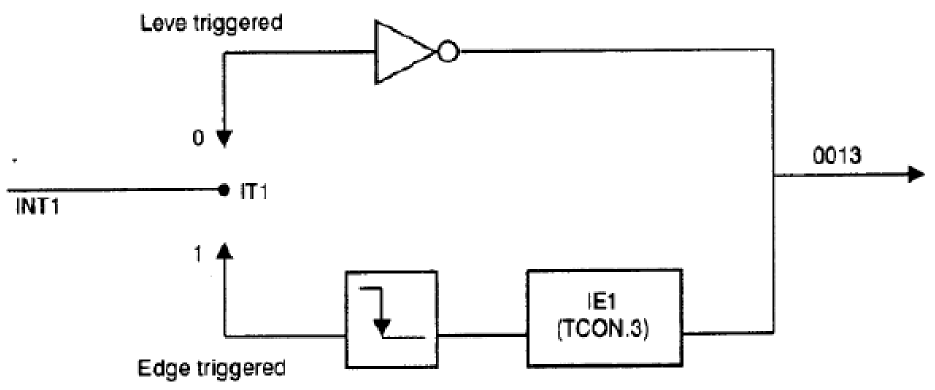


Figure 6: Activation of INT1

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.6.Showsthe serial interrupt is invoke by TI or RI flags.

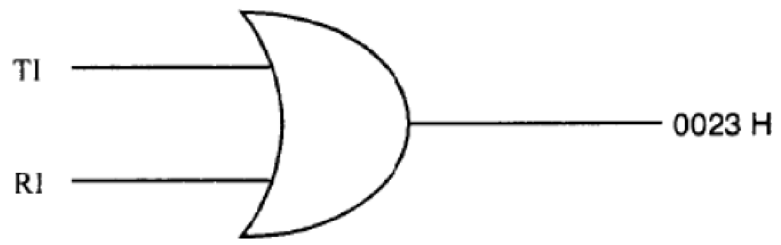


Figure 7: Serial Port Interrupt

INTERRUPTCONTROL

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

• Interrupt Enable Register (IE)

• Interrupt Priority Register (IP) and

• Timer Control Register (TCON).

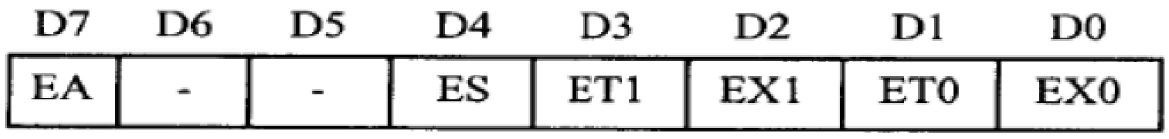
Interrupt Enable Register (IE)

• 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 8: 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.

• Clearedto 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.

• Clearedto 0 to disable**INT0**’ interrupt.

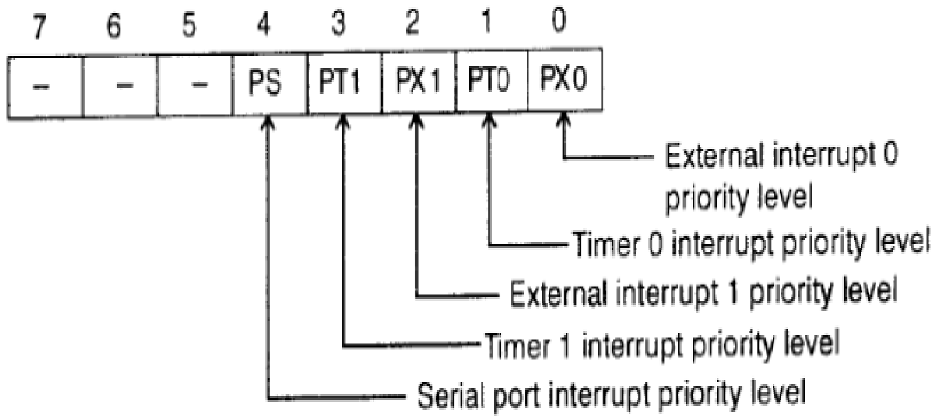
Interrupt Priority Register (IP)

• 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:


7 6 5 4 3 2 1 0
– – – PS PT1 PX1 PT0 PX0
External interrupt 0 priority level
Timer 0 interrupt priority level
External interrupt 1 priority level
Timer 1 interrupt priority level
Serial port interrupt priority level

1. IE0

2. TF0

3. IE1

4. TF1

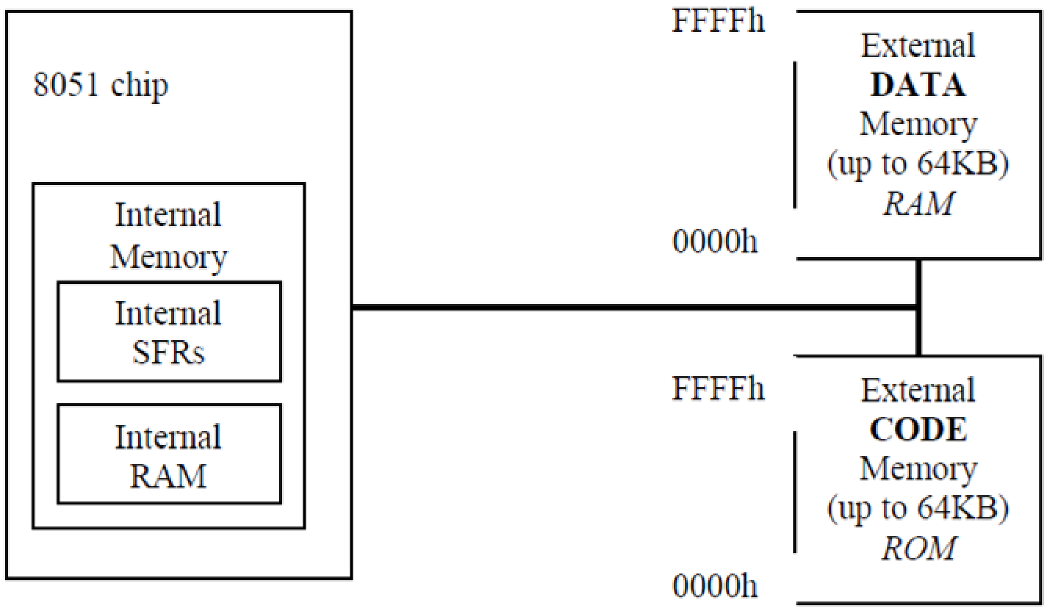
5. RI/TI

• 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.

