Qmm

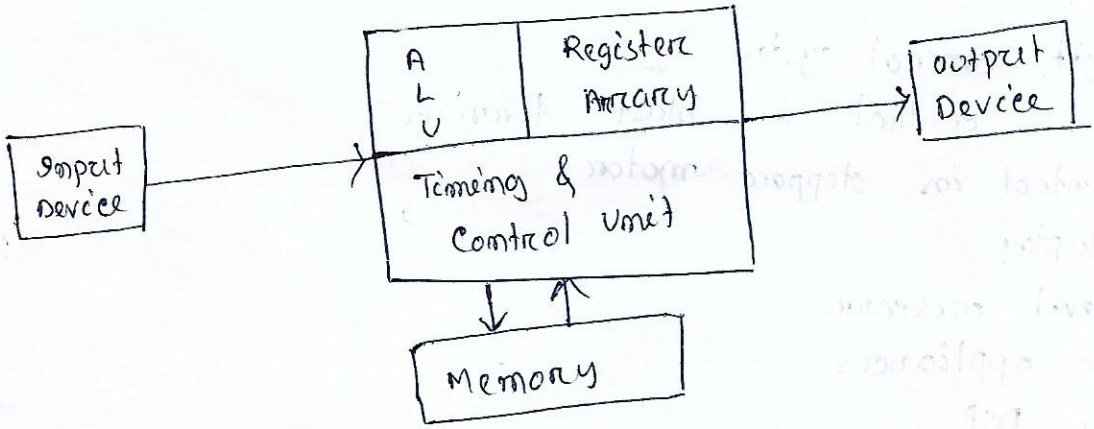
Unit - 01

INTRODUCTION TO MICROPROCESSOR

- **Microprocessor (MP):**
  - → A microprocessor is a multi purpose programmable clock driven register based electronic device, that reads the binary instruction from memory, except the binary data as input and process the data as per the instruction & provide the result as output.

    or

  - → It is a semiconductor device which consists of logic gates manufactured by using LSI or VLSI technology capable of doing arithmetic or logical operation.
  - → Human Brain is the example of processor in this eyes, ears behave as input and hands, legs behaves as outputs.
- **Block Diagram of Micro Computer system (MPU):**

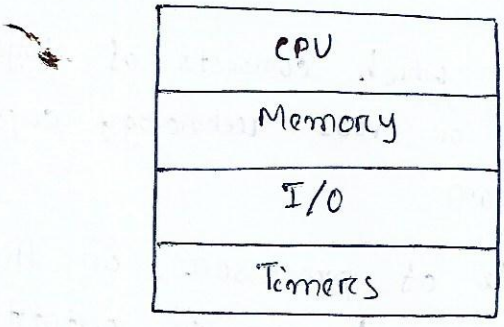


- **Components of MPU:**
  - → CPU (MP)
  - → Input (Keyboard or switch)
  - → Output (Monitor / LED)
  - → Memory

- **Description of MPU :**

→ Microprocessor or CPU consist of ALU to perform arithmatic and logic operations.
→ Register Array for storing the data temporarily.
→ Timing & control unit used for control the operation inside the processor by generating some control signa

- **Micro-controller :**



→ Microcontroller is an electronic device in which there is processor along with the memory inside a single chip.

- **Application :**

→ Traffic light control system
→ Temperature control in blast furnace
→ Speed control in stepper motor.
→ Rolling display
→ Water level indicator
→ Electronic appliances
→ Used in DSP
→ Used in mobile phones
→ Used in nano-technology.
→ Used in security system.
→ Used in automobiles.

• Evaluations of Microprocessor (MP):

MP	No. of bits Process word length	Address line (n)	Data line	Memory Capacity	Clock Frequency
(1971) 4004	4-bit	10-bit	4-bit	$2^n = 2^{10}$ = 1024 bytes = 1 KB	750 KHz
8085	8-bit	16-bit	8-bit	$2^{16} = 2^{10} \cdot 2^6$ = 65536 bytes = 64 KB	3 - 6 MHz
8086	16-bit	20-bit	16-bit	2^{20} = 1 MB	5 - 10 MHz
Penticum	32-bit	32-bit	32-bit	$2^{32} = 2^{30} \cdot 2^2$ = 4 GB	60 - 200 MHz
Penticum 4 2000	64-bit	36-bit	64-bit	$2^{36} = 2^{30} \cdot 2^6$ = 64 GB	1.3 - 3.2 GHz

Q. How many address lines are requires for 4KB Memory ?

⇒ $4KB = 2^{10} \cdot 2^2$

$= 2^{12}$

n = 12-bit = Address line.

$2^{20} = 1MB$

$3MB = 3 \times 2^{20}$

$= 2^2 \times 2^{20}$

$= 2^{22}$

$2^0 \cdot 2^1$

$2^3 \times 2^{32}$

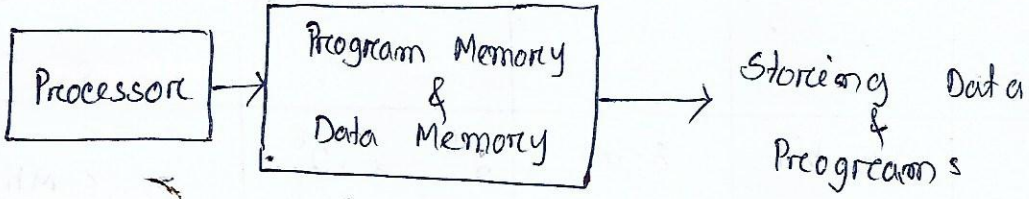
$= 2^{34}$ = 6GB / 8GB

$2^{30} \times 3 = (\quad)$ bytes

• Basic architecture for µp :

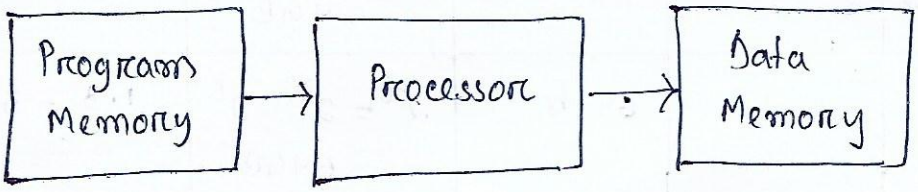
→ Von neemann architecture
→ Harvared architecture.

• Von-neemann :



In Von-neemann architecture to complete one instructio two clock cycle is requried. First clock cycle get the instruction from memory and decode it. Second cloc cycle the data taken from the memory.

• Harvard :



Two separate memories one for program and one fo data and the processor link with both the memories single clock cycle.

Architecture of 8085 :

• Features of 8085 :

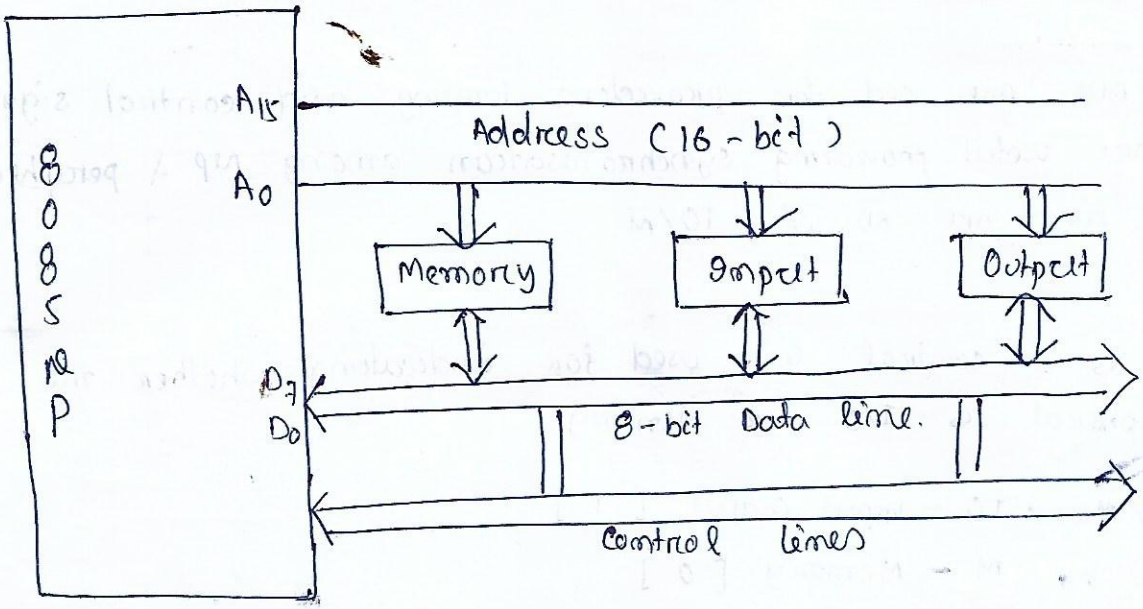
It was developed by n-MOS technology.
→ It is a 8-bit processor.
→ Operating frequency. is 3-5 MHz.
→ It requires 5V DC supply.
→ It has 16 address bus, hence the memory size is 2^{16} i. 64 KB.
→ It has 80 different instructions.
→ It has 246 opcode / machine code / Hex-code.
opcode - Operational code
→ Here all the datas are in hexa-decimal form

• Bus organisations of 8085 µp :

Bus – Inter connecting lines

→ Bus is known as set of interconnecting lines between µp and peripherals. This lines helps the µp to communicates with the peripheral that is I/O or memory.

