8085 microprocessor notes

Book · December 2018

CITATIONS
0

READS
66,351

1 author:

Dr. Moorthi Madhavan
Saveetha Engineering College,Chennai,Tamilnadu,India
45 PUBLICATIONS **65** CITATIONS

SEE PROFILE

UNIT I -THE 8085 MICROPROCESSOR

THE 8085 MICROPROCESSORS

8085 Microprocessor architecture-Addressing modes- Instruction set-Programming the 8085

Part A

1. What is Microprocessor?

It is a program controlled semiconductor device (IC), which fetches, decodes and executes instructions.

2. What are the basic units of a microprocessor?

The basic units or blocks of a microprocessor are ALU, an array of registers and control unit.

3. What is Software and Hardware?

The Software is a set of instructions or commands needed for performing a specific task by a programmable device or a computing machine.

The Hardware refers to the components or devices used to form computing machine in which the software can be run and tested. Without software the Hardware is an idle machine.

4. What is assembly language?

The language in which the mnemonics (short -hand form of instructions) are used to write a program is called assembly language. The manufacturers of microprocessor give the mnemonics.

5. What are machine language and assembly language programs?

The software developed using 1's and 0's are called machine language, programs. The software developed using mnemonics are called assembly language programs.

6. What is the drawback in machine language and assembly language, programs?

The machine language and assembly language programs are machine dependent. The programs developed using these languages for a particular machine cannot be directly run on another machine.

7. Define bit, byte and word.

A digit of the binary number or code is called bit. Also, the bit is the fundamental storage unit of computer memory. The 8-bit (8-digit) binary number or code is called byte and 16-bit binary number or code is called word. (Some microprocessor manufactures refer the basic data size operated by the processor as word).

8. What is a bus?

Bus is a group of conducting lines that carries data, address and control signals.

9. Why data bus is bi-directional?

The microprocessor has to fetch (read) the data from memory or input device for processing and after processing, it has to store (write) the data to memory or output device. Hence the data bus is bi-directional.

10. Why address bus is unidirectional?

The address is an identification number used by the microprocessor to identify or access a memory location or I / O device. It is an output signal from the processor. Hence the address bus is unidirectional.

11. What is the function of microprocessor in a system?

The microprocessor is the master in the system, which controls all the activity of the system. It issues address and control signals and fetches the instruction and data from memory. Then it executes the instruction to take appropriate action.

12. How many machine cycles constitute one instruction cycle in 8085?

Each instruction of the 8085 processor consists of one to five machine cycles.

13. Define opcode and operand.

Opcode (Operation code) is the part of an instruction / directive that identifies a specific operation.

Operand is a part of an instruction / directive that represents a value on which the instruction acts.

14. What is opcode fetch cycle?

The opcode fetch cycle is a machine cycle executed to fetch the opcode of an instruction stored in memory. Every instruction starts with opcode fetch machine cycle.

15. What operation is performed during first T -state of every machine cycle in 8085 ?

In 8085, during the first T -state of every machine cycle the low byte address is latched into an external latch using ALE signal.

16. Why status signals are provided in microprocessor?

The status signals can be used by the system designer to track the internal operations of the processor. Also, it can be used for memory expansion (by providing separate memory banks for program & data and selecting the bank using status signals).

17. How the 8085 processor differentiates a memory access (read/write) and 1/0 access (read/write)?

The memory access and 1/0 access is differentiated using 10 I M signal. The 8085 processor asserts 10 I M low for memory read/write operation and 10 I M is asserted high for 1/0 read/write operation.

18. When the 8085 processor checks for an interrupt?

In the second T -state of the last machine cycle of every instruction, the 8085 processor checks whether an interrupt request is made or not.

19. What is interrupt acknowledge cycle?

The interrupt acknowledge cycle is a machine cycle executed by 8085 processor to get the address of the interrupt service routine in-order to service the interrupt device.

20. How the interrupts are affected by system reset?

Whenever the processor or system is reseted , all the interrupts except TRAP are disabled. in order to enable the interrupts, El instruction has to be executed after a reset.

21. What is Software interrupts?

The Software interrupts are program instructions. These instructions are inserted at desired locations in a program. While running a program, if software interrupt instruction is encountered then the processor executes an interrupt service routine.

22. What is Hardware interrupt?

If an interrupt is initiated in a processor by an appropriate signal at the interrupt pin, then the interrupt is called Hardware interrupt.

23. What is the difference between Hardware and Software interrupt?

The Software interrupt is initiated by the main program, but the Hardware interrupt is initiated by an external device.

In 8085, the Software interrupt cannot be disabled or masked but the Hardware interrupt except TRAP can be disabled or masked.

24. What is vectored and Non- Vectored interrupt?

When an interrupt is accepted, if the processor control branches to a specific address defined by the manufacturer then the interrupt is called vectored interrupt.

In Non-vectored interrupt there is no specific address for storing the interrupt service routine. Hence the interrupted device should give the address of the interrupt service routine.

25. List the Software and Hardware interrupts of 8085?

Software interrupts: RST 0, RSTl, RST 2,RST 3, RST 4, RST 5,RST 6 and RST 7.

Hardware interrupts: TRAP, RST 7.5, RST 6.5,RST 5.5 and INTR.

26. What is TRAP?

The TRAP is non-maskable interrupt of 8085. It is not disabled by processor reset or after reorganization of interrupt.

27. Whether HOLD has higher priority than TRAP or not?

The interrupts including map are recognized only if the HOLD is not valid, hence TRAP has lower priority than HOLD.

28. What is masking and why it is required?

Masking is preventing the interrupt from disturbing the current program execution. When the processor is performing an important job (process) and if the process should not be interrupted then all the interrupts should be masked or disabled.

In processor with multiple 'interrupts, the lower priority interrupt can be masked so as to prevent it from interrupting, the execution of interrupt service routine of higher priority interrupt.

29. When the 8085 processor accept hardware interrupt?

The processor keeps on checking the interrupt pins at the second T -state of last Machine cycle of every instruction. If the processor finds a valid interrupt signal and if the interrupt is unmasked and enabled then the processor accepts the interrupt. The acceptance of the interrupt is acknowledged by sending an OOA signal to the interrupted device.

30. When the 8085 processor will disable the interrupt system?

The interrupts of 8085 except TRAP are disabled after anyone of the following operations

1. Executing El instruction.
2. System or processor reset.
3. After reorganization (acceptance) of an interrupt.

31. What is the function performed by Dl instruction?

The function of Dl instruction is to enable the disabled interrupt system.

32. What is the function performed by El instruction?

The El instruction can be used to enable the interrupts after disabling.

33. How the vector address is generated for the INTR interrupt of 8085?

For the interrupt INTR, the interrupting device has to place either RST opcode or CALL opcode followed by l6-bit address. I~RST opcode is placed then the corresponding vector address is generated by the processor. In case of CALL opcode the given l6-bit address will be

the vector address.

34. How clock signals are generated in 8085 and what is the frequency of the internal clock?

The 8085 has the clock generation circuit on the chip but an external quartz crystal or L C circuit or RC circuit should be connected at the pins XI and X2. The maximum internal clock frequency of 8085A is 3.03 MHz

35. What happens to the 8085 processor when it is resetted?

When the 8085 processor is resetted it executes the first instruction at the OOOOH location. The 8085 resets (clears) instruction register, interrupt mask bits and other registers.

36. What are the operations performed by ALU of 8085?

The operations performed by ALU of 8085 are Addition, Subtraction, Logical AND, OR, Exclusive OR, Compare Complement, Increment, Decrement and Left I Right shift

37. What is a flag?

Flag is a flip flop used to store the information about the status of the processor and the status of the instruction executed most recently.

38. List the flags of 8085

There are five flags in 8085. They are sign flag, zero flag, Auxiliary carry flag, and parity flag and carry flag.

39. What is the Hardware interrupts of 8085?

The hardware interrupts in 8085 are TRAP, RST 7.5, RST 6.5 and RST 5,5.

40. Which interrupt has highest priority in 8085? What is the priority of other interrupts?

The TRAP has the highest priority, followed by RST 7.5, RST 6.5, RST 5.5 and INTR.

41. What is ALE?

The ALE (Address Latch Enable) is a signal used to demultiplex the address and data lines, using an external latch. It is used to enable the external latch.

42. Explain the function of IO/M in 8085.

The IO/M is used to differentiate memory access and I/O access. For IN and OUT instruction it is high. For memory reference instructions it is low.

43. Where is the READY signal used?

READY is an input signal to the processor, used by the memory or I/O devices to get extra time for data transfer or to introduce wait states in the bus cycles.

44. What are HOLD and HLDA and how it is used?

Hold and hold acknowledge signals are used for the Direct Memory Access (DMA) type of data transfer. The DMA controller place a high on HOLD pins in order to take control of the system bus. The HOLD request is acknowledged by the 8085 by driving all its tristated pins to

high impedance state and asserting HLDA signal high.

45. Difference between microprocessor and microcontroller.

Microprocessor:

* It is single VLSI chip holding CPU unit.
* It is dedicated to the specific instruction
* It is based on **Princeton Architecture** i.e. program as well as data stored in same memory location.

Microcontroller:

*It includes microprocessor and memory, peripheral devices on a single unit.
*It is dedicated to implement the specific instruction.
*It is based on **Harvard architecture** i.e. program and data will be stored in different memory location.

46. What is bus contention?

If two devices drive the data bus simultaneously then it is called Bus Contention. It may lead to following undesirable events.

*Damaging one or both the IC chip
*The high current may cause a voltage spike in the supply system leading to data loss.

47. What is an assembler?

An assembler is a program that translates the mnemonics into their machine code. It is generally not available on a single-board microcomputer.

A program can be entered in mnemonics in a microcomputer equipped with an ASCII keyboard. The assembler will translate mnemonics into the 8085 machine code and assign memory locations to each machine code, thus avoiding the manual assembly and the errors associated with it. Additional instructions can be inserted anywhere in the program, and the assembler will assign all the new memory locations and jump instructions.

48. How does the microprocessor differentiate between data and instruction?

When the first machine code of an instruction is fetched and decoded in the instruction register, the microprocessor recognizes the number of bytes required to fetch the entire instruction. For example, in the case of the instruction MVI A, data (3E data), the second byte is always considered data. If that data byte is omitted by mistake, whatever is in that memory location will be considered data. The byte after "Data" will be treated as the next instruction.

49. How does the microprocessor differentiate among positive number, a negative number and a bit pattern?

It does not know the difference. The microprocessor views any data byte as eight binary digits. The programmer is responsible for providing the interpretation.

For example, after an arithmetic or logic operation, if the bits in the accumulator are
1 1 1 1 0 0 1 0 = F2 H

The sign flag is set because D7=1. This does not mean it is a negative number, even if the sign flag is set. The sign flag indicates only that D7=1. the eight bits in the accumulator could be a bit pattern, or a positive number larger that 127_{10}, or the 2's complement of a number.

50. List the components of microprocessor (single board microcomputer) based system.

The microprocessor based system consist of microprocessor as CPU, semiconductor memories like EPROM and RAM, input device, output device and interfacing devices

51. Define machine cycle.

Machine cycle is defined as the time required to complete one operation of accessing memory, input / output or acknowledging an external request. This cycle may consist of 3 to 6 T-states.

52. Define T-state.

T-state is defined as one subdivision of the operation performed in 1 clock period. These subdivisions are internal states synchronized with the system clock, and each T-state is precisely to 1 clock period.

53. What is instruction cycle?

The sequence of operations that a processor has to carry out while executing the instruction is called instruction cycle. Each instruction cycle of a processor consists of a number of machine cycles.