MEMORY ORGANISATION:

• The 8051 has a separate memory space for programs (Code) and data.

• Program memory stores the programs to be executed, while data memory stores the data like intermediate results, variable and constants required for the execution of the program.


8051 chip
Internal Memory
Internal SFRs
Internal RAM
FFFFh
0000h
External DATA Memory (up to 64KB) RAM
FFFFh
0000h
External CODE Memory (up to 64KB) ROM

External Program (Code) Memory

• The executable program is stored in this code memory.

• The code memory size is limited to 64KBytes (in a standard 8051).

• The code memory is read-only in normal operation and is programmed under special conditions. e.g. it is a PROM or a Flash RAM type of memory.

External RAM Data Memory

• This is read-write memory and is available for storage of data. Up to 64KBytes of external RAM data memory is supported (in a standard 8051).

Internal Memory

The 8051's on-chip memory consists of 256 memory bytes organized as follows:

The first 128 bytes of internal memory is organized as shown in figure 2, and is referred to as Internal

RAM, or IRAM.

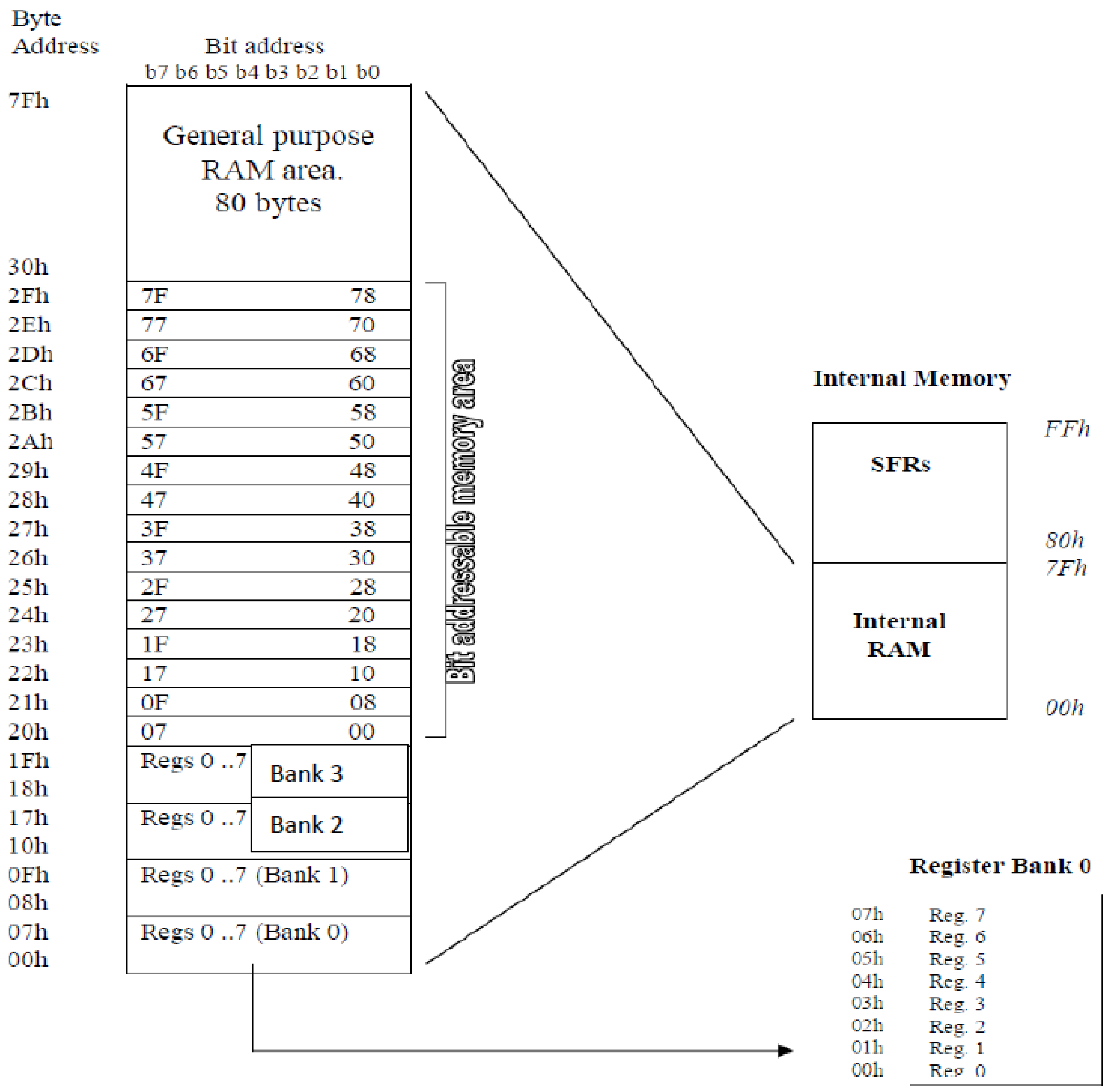


Figure : Organization of Internal RAM (IRAM) memory

Register Banks: 00h to 1Fh

The 8051 uses 8 general-purpose registers R0 through R7 (R0, R1, R2, R3, R4, R5, R6, and R7).

These registers are used in instructions such as:

ADD A, R2 ; adds the value contained in R2 to the accumulator

Bit Addressable RAM: 20h to 2Fh

• The 8051 supports a special feature which allows access to bit variables.

• This is where individual memory bits in Internal RAM can be set or cleared.

• In all there are 128 bits numbered 00h to 7Fh.

• Being bit variables any one variable can have a value 0 or 1.

• A bit variable can be set with a command such as SETB and cleared with a command such as CLR.

General Purpose RAM: 30h to 7Fh

• These 80 bytes of Internal RAM memory are available for general-purpose data storage.

• Access to this area of memory is fast compared to access to the main memory and special
instructions with single byte operands are used.

• However, these 80 bytes are used by the system stack and in practice little space is left for general storage.

• The general purpose RAM can be accessed using direct or indirect addressing modes.

SFR Registers

• The SFR registers are located within the Internal Memory in the address range 80h to FFh, as shown in figure 3.