• Block Diagram of Bus organisation :



• Basic steps for Communication :

→ Identify the peripheral (I/O or Memory)

→ Transfer the data.

→ Provide timing for control.

• Different Bus lines of µp :

→ Address Bus :

→ Address Bus are used to identify the peripheral that is I/O or Memory.

→ It is require for selecting the address of memory or I/O connected to the microprocessor.

→ Communication is µp → peripheral.

Connection of address is unidirectional.

'n' of address line = 2^n memory location

There are 16 address line ($A_0 - A_{15}$) for 8085 µp. It can able to address 2^{16} = 64KB of memory locations.

→ Data BUS :

→ Data BUS is used for transfering binary information (data).

→ These are bidirectional.

→ In 8085 MP, there are set of data line provided in MP chip for transfering the data to and from the processor. 8-bit ($D_0 - D_7$) data line.

→ A 8-bit MP can handle 8-bit data lines i.e. known as word length of MP.

→ Control BUS :

→ Control BUS are used for providing timing and control signals. These are useful providing synchronisation among MP & peripheral.

→ Control lines are $\overline{RD}$, $\overline{WR}$, IO/$\overline{M}$

• IO/$\overline{M}$:

→ This is a control line used for indicating whether the peripheral is I/O or Memory.

→ IO/$\overline{M}$ → IO - Input output [1]
M - Memory [0]

• Reset :

→ When the signal activates, the MP initialize to initial location i.e. 0000.

• Interrupt :

→ Interrupt is the set of control lines through which the external devices command to the MP to suspend the present program.

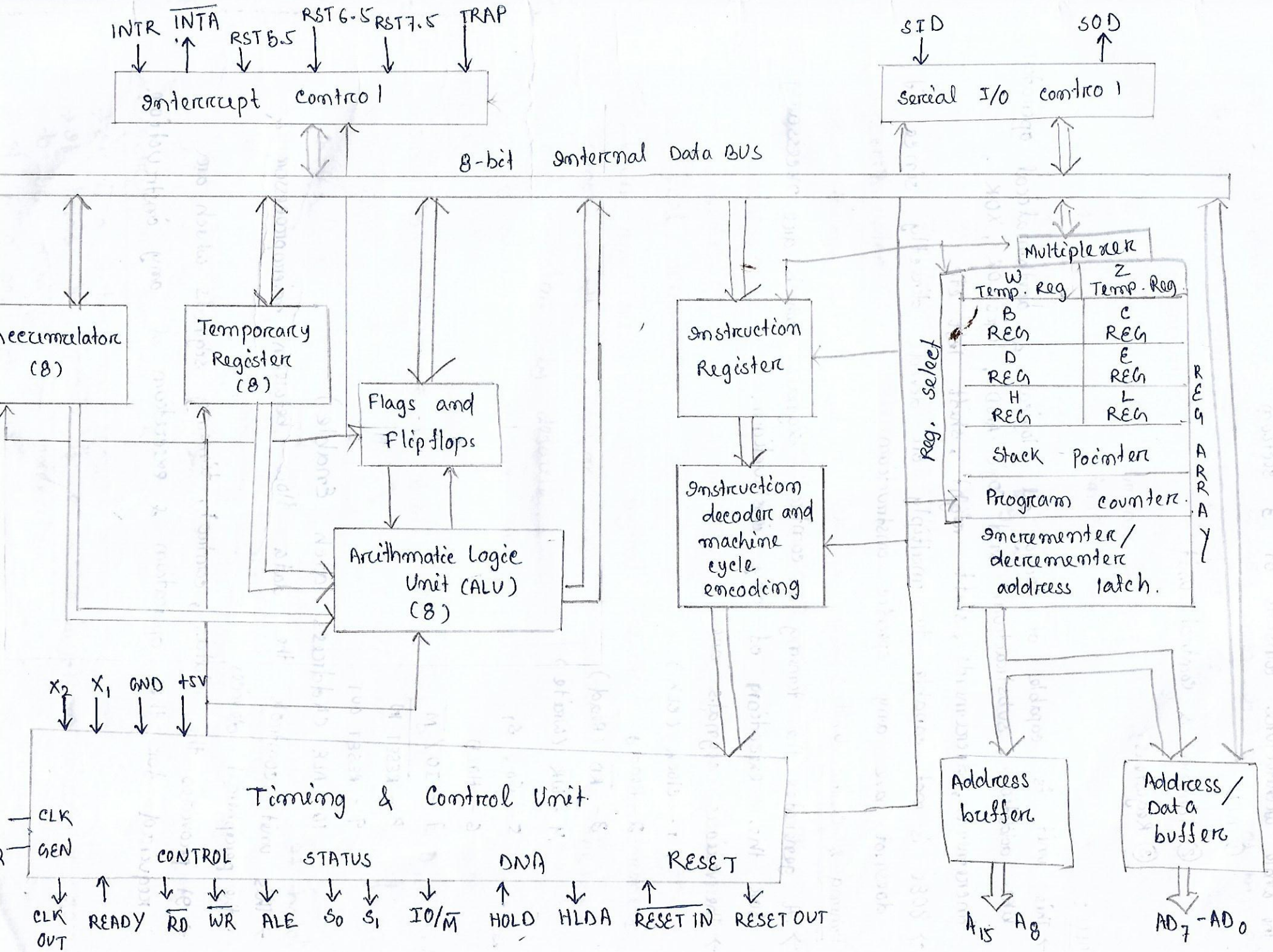
→ Then execution jump to another program and after that it again returns to main program.

2 marks

• Basic Operations by the Control lines between MP & peripheral ?

① Memory Read - Read the data from memory.
② Memory Write - Write / store the data into memory.
③ I/O Read - Read the data from I/O.
④ I/O Write - Write the data in I/O.
⑤ Fetch - Fetching of an instruction.

Architecture of 8085 µP :



→ The whole architecture consists of 3 section

① ALU

② Timing & Control Unit

③ Registers.

① ALU :

→ This unit is capable of doing all logical & arithmatical operation like addition, substraction and logic AND, logic OR, XOR, increements, decrements, shift right, shift left etc.

→ 8085 is not capable to multiply or divide directly since it doesnot have any specific instruction.

② Timing & Control Unit :

→ It generates the timing & control signals which are necessary for the execution of any instruction.

→ The various signals are.

1 - clock (CLK)

2 - Ready

3 - $\overline{RD}$ (Read)

4 - $\overline{WR}$ (Write)

5 - S_0, S_1

6 - HOLD

7 - IO / $\overline{M}$

8 - $\overline{\text{RESET IN}}$

9 - RESET OUT

10 - ALE (Address latch Enable)

→ This unit controls the data flow between microprocessor & the peripheral devices.

→ It provides the states, control, timing signals which are required for the operation & execution of any instruction.

③ Registers :

→ It is a device which stores binary data. It is used by 8085 for temporary storage of registers are

1 - Accumulator
2 - GPRC General Purpose Register)
3 - 16-bit stack pointer
4 - Program counter
5 - Status Register / Flag Register
6 - Temporary Register
7 - Instruction Register.

1 - Accumulator (A - Register) :

→ It is an 8-bit register always associated to ALU.
→ It is used during execution of a program for temporary storage.
→ It holds one of the operand during any arithmatic/ logical operation.
→ when any arithmatic or logical operation is performed, the final result of the operation is stored in the accumulator by default.
→ It is always act as an I/P to ALU.

2. GPR :

→ In 8085, there are 6 8-bit GPRs. They are B, C, D, E, H, L
→ To handel 16-bit of data, the 6 GPRs are combined as B-C, D-E & H-L.
→ The H-L pair is used as memory-pointer. It holds the 16-bit address of any memory location.
→ These registers are used for general purpose as defined by the programmer.

3. Instruction Register (IR) :

→ It is a 8-bit register.
→ It holds the opcode of any instruction which is going to be decoded or executed.

4. Stack Pointer :

→ It is a 16-bit special purpose register.
→ Stack is a sequence of memory location assigned by the programmer to store / retrieve the content of accumulator, flags, PC (Program Counter), GPRS during the execution of any program.
→ It is also called extra area memory location.
→ It works on LIFO process.
→ The operation is faster compared to the normal storage of data in any memory location.
→ During execution of a program it is sometimes necessary to store the content of certain registers because that register is required for some other operation in the subsequent steps. Then the content of such registers can be stored in an assigned / predefined memory location known as stack.
→ The top of the stack is known as stack top.
→ Stack pointer is a 16-bit register which holds the address of the stack top.