54. What does memory-mapping mean?

The memory mapping is the process of interfacing memories to microprocessor and allocating addresses to each memory locations.

55. What is the need for timing diagram?

The timing diagram provides information regarding the status of various signals, when a machine cycle is executed. The knowledge of timing diagram is essential for system designer to select matched peripheral devices like memories, latches, ports etc, to form a microprocessor system.

Part B

1. Explain briefly about bus structure of 8085

The microprocessor unit performs primarily four operations:

i) **Memory Read:** Reads data (or instructions) from memory

ii) **Memory Write:** Writes data (or instructions) into memory

iii) **I/O read:** Accepts data from input devices

iv) **I/O Write:** Sends data to output devices

All these operations are part of the communication process between the MPU and the peripheral devices (including memory). To communicate with a peripheral (or the memory location), the MPU needs he following steps:

Step 1: Identify the peripheral or the memory location (with its address)

Step 2: transfer binary information (data and instructions)

Step 3: Provide timing or synchronization signals.

The 8085 MPU performs these functions using three set of communication lines called buses: the address bus, data bus and control bus.

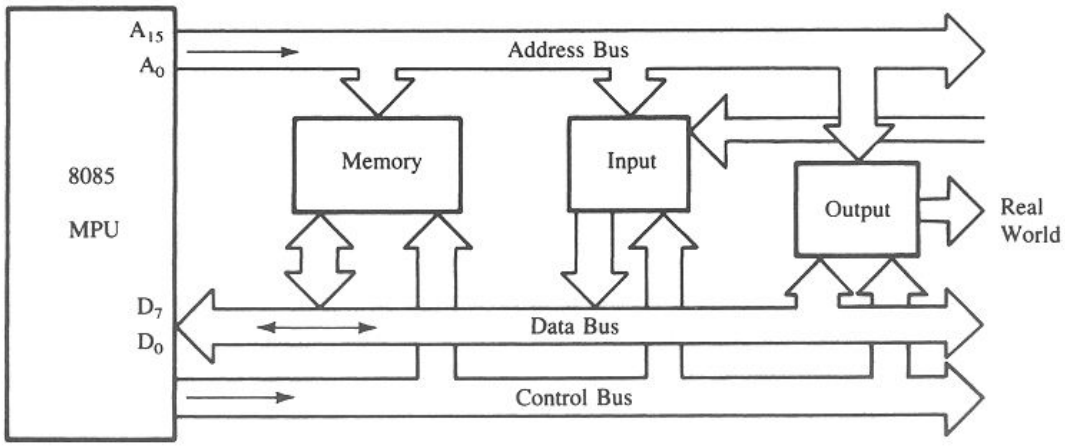


Figure 1-Bus structure of 8085

ADDRESS BUS

The address bus is a group of 16 lines generally identified as A_0 to A_{15}. the address bus is unidirectional: bits flow in one direction – from the MPU to peripheral devices. The MPU uses the address bus to perform the first function: identifying a peripheral or a memory location. In a computer system, each peripheral or memory location is identified by a binary number called an address and the address bus is used to carry a 16-bit address. This is similar to postal address of a house.

DATA BUS

The data bus is a group of eight lines used for data flow. These lines are bidirectional - data flow in both directions between the MPU and memory and peripheral devices. The MPU uses the data bus to perform the second function: transferring binary information. The eight data lines enable the MPU to manipulate 8- bit data ranging from 00 to FF.

CONTROL BUS

The control bus is comprised of various signal lines that carry synchronization signals. The MPU uses such lines to perform the third function: provide timing signals.

The control signals are not group of lines like address or data buses, but individual lines that provide pulse to indicate an MPU operation. The MPU generates specific control signals for every operation (such as memory read or I/O write) it performs. These signals are used to identify a device type with which the MPU intends to communicate.

2. Explain the features of 8085 in detail.

The features of 8085 include:

1. It is an 8-bit microprocessor i.e. it can accept, process or provide 8-bit data simultaneously.
2. It operates on a single +5V power supply connected at Vcc
3. It operates on clock cycle with 50% duly cycle.
4. It has on chip clock generator this internal clock generator requires tuned circuit like LC, RC or crystal. The internal clock generator divides oscillation frequency by 2 and generates clock signal, which can be used for synchronizing external devices.
5. It can operate with 3 MHz clock frequency.
6. It has 16 address buses, hence it can access 2^{16} 64 bytes of memory.
7. It provides 8 bit I/o address to acce4ss (2^8) 256 I / o ports.
8. In 8085, the lower 8-bit address bus (A_0-A_7) and data bus (D_0-D_7) are multiplexed to reduce number of external pins. But due to this, external hardware is required to separate address lines and data lines.
9. It supports 74 instructions with following addressing modes. (a) Immediate, (b) Register, (c) Direct (d) Indirect (e) Implied.
10. The Arithmetic logic unit of 8085 performs a) 8 bit binary addition with or without carry. (b) 16 bit binary addition (c) 2 digit BCD addition (d) 8-bit binary subtraction with or without borrow (e) 8-bit logical AND, OR, EX-OR, complement (NOT) and bit shift operations.
11. It has 8-bit accumulator, flag register, instruction, register, six 8-bit general purpose. Registers (B, C, D, E, H and C) and five 16-bit registers (SP and PC)
12. It provides five hardware interrupts: TRAP, RST 7.5. RST 6.5, RST 5.5 and INTR.
13. It has serial I/O control which allows serial communication.
14. It provides control signals (IO /M, RD, ~~WR)~~ ~~to con~~trol bus cycles.
15. The external hardware (another microprocessor or equivalent master) can detect which machine cycle microprocessor is executing using status signals (IO/M, S_0, S_1) This

feature is useful when more than one processors are using common system resources (memory & I/O devices).

16. It has mechanism by which it is possible to increase its interrupt handling capacity.
17. The 8085 has an ability to share system bus with direct memory access controller. This feature allows to transfer large amount of data from I/O device to memory or from memory to I/O device with high speeds.

3. Draw and explain the architecture of 8085 microprocessorMAY06

ARCHITECTURE OF 8085

It consists of various functions blocks as listed below:

1) Registers
2) Arithmetic and logic unit
3) Instruction decoder and machine cycle encoder
4) Address Buffer
5) Address / Data Buffer
6) Incrementer / Decremented address batch
7) Serial I/O control
8) Timing and control circuitry

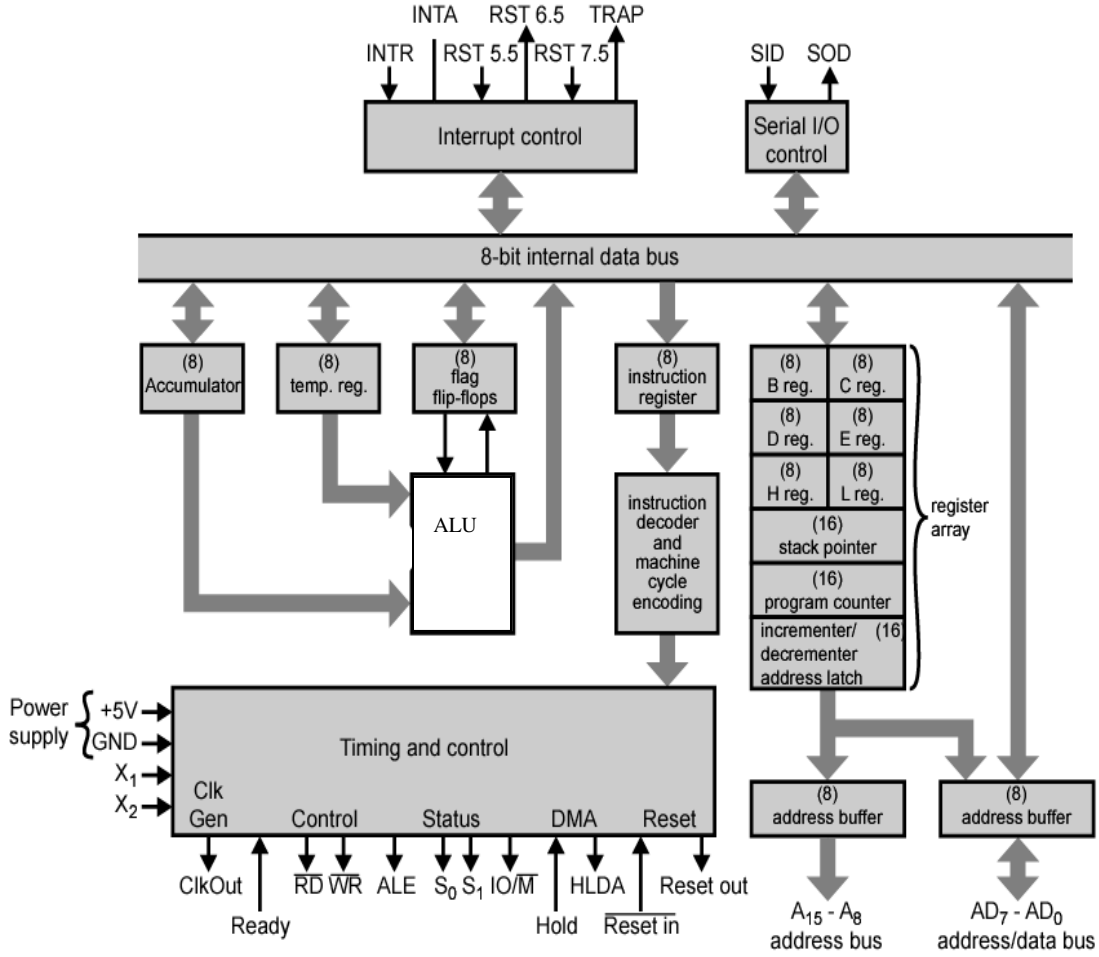


Figure 2-Architecture of 8085

REGISTERS:

It has eight addressable 8-bit registers: A, B, C, D, E, H, L, F and two 16-bit registers PC and SP. These register can be classified as :

1. General purpose registers
2. Temporary registers (a) Temporary data register (b) W and Z registers
3. Special purpose registers (a) Accumulator (b) Flag registers (c) Instruction register
4. Sixteen bit registers (a) Program counter PC (b) Stack pointer (SP)

Accumulator A (8)	S	Z		AC		P		CY
B	C							
D	E							
H	L							
Stack Pointer								
Program Counter								

Figure 3-Registers in 8085

1. General Purpose Registers :

B,C,D,E,H and L are 8-bit general purpose registers can be used as a separate 8-bit registers or as 16-bit register pairs, BC,DE and HL. HL pair also functions as a data pointer or memory pointer. These are also called Scratch pad registers, as user can store data in them. To store and read data from these registers bus access is not required, it is an internal operation. Thus it provides an efficient way to store intermediate results and used them when required.

2. Temporary Registers :

a) Temporary Data Register :

The ALU has two inputs. One input is supplied by the accumulator and other from temporary data register. The programmer cannot access this temporary data register. However, it is internally used for execution of most arithmetic and logical instructions.

b) W and Z registers :

W and Z registers are temporary registers. These registers are used to hold 8-bit data during execution of some instruments. These registers are not available for programmer, since 8085 user them internally.

3. Special Purpose Registers:

a) Register A (Accumulator) :

It is a tri-state eight bit register it is extensively used in arithmetic, logic, load and store operations, as well as, input / output (I/O) operations. Most of the times the result of arithmetic and logical operations is stored in the register A, hence it is also identified as accumulator.

b) Flag Register :

It is an 8-bit register, in which five of the bits carry significant information in the form of flags : 5 (sign flag) Z (Zero flag), AC (Auxillary carry flag)m P (Parity flag) and CY (carry flag).