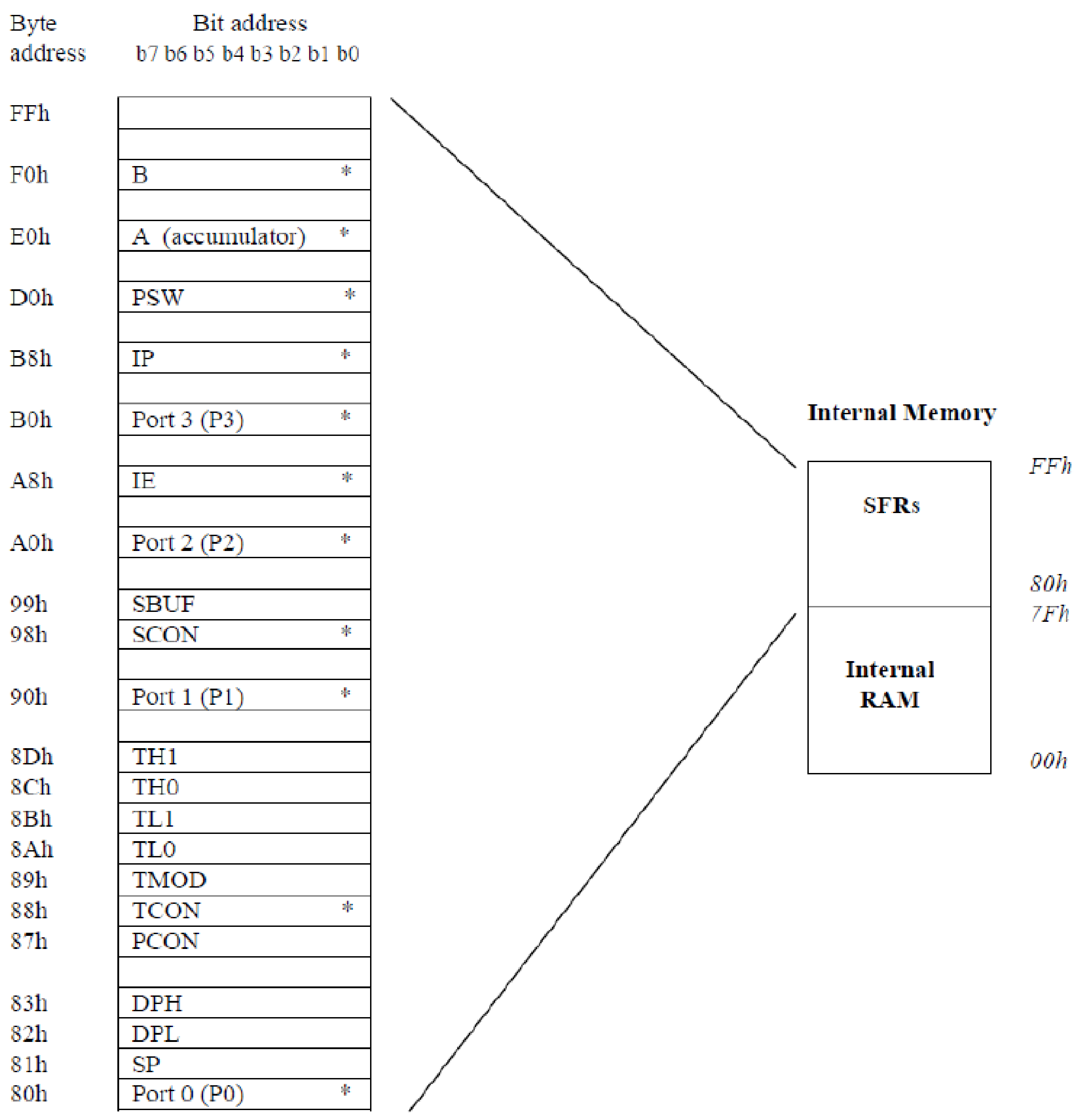
• Each SFR has a very specific function.

• Each SFR has an address (within the range 80h to FFh) and a name which reflects the purpose of the SFR.

• Although 128 byes of the SFR address space is defined only 21 SFR registers are defined
in the standard 8051.

• Undefined SFR addresses should not be accessed as this might lead to some unpredictable results.

• Note some of the SFR registers are bit addressable. SFRs are accessed just like normal Internal RAM locations.



* indicates the SFR registers which are bit addressable

Figure. SFR Register Layouts

Timer/Counters

• The 8051 has two 16-bit timer/counters. These two timer/counters can be programmed independently.

• There is a bit in the TMOD SFR that specifies whether it is a timer or a counter.

• If this bit is set, the timer/counter will work as a counter; and

• if this bit is reset, the timer/counter will work as a timer.

TIMER MODE

• When a timer/counter is functioning as a timer, the timer register (TH 1 and/or TL1 for Timer 1 or TH0 and/or TL0 for Timer 0) is incremented after every machine cycle.

• That is, it will be working at 1/12th of the oscillator frequency because each machine cycle has got 12 oscillator periods.

• As an example, after starting the spin motor in a washing machine, the next operation, i.e. the motor shut down, is performed after a fixed time interval.

MODES OF TIMER:

These timers can function in four different modes, namely: **i) mode 0, ii) mode 1,iii) mode 2 and iv) mode 3.**

• These modes are achieved by setting certain bits in the TMOD-register.

• Mode 0 to mode 2 are common for both the timers but not mode 3

i) Mode 0:

• In this mode, TL0and TH0 for Timer 0 (or TLl and THl for Timer 1) are used as a 13-bit register,

• i.e. all the 8 bits of the TL0 or TLl are utilized and the five lower most bits of the TH0 or THl are used for counting purposes.

• As the count rolls over from all 1s in the register to all 0s, the interrupt flag is set.

• This timer interrupt flag is a bit, namely TF0 (for Timer 0) in the TCON, which is a special function register.

• From Figure 1, it is clear that if C/T = 0, then the register is incremented after every machine cycle.