5. Program Counter (PC) :

→ It is a 16-bit register.
→ It holds the address of the next instruction which is going to be executed.
→ Microprocessor fetches an instruction from memory, executes it & increment the content of PC each time.

6. Flag Register / Status Register :

→ It is a 8-bit register i.e. 8 flipflops are there.
→ Out of these 8 FFs, 5 FFs are used to show the various status of the accumulator after any arithmatic or logical operation performed.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Sign	Zero	X	AC	X	P (Parity)	X	CY (Carry)

D_0 - Carry Flag
D_2 - Parity Flag.
D_7 - Sign Flag

- Carry Flag :

The carry flag / carry bit is set to 1 indicates a carry bit is generating after any arithmatic / logical opercation. Else this bit is set to zero.

- Parity Flag :

The parity flag (P) is set to 1 if the result of any arithmatic or logical operation is having even no. of 1. Else it is set to zero.

- Auxillary Carry Flag :

This flag is set to 1 if any carry is propagating from 3rd bit (D_3) to 4th bit (D_4) during any arithmatic/ logical opercation.

ex :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	0	0	0	0	1	0	1	
+ 0	0	0	0	0	1	1	1	
0	0	0	0	1	1	0	0	

* Here no carry from D_3 to D_4, Hence AC = 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	0	0	0	1	1	0	1	
+ 0	0	0	0	0	1	1	1	
0	0	0	1	0	1	0	0	

* Here '1' carry from D_3 to D_4, Hence AC = 1

- Zero Flag :

This flag is set to 1 if the content of accumulator is 00H after any arithmatic / logical opercation is performed.

ex :

	0000	0101
+	0000	0111
	0000	1100

* Here accumulator content is not zero. Hence zero flag content is zero.

- Sign Flag :

This flag set to one if the result of any arithmatic or logical operation is negative else it is zero.

0	0	0	0	0	1	0	0

Q. Add 77H and 66H in binary and show which flags are affected.

Ans: 77 → 0 1 1 1 0 1 1 1

+ 66 → 0 1 1 0 0 1 1 0

1 1 0 1 1 1 0 1

D_7 is 1, S flag = 1

CY = 0, Z = 0, AC = 0, ~~P flag = 1~~

D_2 is 1, Parity flag = 1

7. Temporary Register :

→ This is a 8-bit register associated to ALU and it holds the data during any arithmatic or logical operation temporarily.

→ It is associated to processor not to user.

Block of Registers Organisation of 8085 MP :

Accumulator (A) 8-bit	Flag Register 8-bit
W(8)	Z(8)
B(8)	C(8)
D(8)	E(8)
H(8)	L(8)
SP(16)	
PC(16)	

Accumulator + Flag Register = PSW (Program status word)

Instructions of 8085 :

→ Instruction is a command to a processor to perform certain operations.

→ Each instruction consists of 2 parts.

- Opcode
- Operand

- **Opcode :**
  → It is the first part of an instruction which specifies the task performed by the processor.

- **Operand :**
  → The second part of an instruction is the operand. The operand may be a 8-bit data, 8-bit address, 16-bit data, 16-bit address.
  → The I/O devices have 8-bit address but memory address is 16-bit.

Types of instructions according to word size :—

→ According to the word size 8085 instructions are catagorised in 3 groups.

- 1 byte instruction.
- 2 byte instruction
- 3 byte instruction.

- **1-byte instructions :**
  → If the opcode and operand of any instructions take 1-byte of memory space, then it is called 1-byte instruction.
  → Ex : MOV A, B, [content of B is added to content of accumulator]
  opcode operand.
  Opcode ADD B
  ADD M

- **2-byte instructions :**
  → In a 2-byte instructions the first byte specifies the opcode & 2nd byte specifies the operand which may be a data or address.
  → Ex : MVI B, 78H

  Here MVI B → opcode
  78H → operand.

- **3-byte Instructions :**

→ The 1st - byte specifies the opcode & the next 2-byte specifies a 16-bit data / address

→ Ex : LDA , 4500

STA , 4600

LDA → Opcode

4500 → 00 Low byte
45 higher byte [address]

Based on Operation :

→ Basing on operation there are 5 types of instructions.

- Data transfer group
- Arithmatic & logic group
- Logical group
- Branch control group
- I/O machine control group

- **Data transfer group :**

1- `MOV R1, R2` :

$[R_2] \longrightarrow R_1$ * Register addressing mode.

Ex : MOV A, B

→ The content of R_2 goes to R_1
→ Here operation is only fetch. It is one-byte inst^n.
→ No. of T-status is 4.

2- `MOV R, M` :

Ex : MOV B, M * Indirect addressing mode.

→ Content of memory location moved to register.
→ It is ~~one~~ two-byte instruction.
→ Here operation is fetch and memory read (i.e. machine cycle is 2).
→ No. of T-status = 7 (4+3)

3- **MOV M, R** :

→ Content of register goes to the memory location which is already predefined through the LXI statement.

→ Machine cycle is 2.

→ operation is fetch & memory write. * Indirect addressing mode.

→ one-byte instruction.

→ T-status = 7

4. **MVI R, data** :

→ Move immidiatly the 8-bit data to the register

→ Ex : MVI B, 55H * Immidiate addressing mode

→ word size 2-byte.

→ Machine cycle - 2

→ Operation - Fetch, memory read.

→ No. of T-status = 4+3 = 7

5. **MVI M; data** :

→ Move immediately the 8-bit data to any memory location.

→ Machine cycle = 3

→ operation - Fetch, memory read and memory write

(Fetch → MR → MW)

→ word size = 2 byte

→ No. of T-status = 10 (4+3+3)

6. **LXI rp, address (16-bit) data** ;

→ Load the H-L pair register pair immediately with the 16 bit data.

→ Ex : LXI H 1000 H → address (H means HL pair)

LXI B 2000 H → data (B means Bc pair)

LXI D 2000 H (D means DE pair)

→ word size = 3-bytes

→ No. of machine cycles - 3

→ Operation - Fetch, memory read, memory ~~write~~ read.

→ Ex : LXI H, 4300 H

Memory	
Opcode	2000

7. **LDA, address** : * Direct addressing mode

→ Load the content of memory location directly on the accumulator.

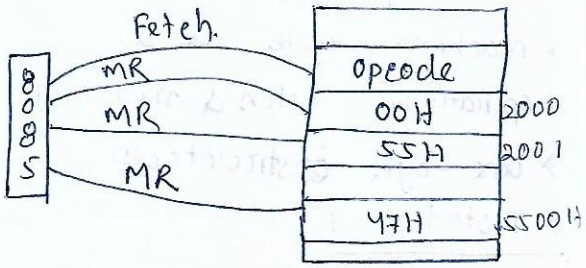
→ Ex : LDA, 5500 H

→ Operation = Fetch, MR, MR, MR

→ Machine cycle = 4

→ word size = 3 bytes

→ T-status = 13



→ First operation is Fetch. Then 2 memory read operation to read the address i.e. 5500H. Then one more memory read operation to read the data i.e. 47H present in the memory location 5500H.

8. **STA, address** :

→ Store the content of accumulator in the specified memory location given in the instruction.

→ Machine cycle = 4

→ Operation = Fetch, MR, MR, MW

→ word size = 3 byte

→ T-state = 13

9. **XCHG** :

→ Exchange the content of HL pair with DE pair.

→ word size = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-state = 4

10. **LHLD address** :

→ Load the HL-pair directly

→ Ex : LHLD 2500H

→ It indicates the content of 2500H will be stored in 'L' and the content of 2501H will be stored in 'H' by default

→ It indicates the content of memory location 2500H will go to register 'L' & the content of next memory location i.e. go to register 'H'.

