

UNIT-I 8BIT EMBEDDED PROCESSOR

Microcontrollers for an Embedded System – 8051 – Architecture – Addressing Modes – Instruction Set – Program and Data Memory – Stacks – Interrupts – Timers/Counters – Serial Ports – Programming.

UNIT-I (8051 MICROCONTROLLER)

2 Marks Questions with Answers

1. What is meant by microcontroller? [APR/MAY 2011]

Microcontroller is a device that includes microprocessor, memory and I/O port lines on a single chip, fabricated using VLSI technology.

2. What is Microcontroller and Microcomputer? [APRIL/MAY/2011]

Microcontroller is a device that includes microprocessor, memory and I/O signal lines on a single chip. Microcomputer is a computer that is designed using microprocessor as its CPU. It includes microprocessor, memory and I/O.

3. Compare Microprocessor and Microcontroller. [May 2009, NOV 2006, NOV 2011]

What are the differences between a microprocessor and a microcontroller? (May 2007, Nov 2011, 2018)

Sl.No	Microprocessor	Microcontroller
1.	A microprocessor is a general purpose device which is called a CPU.	A microcontroller is a dedicated chip which is also called a single chip computer.
2.	A microprocessor does not contain on-chip I/O Ports, Timers, Memories etc.	A microcontroller includes RAM, ROM, serial and parallel interface, timers, interrupt circuitry in a single chip.
3.	Microprocessor is used as the CPU in microcomputer system.	Microcontroller is used to perform control-oriented applications.
4.	Microprocessor instructions are nibble or byte addressable	Microcontroller instructions are both bit addressable as well as byte addressable.

4. List the features of 8051 microcontroller. [MAY 2007] [NOV 2007, NOV 2011]

The 8051 is an 8-bit microcontroller:

- ✓ The CPU can work on only 8 bits of data at a time
- ✓ The 8051 has
 - 128 bytes of RAM
 - 4K bytes of on-chip ROM
 - Two timers

5. List the applications of microcontroller. [MAY/JUNE 2009]

Microcontroller is used in various control applications:

- Fire detection in building.
- Industrial control (process control)
- Motor speed control (stepper motor control)

- Peripheral devices (printer)
- Standalone devices (color Xerox machine)
- Automobile applications (power steering)
- Home applications (washing machine, AC)

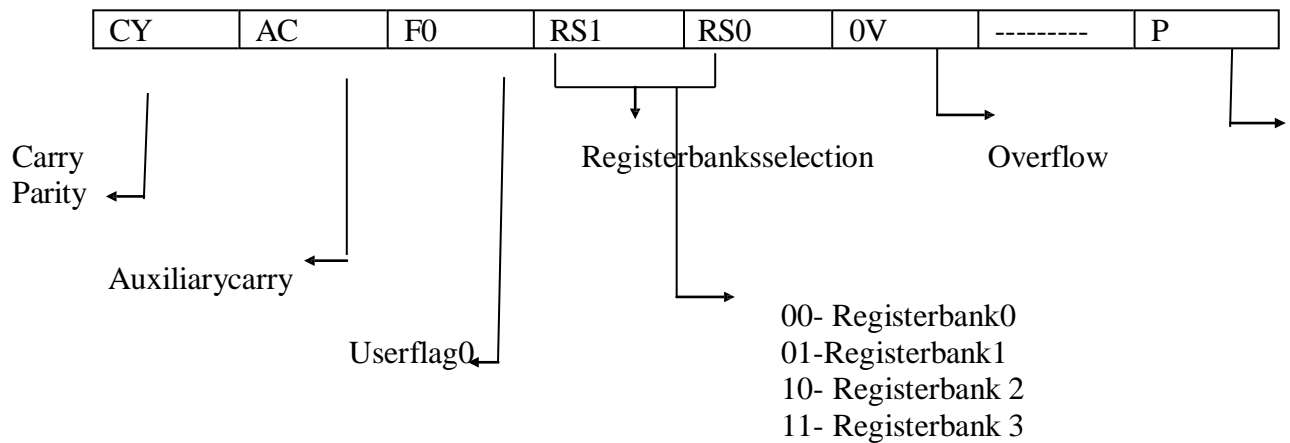
6. Mention the different operand types used in 8051. (NOV 2012)

Register operand, immediate operand, Direct operand, Direct-offset operand and Indirect operand.

7. What are the addressing modes supported by 8051? [May 2010, May 2009, MAY 2008, NOV 2011] What are the different ways of operand addressing in 8051? (May 2016, NOV/Dec 2018)

1. Immediate addressing mode
2. Direct Addressing mode
3. Register addressing mode
4. Register indirect addressing mode
5. Indexed addressing mode

8. Draw the format of PSW of 8051. (May 2015)



9. State the function of RS1 and RS0 bits in the flag register of Intel 8051 microcontroller. [NOV 2011]

How do you select the register bank in 8051 micro-controller? (May 2010, NOV 2008, May 2008, May 2016)

Mention the number of register banks and their addresses in 8051. (NOV 2015)

✓ RS1 & RS0 are used to indicate which bank is currently in use.

RS1	RS0	Registerbank Selection	Addresses of Registerbanks
0	0	RegisterBank0	00H to 07H
0	1	RegisterBank1	08H to 0FH
1	0	RegisterBank2	10H to 17H
1	1	RegisterBank3	18H to 1FH

RS1, RS0 – Register bank select bits

10. In the program status word of 8051, the bits RS0 and RS1 are 1 and 0, then which register bank is selected for operation? (AU May 2013)

Register Bank 1 is selected.

11. Explain the 16-bit registers DPTR of 8051. [MAY/JUNE 2007]

What is the use of DPTR? (May 2009)

DPTR: DPTR stands for data pointer. DPTR consists of a high byte (DPH) and a low byte (DPL). Its function is to hold a 16-bit address.

It may be manipulated as a 16-bit data register or as two independent 8-bit registers. It serves as a base register in external data transfer.

12. What are the advantages of microcontroller over microprocessor?

What are the advantages of using a microcontroller in place of a microprocessor?

(AU May 2011 MAY 2008)

- The overall system cost is low, as the peripherals are integrated in a single chip.
- The size is very small.
- The system is easy to troubleshoot and maintain.
- The system is more reliable.

13. Define XTAL1 and XTAL2. [MAY/JUNE 2009]

- These two pins are connected to Quartz crystal oscillator which runs the on-chip oscillator.
- If used as a source other than the crystal oscillator, XTAL1 and XTAL2 is left unconnected.

14. Name the special function registers available in 8051. [MAY 2007, NOV 2007, May 2008, May 2010]

- Accumulator
- B Register
- Program Status Word
- Stack Pointer

- DataPointer
- Port0,Port1, Port2&Port3
- Interruptprioritycontrol register
- Interruptenablecontrolregister

15. What is the importance of special function registers available in 8051 microcontroller?

- ✓ The 8051 operations that do not use the internal 128 byte RAM address from 00H to 7FH.
- ✓ 128 byte RAM locations used by a group of special internal registers.
- ✓ SFRs (special function registers), which may be addressed like internal RAM.

16. How is stack implemented in 8051? (or) What is stack pointer and write the stack level of 8051? (NOV 2007)

- ✓ The 8051 LIFO: Stack can reside anywhere in the internal RAM.
- ✓ It has 8 bit stack pointer to indicate the top of the stack using PUSH and POP instructions.
- ✓ During PUSH the SP is incremented by one and POP the SP is decremented by one.

17. What is the use of RET and RETI instruction in 8051?

RET – Return to subroutine

Used to return from a subroutine previously entered by CALL instructions
RET – Return to interrupt

Used at the end of interrupt service routine (ISR)

18. List the 8051 instructions that affect the flags. [NOV/DEC 2007]

ADD, ADDC, DIV, MUL and SUBB

19. List the 8051 instructions that always clear the carry flag.

CLRC, DIV, MUL

20. Give the function of the SP register of 8051. [NOV/DEC 2011]

SP: SP stands for stack pointer.

- ✓ SP is an 8-bit wide register.
- ✓ It is incremented before data is stored during PUSH and CALL instructions.
- ✓ The stack pointer is initialized to 07H after a reset.

21. What are the functions of the following signals of 8051? ALE/PROG, PSEN. (AU Nov 2010)

ALE (Address Latch Enable):

- ✓ This is an output pin and is active high.
- ✓ When connecting an 8051 to external memory, Port 0 provides both address and data.
- ✓ If ALE=0, Port 0 (D₀-D₇). If ALE=1, it has (A₀-A₇).

PSEN (Program Store Enable):

- ✓ This is an output pin.

- ✓ In 8051-based system in which an external ROM holds the program code, this pin is connected to the OE pin of the ROM.

22. Give the alternate functions for the port pins of port 3. [APRIL/MAY 2011] Which port is used as a multifunction port? List the signals. (April 2017)

Port 3 is used as a multifunction port and its signals are

- ✓ RD – Read data control output
- ✓ WR – Write data control output
- ✓ T1 – Timer/Counter 1 external input or test pin
- ✓ T0 – Timer/Counter 0 external input or test pin
- ✓ INT1 – Interrupt 1 input pin
- ✓ INT0 – Interrupt 0 input pin
- ✓ TXD – Transmit data pin for serial port in UART mode
- ✓ RXD – Received data pin for serial port in UART mode

23. List the ports available in 8051. (or) **How many ports are bit addressable in 8051? (NOV 2011, NOV 2009)**

- ✓ The four ports are P0 (Port 0), P1 (Port 1), P2 (Port 2) and P3 (Port 3).
- ✓ Four ports are bit addressable.

24. How does the status of EA pin affect the access to internal and external program memory?

- ✓ If EA=0, 8051 can access the external program memory.
- ✓ EA=1, access the internal program memory.

25. What is the size of the on-chip program memory and on-chip data memory of 8051 microcontroller? (AU May 2012, NOV 2011)

- ✓ The size of the on-chip program memory of 8051 microcontroller : 4K
- ✓ The size of the on-chip data memory of 8051 microcontroller : 128 bytes

26. What is the difference between timer and counter operations in 8051?

- ✓ The timer, counts the internal clock pulses whose frequency is 1/12th of oscillator frequency.
- ✓ The counter, counts the external clock pulses which are given through T0 pin and T1 pin of 8051.

27. Define watchdog timer.

- ✓ Watchdog timer is a dedicated timer to take care of system malfunction.
- ✓ It can be used to reset the controller during software malfunction.

28. What is the function of TMOD register?

- ✓ TMOD (timer mode) register is used to set the various timer operation modes.
- ✓ TMOD is dedicated to the two timers (Timer 0 and Timer 1).

29. If a 12 MHz crystal is connected with 8051, how much is the time taken for the count in timer 0 to get incremented by one?

$$\begin{aligned} \text{Baudrate} &= \frac{\text{oscillator frequency}}{12} \\ &= \frac{12 \text{ MHz}}{12} = 1 \text{ MHz} \end{aligned}$$

$$T=1/f=1/1\text{MHz}=1\text{microsec}$$

30. What is the time duration for one state and one machine cycle if a 6MHz crystal is connected to 8051?

$$\text{Clock frequency} = 6\text{MHz}/12 = 0.5\text{MHz}$$

$$\text{One } T_{\text{state}} = 1/\text{clock frequency} = 1/0.5\text{MHz} = 2\text{microsec}$$

The time taken to execute a machine cycle is 12 clock periods.

31. What happens in power-down mode of 8051 microcontroller? (May 2016)

- ✓ The memory location of power-down RAM can be maintained through a separate small battery backup supply.
- ✓ So that the content of these RAM can be preserved during power failure conditions.

32. Define baud rate. (May 2016)

- ✓ Baud rate is used to indicate the rate at which data is being transferred.
- ✓ Baud rate = 1/Time for a bit cell

33. Give the register IE format of 8051. (or)

Mention the use of interrupt enable register in 8051. (May 2009)

EA	—	ET2	ES	ET1	EX1	ET0	EX0
-----------	----------	------------	-----------	------------	------------	------------	------------

- ✓ EA – Enable all control bit
- ✓ ET2 – Timer 2 interrupt enable bit
- ✓ ES – Enable serial port control bit
- ✓ ET1 – Enable Timer 1 control bit
- ✓ EX1 – Enable external interrupt 1 control bit
- ✓ ET0 – Enable Timer control bit
- ✓ EX0 – Enable external interrupt control bit

34. Name the interrupt sources of 8051 for which the priority levels are highest, lowest respectively.

- | | |
|--------------------|------------------|
| 1) IEO | highest priority |
| 2) TFO | ↓ |
| 3) IE1 | |
| 4) TF1 | |
| 5) Serial RI or TI | lowest |

35. What is the function of IP register in 8051? (or) What register keeps track of interrupt priority in the 8051? Explain. (NOV 2009)

The IP register is used to set high priority to one or more interrupts in 8051.