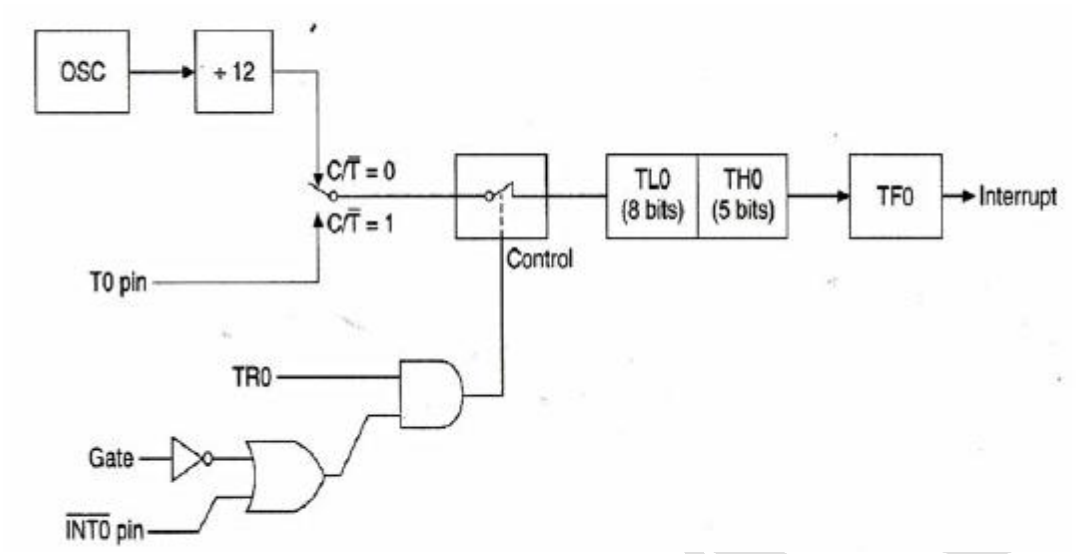


Figure 1: Timer0, mode0 _13-bit Counter

• **Case 1:**

TR0 = 1 (high)

Gate = 0 (low) and INT0= 0 (low)

• **Case 2 :**

TRO = 1 (high)

INTO = 1 (high) Gate = 1 (high)

• Mode 0 for both the timer/counters is the same, the only thing to do is to change TR0 to TR1, INT0' to INTl', and Gate bit for Timer 0 to Gate bit for Timer 1.

ii)Mode 1:

This mode (Figure 2) is similar to mode 0, except that in this, 16 bits, that is, full TL0 and

TH0 for Timer 0 (or TLl andTH1 for Timer 1) are used for counting. So, the interrupt flag will be

set only when the 16 bits go from all 1s to all 0s.

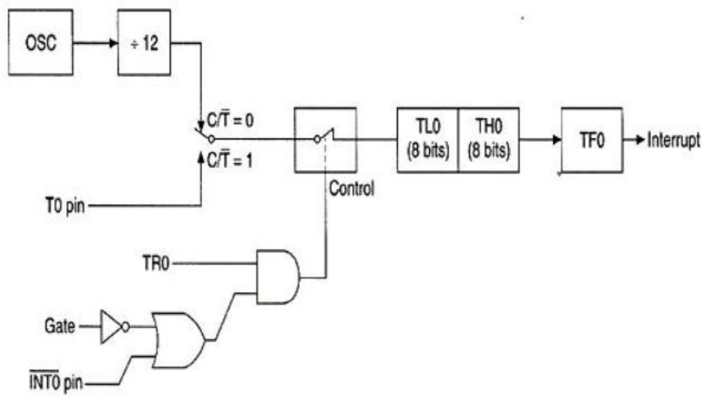


Figure 2: Timer 0, mode 1 _ 16-bit counter.

iii) Mode 2:

• In this mode, the timer register is 8 bits wide. TL0 for Timer 0 (or TLl for Timer 1) is used

for this purpose (Figure 3).

• This mode is also called the auto-reload mode as the timer generates an interrupt on

overflow and after generating the interrupt, will also reload the present value from TH0

into TL0. This present value can be put in the TH0 through software.

• The interrupt flag is set when TL0 goes from all 1s to all 0s.After generating an interrupt,

it also reloads the TL0 with the value from TH0 (80H in this case) and then starts counting

again.

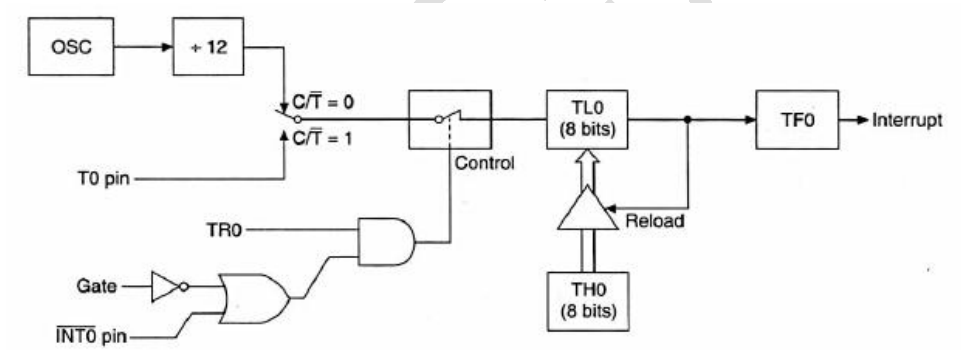


Figure 3: Timer 0, mode 2 _ auto reload.

iv) Mode 3:

• "If the Timer 0 is put into mode 3 (Figure 4), then it acts as two 8-bit counters(TL0 and

TH0 become two separate counters).

• In this case, all the Timer 0 control bits (C / T',Gate, TRO, TFO and INT0') are used by

TL0 itself and TH0 register is locked into a timerfunction.

• TH0 is counting machine cycles and has taken over the use of TRl and TFl from Timer 1.

• Therefore TH0 will now control Timer 1 interrupt. If the Timer 1 is put into mode 3,it just holds the count.

• The effect is same as setting TRI = 0, hence opening the switch.

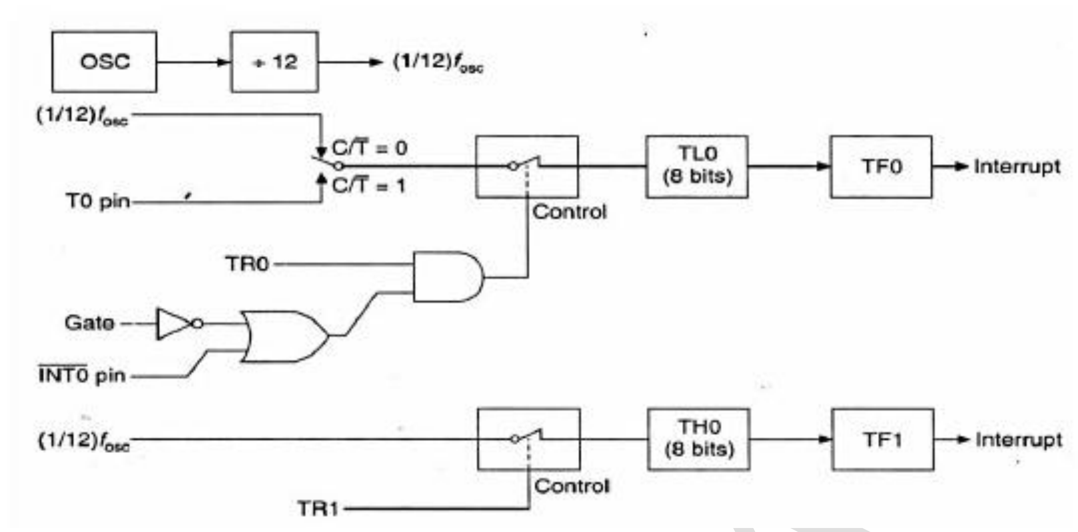


Figure 4: Timer 0, mode 3 _ split to two 8-bit counters

TIMER CONTROL AND STATUS REGISTER:

• The special function registers TMOD and TCON are used to control the

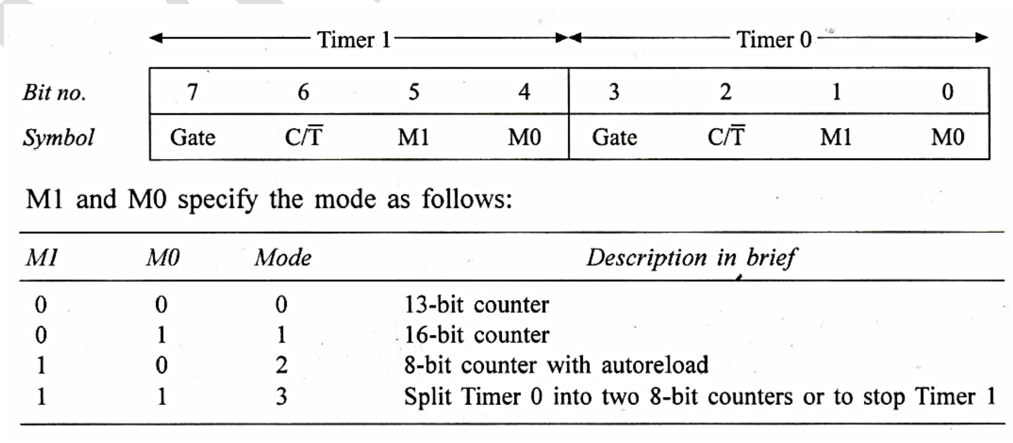
timer/counterfunctions.

• When we write into these registers, the data is latched into them and takes effect atS1P1 of

the next instruction (first cycle).

The bits of these registers are described below:

1. **TMOD: Timer Mode Control Registers**



	Timer 1				Timer 0			
Bit no.	7	6	5	4	3	2	1	0
Symbol	Gate	C/$\overline{T}$	M1	M0	Gate	C/$\overline{T}$	M1	M0

M1 and M0 specify the mode as follows:

M1	*M0*	*Mode*	*Description in brief*
0	0	0	13-bit counter
0	1	1	16-bit counter
1	0	2	8-bit counter with autoreload
1	1	3	Split Timer 0 into two 8-bit counters or to stop Timer 1

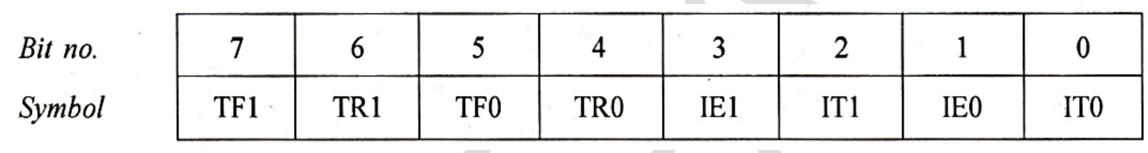
• If C/T = 1, the timers function as counters to count the negative transitions' at T0 or T1 pins,

• If C/T= 0, the timers function as timers, that is, they basically count the number of machine cycles.

• Gate = 0 means that the timer is controlled by TR1 or TR0 only, irrespective of INT0' or INT1'.

• Gate = 1 means that the timer control will depend on INT0'or INT1' and also on TR0 or TR1 bits.

2. TCON: Timer Control Register



Bit no.	7	6	5	4	3	2	1	0
Symbol	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

• **TF 1:**Timer 1 overflow flag. Set by hardware when the timer/counter overflows. Cleared by hardware when the processor vectors to the interrupt routine.

• **TR1:**Timer 1 run control bit. Set/cleared by software to turn the timer/counter on/off.

• **TF0:** Timer 0 overflow flag. Set by hardware when the timer/counter overflows. Cleared by hardware when the processor vectors to the interrupt routine.

• **TR0:** Timer 0 run control bit. Set/cleared by software to turn the timer/counter on/off.

• **IE 1:** Interrupt 1 edge flag. Set by hardware when the external interrupt edge is Detected. Cleared when the interrupt is processed.

• **IT1:** Interrupt 1 type control bit. Set/cleared by software to specify the falling edge/low level triggered external interrupts. When IT1 = 1 then INTl' is falling edge triggered, otherwise when IT1 = 0 the INT1' is low-level triggered.

• **IE0:** Interrupt 0 edge flag. Set by hardware when the external interrupt edge is detected. Cleared when the interrupt is processed.

• **IT0**: Interrupt 0 type control bit. Set/cleared by software to specifiy the falling edge/low level triggered external interrupts. When IT0 = 1 then INT0' is falling edge triggered, otherwise when IT0 = 0 the INT0' is low-level triggered.

SERIAL COMMUNICATION.

- In serial communication, a single data can be used instead of 8-bit data lines.
- It is uses either asynchronous data format or synchronous data format methods.
- Serial transmission and reception can be easily achieved using UART devices.
- 8051 has built in UART, communication circuit use SUBF register to hold the data.
- Port 3, bit-0 act as RXD pin and bit-1 act as TXD for serial port.
- SUBF has two registers, one for write only TXD pin and another one for read only RXD pin.
- It can be programmed in 4 types of modes.
- The modes are: mode 0, mode 1, mode 2, and mode 3.