→ word size = 3 bytes
→ Machine cycle = 5
→ Operation = Fetch, MR, MR, MR, MR
→ T-states = 16

11. **SHLD address**:
→ store the HL pair directly.
→ Ex: SHLD 4500H [L] → 4500 H, [H] → 4501 H]
→ It will store the content of 'L' to the 1st memory location given in the instruction itself & the content of 'H' will be stored in the next memory location.
→ word size = 3 bytes
→ Machine cycle = 5
→ Operation = Fetch, MR, MR, MW, MW.
→ T-state = 16

12. **LDAX rp**:
→ It is an indirect addressing mode.
→ It loads the content of memory location whose address is in the register pair into the accumulator.
→ Ex: [B] = 45 H
[C] = 73 H
B. LDAX BC
Then it will load the content of 4573 H into accumulator
→ word size = 1 byte
→ Machine cycle = 2
→ Operation = Fetch + MR
→ T-states = 7

13. **STAX rp**:
→ The content of accumulator whose address will go to the memory location whose address is given in the register pair.
→ Ex: STAX, DE
→ wordsize = 1 bytes
→ Machine cycle = 2
→ Operation = Fetch, MW

- Arithmatic group :

14. ADD R :
   → Add the content of any register to accumulator and store the result in accumulator.
   → Ex : ADD B
   → word size = 1 byte
   → Machine cycle = 1
   → Operation = Fetch
   → T-state = 4

15. ADD M :
   → Add the content of memory location with the accumulator and the result will be stored in accumulator.
   → word size = 1 byte
   → Machine cycle = 2
   → Operation = Fetch, MR
   → T-states = 7

16. ADC R :
   → Add the content of register with carry to the accumulator & result of the operation is stored in accumulator.
   → word size = 1 byte
   → Machine cycle = 1
   → Operation = Fetch
   → T-states = 4

17. ADC M :
   → Add the content of memory location with carry to the accu & store the result in accumulator.
   → word size = 1 byte
   → Machine cycle = 2
   → Operation = Fetch, MR
   → T-states = 7

18. ADI data : (8-bit)
   → Add immediately the data with accumulator
   → word size = 2 bytes
   → Machine cycle = 2
   → Operation = Fetch, MR
   → T-states = 7

19. **ACI data** : (8-bit)

→ Add immediately the data with carry with accumulator.
→ Word size = 2 bytes
→ Machine cycle = 2
→ Operation = Fetch, MR
→ T-states = 7

20. **DAD rp** :

→ Add the content of register pair with HL pair.
→ Machine cycle = 3
→ Word size = 1 bytes
→ Operation = Fetch, MR, MR
→ T-states = 10

21. **SUB R** :

→ Substract the content of any register from accumulator and stores the result in accumulator.
→ Ex : SUB B
→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

22. **SUB M** :

→ substract the content of memory location from the accumulator and the result will be stored in accumulator.
→ Word size = 1 byte
→ Machine cycle = 2
→ Operation = Fetch, MR
→ T-states = 7

23. **SUI data** :

→ Subtract immediately the data from accumulator.
→ Word size = 2 bytes
→ Machine cycle = 2
→ Operation = fetch, MR
→ T-states = 7

24. **SBB R** :

→ subtract the content of register with borrow from the accumulator and

25. SBB M :

→ Subtract the content of memory location with borrow from the accumulator and stores the result in accumulator.

→ Word size = 1 byte

→ Machine cycle = 2

→ Operation = Fetch, MR

→ T-states = 7

26. SBI data :

→ subtract immediately the data with borrow from accumulator.

→ Word size = 2 bytes

→ Machine cycle = 2

→ Operation = Fetch, MR

→ T-states = 7

27. INR R :

→ Increment the content of register by 1.

→ Wordsize = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

28. INR M :

→ Increment the content of memory location by 1.

→ Wordsize = 1 byte

→ Machine cycle = 3

→ Operation = Fetch, MR, MW

→ T-states = 10

29. DCR R :

→ Decrement the content of register by 1.

→ Wordsize = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

30. DCR M :

→ Decrement the content of memory location by 1.

→ Word size = 1 byte

→ Machine cycle = 3

→ Operation = Fetch, MR, MW

→ T-states = 10

31. INX Rp :

→ Increment the content of register pair by 1.
→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 6 (exceptional case)
→ Ex : INX H

H	L	→	H	L
45	00		45	01

32. DCX Rp :

→ Decrement the content of register pair by 1.
→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 6 (exceptional case)

• Logical Group :

33. ANA R :

→ The content of register will have an 'and' operation with the content of accumulator.
→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

34. ANA M :

→ The content of memory location will have an 'and' operation with accumulator.
→ Word size = 1 byte
→ Machine cycle = 2
→ Operation = Fetch, MR
→ T-states = 7

35. ANI 75H :

→ Immediate 'and' operation will be performed by the given 8-bit data with accumulator.
→ Word size = 2 bytes
→ Machine cycle = 2
→ Operation = Fetch, MR
→ T-states = 7

36. ORA R : [OR - Accumulator]

→ The contents of the accumulator are logically ORed with the conte of the register.

→ Word size = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

37. ORA M :

→ The contents of the accumulator are logically ORed with the conten of the memory.

→ Word size = 1 byte

→ Machine cycle = 2

→ Operation = Fetch, MR

→ T-states = 7

38. ORI data : [OR - Immediate with accumulator]

→ The contents of the accumulator are logically ORed with the 8- data in the operand & the result is stored in accumulator.

→ word size = 2 bytes

→ Machine cycle = 2

→ operation = Fetch, MR

→ T-states = 7

39. XRA R :

→ The contents of the register are XORed with accumulator.

→ Word size = 1

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

40. XRA M :

→ The contents of the memory are XORed with accumulator.

→ Word size = 1

→ Machine cycle = 2

→ Operation = Fetch, MR

→ T-states = 7

41. **XRI data**:

→ The 8-bit data is XORed with the content of accumulator.

→ word size = 2 bytes

→ Machine cycle = 2

→ Operation = Fetch, MR

→ T-states = 7

42. **CMA**:

→ Complement the accumulator content.

→ Ex: CMA

[A] = 75 → 8A

→ word size = 1

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

43. **CMC**:

→ Complement the carry states.

→ The carry flag will get complemented while the other flags remain constant / unchanged.

→ word size = 1

→ Machine cycle = 1

→ Operation = fetch

→ T-states = 4

44. **STC**:

→ Set the carry status flag.

→ This is set to '1' after the instruction is executed.

→ word size = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

45. **CMP R**:

→ Compare the register content with accumulator.

→ word size = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

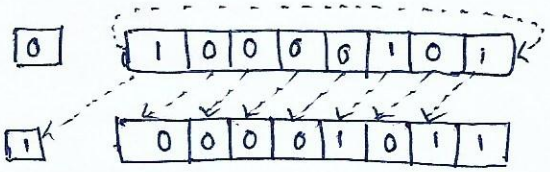
46. CMP M :

→ Compare the content of the memory location with accumulator.
→ Word size = 1 byte
→ Machine cycle = 2
→ Operation = Fetch & MR
→ T-states = 7

47. RLC :

→ Rotate accumulator left.
→ The content of accumulator is rotated left by 1-bit. After RLC instruction executed. The 7th bit of the accumulator move to carry bit as well as to the zeroth bit

→ Ex :



Ex : 1 0 0 1 1 0 0 0 1

0 0 0 1 1 0 0 0 1 0

→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

48. RRC :

→ Rotate accumulator right.
→ After this instruction executed, the 0th bit of 'A' move to the carry bit as well as to the 7th bit.
→ Ex : 0 1 0 0 0 0 1 0 1

1 1 1 0 0 0 0 1 0

→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

49. **RAL**:

→ Rotate accumulator left through carry.

→ After this instruction executed, the 7th bit of the accumulator moves to carry bit & the carry bit moves to the 0th bit.

→ Ex: 1 1 0 0 0 0 1 0 1

1 0 0 0 0 1 0 1 1

0 0 0 1 1 0 0 0 1

0 0 1 1 0 0 0 1 0

6 2

→ word size = 1 byte

→ Machine cycle = 1

→ operation = Fetch

→ T-states = 4

50. **RAR**:

→ Rotate the content of accumulator by 1-bit towards right through carry. The 0th bit will move to the carry bit & carry bit will move to the 7th bit.

1 1 0 0 0 0 1 0 1

1 1 1 0 0 0 0 1 0

e 2

→ word size = 1 byte

→ Machine cycle = 1

→ Operation = Fetch

→ T-states = 4

51. **CPI data**:

→ compare immediate with accumulator.

→ word size = 2 bytes

→ Machine cycle = 2

→ operation = Fetch, MR

→ T-states = 7

• **Branch control Group :**

→ The instruction of this group change the normal sequence of operation of the program. There are two types of branch instruction.

1. Conditional
2. Unconditional.

1. Conditional :

→ This instruction changes the normal sequence of operation when the specified condition given in the instruction is satisfied.

2. Un-conditional :

→ This instruction transfer the program to the given level unconditionally.

→ The various branch control instruction are.

Jump (JUMP / JMP)
call (CALL)
Return (RET)

① **Unconditional Jump :**

→ JMP address level → It is a 3-bytes instruction.

→ Machine cycle = 3
→ operation = Fetch, MR, MR
→ T-states = 9

→ Ex: JMP 4300H

→ The 2nd & 3rd byte of the instruction gives the address of the level where the program jumps.

② **Conditional Jump :**

→ After execution of the conditional jump instruction, the program jumps to the instruction specified in the address level. if the specified condition is satisfied

→ The conditional jump will take 3 machine cycle if the condition is true (10-T) and it takes 2 machine cycle if the condition is false (7-T).

→ 3 Machine cycle = Fetch, MR, MR
2 Machine cycle = Fetch, MR.

→ The various conditional jumps are :—

JZ address level : Jump if result is zero.
JNZ address level : Jump if result is not zero.
JC address level : Jump if result has a carry.
JNC address level : Jump if result has no carry.
JP address level : Jump if result is positive.
JM address level : Jump if result is negative (minus)

③ **Unconditional call** :

→ A subroutine always ends with a return where as the main program ends with a halt.