----	----	---	PS	PT1	PX1	PT0	PX0
------	------	-----	----	-----	-----	-----	-----

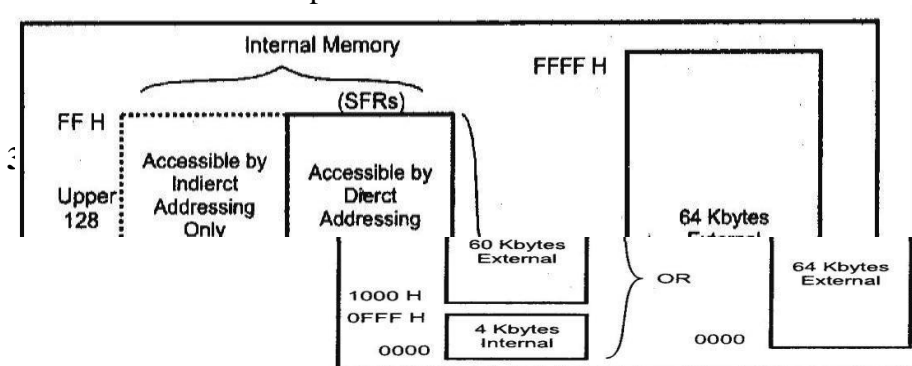
- ✓ Setting a bit, makes the corresponding interrupt to have high priority.
- ✓ Clearing a bit, makes the corresponding interrupt to have low priority.

36. Name the five interrupt sources of 8051. [May 2010, NOV 2009, MAY 2007, MAY 2008]

What is the hardware and software interrupts of 8051? Mention its vector addresses. (NOV 2011)

The interrupts with vector addresses are:

- External interrupt 0: IE0:0003H
- Timer interrupt 0: TF0:000BH
- External interrupt 1: IE1:0013H
- Timer interrupt 1: TF1:001BH
- Serial interrupt



38. Draw the data memory organization in 8051.

39. Define Program Counter.

- ✓ Program counter (PC) is a 16-bit register.
- ✓ It holds the 16-bit address of the instruction to be executed by the processor.
- ✓ PC is automatically incremented after every fetch of instruction byte from the memory.

40. Why all pins of a port are loaded with value “FF” before using it?

- ✓ All ports of 8051 are configured by default as Output port.
- ✓ To make it configured as Input Port, all pins of a port are loaded with value “FF” i.e., 1111 1111.

41. Justify why the crystal oscillator frequency in 8051 is chosen as 11.0592 Mhz.

- ✓ Only XTAL (Crystal Oscillator) of 11.0592 MHz can provide such standard baud rates 4800, 9600, etc., after scaling down by 12, 32 at UART.

42. List the modes of Timer in 8051. (NOV 2008)

The modes of timer in 8051 are chosen with M0 & M1 bits in TMOD register. The different modes of timer are as follows.

M1	M0	Mode	Description of Timer mode
0	0	0	13-bit timer
0	1	1	16-bit timer
1	0	2	8-bit timer with auto-reload
1	1	3	Split timer

43. What is the significance of C/T bit in TMOD register of 8051?

- ✓ The C/T bit in the TMOD register is a selector bit for the type of operation.
- ✓ HIGH in that bit indicates Counter operation and LOW in that bit indicates Timer operation.

44. What is the significance of TRX bit in TCON register of 8051? (May 2015)

TRX bit in the TCON register is used to Start / Stop the timer register for both timer and counter operation, by setting that bit with value „1“ / „0“ respectively.

45. What are the modes of asynchronous serial communication in 8051? (NOV 2008)

- ✓ The mode of serial communication is decided by two bits SM0 & SM1 in

SCON register.

✓ The detail of the various modes is given below.

SM1	SM0	Mode	Serial Mode Description	Baudrate
0	0	0	8-bit Shift register	$F_{Osc}/12$
0	1	1	8-bit UART	Variable
1	0	2	9-bit UART	$F_{Osc}/32$ or 64
1	1	3	9-bit UART	Variable

46. Illustrate the CJNE instruction. (April 2017)

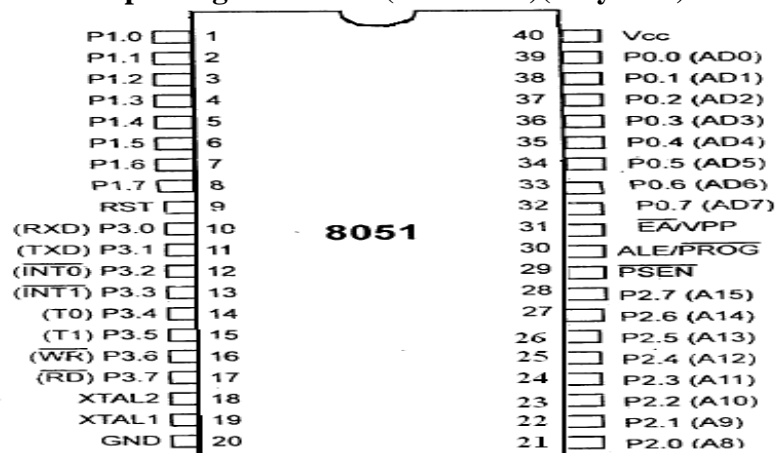
Compare and Jump if Not Equal – CJNE

Compare the magnitude of the two operands and jump if they are not equal. The values are considered to be unsigned.

The Carry flag is set/cleared appropriately. Example:

CJNE A, direct, rel

47. Draw the pin diagram of 8051. (NOV 2016) (May 2018)



48. What is jump range? (NOV 2015)

What is the difference between AJMP and LJMP instruction? (May 2014)

Jump Instruction	Meaning	Jump Range
SJMP	Short jump	256B
AJMP	Absolute jump	2KB
LJMP	Long jump	64KB

- AJMP and LJMP instructions are transfer program control to the specified vector address.
- Program control transfer ranges are 2KB for AJMP and 64KB for LJMP.

49. What are special function registers used for port operation in 8051? (May 2012)

P0, P1, P2 and P3 SFRs used for port operation.

50. What is needed for bitwise instructions in microcontroller? (May 2012)

Bitwise instructions allow manipulating the individual bits of bit-addressable registers and memory locations as well as the CY flag.

51. What are on-chip resources? List those available in the 8051 microcontroller. (NOV 2010)

When various resources are available inside of a chip, it is known as on-chip resources.

On-chip resources available in 8051 are RAM, ROM, Timer, Interrupts, serial ports and parallel ports.

52. A given 8051 chip has a speed of 16 MHz. What is the range of frequency that can be applied to the XTAL1 and XTAL2 pins? (NOV 2009)

16MHz frequency can be applied to the XTAL1 and XTAL2 pins.

53. How do you calculate baud rate for serial communication for 8051? (NOV 2007, MAY 2013, 2015)

8051 divides the crystal frequency by 12 to get machine cycle frequency. 8051 UART circuitry divides the machine cycle frequency by 32.

Timer 1 is used to set baud rate using TH1 register

Baudrate	TH1(decimal)	TH1(Hex)
9600	-3	FD
4800	-6	FA
2400	-12	F4
1200	-24	E8

54. Why is it necessary to have external pullup for port 0 in 8051? [November 2014]

- ✓ When logic-1 is loaded to the latch.
- ✓ This causes port 0 to float to high impedance state and it gets connected to the Pin read buffer.
- ✓ An external pullup resistor is required to supply a high output.

Microcontrollers for Embedded system:

- ✓ A microcontroller is a single chip computer.
- ✓ A CPU with all the peripherals like RAM, ROM, I/O Ports, Timers, and ADC set c. on the same chip. For Ex: Motorola 6811, Intel 8051, Zilog Z8 and PIC 16X etc...

Microprocessor:

- ✓ A CPU built into a single VLSI chip is called a microprocessor.
- ✓ It is a general-purpose device and additional external circuitry is added to make it a microcomputer.
- ✓ The microprocessor contains arithmetic logic unit (ALU), Control unit, Instruction register, Program counter (PC), clock circuit (internal or external), reset circuit (internal or external) and registers.
- ✓ But the microprocessor has no on-chip I/O Ports, Timers, Memory etc.
- ✓ For example, Intel 8085 is an 8-bit microprocessor and Intel 8086/8088 a 16-bit microprocessor.
- ✓ The block diagram of the Microprocessor is shown in Fig. 1

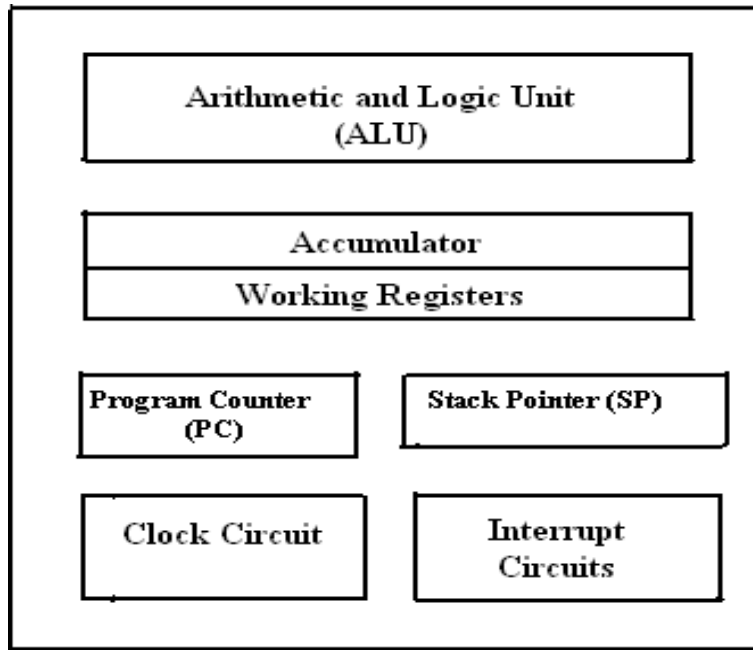


Fig.1:BlockdiagramofaMicroprocessor.

...

MICROCONTROLLER:

- ✓ A microcontroller is an integrated single chip, which consists of CPU, RAM, EPROM/PROM/ROM, I/O ports, timers, interrupt controller.
- ✓ For example, Intel 8051 is 8-bit microcontroller and Intel 8096 is 16-bit microcontroller.
- ✓ The block diagram of Microcontroller is shown in Fig.2.

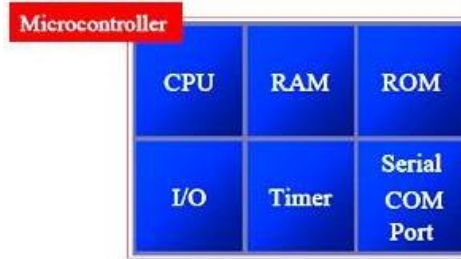


Fig.2. Block Diagram of a Microcontroller

Distinction between Microprocessor and Microcontroller

S.No	Microprocessor	Microcontroller
1	A microprocessor is a general purpose device.	A microcontroller is a dedicated chip which is also called as single chip computer.
2	A microprocessor does not contain on chip I/O Ports, Timers, Memory etc.	A microcontroller includes RAM, ROM, serial and parallel interface, timers, interrupt circuitry in a single chip.
3	Microprocessor is used as the CPU in microcomputer system.	Microcontroller is used to perform control-oriented applications.
4	Microprocessor instructions are nibble or byte addressable	Microcontroller instructions are both bit addressable as well as byte addressable.
5	Microprocessor based system design is complex and expensive	Microcontroller based system design is simple and cost effective
6	The instruction set of microprocessor is complex with large number of instructions.	The instruction set is simple with less number of instructions.

4.2: INTEL8051MICRCONTROLLER:

Drawthearchitecturalblockdiagramof8051microcontrollerandexplain.(NOV2011,MAY2010,NOV2009,NOV2008,May2008,MAY2007,MAY2006,NOV2016,May2016)

Featuresof8051Microcontroller:

The8051isan8-bitController:

- ✓ TheCPUcanworksononly8bitsofdataatatime
- ✓ The8051has
 - 128bytesofRAM
 - 4Kbytesofon-chipROM
 - Twotimers
 - One serialport
 - FourI/Oports,each8bitswide
 - 6interruptsources

ARCHITECTURE&BLOCKDIAGRAMOF8051MICROCONTROLLER:

- ✓ ItashardwarearchitecturewithRISC(ReducedInstructionSetComputer)concept.
- ✓ The blockdiagramof8051microcontrollerisshownin Fig3.
- ✓ 8051has8-bitALU.
- ✓ ALUcanperformallthe8-bitarithmeticalandlogicaloperationsinonemachinecycle.
- ✓ The ALUisassociatedwithtworegistersA&B

AandBRegisters:

- ✓ TheAandBregistersarespecialfunctionregisters.
- ✓ A&Bregistersholdtheresultsofmanyarithmeticalandlogicaloperationsof8051.
- ✓ TheAregisterisalsocalledthe**Accumulator**.
- ✓ Aregisterisusedasageneralregistertoaccumulate theresultsofalargenumberofinstructions.
- ✓ Bydefault,itisusedforallmathematicaloperationsanddatatransferoperationsbetweenCPUandexternalmemory.
- ✓ TheBregisterismainlyusedformultiplicationanddivisionoperationsalong withAregister.
 - Ex: MULAB : DIVAB.
- ✓ Itasnotherfunctionotherthanasastored data.