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

Figure 4-Flag Register

S-Sign flag: After the execution of arithmetic or logic operation, if bit D_7 of the result is 1, the sign flag is set. In a given byte if bit D_7 is 1, the number will be viewed as negative number. If D_7 is 0, the number is considered as positive number.

Z-Zero flag: The zero flag sets if the result of operation in ALU is zero and flag resets if result is non zero. The zero flag is also set if a certain register content becomes zero following an increment or decrement operation of that register.

AC-Auxiliary carry flag: This flag is set if there is an overflow out of bit3 i.e., carry from lower nibble to higher nibble (D_3 to D_4 bit). This flag is used for BCD operations.

P-Parity flag: Parity is defined by the number of ones present in the accumulator. After an arithmetic or logic operation if the result has an even number of ones, i.e., even parity, the flag is set. If the parity is odd, flag is reset.

CY-Carry flag: This flag is set, if there is an overflow out of bit 7. The carry flag also serves as a borrow flag for subtraction.

c) Instruction register:

In a typical processor operation, the processor first fetches the opcode of instruction from memory (i.e. it places an address on the address bus and memory responds by placing the data stored at the specified address on the data bus). The CPU stores this opcode in a register called instruction register. This opcode is further sent to the instruction decoder to select one of the alternatives.

4.Sixteen bit registers

a) Program Counter (PC):

Program is a sequence of instructions. Microprocessor fetches these instructions from the memory and executes them sequentially. The program counter is a special purpose register which, at a given time, stores the address of the next instruction to be fetched. Program counter acts as a pointer to the next instruction. How processor increments the program counter depends upon the nature of instructions: for one byte instruction it increments program counter by one, for two byte instruction it increments program counter by two and for three byte instruction it increments program counter by three such that program counter always points to the address of the next instruction.

b) Stack Pointer:

The stack is a reserved area of the memory in the RAM where temporary information may be stored. A 16-bit stack pointer is used to hold the address of the most recent stack entry.

Arithmetic and Logic Unit (ALU):

The 8085's ALU performs arithmetic and logical functions on eight bit variables. The arithmetic unit performs bitwise fundamental, operation such as addition and subtraction. The logic unit performs the logical operations such as complement, AND, OR and EX-OR as well as rotate and clear. The ALU also looks after branching decisions.

Instruction Decoder

The processor first fetches the opcode of instruction from memory and stores this opcode in the instruction register. It is then sent to the instruction decoder. The instruction decoder decodes it and accordingly gives the timing and control signals which control the register, data buffer, ALU, and external peripheral signals.

The 8085 executes seven different types of machine cycles. It gives the information about which machine cycle is currently executing in the encoded form on the S_0, S_1 and IO/M lines. The task is done by machine cycle encoder.

Address buffer

This is an 8-bit unidirectional buffer. It is used to drive external high order address (A_{15}-A_8). It is also used to tri-state the high order address bus under certain conditions such as reset, hold, halt and when address lines are not in use.

Address / Data buffer

This is an 8-bit bidirectional buffer. It is used to drive multiplexed address/data bus i.e. low order address bus (A_7-A_0) and data bus (D_7-D_0). It is also used to tri-state the multiplexed address/data bus under certain conditions such as reset, hold, halt and when bus is not in use.

Incrementer/Decrementer address latch

This 16-bit register is used to increment or decrement the contents of program counter or stack pointer as a part of execution of instructions related to them.

Interrupt Control

The processor fetches, decodes and executes the instructions in a sequence. Sometimes it is necessary to have the processor automatically execute one of a collection of special routines whenever special condition exists within a program or the microcomputer system. After the execution of special routine, the program control must be transferred to the program which processor was executing before the occurrence of the special condition. The occurrence of this special condition is referred as interrupt. The interrupt control block has five interrupt inputs RST 5.5, RST 6.5, RST 7.5, TRAP and INTR and one acknowledge signal INTA.

Serial I/O Control

In situations like, data transmission over long distance and communication with cassette tapes or CRT terminal, it is necessary to transmit data bit by bit to reduce the cost of cabling. In serial communication one bit is transferred at a time over a single line. The 8085's serial I/O control provides two lines, SID and SOD for serial communication. The Serial Output Data (SOD) line is used to send data serially and Serial Input Data (SID) line is used to receive data serially.

Timing and Control circuitry

The control circuitry in the processor 8085 is responsible for all the operations. The control circuitry and hence the operations in 8085 are synchronized with the help of clock signal. Along with the control of fetching and decoding operations and generating appropriate signals for instruction execution, control circuitry, also generates signals required to interface external devices to the processor 8085.

4. With neat PIN diagram explain the various signals of 8085 microprocessorMAY04

PIN DEFINITIONS OF 8085

The signals of 8085 can be classified into six groups according to their functions:

a) Address bus
b) Data bus
c) Control and status signals
d) Power supply and frequency signals
e) Externally initiated signals and
f) Serial I/O ports.

ADDRESS BUS

The 8085 has eight lines, A_{15}-A_8,which are unidirectional and used as the high order address bus.

MULTIPLEXED ADDRESS / DATA BUS

The signal lines AD_7-AD_0 are bidirectional: they serve a dual purpose. They are used as the low order address bus as well as the data bus. In executing an instruction, during the earlier part of the cycle, these lines are used as the low-order address bus. During the later part of the cycle, these lines are used as the data bus. (This is also known as multiplexing the bus.) However, the low-order address bus can be separated from these signals by using a latch

CONTROL AND STATUS SIGNALS

This group of signals includes two control signals (RD and WR), three status signals (IO/M, S_1, and S_0) to identify the nature of the operation, and one special signal (ALE) to indicate the beginning of the operation. These signals are as follows:

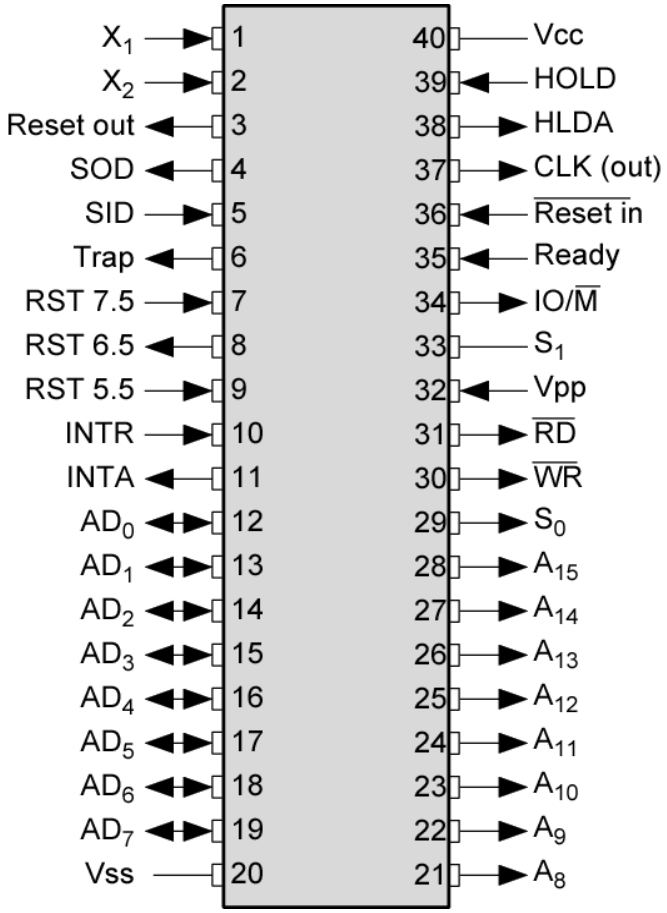


Figure 5-Pin diagram of 8085

- **ALE — Address Latch Enable:** This is positive going pulse generated

every time the 8085 begins an operation (machine cycle); it indicates that the bits on AD_7-AD_0 are address bits. This is used primarily to latch the low-order address from the multiplexed bus and generate a separate set of eight address lines, A_7-A_0.

- **$\overline{RD}$ — Read:** This read control signal (active low). This signal indicates that the selected I/O or memory device is to be read and data are available on the data bus.
- **$\overline{WR}$ — Write:** This is a write control signal (active low). This signal indicates that the data on the data bus are to be written on into a selected memory device or I/O location.
- **IO/$\overline{M}$:** This is status signal used to differentiate between I/O and memory operations. When it is high, it indicates an I/O operation; when it is low, it indicates a memory operation. This signal combines with RD and WR to generate I/O and memory control signals.
- **S_1 and S_0:** These status signals, similar to IO/M, can identify various operations, but they are rarely used in small systems.

Machine Cycle	IO/M	S1	S0	Control Signals
Opcode fetch	0	1	1	$RD^{-}=0$
Memory Read	0	1	0	$RD^{-}=0$
Memory Write	0	0	1	$WR^{-}=0$
I/O Read	1	1	0	$RD^{-}=0$
I/O Write	1	0	1	$WR^{-}=0$
Interrupt Ack	1	1	1	$INTA^{-}=0$
Halt	Z	0	0	
Hold	Z	X	X	$RD^{-},WR^{-}=Z$
Reset	Z	X	X	$INTA^{-}=1$

POWER SUPPLY AND CLOCK FREQUENCY

The power supply and frequency signals are as follows:

- **V_{cc}:** +5V power supply.**V_{ss}:** Ground reference.
- **X_1, X_2:** A crystal is connected at these two pins. The frequency is internally divided by two; therefore, to operate a system at 3MHz and, the crystal should have frequency of 6MHz.
- **CLK (OUT) — Clock out:** This signal is used as the system clock for other devices.

EXTERNALLY INITIATED SIGNALS, INCLUDING INTERRUPTS

The 8085 has five interrupt signals that can be used to interrupt a program execution. The interrupt signals are Interrupt Request (INTR), Restart Interrupts (RST5.5, RST 6.5, RST7.5) and TRAP. The microprocessor acknowledges an interrupt request by the INTA (Interrupt acknowledge) signal. In addition to the interrupts, three pins — RESET, HOLD, and READY — accept the externally initiated signals as inputs. To respond to the HOLD request, the 8085 has one signal called HLDA (Hold Acknowledge).

INTR(input)	Interrupt Request: This is used as a general-purpose interrupt.
INTA(Output)	Interrupt Acknowledge: This is used to acknowledge an interrupt.
RST 7.5(inputs) RST 6.5 RST 5.5	Restart Interrupts: These are vectored interrupts and transfer the program control to specific memory locations. They have higher priorities than the INTR interrupt. Among these three, the priority order is 7.5, 6.5 and 5.5
TRAP(input)	This is a non maskable interrupt and has the highest priority
HOLD(input)	This signal indicates that a peripheral such as a DMA controller is requesting the use of the address and data buses.
HLDA	Hold Acknowledge: This signal acknowledges the HOLD request.

READY	This signal is used to delay the microprocessor Read or Write cycles until a slow-responding peripheral is ready to send or accept data. When this signal goes low, the microprocessor waits for an integral number of clock cycles until it goes high.

RESET, HOLD and READY are additional interrupts. They accept the externally initiated signals as inputs. To respond to the HOLD request, it has one signal called HLDA.

RESET IN: When the signal on this pin goes low, the program counter is set to zero, the buses are tri-stated, and the microprocessor is reset.

RESET OUT: This signal indicates that the microprocessor is being reset. The signal can be used to reset other devices.

SERIAL I/O PORTS

The microprocessor has 2 pins specially designed for software –controlled serial I/O. one is called SOD (Serial Output Data) and the other is called SID (Serial Input Data). Data transfer is controlled through 2 instructions: SIM and RIM. The instruction SIM is necessary to output data serially from the SOD line. Similarly instruction RIM is used to input serial data through the SID line.