→ Machine cycle = 5

→ Operation = Fetch, MW, MW, MR, MR

→ T-states = 6+3+3+3+3 = 18

④ **Conditional call** :

→ If condition is true 5 MC, T = 18

→ If condition is false 2 MC, T = 9

→ The various conditional calls are

CC : call on carry

CNC : call with no carry

CZ : call on zero

CNZ : call on no zero

CP : call on positive

CM : call on minus

CPE : call on parity even

CPO : call on parity odd.

⑤ **Unconditional Return** :

→ The program sequence is transferred from the subroutine to the calling program. This instruction is used in conjunction with CALL or conditional CALL instructions.

→ word size = 1 byte

→ Machine cycles = 3

→ operation : Fetch, MR, MR

→ T-states = 10

⑥ **Conditional Return** :

→ If the condition is true, MC = 1, Operation = Fetch, T-states = 6

→ If the condition is false, MC = 3, operation = F, MR, MR, T-states = 12

→ RC : Return on carry

RNC : Return with no carry

RZ : Return on zero

RNZ : Return on no zero

RP : Return on positive

RM : Return on minus

RPE : Return on parity even

• I/O Machine Control Group :

① IN PORT address (8-bit) :

→ Ex : IN 07H

→ It indicates, the i/p to the accumulator is from a I/O port whose address is given. The data available on the port address will to the accumulator

→ word size = 2 bytes

→ Machine cycle = 3

→ Operation = Fetch, MR, I/O R

→ T-states = 10

② OUT PORT address (8-bit) :

→ It indicates data from the accumulator will move to the port addr given in instruction.

→ Ex : OUT 75 H

→ word size = 2 bytes

→ Machine cycle = 3

→ Operation = Fetch, MR, I/O W

→ T-states = 10

③ PUSH :

→ Push the content of register, accumulator, program counter to stack.

→ Ex : PUSH rp

→ The content of register pair move to stack.

→ word size = 1 byte

→ Machine cycle = 3

→ Operation = Fetch, MW, MW

→ T-states = 12

→ stack is a location present in memory but stack pointer is in processor.

④ POP :

→ Move back the contents of stack to the register or register pair which is already earlier stored in stack.

→ Wordsize = 1 byte

→ Machine cycle = 3

→ Operation = Fetch, MR, MR

⑤ **HALT** :

→ stop the execution of the processor.
→ word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

⑥ **XTHL** :

→ Exchange the stack Top data with HL pair content
→ word size = 1 byte
→ Machine cycle = 5
→ Operation = Fetch, MR, MR, MW, MW
→ T-states = 16

[SP] → L
[[SP] + 1] → H

⑦ **SPHL** :

→ Move the content of HL pair to stack pointer.
→ word size = 1 byte
→ Machine cycle = 1
→ operation = Fetch
→ T-states = 4

⑧ **EI** :

→ Enable the interrupt
→ word size = 1 byte
→ Machine cycle = 1
→ operation = Fetch
→ T-states = 4

⑨ **DI** :

→ Disable the interrupt
→ wordsize = 1 byte
→ Machine cycle = 1
→ opereation = Fetch
→ T-states = 4

⑩ **RIM** :

→ Read interrupt mask.
→ word size = 1 byte
→ Machine cycle = 1
→ operation = Fetch

(11) SIM :

→ Set interrupt mask.
→ Word size = 1 byte
→ Machine cycle = 1
→ Operation = Fetch
→ T-states = 4

8085 PIN DESCRIPTION / DIAGRAM :

→ It has 40 pin IC package.
→ The logic pin layout and signal groups of the 8085 µp are clas into six groups :

1. Address bus
2. Data bus
3. Control & status signals
4. Power supply and frequency signals
5. Externally initiated signals
6. Serial I/O signals

① Address Bus :
→ It has 16-bit address bus.
→ The pins are AD_0 to AD_7, A_8 to A_{15}.
→ Out of these 16 pins, the 8 MSB (Most significant bit) i.e. A_8 t are only for carrying address whereas the LSB (least signf. bi i.e. AD_0 to AD_7 may carry address or data.

② Data Bus :
→ It also known as multiplexed address bus
→ The data pins are AD_0 to AD_7
→ These pins are bi-directional.
→ On executing an instruction during the 1st clock cycle (i.e. T_1 st the 8-bit (LSB) as an address bit and for the remaining clk cycle it acts as a data bit.

③ Controls and status signals :
→ Under this section, there are two control signals and four status signals.
→ The control signals are $\overline{RD}$ (read) & $\overline{WR}$ (write)
→ It is an active low signal.
→ The status signals are S_0, S_1, $IO/\overline{M}$, ALE

$\overline{RD}$ - It is an active low signal.
- This signal indicates a selected I/O location or memory location will be read when this signal is low.

$\overline{WR}$ - Active low
- Processor will write on a certain memory or I/O location when this signal is low.

ALE - This bit indicates whether the bits on AD_0 to AD_7 is a data bit or an address bit
- When ALE is high (i.e. 1), the bits on AD_0 to AD_7 are address bits else it is a data bit.

$IO/\overline{M}$ - It is a status signal which distinguishes, whether the address is for an I/O location or for a memory location.
- When it goes high, that implies $IO/\overline{M} = 1$ [$I/O = 1$, $\overline{M} = 1 \Rightarrow M = 0$] then I/O will be selected.
- When it goes low, $IO/\overline{M} = 0$ [$I/O = 0$, $\overline{M} = 0 \Rightarrow M = 1$] then memory location will be selected.

S_0, S_1 - There are 2 status signals used to indicate various types of processor operation.

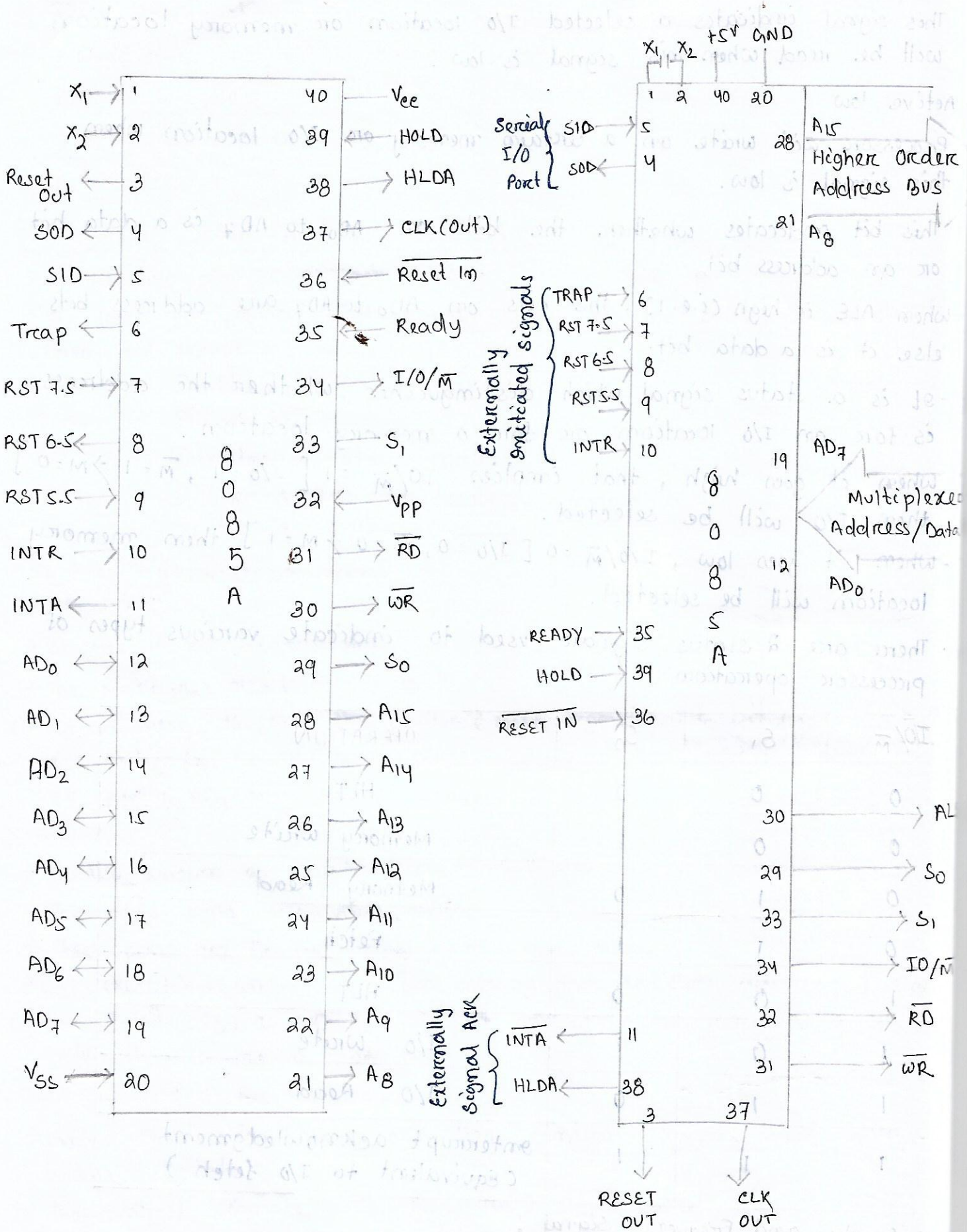
$IO/\overline{M}$	S_1	S_0	OPERATION
0	0	0	HLT
0	0	1	Memory write
0	1	0	Memory Read
0	1	1	Fetch
1	0	0	HLT
1	0	1	I/O write
1	1	0	I/O Read
1	1	1	Interrupt acknowledgment (Equivalent to I/O fetch)