Rregisters:

- ✓ "R"registersareasetofeightregistersthatarenamedR0,R1,etc uptoR7.
- ✓ Theseregistersareusedasauxiliaryregistersin manyoperations.
- ✓ The"R"registersarealsousedtotemporarilystorevalues.

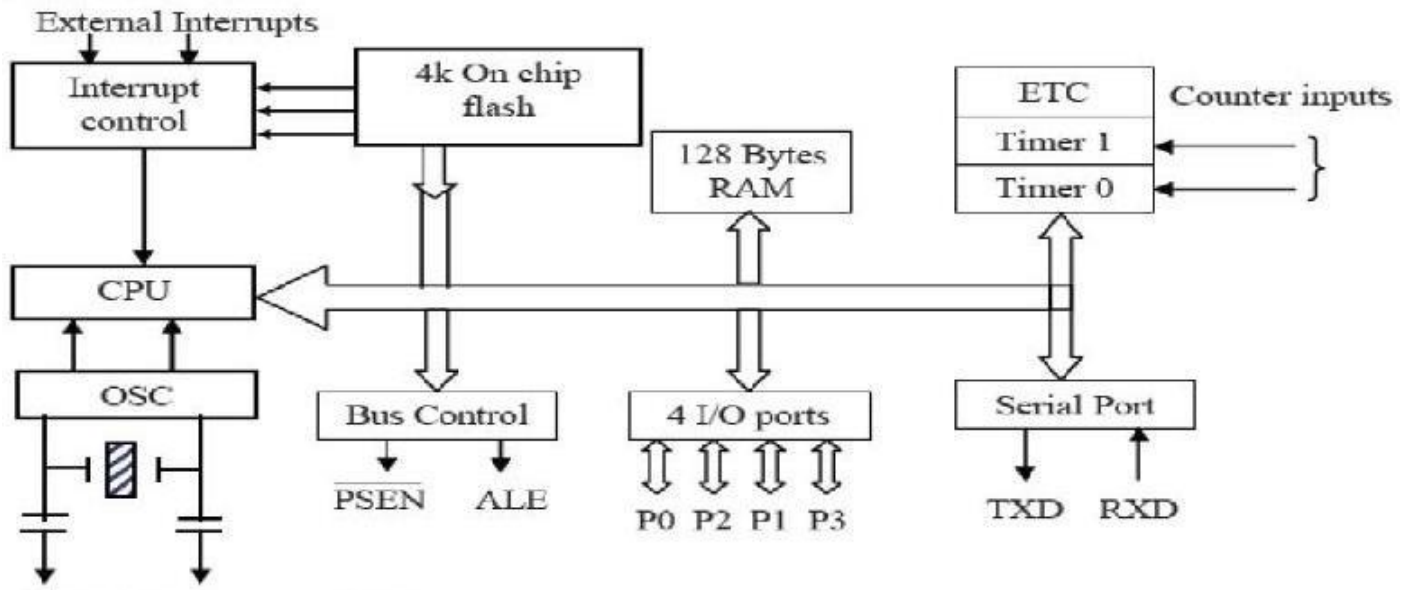


Fig.3. Block Diagram of 8051 Microcontroller

Program Counter (PC):

- ✓ 8051 has a 16-bit program counter.
- ✓ The program counter holds the address of the next instruction to be executed.
- ✓ After execution of one instruction, the program counter is incremented.

Data Pointer Register (DPTR):

- ✓ It is a 16-bit register which is the only user-accessible.
- ✓ DPTR is used to point to the data. 8051 will access external memory at the address indicated by DPTR.
- ✓ DPTR can also be used as two 8-bit registers DPH and DPL.

Stack Pointer Register (SP):

- ✓ It is an 8-bit register which stores the address of the stack top.
- ✓ When a value is pushed onto the stack, the 8051 first increments the value of SP and then stores the value.
- ✓ Similarly when a value is popped off the stack, the 8051 returns the value from the memory location indicated by SP and then decrements the value of SP.
- ✓ Since the SP is only 8-bit wide.
- ✓ It is incremented or decremented by two.
- ✓ SP is modified directly by the 8051 by six instructions: PUSH, POP, ACALL, LCALL, RET, and RETI.
- ✓ It is also used intrinsically whenever an interrupt is triggered.

Block Diagram

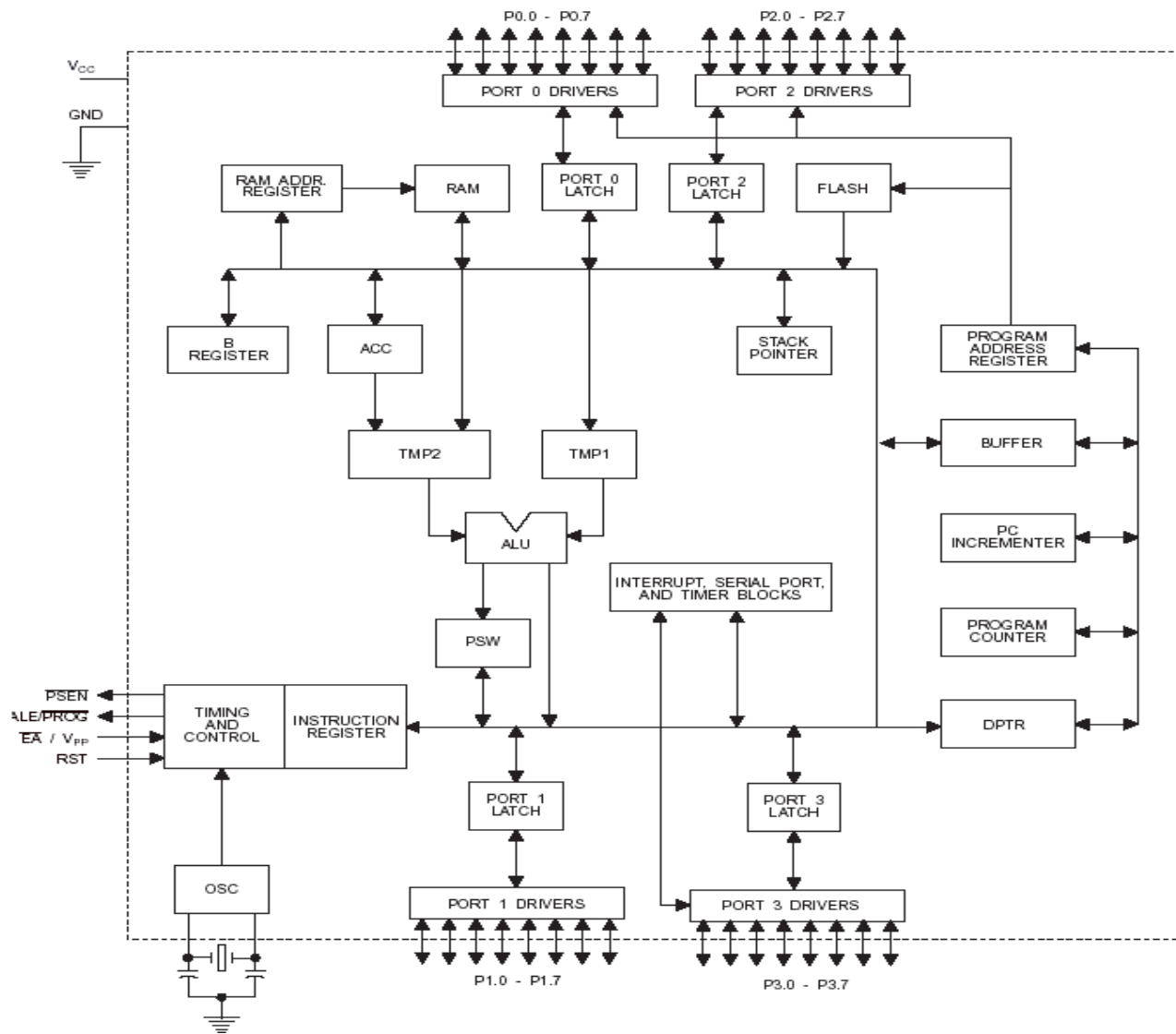


Fig3a: Internal architecture diagram of 8051 Microcontroller

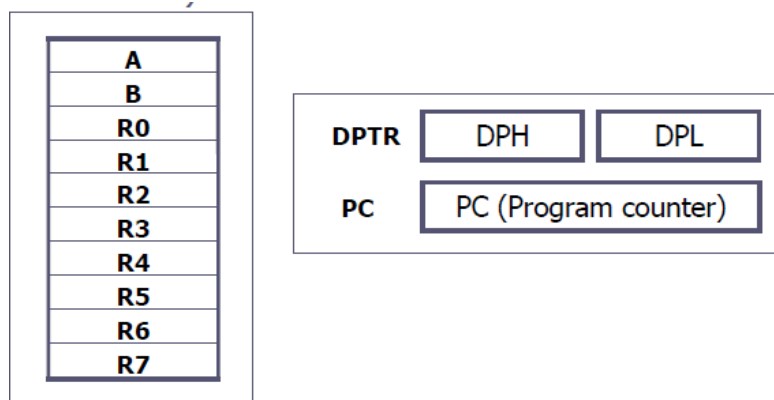
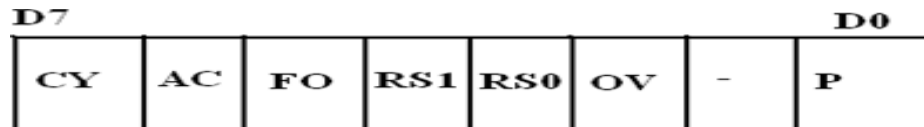


Fig: Structure of registers

Program Status Register (PSW):**Give PSW of 8051 and describe the use of each bit in PSW. (NOV 2015)**

- ✓ The 8051 has an 8-bit PSW register which is also known as Flag register.
- ✓ In the 8-bit register only 6-bits are used by 8051. The two unused bits are user-definable bits.
- ✓ In the 6-bits, four of them are conditional flags. They are Carry –CY, Auxiliary Carry-AC, Parity-P, and Overflow-OV.
- ✓ These flag bits indicate some conditions of the result after an instruction was executed.



- ✓ The bits PSW3 and PSW4 are denoted as RS0 and RS1.
- ✓ These bits are used to select the bank registers of the RAM location.
- ✓ The meaning of various bits of PSW register is shown below.

CY	PSW.7	Carry Flag
AC	PSW.6	Auxiliary Carry Flag
FO	PSW.5	Flag 0 available for general purpose
RS1	PSW.4	Register Bank select bit 1
RS0	PSW.3	Register bank select bit 0
OV	PSW.2	Overflow flag
---	PSW.1	User definable flag
P	PSW.0	Parity flag. set/cleared by hardware.

- ✓ The selection of the register banks and their addresses are given below.

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

RAM&ROM:

- ✓ The 8051 microcontroller has 128 bytes of Internal RAM and 4KB of on-chip ROM.
- ✓ The RAM is also known as Data memory and the ROM is known as program (Code) memory.
- ✓ Code memory holds program that is to be executed.
- ✓ Program Address Register holds address of the ROM/Flash memory.
- ✓ Data Address Register holds address of the RAM.

I/O ports:

- ✓ The 8051 microcontroller has 4 parallel I/O ports, each of 8-bits.
- ✓ So, it provides 32 I/O lines for connecting the microcontroller to the peripherals.
- ✓ The four ports are P0 (Port 0), P1 (Port 1), P2 (Port 2) and P3 (Port 3).

ADDRESSING MODES OF 8051:

Explain different types of addressing modes of 8051 microcontroller. (NOV 2008, NOV 2015, April 2017)

- ✓ The way in which the data operands are specified is known as the addressing modes. There are various methods of denoting the data operands in the instruction.
- ✓ The 8051 microcontroller supports 5 addressing modes. They are
 1. Immediate addressing mode
 2. Direct Addressing mode
 3. Register addressing mode
 4. Register indirect addressing mode
 5. Indexed addressing mode

Immediate addressing mode:

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

DPTR. Examples:

- `MOVA, #27H` : The data (constant) 27 is moved to the accumulator register
- `ADDR1, #45H` : Add the constant 45 to the contents of the accumulator
- `MOVDPTR, # 8245H` : Move the data 8245 into the data pointer register.

Direct addressing mode:

- ✓ In the addressing mode, the data operand is in the RAM location (00-7FH) and the address of the data operand is given in the instruction.
- ✓ The direct addressing mode uses the lower 128 bytes of Internal RAM and the SFRs Example

s:

- `MOV R1, 42H` : Move the contents of RAM location 42 into R1 register
- `MOV 49H, A` : Move the contents of the accumulator into the RAM location 49.
- `ADD A, 56H` : Add the contents of the RAM location 56 to the accumulator

Register addressing mode:

- ✓ In the addressing mode, the data operands are available in the registers. Exam

ples:

- `MOVA, R0` : Move the contents of the register R0 to the accumulator
- `MOV P1, R2` : Move the contents of the R2 register into port 1
- `MOV R5, R2` : This is invalid. The data transfer between the registers is not allowed.