The SID and SOD lines in the 8085 eliminate the need for an input port and an output port in the software-controlled serial I/O. Essentially, the SID is a 1-bit port and SOD is a 1-bit output port.

5. Explain the various addressing modes of 8085 microprocessor with example NOV04

ADDRESSING MODES:

The different ways that a microprocessor can access data are referred to as addressing modes. The 8085 has 5 addressing modes. These are:

1. **Immediate addressing mode**: In an immediate addressing mode, 8 or 16 bit data can be specified as a part of instruction. In 8085, the instructions having 'I' letter fall under this category. "I' indicates immediate addressing mode.

   Example: MVI A,20H ; moves 8-bit immediate data(20H) into accumulator.

   LXI D,10FF H ; moves 16-bit immediate data into DE register pair.

2. Register addressing mode: The register addressing mode specifies the source operand, destination operand or both to be contained in an 8085 registers. This results in faster execution, since it is not necessary to access memory locations for operands.

Example: MOV A, B; moves the contents of register B into the accumulator.

SPHL ; moves the contents of HL register pair into stack pointer.

3. Direct addressing mode: The direct addressing mode specifies the 16- bit address of the operand within the instruction itself. The second and third bytes of instruction contain this 16 bit address.

Example: LDA 2000H ; loads the 8bit contents of memory location 2000H into the accumulator

SHLD 3000H ; Stores the HL register pair into two consecutive memory locations. Lower contents of L register into memory location 3000H and higher contents of H register into memory location 3001H.

4. Indirect addressing mode: In indirect addressing mode, the memory address where the operand located is specified by the contents of a register pair.

Example: LDAX B ; loads the accumulator with the contents of memory location pointed by BC register pair.

MOV M, A ; Stores the contents of accumulator into the memory location pointed by HL register pair

5. Implied addressing mode: In implied addressing mode, Opcode specifies the address of the operands.

Example: CMA; Complements contents of accumulator.

RAL;Rotates the contents of accumulator left through the carry.

6. Explain how the instruction word size are classified in 8085 microprocessorMAY05
INSTRUCTION WORD SIZE:
The 8085 instruction set is classified into the following three groups according to word size:

1. One-word or 1-byte instructions
2. Two-word or 2-byte instructions
3. Three-word or 3-byte instructions

In the 8085, “byte” and “word” are synonymous because it is an 8-bit microprocessor.

ONE-BYTE INSTRUCTIONS

A 1-byte instruction includes the opcode and the operand in the same byte. For example:

Task	Opcode	Operand	Binary Code	Hex code
Copy the contents of the accumulator in register C.	MOV	C,A	0100 1111	4FH

TWO-BYTE INSTRUCTIONS

In a 2-byte instruction, the first byte specifies the operation code and the second byte specifies the operand. For example:

Task	Opcode	Operand	Binary Code	Hex code
Load an 8-bit data byte in the accumulator	MVI	A, Data	0011 1110 DATA	3E-First byte Data- Second byte

This instruction would require two memory locations to store in memory.

THREE-BYTE INSTRUCTIONS

In a 3-byte instruction, the first byte specifies the opcode and the following two bytes specify the 16-bit address. Note that the second byte is the low-order address and the third byte is the high-order address. For example:

Task	Opcode	Operand	Binary Code	Hex code
Transfer the program sequence to the memory location 2085H.	JMP	2085H	1100 0011 1000 0101 0010 0000	C3 First byte 85Second byte 20 Third byte

This instruction would require three memory locations to store in memory.

7. Explain how the instructions are classified in 8085 microprocessor?MAY05

THE 8085 INSTRUCTION SET

The 8085 instructions can be classified into the following five functional categories:

1. Data transfer (copy) operations
2. Arithmetic operations

3. logical operations
4. Branching operations and
5. Machine control operations.

An instruction is a command to the microprocessor to perform a given task on specified data. Each instruction has two parts: one is the task to be performed, called the operation code (opcode), and the second is the data to be operated on, called the operand (or data) can be specified in various ways. It may include 8-bit (or 16-bit) data, an internal register, a memory location, or an 8-bit (or 16-bit) address.

1.DATA TRANSFER (COPY) OPERATIONS

This group of instructions copies data from a location called a source to another location called destination, without modifying the contents of the source. The various types of data transfer are listed below together with examples of each type:

Types	Examples
• Between registers	Copy the contents of register B into Register D.
• Specific data byte to a register or a memory location	Load register B with the data byte
• Between a memory location and a register	From the memory location 2000H to register B.
• Between an I/O device and the accumulator.	From an input keyboard to the accumulator.

INSTRUCTIONS

The data transfer instructions copy data from a source into a destination without modifying the contents of the source. The previous contents of the destination are replaced by the contents of the source.

Opcode	Operand	Description
MOV	R_d,R_s	Move • This is a 1-byte instruction • Copies data from source register R_s to destination register R_d
MVI	R, 8-bit	Move Immediate • This is a 2- byte instruction • Loads the 8 bits of the second byte into the register specified.
OUT	8-bit port address	Output to port • This is a 2- byte instruction • Sends the contents of the accumulator (A) to the output port specified in the second byte

IN	8-bit port address	Input from port • This is a 2- byte instruction • Accepts data from the input port specified in the second byte, and loads into the accumulator.
HLT		Halt • This is a 1-byte instruction • The processor stops executing and enters wait state • The address bus and data bus are placed in high impedance state. No register contents are affected.
NOP		No operation • This is a 1-byte instruction • No operation is performed • Generally used to increase processing time or substitute in place of an instruction.

2.ARITHMETIC OPERATIONS

These instructions perform arithmetic operations such as addition, subtraction, increment and decrement.

- **Addition** — Any 8-bit number, or the contents of a register, or the contents of a memory location can be added to the contents of the accumulator and the result is stored in the accumulator. No two other 8-bit registers can be added directly. The instruction DAD is an exception.
- **Subtraction** — Any 8-bit number, or the contents of a register, or the contents of a memory location can be subtracted from the contents of the accumulator and the result is stored in the accumulator. The subtraction is performed in 2's complement and the results, if negative, are expressed in 2's complement. No two other 8-bit registers can be subtracted directly.
- **Increment / Decrement** — The 8-bit contents of a register or a memory location can be incremented or decremented by one. Similarly, the 16- bit contents of a register pair can be incremented or decremented by 1. These increment and decrement operation differ from the addition and subtraction in an important way; i.e., they can be performed in one of the registers pr in a memory location.

These arithmetic instructions (except INR and DCR)

1. Assume implicitly that the accumulator is one of the operands.
2. Modify all the flags according to the data conditions of the result.
3. place the result in the accumulator
4. Do not affect the contents of the operand register.

The instructions INR and DCR

1. Affect the contents of the specified register.
2. Affect all flags except the CY flag.

The descriptions of the instructions are as follows:

Opcode	Operand	Description
ADD	R	Add • This is a 1-byte instruction • Adds the contents of register R to the contents of the accumulator
ADI	8-bit	Add Immediate • This is a 2-byte instruction • Adds the second byte to the contents of the accumulator
SUB	R	Subtract • This is a 1-byte instruction • Subtracts the contents of register R from the contents of the accumulator
SBI	8-bit	Subtract Immediate • This is a 2-byte instruction • Subtracts the second byte from the contents of the accumulator
INR	R	Increment • This is a 1-byte instruction • Increases the contents of the register R by 1. Caution: All flags except CY are affected.
DCR	R	Decrement • This is a 1-byte instruction • Decreases the contents of the register R by 1. Caution: All flags except CY are affected.

3.LOGICAL OPERATIONS

These instructions perform various logical operations with the contents of the accumulator.

- **AND,OR, Exclusive-OR** — Any 8-bit number, or the contents of a register, or of a memory location can be logically ANDed, ORed, or Exclusive-ORed with the contents of the accumulator. The results are stored in the accumulator.
- **Rotate** — Each bit in the accumulator can be shifted either left or right to the next position.
- **Compare** — Any 8-bit number, or the contents of a register, or of a memory location can be compared for equality, greater than, or less than, with the contents of the accumulator.
- **Complement** — The contents of the accumulator can be complemented; all 0s are replaced by 1s and all 1s are replaced by 0s.

INSTRUCTIONS

The logic instructions

1. implicitly assume that the accumulator is one of the operands.

2. reset (clear) the CY flag. The instruction CMA is an exception; it does not affect any flags.
3. modify the Z, P, and S flags according to the data conditions of the result.
4. place the result in the accumulator.
5. do not affect the contents of the operand register.

Opcode	Operand	Description
ANA	R	Logical AND with accumulator • This is a 1-byte instruction • Logically ANDs the contents of the register R with the contents of the accumulator • 8085: CY is reset and AC is set
ANI	8-bit	AND Immediate with accumulator • This is a 2-byte instruction • Logically ANDs the second byte with the contents of the accumulator • 8085: CY is reset and AC is set
ORA	R	Logically OR with Accumulator • This is a 1-byte instruction • Logically ORs the contents of the register R with the contents of the accumulator
		• ORI 8-bit OR Immediate with the accumulator • This is a 2-byte instruction • Logically ORs the second byte with the contents of the accumulator
XRA	R	Logically Exclusive-OR with Accumulator • This is a 1-byte instruction • Exclusive-ORs the contents of the register R with the contents of the accumulator
XRI	8-bit	Exclusive-OR Immediate with the accumulator • This is a 2-byte instruction • Exclusive-ORs the second byte with the contents of the accumulator
CMA		Complement Accumulator • This is a 1-byte instruction that complements the contents of the accumulator • No flags are affected

4.BRANCH OPERATIONS

This group of instructions alters the sequence of program execution either conditionally or unconditionally.

- Jump — Conditional jumps are an important aspect of the decision making process in programming. These instructions test for a certain condition and alter the program sequence when the condition is met. In addition, the instruction set includes an instruction called unconditional jump.
- **Call, Return, and Restart** — These instructions change the sequence of a program either by calling a subroutine or returning from a subroutine. The conditional Call and Return instructions can also test condition flags.

INSTRUCTIONS

The branch instructions are classified in three categories:

1.Jump instructions

2. Call and Return instructions

3.Restart instructions

Jump instructions are classified into two categories: Unconditional Jump and Conditional Jump.

UNCONDITIONAL JUMP

The 8085 instruction set includes one unconditional Jump instruction. The unconditional jump instruction enables the programmer to set up continuous loops.

Opcode	Operand	Description
JMP	16-bit	Jump • This is a 3-byte instruction • The second and third bytes specify the 16-bit address. However, the second byte specifies the low-order and the third byte specifies the high-order memory address.

CONDITIONAL JUMPS

Conditional jump instructions allow the microprocessor to make decisions based on certain conditions indicated by the flags. After logic and arithmetic operations, flip-flops are set or reset to reflect data conditions. The conditional jump instructions check the flag conditions and make decisions to change or lot to change the sequence of a program.

The four flags used by the Jump instructions are:

1.Carry flag

2.Zero flag

3.Sign flag

4.Parity flag

Two Jump instructions are associated with each flag. The sequence of a program can be changed either because the condition is present or because the condition is absent.

INSTRUCTIONS

All conditional Jump instructions in the 8085 are 3-byte instructions; the second byte specifies the low-order memory address and the third byte specifies the high-order memory address. The

following instructions transfer the program sequence to the memory location specified under the given conditions:

Opcode	Operand	Description
JC	16-bit	Jump on Carry (if result generates carry and CY = 1)
JNC	16-bit	Jump on No Carry (CY = 0)
JZ	16-bit	Jump on Zero (if result is Zero and Z = 1)
JNZ	16-bit	Jump on No Zero (Z = 0)
JP	16-bit	Jump on Plus (if D_7 = 0 and S = 0)
JM	16-bit	Jump on Minus (if D_7 = 1 and S =1)
JPE	16-bit	Jump on Even Parity (P = 1)
JPO	16-bit	Jump on odd Parity (P = 0)

5.MACHINE CONTROL OPERATIONS

These instructions control machine functions such as Halt, Interrupt or do nothing.