Mode 0:

- In serial communication are done through RXD pin.
- TXD pin acts as shift register clock and rising edge of the clock.
- Baud rate of the mode 0 is fixed to ½ of oscillator frequency

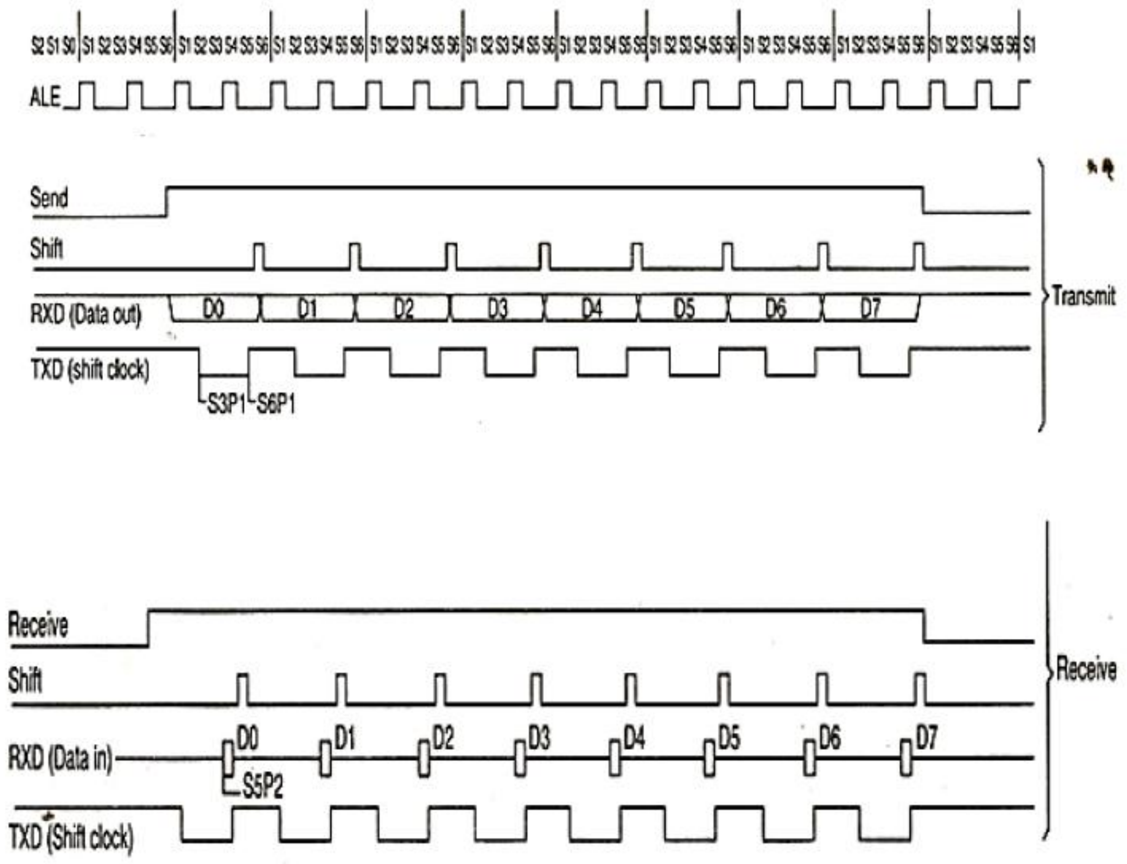


Figure 5:Mode 0

Mode 1: Standard UART

• It is a standard asynchronous serial communication mode.10-bits are transmitted or received, Baud rate is variable,

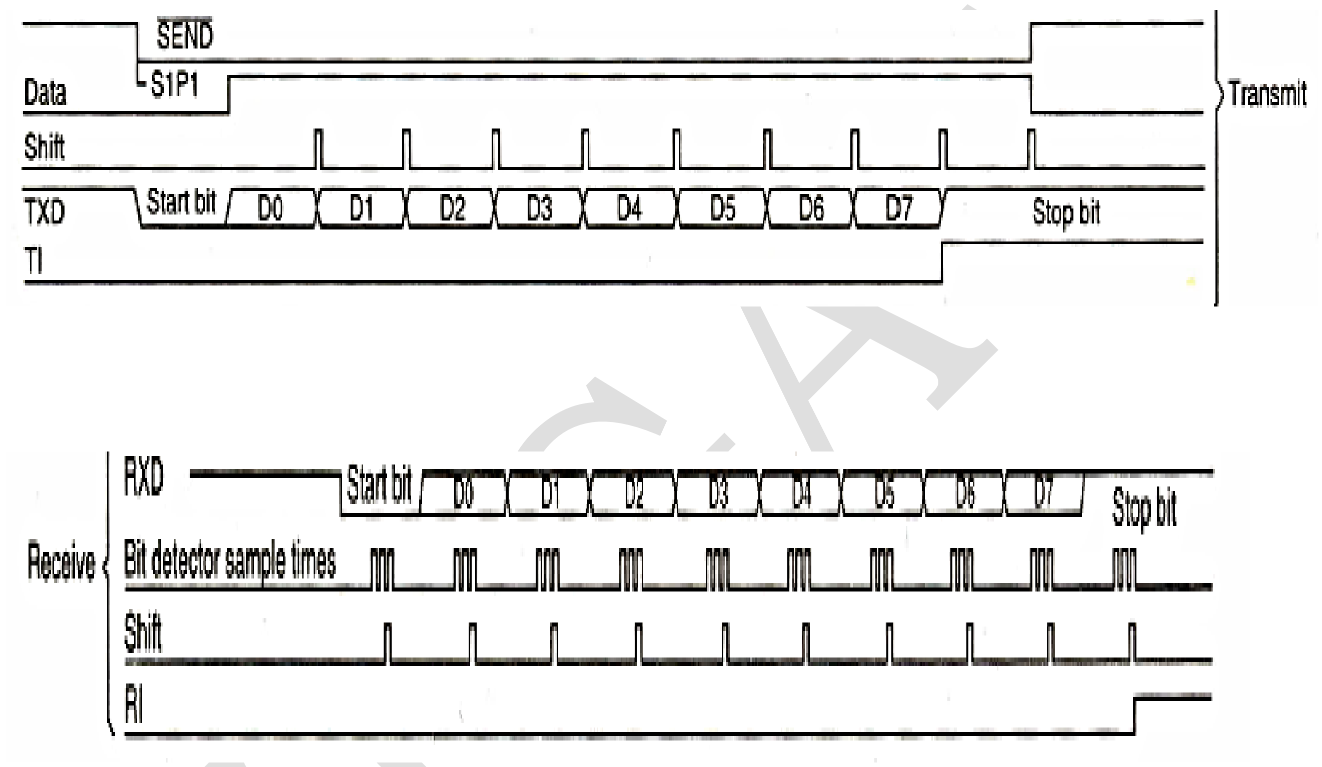


Figure 6: mode1

Mode2: Multiprocessor Mode

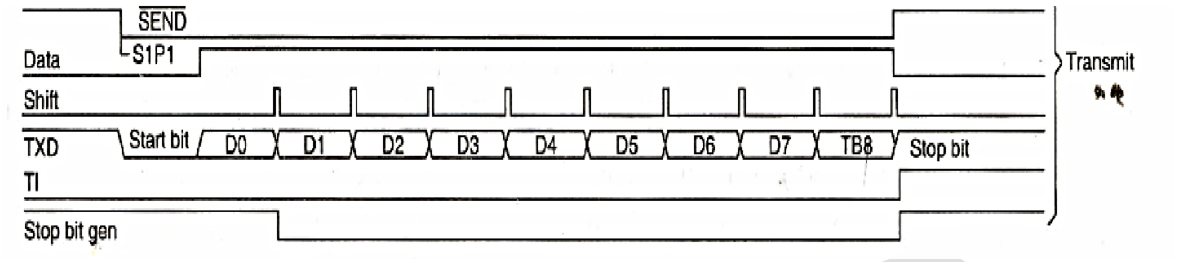
• 11-bits are transmitted through TXD or received through RXD.