Register Indirect addressing mode:

- ✓ In the addressing mode, a register is used as a pointer to the

data memory block. Examples:

- `MOVA, @R0`: Move the contents of RAM location whose address is in R0 into A (accumulator)
- `MOV @R1, B`: Move the contents of B into RAM location whose address is held by R1
- When R0 and R1 are used as pointers, they must be preceded by @ sign

- ✓ **Advantage: It makes accessing the data more dynamic than static as in the case of direct addressing mode.**

Indexed addressing mode:

- ✓ This addressing mode is used in accessing the data elements of lookup table entries, located in program ROM.

Example: `MOVCA, @A+DPTR`

- The 16-bit register DPTR and register A are used to form the address of the data element stored in on-chip ROM.

INSTRUCTION SET OF 8051:**Discuss in detail the 8051 instruction set. (NOV 2008)****Arithmetic instructions:**

...

With example, explain arithmetic instructions in 8051 microcontroller. (NOV 2012)

- ✓ ADD
 - 8-bit addition between the accumulator (A) and a second operand.
 - The result is always in the accumulator.
 - The CY flag is set/reset appropriately.
- ✓ ADDC
 - 8-bit addition between the accumulator, a second operand and the previous value of the CY flag.
 - Useful for 16-bit addition in two steps.
 - The CY flag is set/reset appropriately.
- ✓ DAA
 - Decimal adjust the accumulator.
 - Format the accumulator into a proper 2-digit packed BCD number.
 - Operates only on the accumulator.
 - Works only after the ADD instruction.
- ✓ SUBB
 - Subtract with Borrow.
 - Subtract an operand and the previous value of a borrow (carry) flag from the accumulator.
 - $A \leftarrow A - \langle \text{operand} \rangle - \text{CY}$.
 - The result is always saved in the accumulator.
 - The CY flag is set/reset appropriately.
- ✓ INC
 - Increment the operand by one.
 - The operand can be a register, a direct address, an indirect address, the data pointer.
 - Decrement the operand by one.
- ✓ DEC
 - The operand can be a register, a direct address, an indirect address.
- ✓ MULAB/DIVAB
 - Multiply A by B and place result in A and B registers.
 - Divide A by B and place quotient in A register & remainder in B register.
 -

...

Logical instructions in 8051.

- ✓ ANL: It performs AND logical operation between two operands.
 - Work on byte sized operands or the CY flag.
 - ANLA, Rn
 - ANLA, direct
 - ANLA, @Ri
 - ANLA, #data
 - ANL direct, A
 - ANL direct, #data
 - ANLC, bit
 - ANLC, /bit
- ✓ ORL: It performs OR logical operation between two operands.
 - Work on byte sized operands or the CY flag.
 - ORL A, Rn
 - ORLA, direct
 - ORLA, @Ri
 - ORLA, #data
- ✓ XRL
 - Work on bytes only.
 - XRL A, Rn
 - XRLA, direct
- ✓ CPL/CLR
 - Complement/Clear.
 - Work on the accumulator or a bit.
 - CLR P1.2
 - CPL Rn
- ✓ RL/RLC /RR/RRC
 - Rotate the accumulator.
 - RL and RR without the carry
 - RLC and RRC rotate through the carry.
 - SWAP A: Swap the upper and lower nibbles of the accumulator.

Datatransferinstructionsin8051.**Brieflyexplainthedatatransferinstructionsavailablein8051microcontroller.(NOV2014)****MOV**

➤ 8-bitdatatransferforinternalIRAMandtheSFR.

- MOVA,Rn
- MOVA,direct
- MOVA,@Ri
- MOVA,#data
- MOVRn,A
- MOVRn,direct
- MOVRn,#data
- MOVdirect,A
- MOVdirect,Rn
- MOVdirect,direct
- MOVdirect,@Ri
- MOVdirect,#data
- MOV@Ri,A
- MOV@Ri,direct
- MOV@Ri,#data

✓ MOV

➤ 1-bitdatatransferinvolvingtheCYflag

- MOVC,bit
- MOVbit,C

✓ MOV

➤ 16-bitdatatransferinvolvingtheDPTR

- MOVDPTR,#data

✓ MOVC

➤ MoveCodeByte

- Loadtheaccumulatorwithabytefromprogrammery.
- Mustuseindexedaddressing
- MOVCA,@A+DPTR
- MOVCA,@A+PC

✓ MOVX

- Datatransferbetween the accumulatorand abytefromexternaldatamemory.
 - MOVXA,@Ri
 - MOVXA,@DPTR
 - MOVX@Ri,A
 - MOVX@DPTR,A

✓ PUSH/POP

- PushandPopadatabyteonto thestack.
- ThedatabyteisidentifiedbyadirectaddressfromtheinternalRAMlocations.
 - PUSH DPL
 - POP 40H

✓ XCH

- Exchangeaccumulatorandabyteoperand
 - XCH A,Rn
 - XCH A,direct
 - XCH A,@Ri

✓ XCHD

- Exchangelowerdigitofaccumulatorwiththelowerdigitofthememorylocationspecified.
 - XCHDA,@Ri
 - Thelower4-bitsoftheaccumulatorareexchangedwith thelower4-bitsoftheinternalmemory location identifiedindirectly bytheindexregister.
 - Theupper4-bitsofeach arenotmodified.

Boolean(or)Bitmanipulationinstructions in8051.

- ✓ Thisgroupofinstructions isassociatedwiththesingle-bitoperationsofthe8051.
- ✓ Thisgroupallowsmanipulatingtheindividualbitsofbitaddressableregistersandmemorylocationsaswellasthe CYflag.
 - TheP,OV,andACflagscannotbe directlyaltered.
- ✓ Thisgroupincludes:
 - Set,clear,and,orcomplement,move.
 - Conditionaljumps.
- ✓ CLR

- Clear a bit or the CY flag.
- CLR P1.1
- CLRC
- ✓ SETB
 - Set a bit or the CY flag.
 - SETBA.2
 - SETBC
- ✓ CPL
 - Complement a bit or the CY flag.
 - CPL 40H; Complement bit 40 of the bit addressable memory
- ✓ ORL/ANL
 - OR/AND a bit with the CY flag.
 - ORL C, 20H; OR bit 20 of bit addressable memory with the CY flag
 - ANL C, 34H; AND bit 34 of bit addressable memory with the CY flag.
- ✓ MOV
 - Data transfer between a bit and the CY flag.
 - MOV C, 3FH; Copy the CY flag to bit 3F of the bit addressable memory.
 - MOV P1.2, C; Copy the CY flag to bit 2 of P1.
- ✓ JC/JNC
 - Jump to a relative address if CY is set/cleared.
- ✓ JB/JNB
 - Jump to a relative address if a bit is set/cleared.
 - JB ACC.2, <label>
- ✓ JBC - Jump to a relative address, if a bit is set and clear the bit.

Branching instructions:

With example, explain branching instructions in 8051 microcontroller. (May 2010, NOV

2012) Explain the working of program control transfer instructions of 8051. (May 2012)

- ✓ The 8051 provides four different types of unconditional jump instructions:
 - Short Jump – SJMP
 - Uses an 8-bit signed offset relative to the 1st byte of the next instruction.
 - Long Jump – LJMP
 - Uses a 16-bit address.

- 3byteinstructioncapableofreferencinganylocationintheentire64Kofprogrammmemory.
- AbsoluteJump–AJMP
 - Usesan11-bitaddress.
 - 2byteinstruction
 - The11-bitaddressissubstitutedforthe lower11-bitsofthePCto calculatethe16-bitaddressofthetarget.
 - Thelocationreferencedmustbewithinthe2KBytememory.
- IndirectJump–JMP
 - JMP@A+DPTR
- ✓ The8051provides2formsfortheCALLinstruction:
 - AbsoluteCall–ACALL
 - Usesan11-bitaddresssimilar toAJMP
 - Thesubroutinemustbewithinthesame2Kpage.
 - LongCall–LCALL
 - Usesa16-bitaddresssimilar toLJMP
 - Thesubroutinecanbe anywhere.
 - Bothforms pushthe16-bitaddressofthenextinstructiononthestackandupdatethestackpointer.
- ✓ The8051provides2formsforthereturninstruction:
 - Returnfromsubroutine–RET
 - Popthereturnaddressfromthestackand continueexecutionthere.
 - ReturnfromInterruptServiceRoutine–RETI
 - Popthereturnaddressfromthestack.
 - Continueexecutionattheaddressretrievedfromthestack.
 - ThePSWisnotautomaticallyrestored.
- ✓ The8051supports5differentconditionaljumpinstructions.
 - ALLconditionaljumpinstructionsusean8-bitsigned offset.
 - JumponZero–JZ/JNZ
 - JumpiftheA==0/A!=0
 - Thecheckisdone atthetime oftheinstructionexecution.

...

- Jump on Carry– JC/JNC
 - Jump if the C flag is set/cleared.
- Jump on Bit–JB/JNB
 - Jump if the specified bit is set/cleared.
 - Any addressable bit can be specified.
- Jump if the Bit is set then Clear the bit–JBC
 - Jump if the specified bit is set.
 - Then clear the bit.
- ✓ Compare and Jump if Not Equal–CJNE
 - Compare the magnitude of the two operands and jump if they are not equal.
 - The values are considered to be unsigned.
 - The Carry flag is set/cleared appropriately.
 - CJNE A, direct, rel
 - CJNE Rn, #data, rel
 - CJNE @Ri, #data, rel
- ✓ Decrement and Jump if Not Zero–DJNZ
 - Decrement the first operand by 1 and jump to the location identified by the second operand if the resulting value is not zero.
 - DJNZ Rn, rel
 - DJNZ direct, rel
 - NOP –No operation

Memory organization:

Explain in detail the internal memory organization of 8051 microcontroller (NOV 2014, May 2012, NOV 2011, NOV 2010, May 2010, MAY 2009, NOV 2008, NOV 2007)

- ✓ The 8051 microcontroller has 128 bytes of Internal RAM and 4kB of on-chip ROM.
- ✓ The RAM is also known as Data memory and the ROM is known as program (Code) memory.
- ✓ Code memory holds the actual 8051 program to be executed.
- ✓ In 8051, memory is limited to 64KB.
- ✓ Code memory may be found on-chip, as ROM or EPROM.
- ✓ It may also be stored completely off-chip in an external ROM/EPROM.
- ✓ The 8051 has only 128 bytes of Internal RAM but it supports 64KB of external RAM.

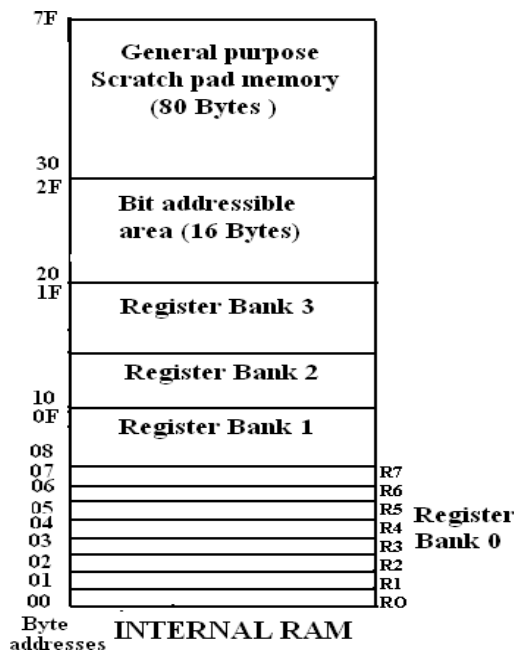
- ✓ Since the memory is off-chip, it is not as flexible for accessing and is also slower.

Structure of Internal RAM of 8051 (Data Memory):

Explain the Data memory structure of 8051. (NOV 2011)

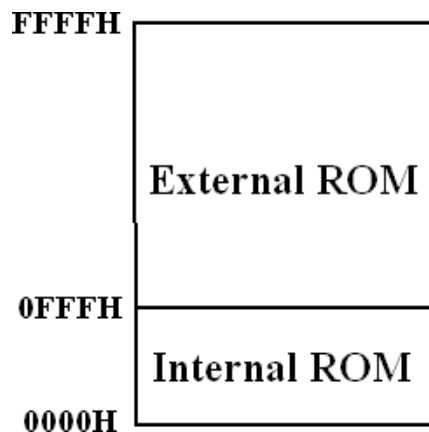
- ✓ Internal RAM is found on-chip on the 8051. So it is the fastest RAM available.
- ✓ It is flexible in terms of reading, writing and modifying its contents.
- ✓ Internal RAM is volatile.
- ✓ When the 8051 is reset, internal RAM is cleared.
- ✓ The 128 bytes of internal RAM is organized as below.
- ✓ Four register banks (Bank 0, Bank 1, Bank 2 and Bank 3) each of 8-bits (total 32 bytes).
- ✓ The default bank register is Bank 0.
- ✓ The remaining banks are selected with the help of RS0 and RS1 bits of PSW Register.
- ✓ 16 bytes of bit addressable area and
- ✓ 80 bytes of general purpose area (Scratch pad memory) of internal RAM as shown in the diagram below.
- ✓ This area is utilized by the microcontroller as a storage area for the operating stack.
- ✓ The 32 bytes of RAM from address 00 H to 1FH are used as working registers organized as four banks of eight registers each.
- ✓ The registers are named as R0-R7.
- ✓ Each register can be addressed by its name or by its RAM

address. For example: `MOVA, R7` or `MOVR7, #05H`



Structure of Internal ROM (On-chip ROM / Program Memory / Code Memory):

- ✓ The 8051 microcontroller has 4KB of on-chip ROM, but it can be extended up to 64KB.
- ✓ This ROM is also called program memory or code memory.
- ✓ The CODE segment is accessed using the program counter (PC) for opcode fetches and by DPTR for data.
- ✓ The external ROM is accessed when the EA pin is connected to ground or the contents of program counter exceeds 0FFFH.
- ✓ When the internal ROM address is exceeded, the 8051 automatically fetches the code bytes from the external program memory.