1. EI : Enable Interrupt.
2. DI : Disable Interrupt.
3. NOP : No Operation.
4. HLT : Halts the processor.
5. SIM : Set interrupt Mask
6. RIM : Read interrupt Mask

One can transfer data to or from the SID or SOD lines using the instruction RIM and SIM. After executing the RIM instruction, the bits in the accumulator are interpreted as follows:

1. Serial input bit is bit 7 of the accumulator.
2. Bits 0 to 6 are interrupt masks, the interrupt enable bit and pending interrupts.

The SIM instruction sends the contents of the accumulator to the interrupt mask register and serial input line. Therefore, before executing the SIM, the accumulator must be loaded with proper data. The contents of the accumulator are interpreted as follows:

1. Bit 7 of the accumulator is the serial output bit.
2. The SOD enable bit is bit 6 of the accumulator. This bit must be 1 in order to output bit 7 of the accumulator to the SOD line.
3. Bits 0 to 5 are interrupt masks, enables and resets.

7 (i).Explain the multiple interrupts and priorities of 8085 microprocessor

(ii)Explain the Vectored interrupts of 8085 microprocessorNOV04

INTERRUPT METHOD: It provides an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is completed, the processor would resume exactly where it left off.

- Interrupt is signals send by an external device to the processor, to request the processor to perform a particular task or work.
- Mainly in the microprocessor based system the interrupts are used for data transfer between the peripheral and the microprocessor.
- The processor will check the interrupts always at the 2^{nd} T-state of last machine cycle.
- If there is any interrupt it accept the interrupt and send the INTA (active low) signal to the peripheral.
- The vectored address of particular interrupt is stored in program counter.
- The processor executes an interrupt service routine (ISR) addressed in program counter.
- It returned to main program by RET instruction.

Vectored Interrupt: When interrupt signal is activated, the internal control circuit of the microprocessor produces a CALL to a predetermined memory location. This memory location, where the subroutine starts is referred to as vector location and such interrupts are called vectored interrupts.

TRAP, RST 7.5, RST 6.5, RST 5.5, RST 0 to 7

Non vectored interrupt: INTR

Maskable Interrupt: In the microprocessor those interrupts which can be enabled and disabled under program control is called maskable interrupts.

INTR, RST 7.5, RST 6.5, RST 5.5, RST 0 to 7

Non Maskable Interrupt : TRAP

Hardware Interrupt: In the microprocessor, those pins which allow peripheral device to interrupt the main program for I/O operations is called Hardware interrupt.

When an interrupt occurs, the 8085 completes the instruction it is currently executing and transfers the program control to a subroutine that services the peripheral device. Upon completion of the service routine, the CPU return to the main program.

TRAP, RST 7.5, RST 6.5, RST 5.5 , INTR

Types of Interrupts:

It supports two types of interrupts.

1. Hardware
2. Software

HARDWARE INTERRUPT: An external device initiates the hardware interrupts and placing an appropriate signal at the interrupt pin of the processor.

- If the interrupt is accepted then the processor executes an interrupt service routine.
- The 80S5 has five hardware interrupts

(1) TRAP (2) RST 7.5 (3) RST 6.5 (4) RST 5.5 (5) INTR

TRAP: This interrupt is a nonmaskable interrupt. It is unaffected by any mask or interrupt enable. TRAP has the highest priority. TRAP interrupt is edge and level triggered. This means that the TRAP must go high and remain high until it is acknowledged. This avoids false triggering caused by noise and transients.

Once the TRAP is acknowledged, the 8085 completes its current instruction. It then pushes the address of the next instruction i.e. return address onto the stack and loads PC with fixed vector address 0024H. Due to this, 8085 starts execution of instructions from address 0024H which is the starting address of an interrupt service routine for TRAP.

RST 7.5 : The RST 7.5 interrupt is a maskable interrupt. It has the second highest priority. It is positive edge triggered.If the mask bit M 7.5 is 0 i.e. RST 7.5 is unmasked then 8085 completes its current instruction. It then pushes the address of the next instruction onto the stack and loads PC with fixed vector address 003CH. Due to this, 8085 starts execution of instructions from address 003CH which is the starting address of an interrupt service routine for RST 7.5.

RST 6.5 and RST 5.5 : The RST 6.5 and RST 5.5 both are level triggered. These interrupts can be masked using SIM instruction. The RST 6.5 has the third priority whereas RST 5.5 has the fourth priority. The vector addresses of RST 6.5 and RST 5.5 are 0034H and 002CH respectively. After recognition of RST 6.5 or RST 5.5 interrupt, 8085 completes its current instruction; pushes the address of next instruction onto the stack and loads PC with corresponding vector address.

INTR : INTR is a maskable interrupt, but not the vector interrupt. It is also called hand shake interrupt. INTR is high level sensitive. It has the lowest priority. The following sequence of events occur when INTR signal goes high.

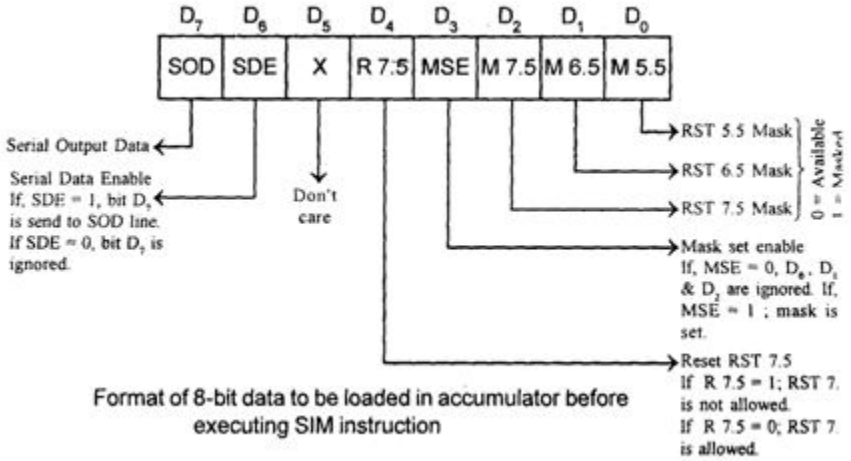
1. The 8085 checks the status of INTR signal during execution of each instruction.
2. If INTR signal is high, then 8085 completes its current instruction and sends an active low interrupt acknowledge signal (INTA) if the interrupt is enabled.
3. The 8085 then expects either a 1-byte CALL (RST 0 to RST 7) or a 3-byte CALL. This instruction must be provided by external hardware. In other words, the INTA can be used to enable a tristate buffer. The output of this buffer can be connected to the 8085 data lines. The buffer can be designed to provide the appropriate Opcode on the data lines.
4. On receiving the instruction, the 8085 saves the address of next instruction on stack and executes received instruction.

Interrupt type	Trigger	Priority	Maskable	Vector Address
TRAP	Edge And Level	1st (Highest)	No	0024H
RST 7.5	Edge	2nd	Yes	003CH
RST 6.5	Level	3rd	Yes	0034H
RST 5.5	Level	4th	Yes	002CH

INTR	Level	5th (Lowest)	Yes	-

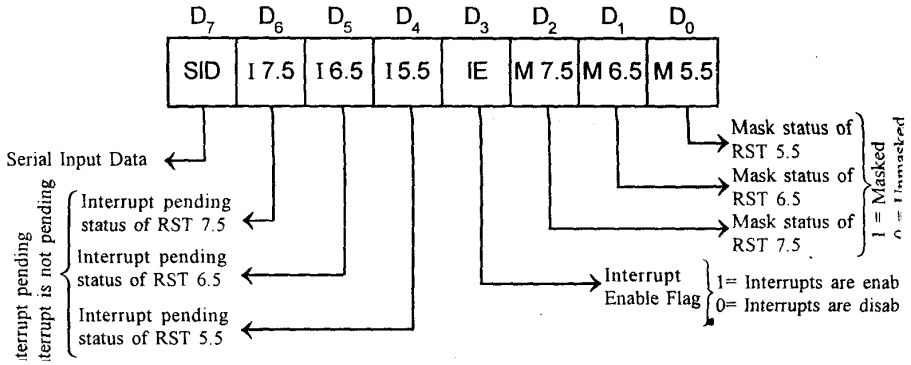
SIM and RIM for interrupts:

- The 8085 provide additional masking facility for RST 7.5, RST 6.5 and RST 5.5 using SIM instruction.
- The status of these interrupts can be read by executing RIM instruction.
- The masking or unmasking of RST 7.5, RST 6.5 and RST 5.5 interrupts can be performed by moving an 8-bit data to accumulator and then executing SIM instruction.
- The format of the 8-bit data is shown below.



Format of 8-bit data to be loaded in accumulator before executing SIM instruction

- The status of pending interrupts can be read from accumulator after executing RIM instruction.
- When RIM instruction is executed an 8-bit data is loaded in accumulator, which can be interpreted as shown in fig.



SOFTWARE INTERRUPTS: The software interrupts are program instructions. These instructions are inserted at desired locations in a program.

The 8085 has eight software interrupts from RST 0 to 7. If the external device places an Opcode for any one of the RST instruction (RST 0 to RST 7), then 8085 pushes the contents of

PC onto the stack. It then branches the program control to the vector address of the corresponding RST instruction. The vector address for these interrupts can be calculated as follows:

Interrupt number x 8 = vector address

Instruction	Vector Address
RST 0	0000H
RST 1	0008H
RST 2	0010H
RST 3	0018H
RST 4	0020H
RST 5	0028H
RST 6	0030H
RST 7	0038H

8. Explain the machine control instructions with 8085 with example.

These instructions are related to interrupts and are used to halt program execution.

1. DI : Disable interrupts

Description: The Interrupt enable flip – flop is reset and all the interrupts except the TRAP (8085) are disabled.

One byte instruction.

One machine cycle: OPcode fetch – 4 T

No flags are affected.

[This instruction is commonly used when the execution of a code sequence cannot be interrupted. For example, in critical time delays, this instruction is used at the beginning of the code and the interrupts are enabled at the end of the code. Te 8085 TRAP connect be disabled.]

2. EI : Enable Interrupts

Description:The interrupts enable flip – flop is set and all interrupts are enabled.

One byte instruction

One machine cycle: Opcode fetch – 4 T

No flags are affected

[After a system reset or the acknowledgment of an interrupt the Interrupt enable flip flop is reset, thus disabling the interrupts. This instruction is necessary to reenable the interrupts (except TRAP)]

HLT : Halt and Enter Wait State:

Description:

The MPU finished executing the current instruction and halfs any further execution. The MPU enters the Halt Acknowledge machine cycle and Wait states re inserted in every clock period. The address and the data bus are placed in the high impedance state. The contents of the registers are unaffected during the HLT state. An interrupt or reset is necessary to exit from the Halt state.

One byte instruction.

Two machine cycle: Opcode fetch – 3 T

Bus idle -2T

No flags are affected. 5 T

4. NOP: No Operation

Description:

No operation is performed. The instruction is fetched and decoded; however, no operation is executed. This is an useful instruction for producing software delay and reserve memory space for future software modifications.