④ Power Supply and Frequency Signal :

$V_{CC} = +5V$

$V_{SS} = GND$

external crystal oscillator


8085A
X1
X2
Reset Out
SOD
SID
Trap
RST 7.5
RST 6.5
RST 5.5
INTR
INTA
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7
VSS
Vcc
HOLD
HLDA
CLK(Out)
Reset In
Ready
IO/M
S1
VPP
RD
WR
S0
A15
A14
A13
A12
A11
A10
A9
A8
Serial I/O Port
SID
SOD
Externally Initiated signals
TRAP
RST 7.5
RST 6.5
RST 5.5
INTR
READY
HOLD
RESET IN
Externally Signal Acknowledge
INTA
HLDA
+5V
GND
Higher Order Address Bus
Multiplexed Address/Data
AL
RESET OUT
CLK OUT

⑤ Externally initiated signal :

INTR : Interrupt request

- It is used as a general purpose interrupt.
- It is an interrupt requests signal. When this pin goes high, the program counter doesn't increment its content.
- The MP performs the normal sequence of operation, after performing all instruction it executes a CALL instruction for executing the INTR interrupt instruction.

$\overline{\text{INTA}}$: Interrupt Acknowledgment

- It is sent by the processor after receiving the interrupt signal to the corresponding device.

RST 7.5, RST 6.5, RST 5.5 : There are maskable interrupt.

- It has priority than INTR
- The priority of RST signals are 7.5 > 6.5 > 5.5.

TRAP : Non-maskable interrupt having the higher priority.

HOLD : It indicates that a peripheral device is requesting for the use of system address bus or data bus.

- When a hold signal is received by the processor, it complete its current machine cycle and releases the system address bus and data bus.
- During this period, the internal processing (execution) of the processor will continue.

HLDA : It indicates that the hold signal is received by the processor.

- It stands for hold acknowledgement.

READY : It is used by the MP to check whether the peripheral is ready to transsent data or not.

- If the ready pin is high ⇒ Peripheral is ready to transmit or receive data.
- If it is low ⇒ then the processor has to wait till the ready pin goes high.

CLK : It is a triggering pulse. Any triggering pulse can be applied through this pin.

$\overline{\text{RESET IN}}$: It resets the program counter to zero.

- The CPU is hold in rest condition as long as reset to applied.

RESET OUT : It indicates that the CPU is in being reset condition.

⑥ Serial I/O signals :

SID : serial i/p Data

- It is a dataline for serial i/p.
- The data on these line is loaded on the 7th bit of the accumulator by getting on RIM instruction.

SOD : Serial o/p Data

- It is a dataline for serial o/p.
- The 7th bit of the accumulator will transfer to the SOD by getting SIM instruction.

VARIOUS ADDRESSING MODES :

1. Direct addressing mode
2. Register addressing mode
3. Register direct addressing mode
4. Register indirect mode
5. Immediate addressing mode
6. Implicit addressing mode.

1. Direct Addressing Mode :

→ When data is in the memory, that must have a 16-bit addr if that address is given directly in the instruction then the is called direct addressing mode.

→ The different technique used to specify the operand in the instructions is called addressing mode.

→ Ex : LDA 6000H
STA 5000 H
LHLD 6000 H

2. Register Addressing Mode :

~~→ In this mode the address of data is~~

→ In this mode the operand is specified through regis

→ Ex : MOV A, B
ADD B
ANA C
XRA L

3. Register Indirect Addressing Mode :

→ In this mode the address of data is indirectly specified.

→ Ex : LDAX B
STAX D
MOV A, M
ADD M

4. Immediate Addressing Mode:

- In immediate addressing mode the source operand is always data.
- If data is 8-bit, then the instruction will be of 2 bytes, if the data is 16-bit then the instruction will be 3 bytes
- Ex: MVI B 45

  LXI H 3050

  IMP address

5. Implied / Implicit Addressing Mode:

- In implied / implicit addressing mode the operand is hidden and the data to be operated is available in the instruction itself.
- Ex: CMA

  RAL

  RLC

  RAR

  RRC

INSTRUCTION CYCLE:

→ For executing an instruction a microprocessor fetches the instruction and executes it. The time taken for the execution of an instruction is known as instruction cycle.

→ An instruction cycle is consists of a Fetch Cycle and Execute cycle.

- The execution of any 8085 program consists of a sequence of READ and WRITE operations of which each transfer a byte of data between the 8085 and a particular I/O device address or memory address.

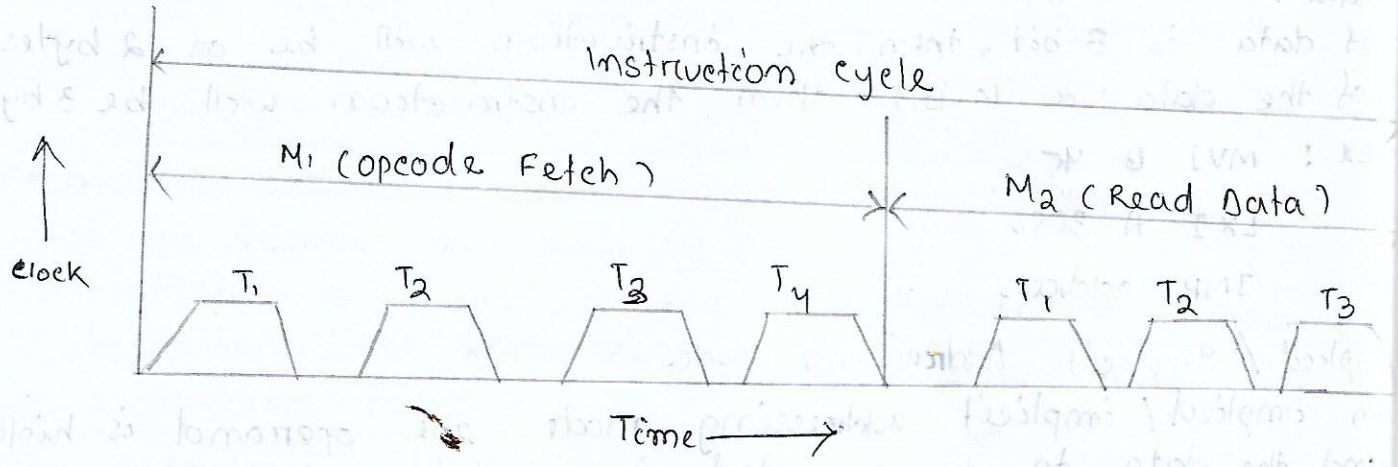
→ These READ and WRITE operations are the only communication between the processor and the other components ~~and the other components~~ and are all that is necessary to execute any instruction or program.

Machine Cycle:

→ Each READ or WRITE operations of the 8085 is referred to as Machine cycle. An 8085 instruction's execution consists of a number of machine cycles.

→ These cycles vary from one to five (M_1 to M_5) depending on the instruction. Each machine cycle contains a number of clock cycles (also referred to as T-states)

→ The machine cycles that follow will have three clock periods or T-state. The 8085 machine cycle has been shown in figure below



[Opcode Fetch and Read Machine cycle]

→ Machine cycle is the sequence of steps in performing the various operation like opcode fetch, memory read, memory writes, I/O Re I/O write etc.

T-state :

→ T-state may be defined as the time taken by the clock to complete one period. To perform a particular task by a programmer on a computer.

→ A programmer writes a set of instructions called a program and d are stored in the memory.

→ The microprocessor fetches one instruction from memory at a tim and executes it. It fetches and executes all the instructions of the program from the memory one by one to produce the final results.

Instruction Cycle :

→ Instruction is a command to CPU to perform a given operation specifies on the given data.

→ The sets of instruction given by the programmer to perform a specific task is known as a program.

→ The necessary steps that the CPU carrees out to fetch an instruction and take the necessary data from the memory and executes it constitute an instruction cycle.