**SPECIAL FUNCTION REGISTERS (SFRs)**

Write the available special function registers in 8051. Explain each register with its format and functions. (April 2017, NOV 2015)

- ✓ In 8051 microcontroller, there are registers which use the RAM addresses from 80H to FFH.
- ✓ They are used for certain specific operations. These registers are called Special Function Registers (SFRs).
- ✓ Most of SFRs are bit addressable and other few registers are byte addressable.
- ✓ In these SFRs, some of them are related to I/O ports (P0, P1, P2 and P3) and some of them are for control operations (TCON, SCON & PCON).
- ✓ Remaining are the auxiliary SFRs, that they don't directly configure the 8051.
- ✓ The list of SFRs and their functional names are given below.
- ✓ **The * indicates the bit addressable SFRs**

...

S.No	Symbol	Name of SFR	Address (Hex)
1	ACC*	Accumulator	0E0
2	B*	B-Register	0F0
3	PSW*	Program Status word register	0D0
4	SP	Stack Pointer Register	81
5	DPL	Data pointer low byte	82
	DPH	Data pointer high byte	83
6	P0*	Port 0	80
	P1*	Port 1	90
8	P2*	Port 2	0A
9	P3*	Port 3	0B
10	IP*	Interrupt Priority control	0B8
11	IE*	Interrupt Enable control	0A8
12	TMOD	Timer mode register	89
13	TCON*	Timer control register	88
14	TH0	Timer 0 Higher byte	8C
15	TL0	Timer 0 Lower byte	8A
16	TH1	Timer 1 Higher byte	8D
17	TL1	Timer 1 lower byte	8B
18	SCON*	Serial control register	98
19	SBUF	Serial buffer register	99
20	PCON	Power control register	87

Table: Special Function Registers

STACK in 8051 Microcontroller:

- ✓ The stack is a part of RAM used by the CPU to store information temporarily.
- ✓ This information may be either data or an address.

- ✓ The register used to access the stack is called the Stack pointer (SP).
- ✓ SP is an 8-bit register. So, it can take values of 00 to FFH.
- ✓ When the 8051 is powered up, the SP register contains the value 07. i.e. the RAM location value 08 is the first location being used for the stack by the 8051 controller.
- ✓ There are two important instructions to handle stack. One is the PUSH and the other is the POP.
- ✓ The loading of data from CPU registers to the stack is done by PUSH.
- ✓ The loading of the contents of the stack back into a CPU register is done by POP.

Interrupts:

Explain interrupt structure of 8051 microcontroller. (NOV 2011, MAY

2009) Interrupt Structure:

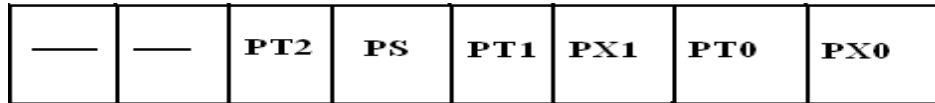
- ✓ An interrupt is an external or internal event that disturbs the microcontroller to inform it that a device needs its service.
- ✓ The program which is associated with the interrupt is called the **interrupt service routine (ISR)** or **interrupt handler**.
- ✓ Upon receiving the interrupt signal, the microcontroller finishes current operation and saves the PC on stack.
- ✓ Jump to a fixed location in memory depending on type of interrupt.
- ✓ Start to execute the interrupt service routine until RETI.
- ✓ Upon executing the RETI the microcontroller returns to the place where it was interrupted. Get pop PC from stack.
- ✓ The 8051 microcontroller has **FIVE** interrupts in addition to Reset. They are
 - Timer 0 overflow Interrupt
 - Timer 1 overflow Interrupt
 - External Interrupt 0 (INT0)
 - External Interrupt 1 (INT1)
 - Serial Port Interrupt
- ✓ Each interrupt has a specific place in code memory where program execution begins.
 - External Interrupt 0: 0003 H
 - Timer 0 overflow: 000BH

- ExternalInterrupt1: 0013 H
 - Timer1 overflow: 001BH
 - SerialInterrupt: 0023H
- ✓ UponresetallInterruptsaredisabled &donotrespondto theMicrocontroller.
- ✓ Theseinterruptsmustbeenabledbysoftware.Thisisdonebyan8-bitregistercalledInterruptEnableRegister(IE).

InterruptEnableRegister:

EA	---	ET2	ES	ET1	EX1	ET0	EX0
-----------	------------	------------	-----------	------------	------------	------------	------------

- EA:Globalenable/disable.Toenabletheinterrupts,thisbitmustbesethigh.
 - --- :Undefined-reservedforfutureuse.
 - ET2:Enable/disableTimer2overflow interrupt.
 - ES:Enable/disableSerialportinterrupts.
 - ET1:Enable/disableTimer1overflow interrupt.
 - EX1: Enable/disableExternalinterrupt1.
 - ET0:Enable/disableTimer0overflowinterrupt.
 - EX0:Enable/disableExternalinterrupt0
- ✓ Uponreset,theinterruptshavethefollowingpriorityfromtopdown.TheinterruptwiththehighestPRIORITYgetsservicedfirst.
1. Externalinterrupt0(INT0)
 2. Timerinterrupt0(TF0)
 3. Externalinterrupt1(INT1)
 4. Timerinterrupt1(TF1)
 5. Serialcommunication(RI+TI)
- ✓ Prioritycanalso besetto“high”or“low”by8-bitIPregister.

Interrupt priority register:

- IP.7:reserved
- IP.6:reserved
- IP.5:Timer2interruptprioritybit(8052only)
- IP.4:Serialportinterruptprioritybit
- IP.3:Timer1interruptprioritybit
- IP.2:Externalinterrupt1prioritybit
- IP.1:Timer0interruptprioritybit
- IP.0:Externalinterrupt0prioritybit

PROGRAMMING TIMERS OF 8051

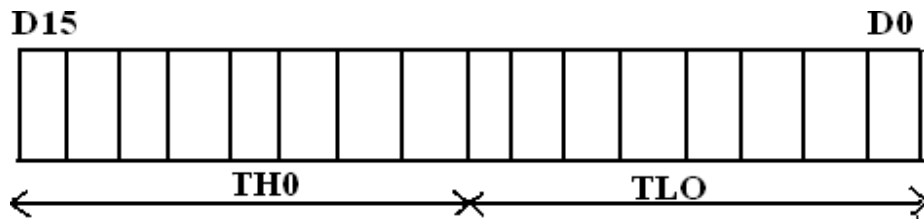
1. Explain the different modes of operation of timers in 8051 in detail with its associated registers. Describe different modes of operation of timers/counters in 8051 with its associated registers. (NOV 2009, MAY 2009, May 2007, May 2016)
 Draw and explain the functions of TCON and TMOD registers of 8051. (Dec 2008)
 Explain the on-chip timer modes of an 8051 Microcontroller. (April 2010, NOV 2016)

Timer Registers.

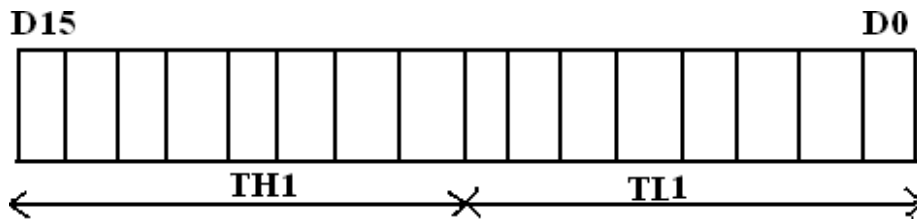
✓ The 8051 has two timers/counters, they can be used either as timers (used to generate a time delay) or as event counters.

TIMER 0:

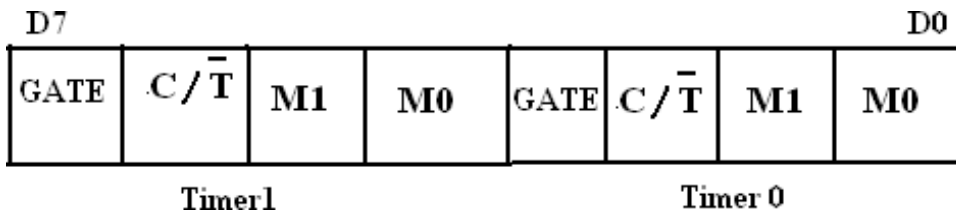
- ✓ Timer 0 is a 16-bit register and can be treated as two 8-bit registers (TL0 & TH0).
- ✓ These registers can be accessed similar to any other registers like A, B, or R1 etc.
- ✓ Ex: The instruction MOV TL0, #07 moves the value 07 into the lower byte of Timer 0.
- ✓ Similarly MOV R1, TH0 saves the contents of TH0 in the R1 register.

**TIMER 1:**

- ✓ Timer 1 is also a 16-bit register and can be treated as two 8-bit registers (TL1 & TH1).
- ✓ These registers can be accessed similar to any other registers like A, B, or R1 etc
- ✓ Ex: The instruction `MOV TL1, #05` moves the value 05 into the lower byte of Timer 1.
- ✓ Similarly `MOV R0, TH1` saves the contents of TH1 in the R0 register.

**TMOD (Timer mode Register):**

- ✓ The various operating modes of both the timers T0 and T1 are set by the TMOD register.
- ✓ TMOD is an 8-bit register.
- ✓ The lower 4 bits are for Timer 0
- ✓ The upper 4 bits are for Timer 1
- ✓ In each case,
 - The lower 2 bits are used to set the timer mode
 - The upper 2 bits specify the operation



GATE:

- ✓ This bit is used to start or stop the timers by hardware.
- ✓ When GATE= 1, the timers can be started/stopped by the external sources.
- ✓ When GATE=0, the timers can be started or stopped by software instructions like SETBTR_x or CLRTR_x.

C/T(Counter/Timer):

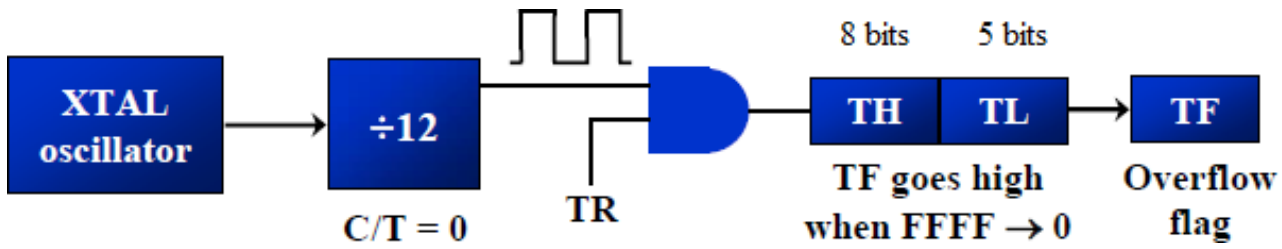
- ✓ This bit decides whether the timer is used as a delay generator or event counter.
- ✓ When $C/\bar{T}=0$, timer is used as a delay generator.
- ✓ When $C/\bar{T}=1$, timer is used as an event counter.
- ✓ The clock source for the timer delay is the crystal frequency of 8051.
- ✓ The clock source for the event counter is the external clock source.

M1,M0(Mode):

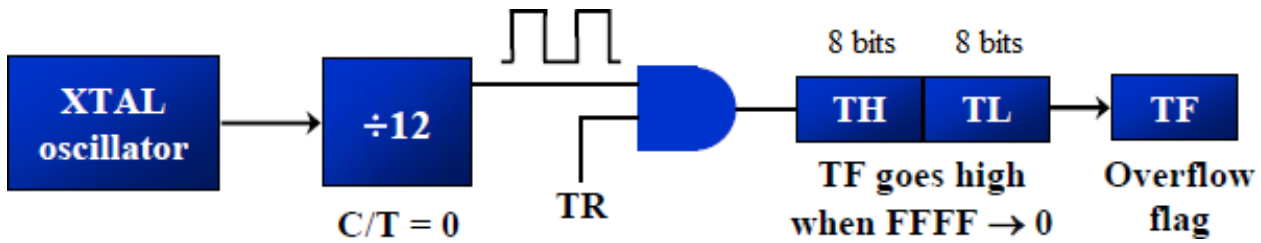
- ✓ These two bits are the timer mode bits.
- ✓ The timers of the 8051 can be configured in four modes Mode0, Mode1, Mode2 & Mode3.
- ✓ The selection and operation of the modes is shown below.