One byte instruction

One machine cycle: Opcode fetch – 4T

5. RIM : Read Interrupt Mask

Description:

This is a multipurpose instruction used to read the status of interrupts 7.5, 6.5, 5.5 and to read serial data input bit. This instruction loads 8 bits in the accumulator with the interpretations as,

* Bits D0, D1, D2 provide the mask status of RST interrupts.
* If the interrupt enable bit (IE) D3 is "O", the 8085's maskable interrupts are disabled. The interrupts are enabled if this bit is 1. interrupt
* If a particular pending bit is 1, s an interrupt is being requested on the identified RST line. When this bit is 'O', no interrupt is waiting to be serviced.

6. SIM: Set Interrupt Mask

Description:

This is a multipurpose instruction and used to implement the 8085 interrupts (RST 7.5, 6.5 and 5.5) and serial data output.

The instruction interprets the accumulator content as,

* SOD: Serial Output Data: Bit D_7 of the accumulator is latched into the SOD output line and made available to a serial peripheral if bit $D_6 = 1$
* SDE : Serial Data Enable: If this bit =1, if enables the serial output. To implement serial output. This bit needs to be enabled.
* XXX = Don't care.
* R7.5 = Reset RST 7.5 : If this bit =1, RST 7.5 flip – flop is reset. T is is an additional control to reset RST 7.5.
* MSE: Mast Set Enable: If this it is high it enables the functions of bits D_2, D_1, D_0. This is the master control over all the interrupt masking bit 2. If this bit is low, bits D_2, D_1 and D_0 do not have any effect on the masks.
* M7.5 – D_2 = 0, RST 7.5 is enabled.
  = 1, RST 7.5 is masked or disabled.
* M6.5 – D_1 = 0,RST 6.5 is enabled.
  = 1, RST 6.5 is masked or disabled.
* M5.5 – D_0 = 0, RST 5.5 is enabled
  = 1, RST 5.5 is masked or disabled.

9.Explain timing diagram in details

Instruction Cycle:

The time required to execute an instruction is called instruction cycle.

Machine Cycle:

The time required to access the memory or input/output devices is called machine cycle.

T-State:

- The machine cycle and instruction cycle takes multiple clock periods.
- A portion of an operation carried out in one system clock period is called as T-state.

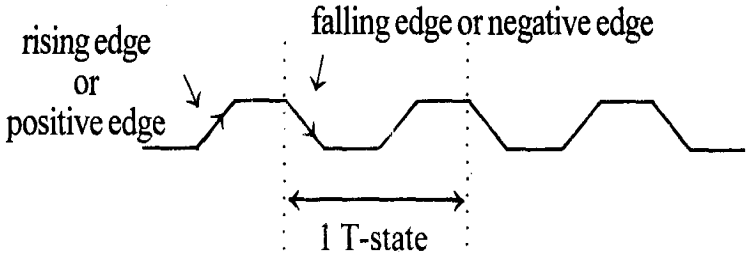
MACHINE CYCLES OF 8085:

The 8085 microprocessor has 5 (seven) basic machine cycles. They are

1. Opcode fetch cycle (4T)
2. Memory read cycle (3 T)
3. Memory write cycle (3 T)
4. I/O read cycle (3 T)
5. I/O write cycle (3 T)

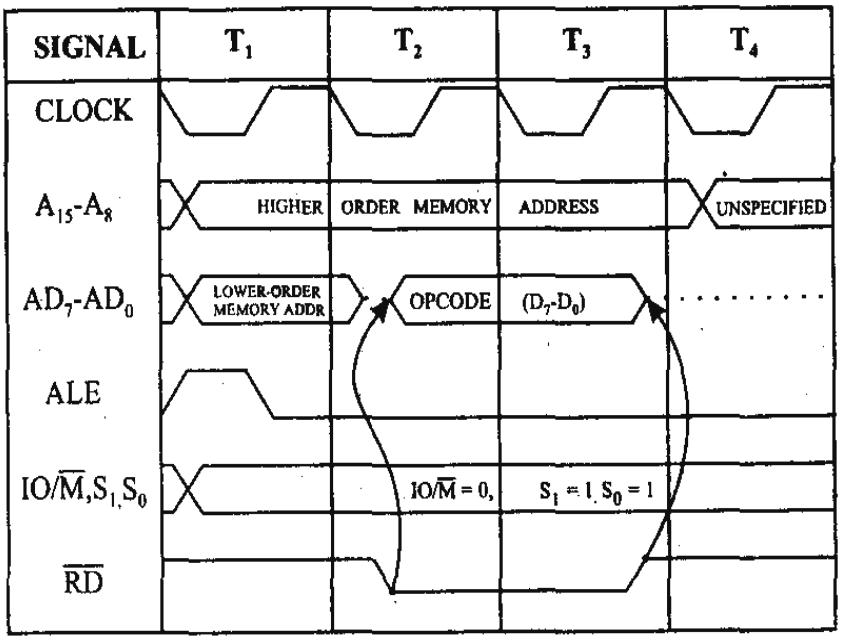
- Each instruction of the 8085 processor consists of one to five machine cycles, i.e., when the 8085 processor executes an instruction, it will execute some of the machine cycles in a specific order.
- The processor takes a definite time to execute the machine cycles. The time taken by the processor to execute a machine cycle is expressed in T-states.
- One T-state is equal to the time period of the internal clock signal of the processor.
- The T-state starts at the falling edge of a clock.

Note : Time period, T = 1/f ; where f = Internal clock frequency



Opcode fetch machine cycle of 8085 :

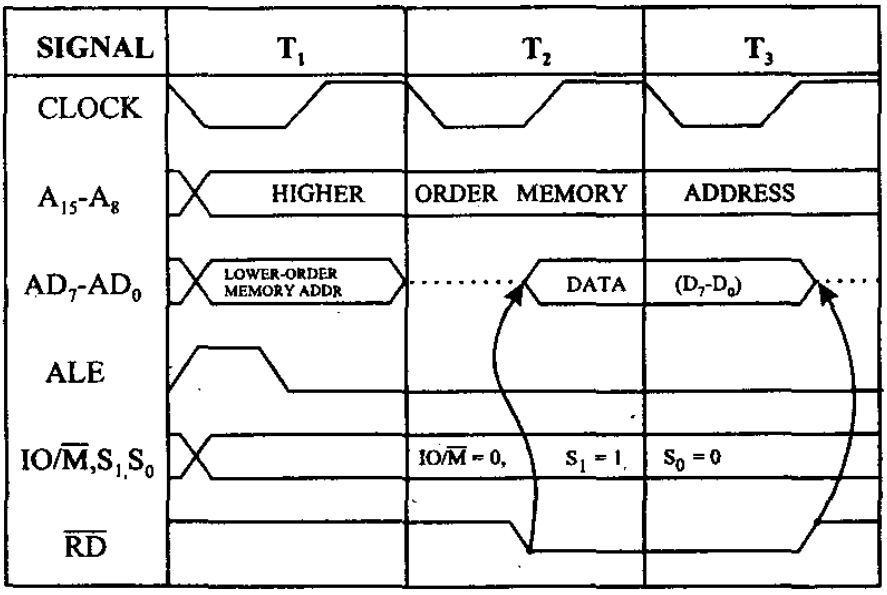
- Each instruction of the processor has one byte opcode.
- The opcodes are stored in memory. So, the processor executes the opcode fetch machine cycle to fetch the opcode from memory.
- Hence, every instruction starts with opcode fetch machine cycle.
- The time taken by the processor to execute the opcode fetch cycle is 4T.
- In this time, the first, 3 T-states are used for fetching the opcode from memory and the remaining T-states are used for internal operations by the processor.



Timing Diagram for Opcode Fetch Machine Cycle

Memory Read Machine Cycle of 8085:

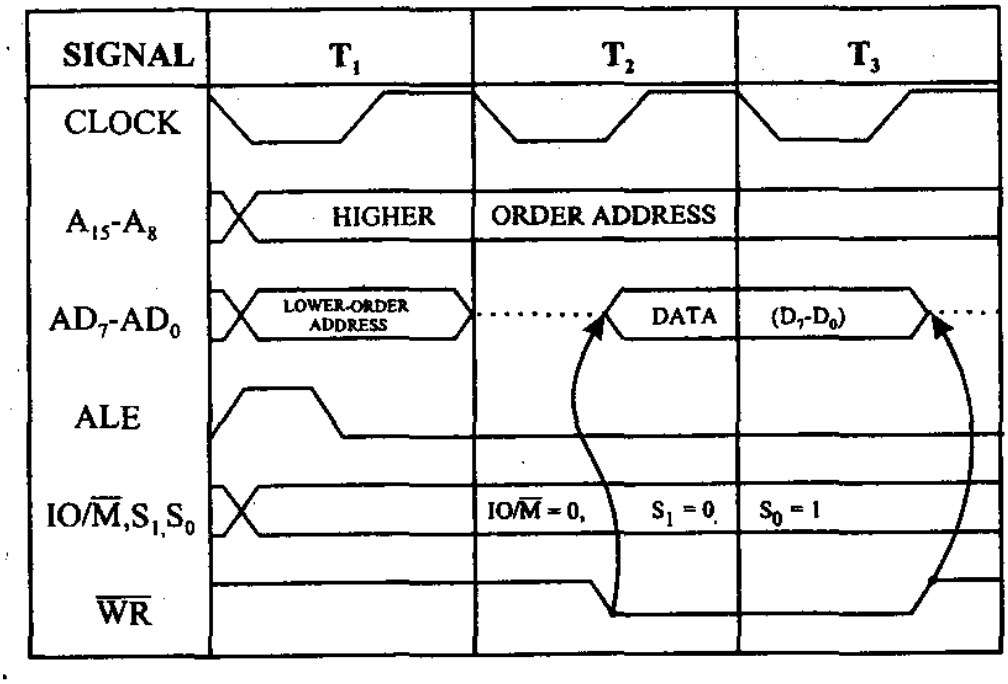
- The memory read machine cycle is executed by the processor to read a data byte from memory.
- The processor takes 3T states to execute this cycle.
- The instructions which have more than one byte word size will use the machine cycle after the opcode fetch machine cycle.



Timing Diagram for Memory Read Machine Cycle

Memory Write Machine Cycle of 8085:

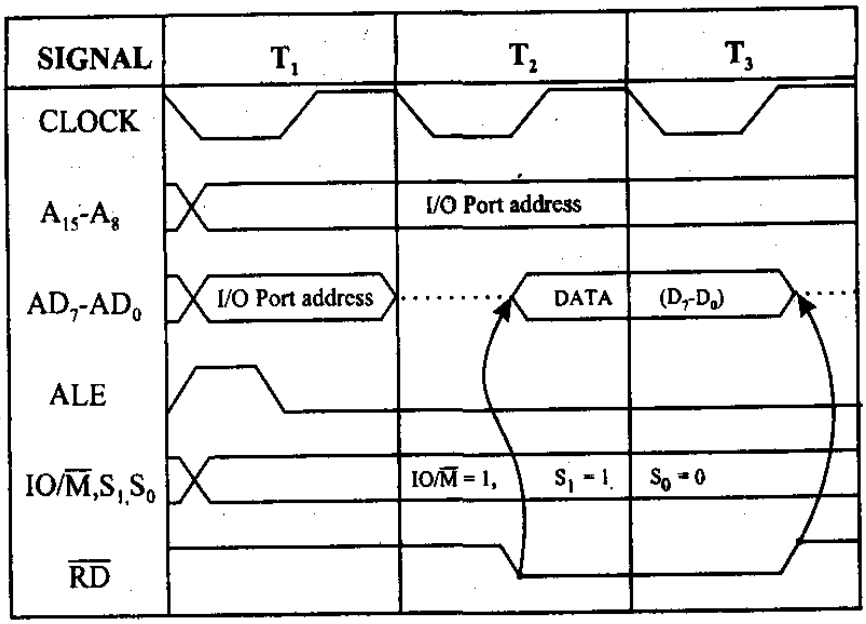
- The memory write machine cycle is executed by the processor to write a data byte in a memory location.
- The processor takes, 3T states to execute this machine cycle.