→ The instruc^{n} cycle consists of a fetch cycle and an execution cycle

$$IC = FC + EC$$

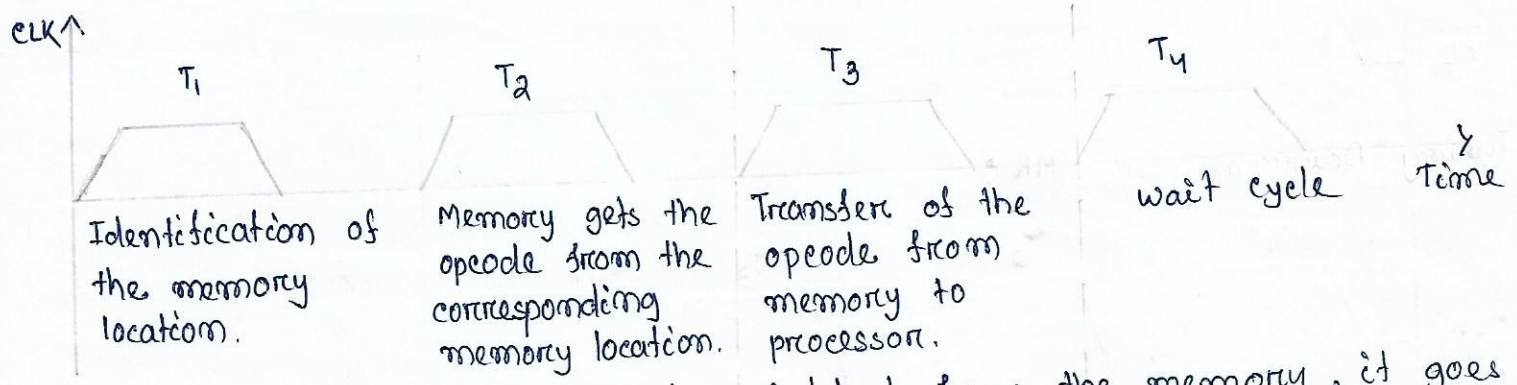
IC : Instruction Cycle.

FC : Fetch Cycle.

EC : Execution Cycle

Fetch : The necessary steps which are carried out to fetch an instruction from memory constitute a fetch cycle.

- Generally the 1st byte of any instruction is the opcode and the remaining bytes may be a 8-bit data / address or 16-bit data / address (operand).
- On the begining of the fetch cycle, the content of the program counter which is the address of the memory location where opcode is available is sent to the processor.
- The memory places the opcode on the data bus and send it to the processor.
- The processor stores this opcode in the IRs from where it is further fed for decoding and execution.
- The entire process is known as fetching i.e. reading of opcode (not data & opcode) & hexcode or machine code. (
- The fetching oprn takes 3 clk pulses or 3 T-states but one more extra T-state is available if the processor or the memory is slower by nature. This extra cycle is known as wait cycle.



Execution : After the opcode is being fetched from the memory, it goes to the IRs.

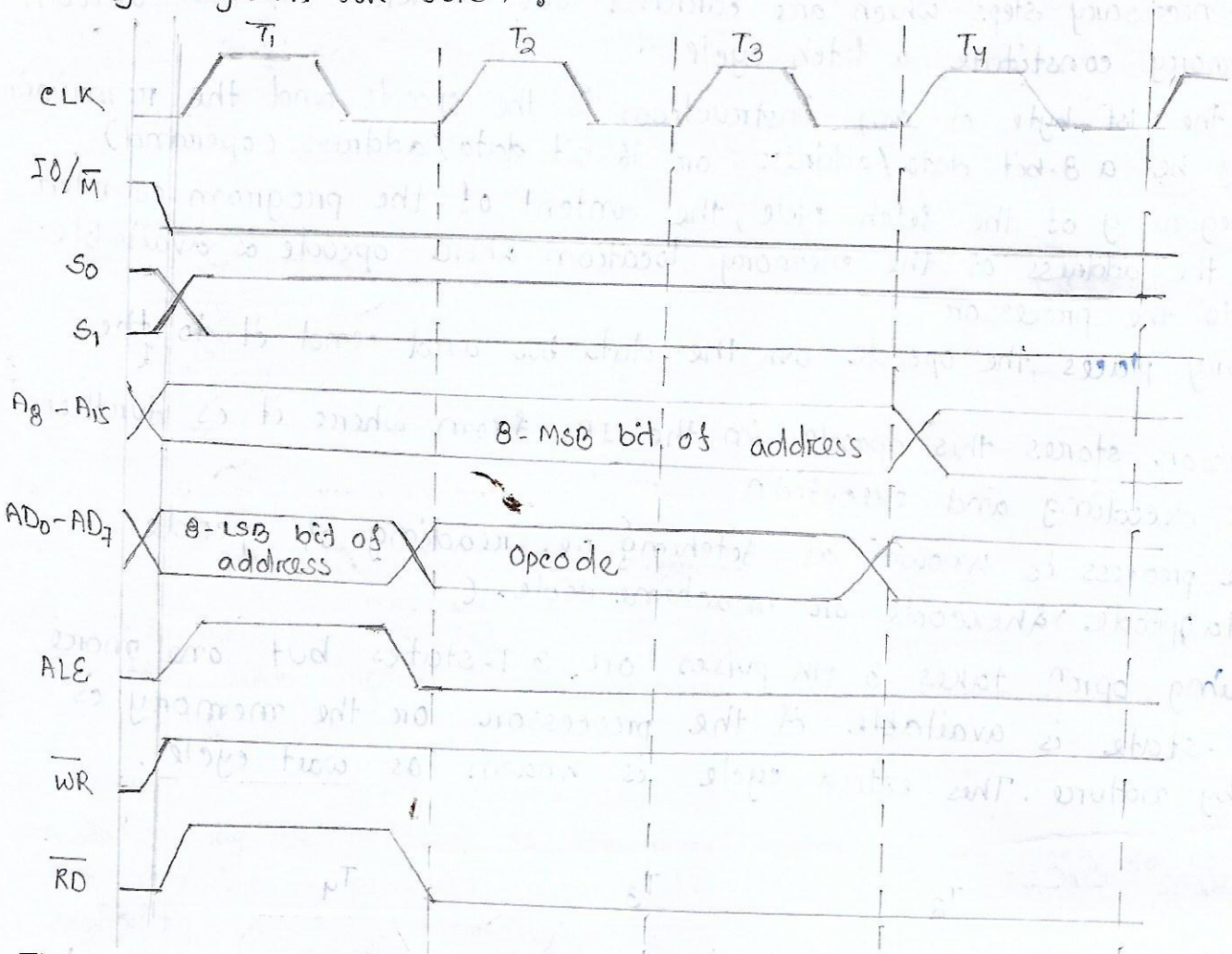
- From the IRs it goes to the decoding unit. After the instruction being decoded execution takes place.
- During the time of execution, if the operand is availabel in the GPR then the execution takes place immediately.
- The time takes to decode and execute is 1 clk cycle or 1 T-state if the operand is in GPR.
- If the operand is in certain memory / I/O loc^{n} then the processor has to perform certain operation like MR / MW / I/O R / I/O W.

Timing Diagram : It is a graphical representation of certain pins, where status get changed for various processor operation during each T-states.