S.No	M0	M1	Mode	Operation
1	0	0	Mode0	13-bit Timer mode. 8-bit Timer/counter THx with TLx as 5-bit prescaler
2	0	1	Mode1	16-bit Timer mode. 16-bit timer /counter THx and TLx are cascaded. There is no prescaler
3	1	0	Mode2	8-bit autoreload. 8-bit autoreload timer/counter. THx holds a value which is to be reloaded TLx each time it overflows
4	1	1	Mode3	Split timer mode

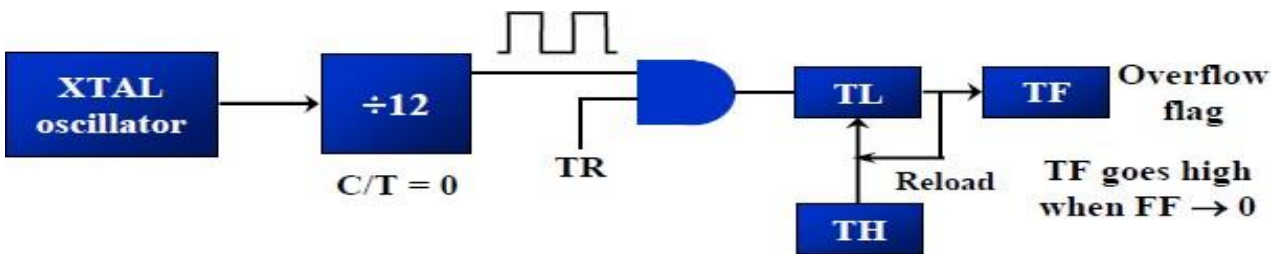
Mode0:13bit Timermode



Mode1:16bit Timermode



Mode2:8bit autoreloadmode



Mode3:Split Timermode

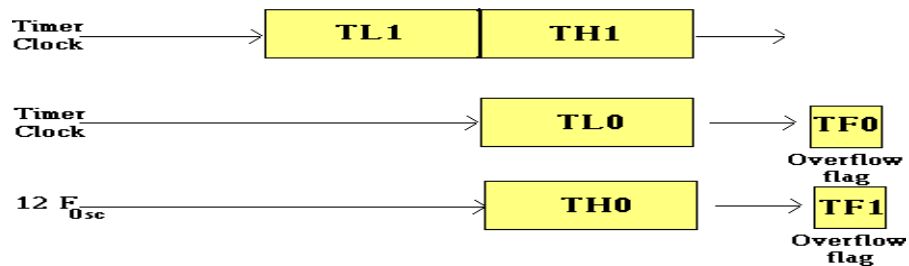


Figure: Modes of operation of Timer

...

TCON(Timer control register)

- ✓ TCON(timer control) register is an 8-bit register. TCON register is a bit-addressable register.

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Bit Number	Bit Mnemonic	Description
7	TF1	Timer 1 overflow flag Cleared by hardware when processor vectorsto interrupt routine. Set by hardware on timer/counter overflow, when the timer 1 register overflows.
6	TR1	Timer 1 run control bit Clear to turn off time/counter 1. Set to turn on timer/counter 1.
5	TF0	Timer 0 overflow flag Cleared by hardware when processor vectorsto interrupt routine. Set by hardware on timer/counter overflow, when the timer 0 register overflows.
4	TR0	Timer 0 run control bit Clear to turn off time/counter 0. Set to turn on timer/counter 0.
3	IE1	External interrupt 1 edge flag. Cleared by hardware when interrupt is processed if edge-triggered. Set by hardware when external interrupt is detected on INT1 pin.
2	IT1	External interrupt 1 type control bit Clear to select low level active (level triggered) for external interrupt 1. Set to select falling edge active (edge triggered) for external interrupt 1.
1	IE0	External interrupt 0 edge flag Cleared by hardware when interrupt is processed if edge-triggered. Set by hardware when external interrupt is detected on INT0 pin.
0	IT0	External interrupt 0 type control bit Clear to select low level active (level triggered) for external interrupt 0. Set to select falling edge active (edge triggered) for external interrupt 0.

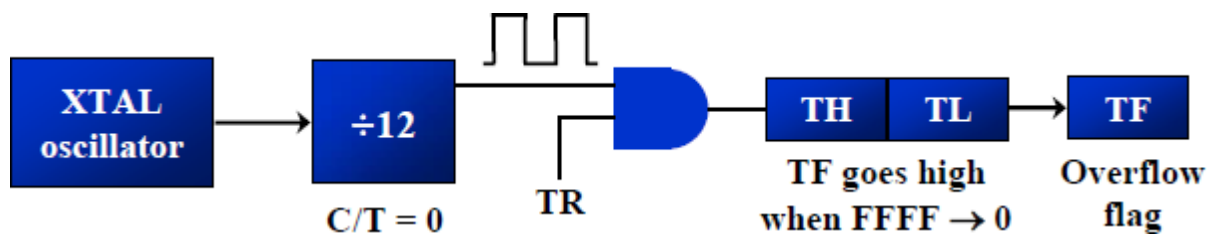
Timers of 8051 do starting and stopping by either software or hardware control

- ✓ For using software to start and stop the timer where GATE=0
- ✓ The start and stop of the timer are controlled by software using TR(timer start) bits TR_x and CLR_x
- ✓ The SETB instruction starts it, and it is stopped by the CLR instruction.
- ✓ These instructions start and stop the timers as long as GATE=0 in the TMOD register
- ✓ The hardware way of starting and stopping the timer is achieved by making GATE=1 in the TMOD register.



The following are the characteristics and operations of mode 1:

1. It is a 16-bit timer.
2. It allows value from 0000 to FFFFH.
3. Value to be loaded into the timer register TH and TL.
4. After TH and TL are loaded with a 16-bit initial value, the timer must be started
 - This is done by SETB TR0 for timer 0 and SETB TR1 for timer 1
5. After the timer is started, it starts to count up
 - It counts up until it reaches its limit of FFFFH
 - When it rolls over from FFFFH to 0000, it sets high a flag bit called TF (timer flag)
 - Each timer has its own timer flag.
 - There are TF0 for timer 0, and TF1 for timer 1.



6. Timer flag can be monitored,
 - When the timer flag is raised, to stop the timer with the CLR instructions.
 - CLR TR0 and CLR TR1, for timer 0 and timer 1 respectively.
 - After the timer reaches its limit and rolls over.
 - In order to repeat the process, TH and TL must be reloaded with the original value and TF must be reloaded to 0.

To generate a timedelay

1. Load the TMOD register indicating which timer is to be used and which timer mode is selected.
2. Load registers TH and TL with initial count value.
3. Start the timer
4. Keep monitoring the timer flag (TF) with the JNB TFx, target to see if it is raised
 - Get out of the loop when TF becomes high
5. Stop the timer
6. Clear the TF flag for the next round

7. Go back to Step 2 to load TH and TL again.

Example 1:

In the following program, we create a square wave of 50% duty cycle (with equal portions high and low) on the P1.

5 bit. Timer 0 is used to generate the time delay. Analyze the program. (Nov 2014)

```

MOV TMOD, #01      ;Timer0, mode 1 (16-bit mode)
HERE: MOV TL0, #0F2H ;TL0=F2H, the low byte
      MOV TH0, #0FFH ;TH0=FFH, the high byte
      MOV P1.5      ;toggle P1.5
      ACALL DELAY
DELAY: SJMP HERE    ;start the timer 0
      HERE SETB TR0
AGAIN: JNB TF0, AGAIN ;monitor timer flag 0 until it rolls over
      CLR TF0      ;stop timer 0
      TR0 CLR      ;clear timer 0 flag
      RET

```

In the above program, notice the following steps.

1. TMO is loaded.
2. FFF2H is loaded into TH0-TL0.
3. P1.5 is toggled for the high and low portions of the pulse.
4. The DELAY subroutine using the timer is called.
5. In the DELAY subroutine, timer 0 is started by the SETB TR0 instruction.
6. Timer 0 counts up with the passing of each clock, which is provided by the crystal oscillator.
 - As the timer counts up, it goes through the states of FFF3, FFF4, FFF5, FFF6, and so on until it reaches FFFFH.
 - One more clock roll sits to 0, raising the timer flag (TF0=1). At that point, the JNB instruction falls through.
7. Timer 0 is stopped by the instruction CLR TR0.
 - The DELAY subroutine ends and the process is repeated.

Notice that to repeat the process, we must reload the TL and TH registers, and start the process is repeated.



Example 2:

In Example 1, calculate the amount of time delay in the DELAY subroutine generated by the timer.

Assume XTAL = 11.0592 MHz.

Solution:

- ✓ The timer works with a clock frequency of 1/12 of the XTAL frequency, we have $11.0592 \text{ MHz} / 12 = 921.6 \text{ kHz}$ as the timer frequency.
- ✓ As a result, each clock has a period of $T = 1 / 921.6 \text{ kHz}$, $T = 1.085 \mu\text{s}$.
- ✓ In other words, Timer 0 counts up each $1.085 \mu\text{s}$ resulting in delay = number of counts $\times 1.085 \mu\text{s}$.
- ✓ The number of counts for the rollover is $\text{FFFFH} - \text{FFF2H} = 0\text{DH}$ (13 decimal).
- ✓ Add on to 13 because of the extra clock needed when it rolls over from FFFF to 0 and raises the TF flag.
- ✓ This gives $14 \times 1.085 \mu\text{s} = 15.19 \mu\text{s}$ for half the pulse. For the entire period it is $T = 2 \times 15.19 \mu\text{s} = 30.38 \mu\text{s}$ as the time delay generated by the timer.

(a) In hexadecimal

$(\text{FFFF} - \text{YYXX} + 1) \times 1.085 \mu\text{s}$, where YYXX are TH, TL initial values respectively. Notice that value YYXX are in hex.

(b) In decimal

Convert YYXX values of the TH, TL register to decimal to get a NNNN decimal, then $(65536 - \text{NNNN}) \times 1.085 \mu\text{s}$

Example 3:

In Example 1, calculate the frequency of the square wave generated on pin P1.5.

Solution:

- ✓ In the timer delay calculation of Example 1, we did not include the overhead due to instruction in the loop.
- ✓ To get a more accurate timing, we need to add clock cycles due to these instructions in the loop.
- ✓ To do that, we use the machine cycle as shown below.

		Cycles
HERE:	MOVTL0, #0F2H	2
	MOVTH0, #0FFH	2
	CPLP1.5	1
	ACALLDELAY	2
	SJMPHERE	2
	...	

DELAY:	SETBTR0	1
AGAIN:	JNBTF0,AGAIN	14
	CLRTR0	1
	CLRTF0	1
	RET	2
	Total	28

$$T=2 \times 28 \times 1.085 \mu\text{s} = 60.76 \mu\text{s} \text{ and } F = 16458.2 \text{ Hz}$$

Example4:

Find the delay generated by

timer0 in the following code, using both of the Methods. Do not include the overhead due to instruction.

```

                CLRP2.3           ;Clear P2.3
                MOV                ;Timer0, 16-bit mode
HERE:          TMOD,#01MOV        ;TL0=3EH, the low byte
                TL0,#3EH
                MOV                ;TH0=B8H, the high byte
                TH0,#0B8HSETB      ;SET high timer0
                P2.3              ;Start the timer0
AGAIN:         SETBTR0            ;Monitor timer flag0
                JNB                ;Stop the timer0
                TF0,AGAINCLR      ;Clear TF0 for next round
                TR0
                CLRTF0
                CLRP2.3

```

Solution:

$$(FFFFH - B83EH + 1) = 47C2H = 18370 \text{ in decimal and } 18370 \times 1.085 \mu\text{s} = 19.93145 \text{ ms}$$

The following are the characteristics and operations of mode 2:

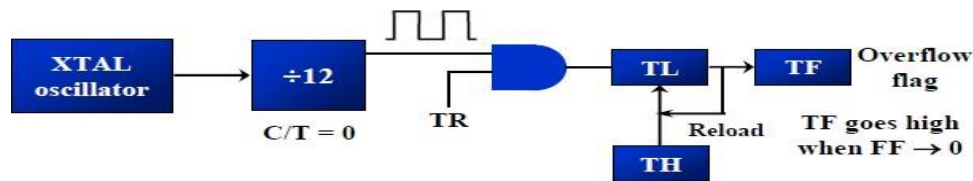
1. It is an 8-bit timer. It allows only values of 00 to FFH to be loaded into the timer register TH.
2. After TH is loaded with the 8-bit value, the 8051 copies the value to the TL register.
 - The timer must be started.
 - This is done by the instruction SETBTR0 for timer 0 and SETBTR1 for timer 1.
3. After the timer is started, it starts to count up by incrementing the TL register.
 - It counts up until it reaches its limit of FFH.
 - When it rolls over from FFH to 00, it sets the TF (timer flag).
 - When the TL register rolls from FFH to 00 and TF is set to 1.
 - TL is reloaded automatically with the original value kept by the TH register.

- Torepeattheprocess,simplyclearTF.

4. Thismakesmode2anauto-reload,incontrastwith mode1inwhichtheprogrammerhasto reload

...