Timing Diagram For Memory Write Machine Cycle

I/O Read Cycle of 8085:

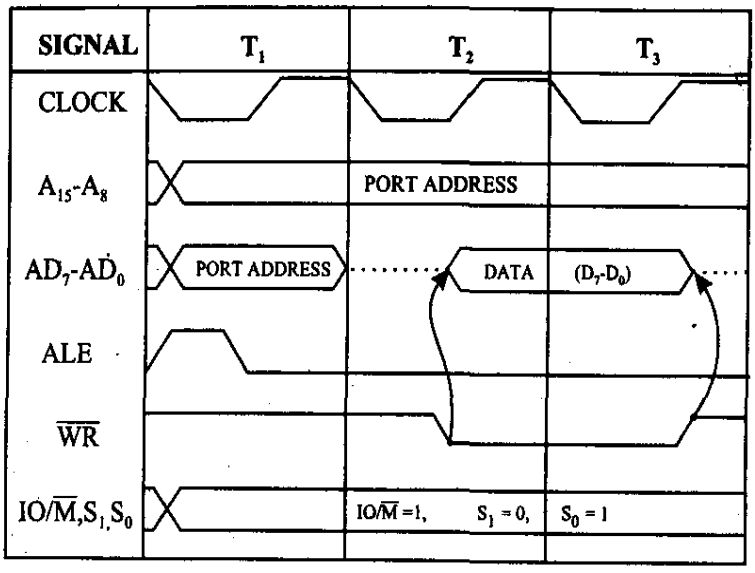
- The I/O Read cycle is executed by the processor to read a data byte from I/O port or from the peripheral, which is I/O, mapped in the system.
- The processor takes 3T states to execute this machine cycle.
- The IN instruction uses this machine cycle during the execution.



Timing Diagram for I/O Read Machine Cycle

I/O Write Cycle of 8085:

- The I/O write machine cycle is executed by the processor to write a data byte in the I/O port or to a peripheral, which is I/O, mapped in the system.
- The processor takes, 3T states to execute this machine cycle.

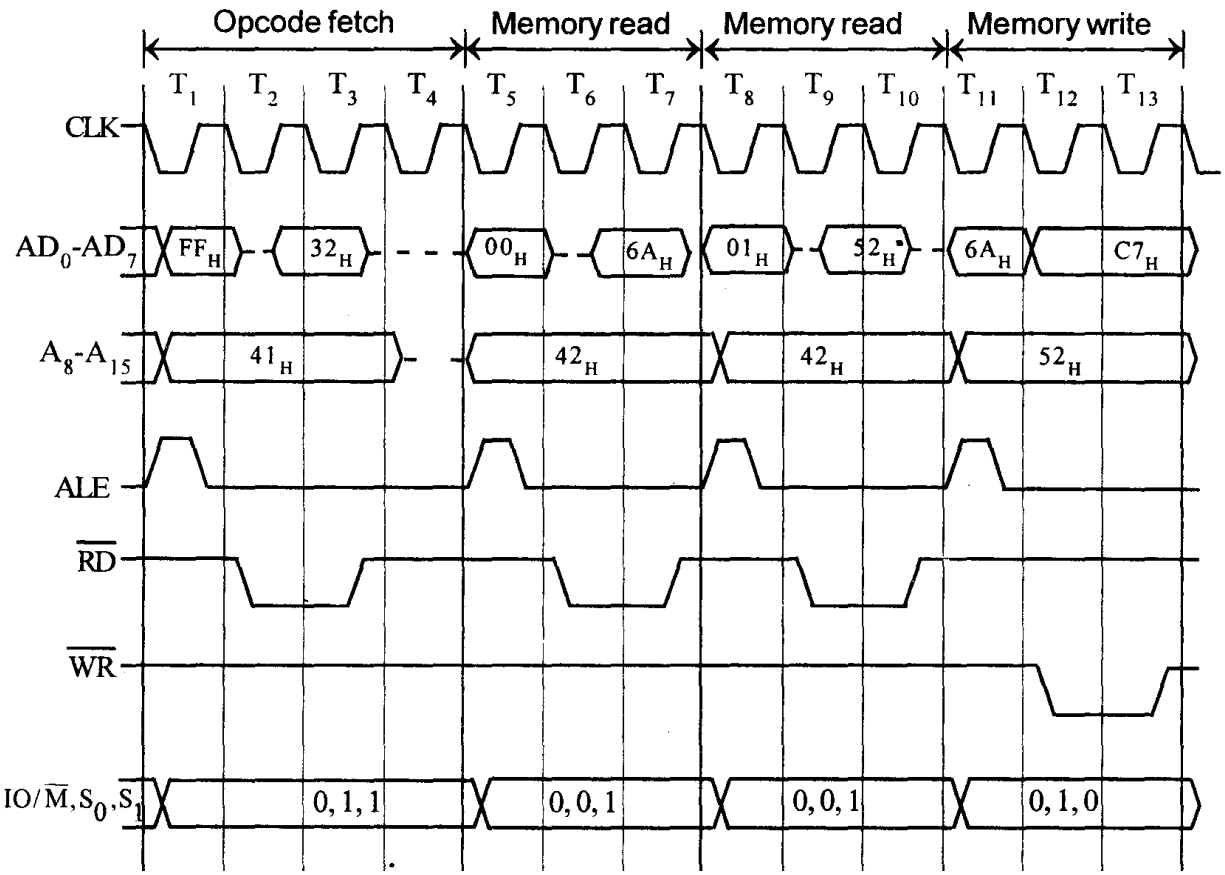


Timing Diagram for I/O Write Machine Cycle

10.Draw the Timing diagram for STA $526A_H$.

- Fetching the Opcode 32_H from the memory $41FF_H$.
- Read the lower order memory address.
- Read the higher order memory address.
- Write the accumulator content into memory location $526A_H$.
- Assume the memory address for the instruction and let the content of accumulator is $C7_H$.

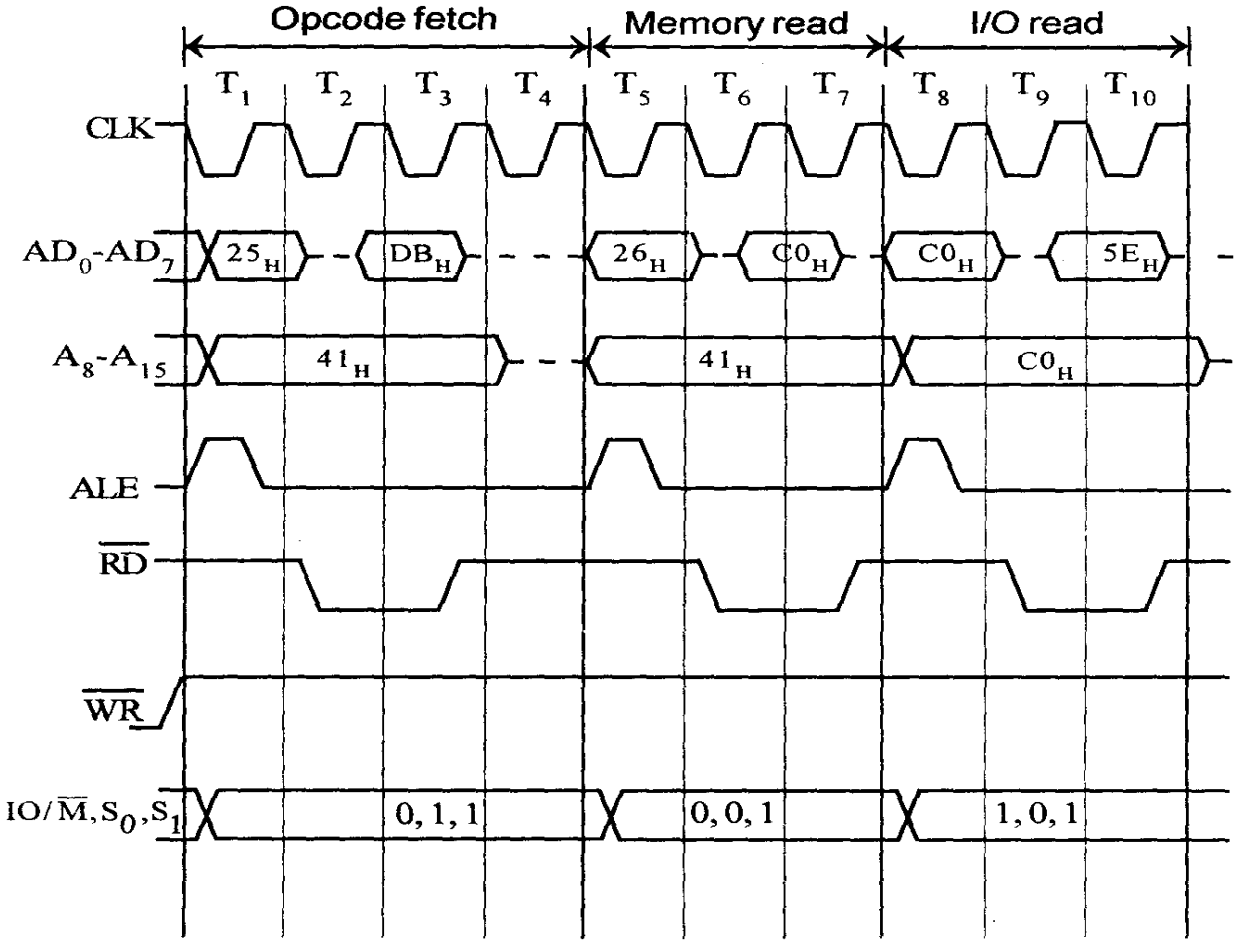
Address	Mnemonics	Opcode
41FF	STA $526A_H$	32_H
4200		$6A_H$
4201		52_H



11.Draw the Timing diagram for IN $C0_H$.

- Fetching the Opcode DB_H from the memory 4125_H.
- Read the port address $C0_H$ from 4126_H.
- Read the content of port $C0_H$ and send it to the accumulator.
- Let the content of port is $5E_H$.