• it consists of one start bit(0),8 data bits, programmable 9thbit and a stop bit(1).

• On transmission 9th data bit is 0/1, 8-bit is loaded into SBUF and copied from TB8 in SCON register. On reception it goes into RB8 of SCON register of SFR.

Mode 3:

• it is same as mode 2 except baud rate, since mode 3 supports variable baud rate and calculated similar to mode1 using timer1.



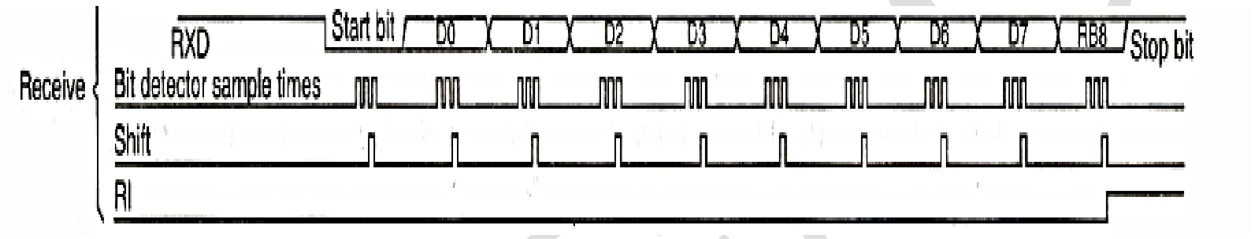


Figure 7: mode 2 and mode 3

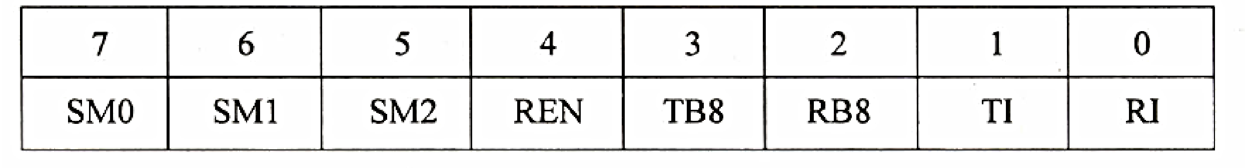
Serial Control Register (SCON)

• This special function register (SFR) controls the operations of the serial port.

• This register is used to define the operatingmodes.

• This also receives the 9th bit and contains the transmit and receive interrupt flags aswell.

Serial Control Register (SCON):



7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

I /O PORTS

Each port of 8051 has bi-directional capability.

Port 0 is called 'true bidirectional port' as it floats (tri-stated) when configured as input.

Port-1, 2, 3 are called 'quasi bidirectional port'**.**

• To communicate data with the external world the microcontroller needs ports.

• The ports may support either parallel or serial data transfer.

• It has 4 I/O ports namely, **Port 0, Port 1, Port 2 & Port 3**

• Port 1: is exclusively for input & output functions.

• Port 0, 2 & 3: perform functions other than parallel data transfer.

• All 4 ports are bidirectional.

• The 8 port pins are connected through 8 D type port latches.

PORT 0:

Port -0 has 8 pins (P0.0-P0.7).

• Port -0 has 8 pins (P0.0-P0.7).

• 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 bi-directional I/O port.

• Port-0 latch is written to with 1's when used for external memory access.

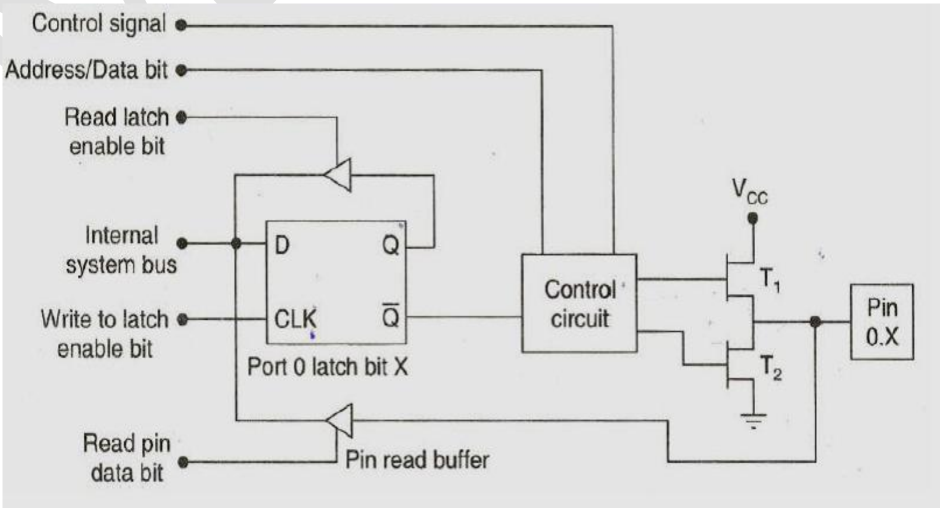


Figure 1: PORT 0 Configuration

Port 1:

• Port-1 has 8 pins (P1.1-P1.7)

• 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 to the pin by the external device then the external source must sink current due to internal pull-up.

• If the external device is not able to sink the current the pin voltage may rise, leading to a possible wrong reading.