THandTL

**To generate a time delay**

1. Load the TMOD value register indicating which timer is to be used, and the timer mode (mode 2) is selected.
2. Load the TH register with the initial count value.
3. Start timer.
4. Keep monitoring the timer flag (TF) with the JNBTFx, target, to see whether it is raised. Get out of the loop when TF goes high.
5. Clear the TF flag.
6. Go back to Step 4, since mode 2 is auto-reload.

Example 5:

Assume XTAL = 11.0592 MHz, find the frequency of the square wave generated on pin P1.0.

```

MOV TMOD, #20H    ; T1/8-bit/autoreload
MOV TH1, #5       ; TH1=5
SETB TR1          ; start the timer 1
BACK: JNB TF1, BACK ; till timer rolls over
CPL P1.0          ; P1.0 to hi, lo
CLR TF1           ; clear Timer 1 flag
SJMP BACK         ; mode 2 is auto-reload

```

Solution:

- ✓ In mode 2, no need to reload TH since it is auto-reload.
- ✓ Now $(256 - 05) \times 1.085 \mu\text{s} = 251 \times 1.085 \mu\text{s} = 272.33 \mu\text{s}$ is the high portion of the pulse.
- ✓ Since it is a 50% duty cycle square wave, the period T is twice.
- ✓ As a result $T = 2 \times 272.33 \mu\text{s} = 544.67 \mu\text{s}$ and the frequency = 1.83597 kHz

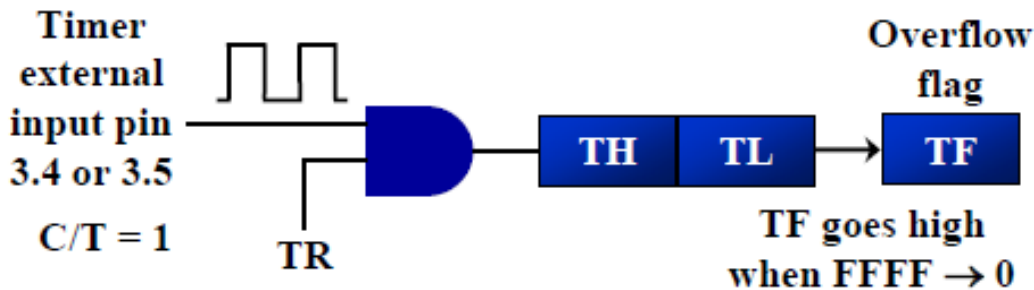
5.2 : Timers as counters

Timers can also be used as counters.

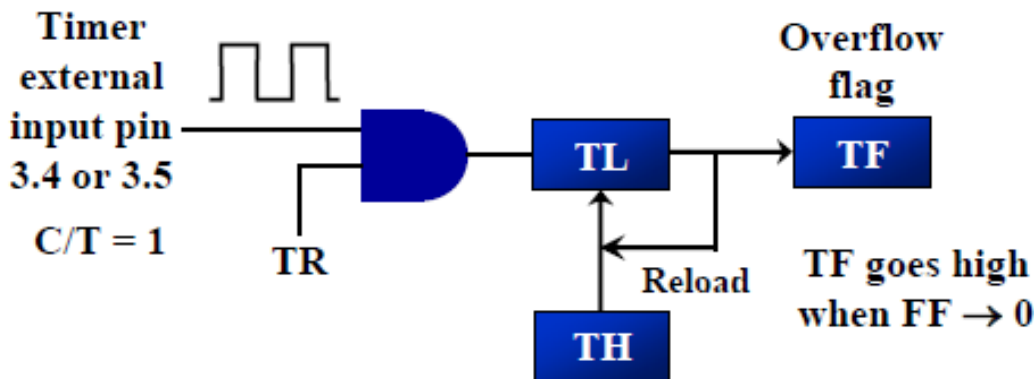
Which are used for counting events happening outside the 8051.

- When it is used as a counter, it is a pulse outside of the 8051 that increments the TH, TL register.
- TMOD and TH, TL registers are the same as in timer concept, except the source of the frequency.
- The C/T bit in the TMOD register decides the source of the clock for the timer
- When C/T=1, the timer is used as a counter and gets its pulses from outside the 8051.
- The counter counts up as pulses are fed from pins 14 and 15.
- these pins are called T0 (timer 0 input) and T1 (timer 1 input)

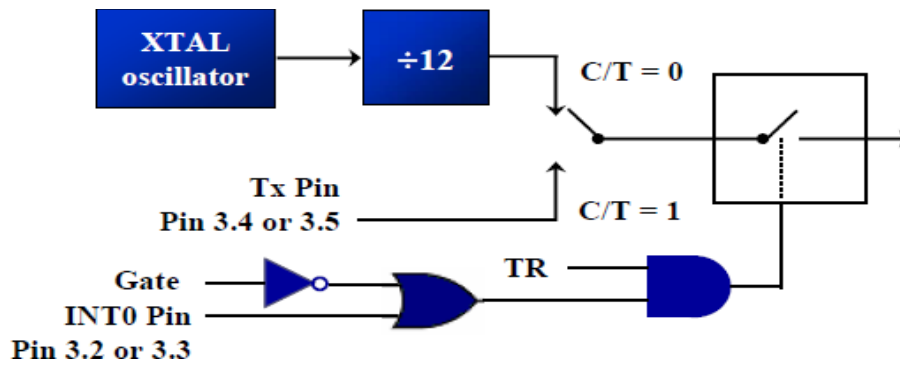
Timer with external input (Mode 1)



Timer with external input (Mode 2)



- ✓ If GATE = 1, the start and stop of the timer are done externally through pins P3.2 and P3.3 for timers 0 and 1, respectively
- ✓ This hardware allows starting or stopping the timer externally at any time via a simple switch



- ✓ The frequency for the timer is always 1/12th the frequency of the crystal attached to the 8051.

Port 3 pins used for Timers 0 and 1

Pin	Port Pin	Function	Description
14	P3.4	T0	Timer/counter 0 external input
15	P3.5	T1	Timer/counter 1 external input

Example 6:

Assuming that clock pulses are fed into pin T1, write a program for counter 1 in mode 2 to count the pulses and display the state of the TL1 counter on P2, which connects to 8 LEDs.

Solution:

```

MOV TMOD, #01100000B    ;counter 1, mode 2, C/T=1 external pulses
MOV TH1, #0             ;clear TH1
SETB P3.5               ;make T1 input
AGAIN: SETB TR1         ;start the counter
BACK:  MOVA, TL1        ;get copy of TL
       MOV P2, A        ;display it on port 2
       JNB TF1, BACK   ;keep doing, if TF=0
       CLR TR1         ;stop the counter 1
       CLRTF1          ;make TF=0
       SJMP AGAIN      ;keep doing it

```

- ✓ Notice in the above program the role of the instruction SETBP3.5.
- ✓ Since ports are set up for output when the 8051 is powered up.
- ✓ So, we make P3.5 an input port by making it high.

- ✓ In other words, we must configure (set high) the T1 pin (pin P3.5) to allow pulses to be fed into it.

5.3 : SERIAL COMMUNICATION

2. Explain the serial programming

of 8051 with its associated registers. (May 2014, 2013) (Or) Explain how to program for sending and receiving data serially using 8051 (April 2010, 2011) Explain 8051

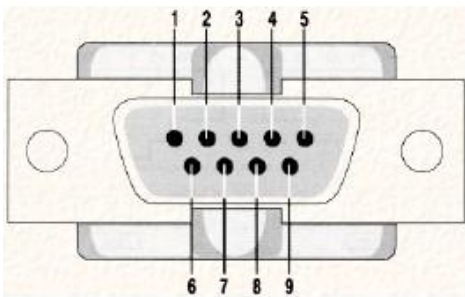
serial port programming with examples. (May 2016, NOV 2012)

Explain the serial modes of operation of 8051 microcontroller. (May 2007)

RS232

- ✓ It is an interfacing standard RS232.
- ✓ It was set by the Electronics Industries Association (EIA) in 1960.
- ✓ The standard was set long before the advent of the TTL logic family.
- ✓ Its input and output voltage levels are not TTL compatible.
- ✓ In RS232, a 0 is represented by -3 to -25V, while a 1 bit is +3 to +25V.
- ✓ IBM introduced the DB-9 version of the serial I/O standard.

RS232 Connector DB-9



RS232 DB-9 Pins

Pin	Description
1	Data carrier detect (-DCD)
2	Received data (RxD)
3	Transmitted data (TxD)
4	Data terminal ready (DTR)
5	Signal ground (GND)
6	Data set ready (-DSR)
7	Request to send (-RTS)
8	Clear to send (-CTS)
9	Ring indicator (RI)

Handshake signals of MODEM

TR (data terminal ready)

- When DTR=1, indicate that it is ready for communication.

DSR (data set ready)

- When DSR=1, indicate that it is ready for communication.

RTS (request to send)



- It asserts RTS signal to the modem that it has a byte of data to transmit.

CTS (clear to send)

- It is to receive, it sends out signal CTS,

DCD (data carrier detect)

- The modem asserts signal DCD to inform the DTE that a valid carrier has been detected.

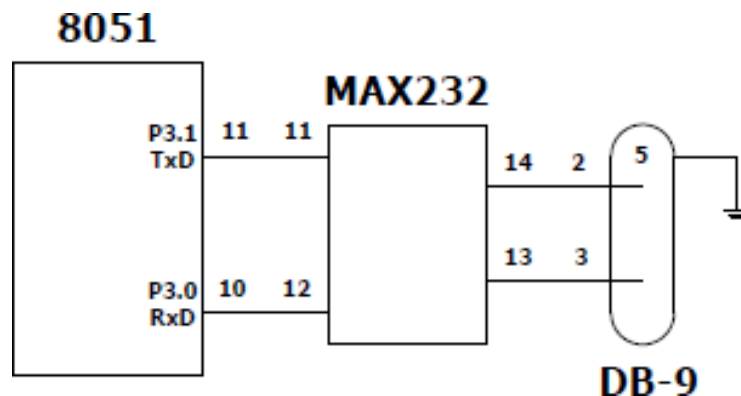
RI (ring indicator)

- An output from the modem and an input to a PC indicates that the telephone is ringing.

MAX232

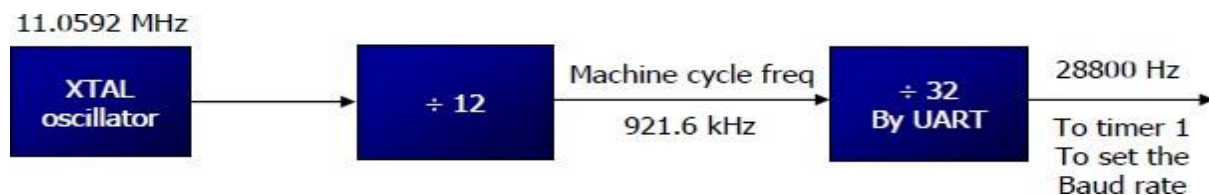
A line driver (MAX232) is required to convert RS232 voltage levels to TTL levels, and vice versa.

- 8051 has two pins that are used specifically for transferring and receiving data serially.
- These two pins are called Tx D and Rx D and are part of the port 3 (P3.0 and P3.1).
- These pins are TTL compatible.
- They require a line driver to make them RS232 compatible.



Baudrate:

- The baud rates in 8051 are programmable.
- 8051 divides the crystal frequency by 12 to get machine cycle frequency.
- 8051 UART circuitry divides the machine cycle frequency by 32.



- Timer 1 is used to set baud rate using TH1 register

Baudrate	TH1(decimal)	TH1(Hex)
9600	-3	FD
4800	-6	FA
2400	-12	F4
1200	-24	E8

Explain in detail the serial communication registers of the 8051. (NOV 2009) SBUF:

- It is an 8-bit register used for serial communication.
- For a byte of data to be transferred via the Tx Dline:
- Byte must be placed in the SBUF register.
- Bytes are framed with the start and stop bits and transferred serially via the Tx Dline.
- SBUF holds the byte of data when it is received by 8051 Rx Dline.
- When the bits are received serially via Rx D.
- 8051 de-frames byte by eliminating the stop and start bits.

SCON:

- It is an 8-bit register used to program the start bit, stop bit and data bits of data framing.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

Bit Number	Bit Mnemonic	Description
SCON.7	SM0	Serial port mode specifier
SCON.6	SM1	Serial port mode specifier
SCON.5	SM2	Used for multiprocessor communication
SCON.4	REN	Set/Cleared by software to enable/disable reception
SCON.3	TB8	Not widely used
SCON.2	RB8	Not widely used
SCON.1	TI	Transmit interrupt flag. Set by hardware at the begin of the stop bit mode 1. And cleared by software
SCON.0	RI	Receive interrupt flag. Set by hardware at the begin of the stop bit mode 1. And cleared by software

SM0,SM1:Serial port modespecifiers

SM0	SM1	
0	0	SerialMode0
0	1	SerialMode1;8-bitdata,1stopbit,1startbit
1	0	SerialMode2
1	1	SerialMode3

In programming the 8051 to transfer character bytes serially

1. TMOD register is loaded with the value 20H, indicating the use of timer 1 in mode 2 (8-bit auto-reload) to set baud rate.
2. The TH1 is loaded with one of the values to set baud rate for serial data transfer.
3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8-bit data is framed with start and stop bits.
4. TR1 is set to 1 to start timer 1
5. TI is cleared by CLR TI instruction.
6. The character byte to be transferred serially is written into SBUF register.
7. The TI flag bit is monitored with the use of instruction
JNB TI, xx, to see if the character has been transferred completely.
8. To transfer the next byte, go to step 5.