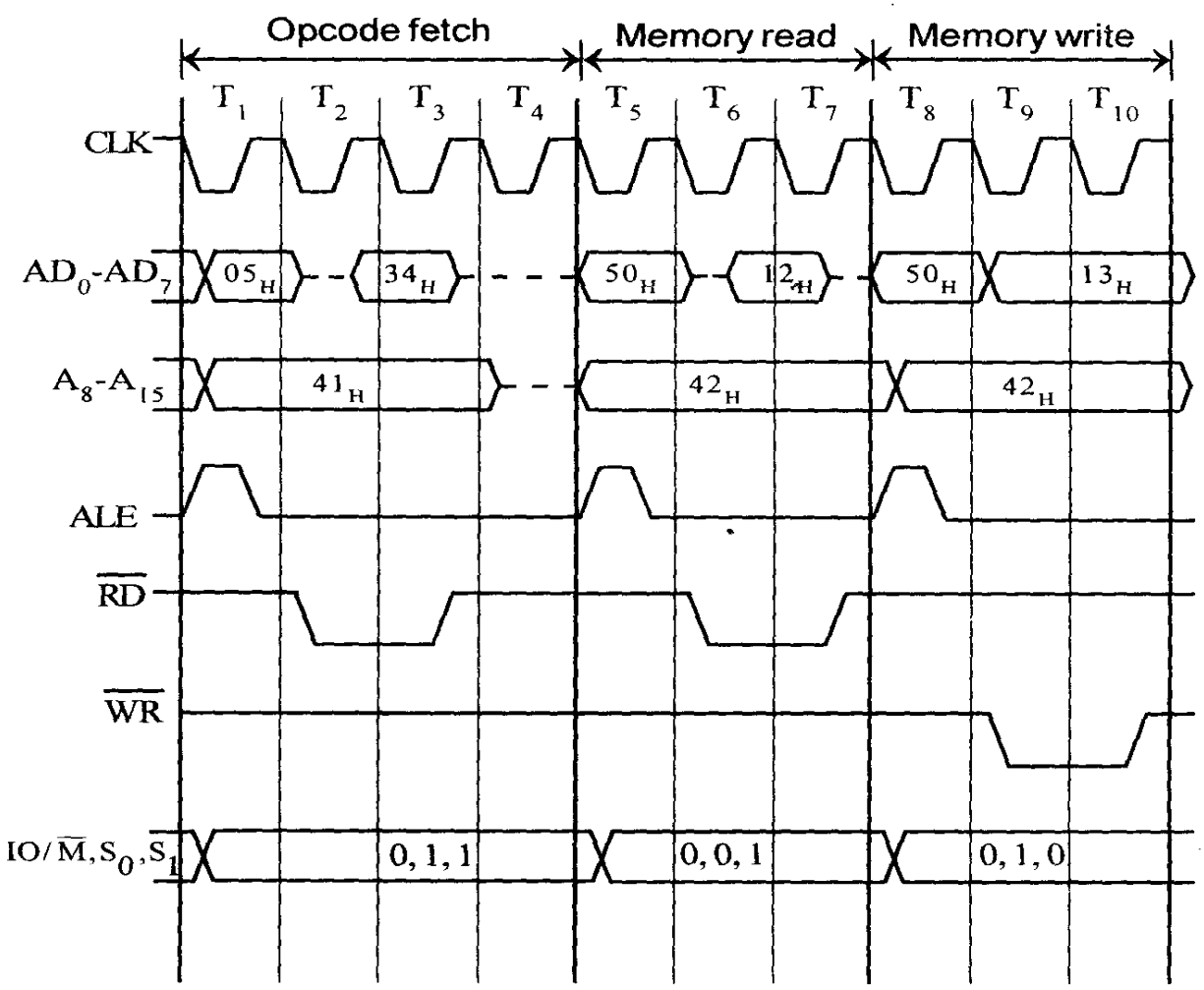
Address	Mnemonics	Opcode
4125	IN $C0_H$	DB_H
4126		$C0_H$



12. Draw the Timing diagram for INR M.

- Fetching the Opcode 34_H from the memory 4105_H.
- Let the memory address (M) is 4250_H.
- Let the content of that memory is 12_H.
- Increment the memory content from 12_H to 13_H.

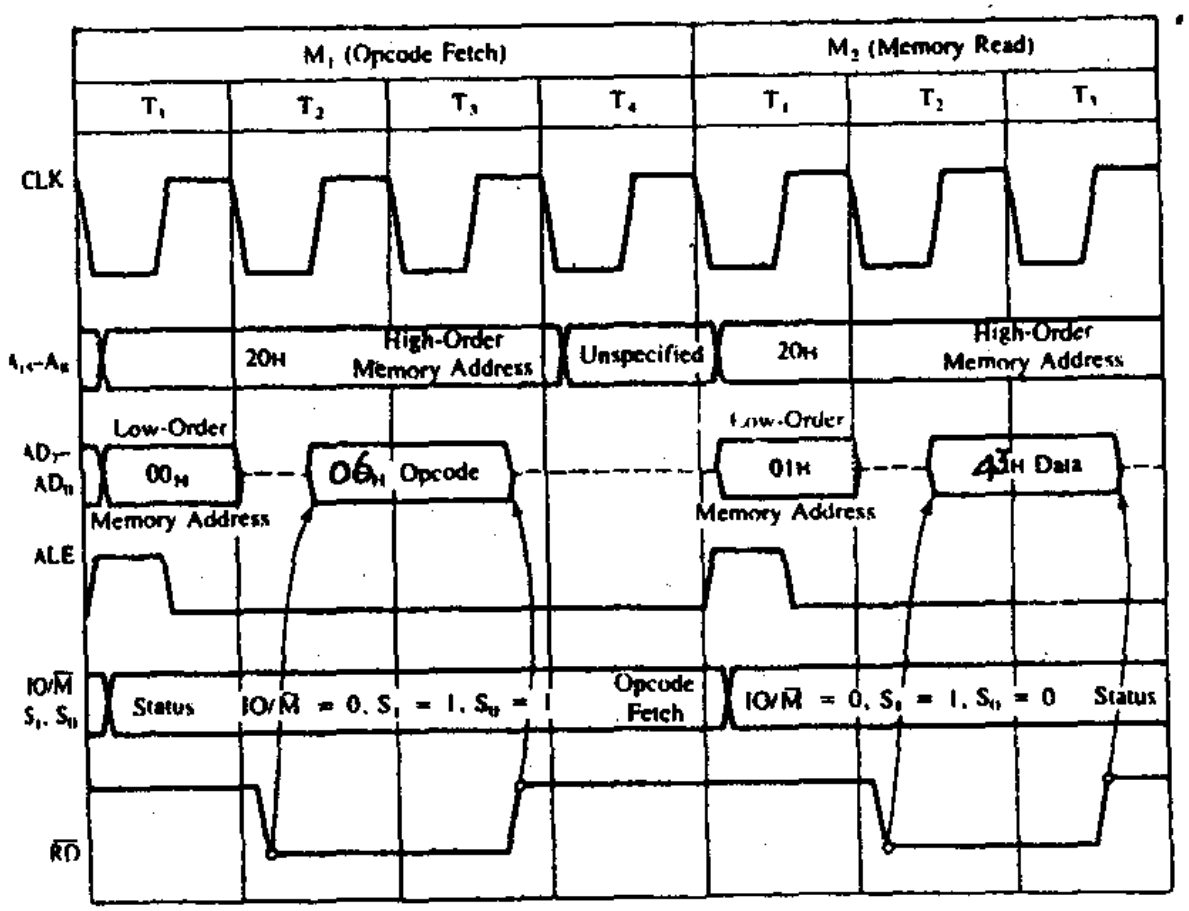
Address	Mnemonics	Opcode
4105	INR M	34_H



13. Draw the Timing diagram for MVI B, 43_H.

- Fetching the Opcode 06_H from the memory 2000_H.
- Read the data 43H from memory 2001_H.

Address	Mnemonics	Opcode
2000	MVI B, 43_H	06_H
2001		43_H



14.simple 8085 programming

1.Write a program to transfer a block of data from one location to the other.

```
5000 Start   LXI   B, 4 A01
             LXI   H, 5101
             MVI   D,05
Loop         MOV   A, M
             STAX  B
             INX   H
             INX   B
             DCR   D
             JNZ           Loop
             HLT
```

2.Write an assembly language program to add or subtract two 8 bit umbers.

8 bit addition:

```
 4100 Start  XRA   A,
             MOV   C, A
             LXI   H,4200
             MOV   A,M
             INX   H
             ADD   M
             JNC   LOOP
             INR   C
Loop         STA   4300
             MOV   A,C
             STA   4301
             MLT
```

8 bit subtraction:

```
4400 Start   XRA   A
             MOV   C,A
             LXIH,  4800
             MOV   A,M
             INX   H
             SUB   M
             JNC loop
             INR   C
             CMA
             ADI   01
Loop         STA   4500
             MOV   A,C
             STA   4501
             HLT
```

3.Write an assembly language program to multiply and divide two 8 – bit numbers.

8 – bit multiplication:

```
Start       LXI   H, 5500
            MVI   C,00
            MOV   B,M
            INX   H
            MVI,  A,00
Loop 1      ADD   M
            JNC   Loop 2
            INR   C
Loop 2      DCR   B
            JNZ   Loop 1
            STA   5600
            MOV   A, C
            STA   5601
            HLT
```

8 – bit division

```
5700  Start     MVI   C, 00
                MVI   A,  Data
                MVI   B, Data
                CMP   B
                JC    Loop
                SUB   B
                INR   C
                CMP   B
      (Loop)    MOV   A, C
                STA   5751
                HLT
```

4. Write an assembly language program to convert a two digit BCD(8-bit) data to binary data.

```
      LDA   4200H
      MOV   E,A
      ANI   F0H
      RLC
      RLC
      RLC
      RLC
      MOV   B,A
      XRA   A
      MVI   C,0AH
REP:  ADD   B
      DCR   C
      JNZ   REP
      MOV   B,A
      MOV   A,E
```

```
        ANI   0FH
        ADD   B
        STA   420H
        HLT
```

5.Write an assembly language program to convert an 8-bit binary data to BCD. The binary data is stored in 4200H. Store the hundred's digit in 4251H. store the ten's and unit's digits in 4250H

```
        MVI   E,00H
        MOV   D,E
        LDA   4200H
HUND CPI   64H
        JC    TEN
        SUI   64H
        INR   D
        JMP   HUND
TEN:  CPI   0AH
        JC    UNIT
        SUI   0AH
        INR   D
        JMP   TEN
UNIT: MOV   C,A
        MOV   A,D
        RLC
        RLC
        RLC
        RLC
        ADD   C
        STA   4250H
        MOV   A,E
        STA   4251H
        HLT
```

6. Write an assembly language program to find the 7-segment LED code for a 2-digit BCD data, by using look up table. The BCD data is stored in 4200H. Store the 7-segment code in 4201H and 4202H.

PROGRAM TO FIND THE 7-SEGMENT LED CODE FOR A BCD DATA.

```
        LDA   4200H
        MOV   B,A
        ANI   0FH
        MOV   L,A
        MVI   H,50H
        MOV   A,M
        STA   4201H
        MOV   A,B
        ANI   F0H
```

```
RLC
RLC
RLC
RLC
MOV  L,A
MOV  A,M
STA   4202H
HLT
```

7.Write an assembly language program to add or subtract two 16 bit numbers.

16 bit addition:

```
LXI H, D000H
MOV  A,M
INX    H
MOV  B,M
INX    H
MOV  C,M
INX    H
MOV  D,M
ADD   C
MOV   L,A
MOV   A,B
ADD    D
MOV    H,A
SHLD   2004 H
```

16 bit subtraction:

```
LXI  H,D000H
MOV  A,M
INX    H
MOV  B,M
INX    H
MOV  C,M
INX    H
MOV  D,M
SUB  C
MOV   L,A
MOV   A,B
SUB    D
MOV    H,A
SHLD   2004H
```

8.Write a program to sort the numbers in ascending order.

```
        MVI B, 09
START:  LXI H, D000H
        MVI C, 09H
```

```
BACK:  MOV A,M
       INX  H
       CMP M
       JC   SKIP
       JZ    SKIP
       MOV D,M
       MOV M,A
       DCX   H
       MOV  M,D
       INX    H
SKIP:  DCR C
       JNZ   BACK
       DCR B
       JNZ   START
       HLT
```

9.Write a program to sort the numbers in descending order.

```
       MVI B, 09
START: LXI H, D000H
       MVI C, 09H
BACK:  MOV A,M
       INX  H
       CMP M
       JNC   SKIP
       JZ    SKIP
       MOV D,M
       MOV M,A
       DCX   H
       MOV  M,D
       INX    H
SKIP:  DCR C
       JNZ   BACK
       DCR B
       JNZ   START
       HLT
```