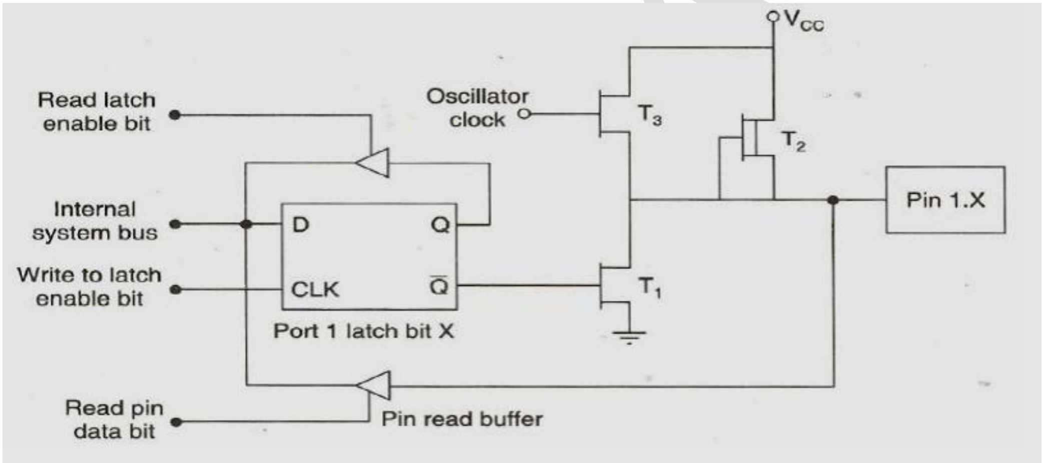


Figure 2: PORT 1 Configuration

Port 2 :

• Port-2 has 8-pins (P2.0-P2.7)

• 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.

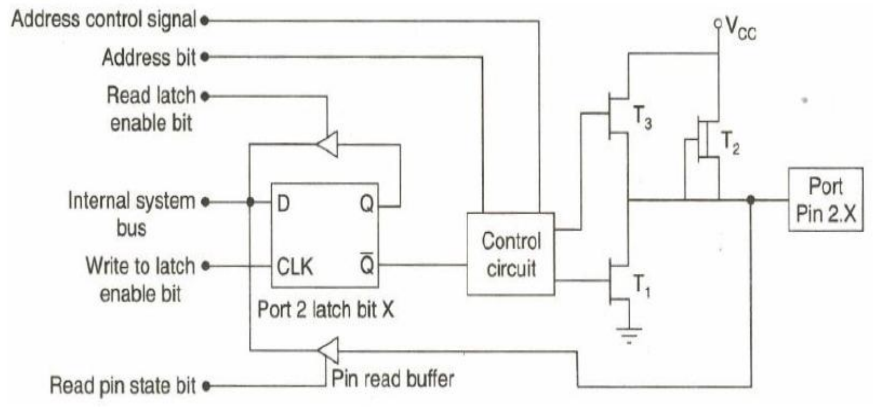


Figure 3: PORT 2 Configuration

PORT 3:

• Port-3 has 8 pin (P3.0-P3.7)

• Port-3 pins have alternate functions.

• 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

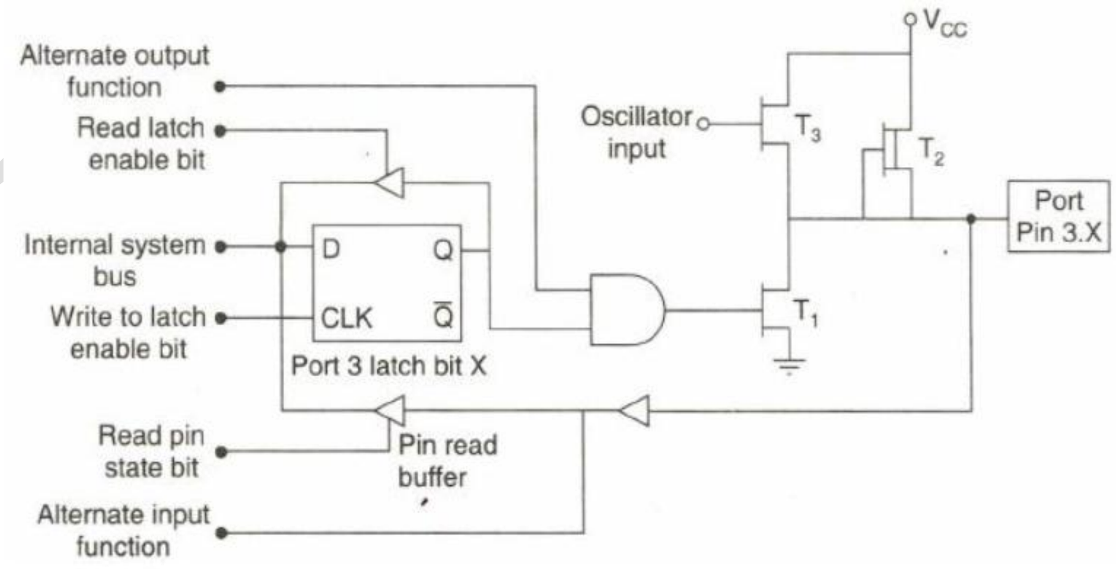


Figure 3: PORT 2 Configuration

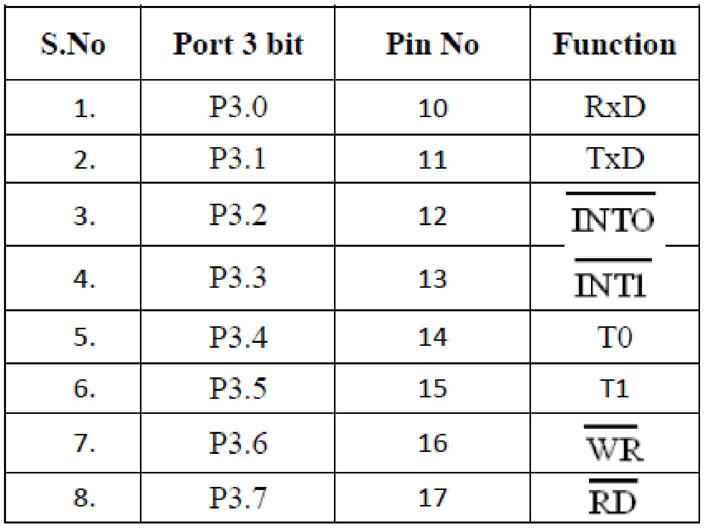
Alternate Functions of Port 3:

• P3.0 and P3.1 are used for the RxD (Receive Data) and TxD (Transmit Data) serial communications signals.

• Bits P3.2 and P3.3 are meant for external interrupts.

• Bits P3.4 and P3.5 are used for Timers 0 and 1 and P3.6 and P3.7 are used to provide the write

and read signals of external memories connected in 8031 based systems



S.No	Port 3 bit	Pin No	Function
1.	P3.0	10	RxD
2.	P3.1	11	TxD
3.	P3.2	12	$\overline{\text{INT0}}$
4.	P3.3	13	$\overline{\text{INT1}}$
5.	P3.4	14	T0
6.	P3.5	15	T1
7.	P3.6	16	$\overline{\text{WR}}$
8.	P3.7	17	$\overline{\text{RD}}$

Table1: PORT 3 alternate functions