Write a program for the 8051 to transfer letter "A" serially at 4800 baud, continuously.

Solution:

```

MOV TMOD, #20H; timer 1, mode 2 (auto reload)
MOV TH1, #-6          ; 4800 baudrate
MOV SCON, #50H; 8-
bit, 1 stop, REN enabled
SETB TR1      ; start timer 1
AGAIN: MOV SBUF, #'A'
        ; letter "A" to transfer
HERE: JNB TI, HERE      ; wait for the last bit
        CLR TI          ; clear TI for next char
        SJMP AGAIN     ; keep sending A

```

The steps that 8051 goes through in transmitting a character via Tx



1. The byte character to be transmitted is written into the SBUF register
2. The start bit is transferred
3. The 8-bit character is transferred on bit at a time
4. The stop bit is transferred
 - It is during the transfer of the stop bit that 8051 raises the TI flag, indicating that the last character was transmitted
5. By monitoring the TI flag, we make sure that we are not overloading the SBUF
 - If we write another byte into the SBUF before TI is raised, the un-transmitted portion of the previous byte will be lost.
6. After SBUF is loaded with a new byte, the TI flag bit must be forced to 0 by CLR TI in order for this new byte to be transferred

By checking the TI flag bit, we know whether or not the 8051 is ready to transfer another byte

- It must be noted that TI flag bit is raised by 8051 itself when it finishes data transfer
- It must be cleared by the programmer with instruction CLR TI
- If we write a byte into SBUF before the TI flag bit is raised, we risk the loss of a portion of the byte being transferred
- The TI bit can be checked by the instruction JNB TI, xx Using an interrupt.

Write a program for the 8051 to transfer "YES" serially at 9600 baud, 8-bit data, 1 stop bit do this continuously. (May 2006)

Solution:

```

MOV TMOD, #20H; timer 1, mode 2 (autoreload)
MOV TH1, #-3          ; 9600 baud rate
MOV SCON, #50H; 8-bit, 1 stop, REN enabled
SETB TR1             ; start timer 1
AGAIN:  MOV A, #'Y'      ; transfer "Y"
        ACALL TRANS
        MOV A, #'E'
        ; transfer "E"
        ACALL TRANS
        MOV A, #'S'
        ; transfer "S"

```



```

        SJMPAGAIN        ;keepdoingit
;serialdata transfer subroutine
TRANS:  MOV  SBUF,A      ;loadSBUF
HERE:   JNB  TI,HERE    ;waitforthelastbit
        CLR  TI         ;getreadyfornextbyte
        RET  `

```

In programming the 8051 to receive character bytes serially

1. TMOD register is loaded with the value 20H, indicating the use of timer 1 in mode (8-bit auto-reload) to set baud rate
2. TH1 is loaded to set baud rate
3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8-bit data is framed with start and stop bits
4. TR1 is set to 1 to start timer 1
5. RI is cleared by CLR RI instruction
6. The RI flag bit is monitored with the use of instruction
JNB RI, x to see if a new character has been received yet
7. When RI is raised, SBUF has the byte, its contents are removed into a safe place.
8. To receive the next character, go to step 5.

Write a program for the 8051 to receive bytes of data serially and put them in P1, set the baud rate at 4800, 8-bit data and 1 stop bit. (NOV 2016)

Solution:

```

        MOV TMOD, #20H; timer 1, mode 2 (autoreload)
        MOV TH1, #-6          ;4800 baudrate
        MOV SCON, #50H; 8-
        bit, 1 stop, REN enabled
        SETB TR1      ;start timer 1
HERE:   JNB  RI, HERE    ;wait for char to come
        in MOVA, SBUF
        ;saving incoming byte in A
        MOV P1, A
        ;send to port 1
        CLR  RI         ;get ready to receive next byte
        SJMP HERE      ;keep getting data

```



In receiving bit via its RxD pin, 8051 goes through the following steps.

1. It receives the start bit

- Indicating that the next bit is the first bit of the character byte it is about to receive

2. The 8-bit character is received one bit at a time

3. The stop bit is received

- When

receiving the stop bit 8051 makes RI=1, indicating that an entire character byte has been received.

5. After the SBUF contents are copied into a safe place.

- The RI flag bit must be forced to 0 by CLRRI in order to allow the next received character byte to be placed in SBUF.
- Failure to do this causes loss of the received character.

There are two ways to increase the baud rate of data transfer

- To use a higher frequency crystal
- To change a bit in the PCON register

PCON

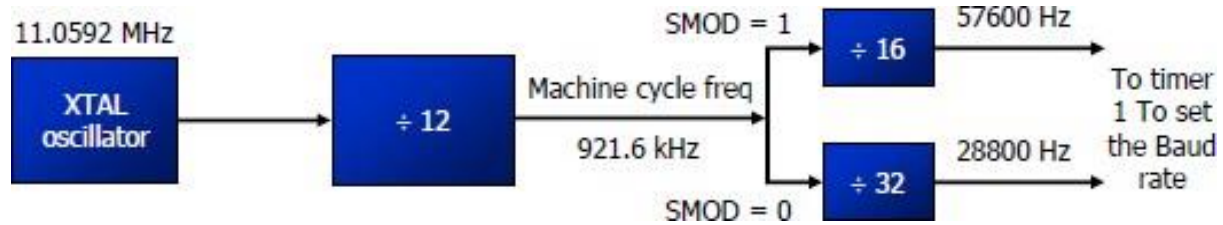
- PCON register is an 8-bit register
- When 8051 is powered up, SMOD is zero.
- We can set it to high by software and thereby double the baud rate.
- GF1, GF0: General flag bits
- PD: Power down mode
- IDL: Idle mode



```

MOV  A,PCON      ;place a copy of PCON in ACC
SETB ACC.7      ;make D7=1
MOV  PCON,A     ;changing any other bits

```



PIN Diagram of 8051 Microcontroller:

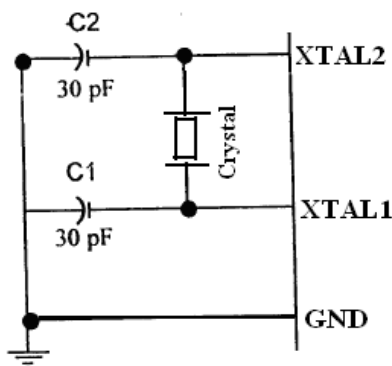
Explain Pin details of 8051 microcontroller. (MAY 2006)

Describe the functions of the following signals in 8051. RST, EA, PSEN and ALE. (NOV 2015)

- ✓ The 8051 microcontroller is available as a 40-pin DIP chip and it works at +5 volts DC.
- ✓ Among the 40 pins, a total of 32 pins are allotted for the four parallel ports P0, P1, P2 and P3. Each port occupies 8-pins.
- ✓ The remaining pins are VCC, GND, XTAL1, XTAL2, RST, EA, PSEN.

XTAL1, XTAL2:

- ✓ These two pins are connected to a Quartz crystal oscillator which runs the on-chip oscillator.
- ✓ The quartz crystal oscillator is connected to the two pins along with a capacitor of 30 pF as shown in the circuit.
- ✓ If used as a source other than the crystal oscillator, it will be connected to XTAL1 and XTAL2 is left unconnected.



RST:

- ✓ The RESET pin is an input pin and it is an active high pin.
- ✓ When a high pulse is applied to this pin, the microcontroller will reset and terminate all activities.
- ✓ Upon reset, all the registers will reset to 0000 value and SP register will reset to 0007 value.

...

\overline{EA} (External Access):

- ✓ This pin is an active low pin.
- ✓ This pin is connected to ground when microcontroller is accessing the program code stored in the external memory.
- ✓ This pin is connected to V_{cc} when it is accessing the program code in the on-chip memory.

 \overline{PSEN} (Program Store Enable):

- ✓ This is an output pin which is active low.
- ✓ When the microcontroller is accessing the program code stored in the external ROM, this pin is connected to the OE (Output Enable) pin of the ROM.

ALE (Address Latch Enable):

- ✓ This is an output pin, which is active high.
- ✓ This ALE pin will demultiplex the address and data bus.
- ✓ When the pin is high, the Address/Data bus will act as an address bus, otherwise the Address/Data bus will act as a Data bus.

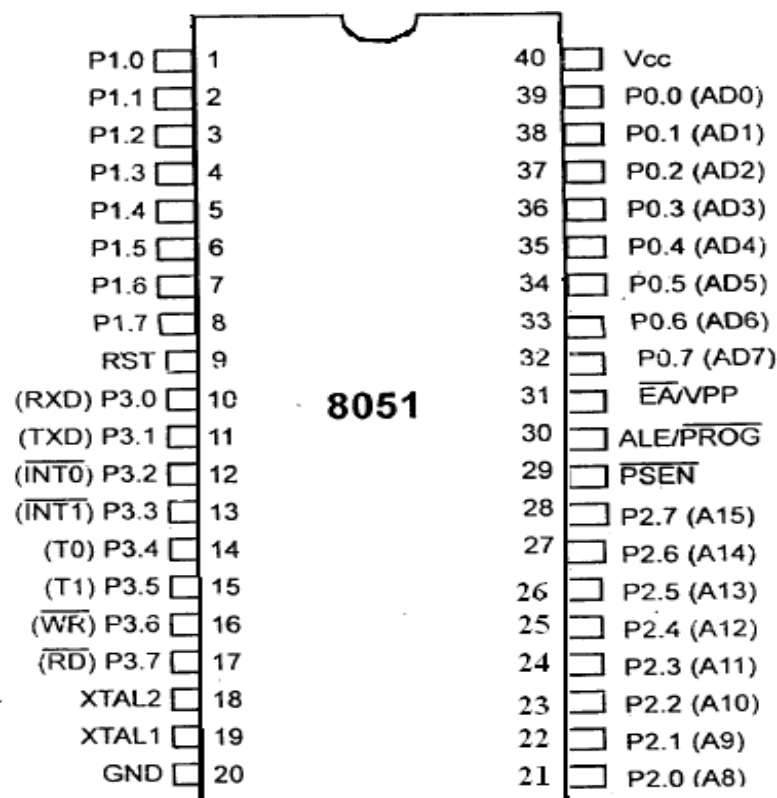


Figure: Pin diagram of 8051

P0.0-P0.7(AD0-AD7):

- ✓ The port 0 pins are multiplexed with address/data pins.
- ✓ If the microcontroller is accessing external memory, these pins will act as address/data pins, otherwise they are used for Port 0 pins.

P2.0-P2.7(A8-A15):

- ✓ The port 2 pins are multiplexed with the higher order address pins.
- ✓ When the microcontroller is accessing external memory, these pins provide the higher order address byte, otherwise they act as Port 2 pins.

P1.0-P1.7:

- ✓ These 8-pins are dedicated to perform input or output port operations.

P3.0-P3.7:

- ✓ These 8-pins are meant for Port 3 operations and also for some control operations like read, Write, Timer 0, Timer 1, INT0, INT1, RxD and TxD.



Program 1: Using timers in 8051 write a program to generate square wave 100ms, 50% duty cycle. (NOV 2014, May 2016, May 2012)

MOV TMOD, #01 Here:

MOV TL0,

#D7 MOV TH0, #B4

CPL

P1.3 SETB

TRO

Again: JNB TF0, Again C

LRTR0

CLRTF0

SJMP Here



Program2: Writean8051ALPtomultiplythegivennumber48Hand30H.(April2017)

Mnemonics		Comments
Opcode	Operand	
MOV	A,#48	;Storedata1inaccumulator
MOV	B,#30	;Storedata2inBregister
MUL	AB	;Multiplyboth
MOV	DPTR,#4500	;Initializememorylocation
MOVB	@DPTR,A	;Storelowerorderresult
INC	DPTR	;Gotonextmemorylocation
MOV	A,B	;Storehigherorderresult
MOVB	@DPTR,A	
L1:SJMPL	L1	;Stoptheprogram

Program3: Writeaprogramtoaddtwo16bitnumbers.Thenumbersare8C8Dand8D8C. PlacethesuminR7andR6.R6shouldhavethelowerbyte. (NOV 2010)

Mnemonics		Comments
Opcode	Operand	
MOV	A,#8D	;StoreLSBdata1inaccumulator
MOV	B,#8C	;StoreLSBdata2inBregister
ADD	A,B	;Addboth
MOV	R6,A	;StoreLSBresult
MOV	A,#8C	;StoreMSBdata1inaccumulator
MOV	B,#8D	;StoreMSBdata2inBregister
ADD	A,B	;Addboth
MOV	R7,A	;StoreMSBresult
L1:SJMPL	L1	;Stoptheprogram