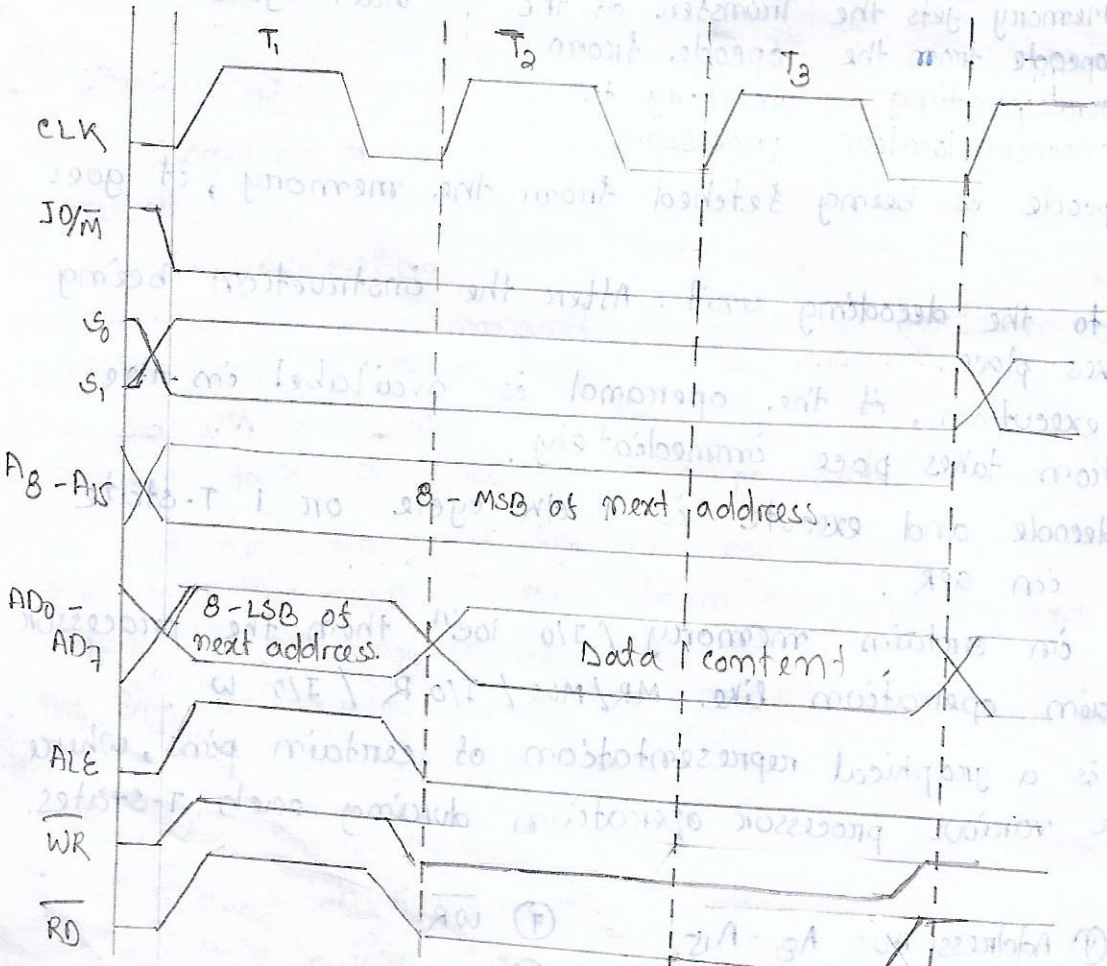
- The pins are

① IO / $\overline{M}$ ② Address Bus $A_8 - A_{15}$ ⑦ $\overline{WR}$

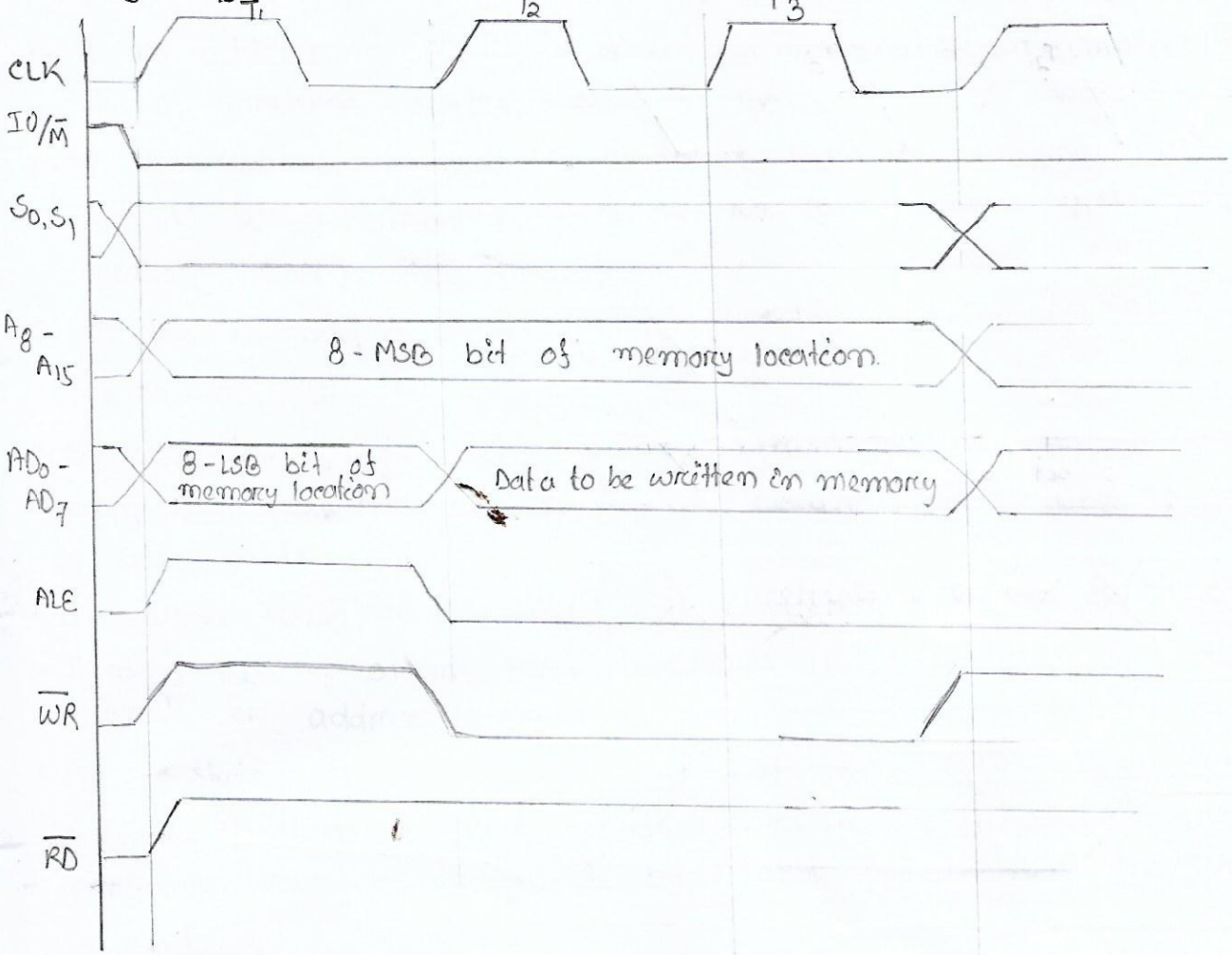
Timing Diagram for fetch :



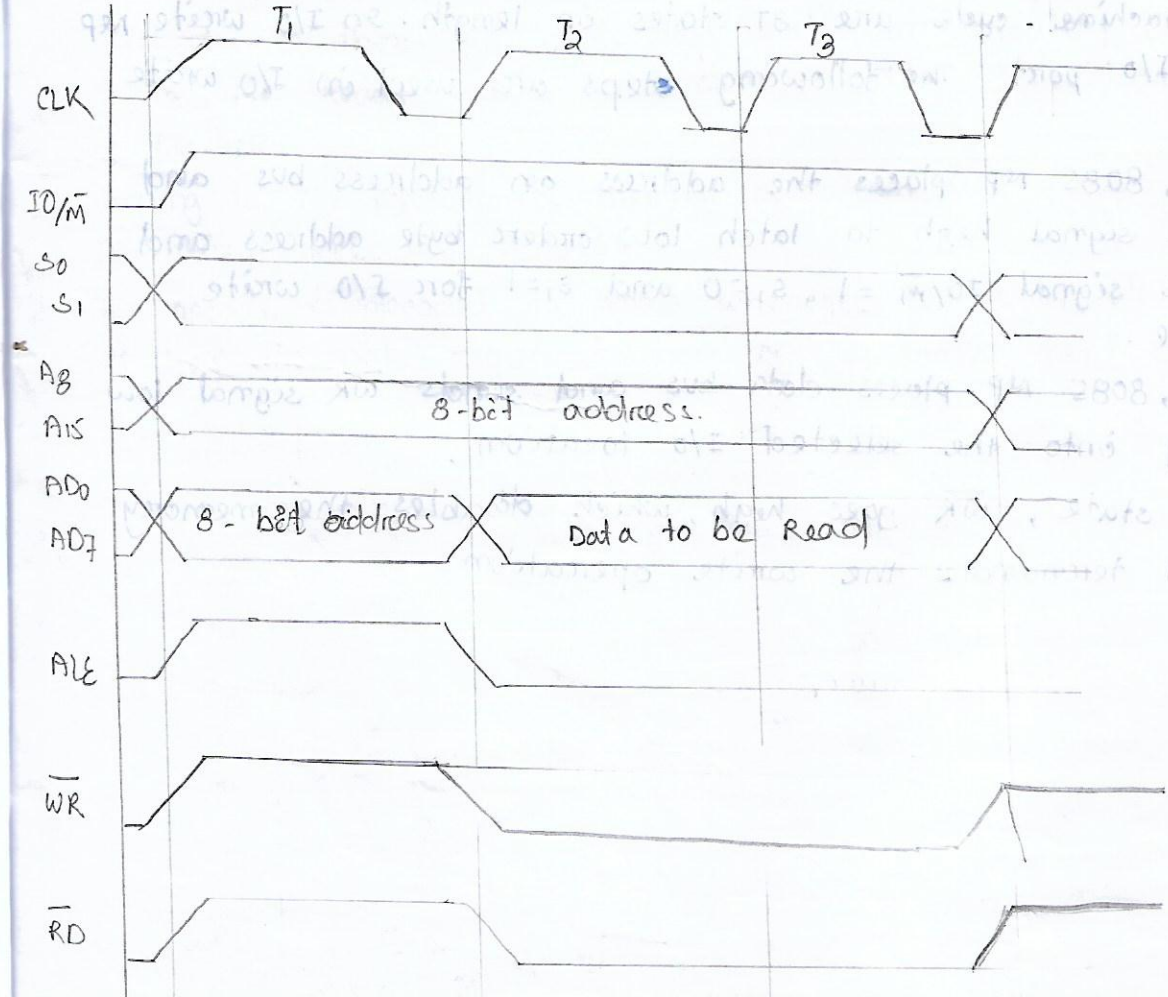
Timing Diagram for MR :



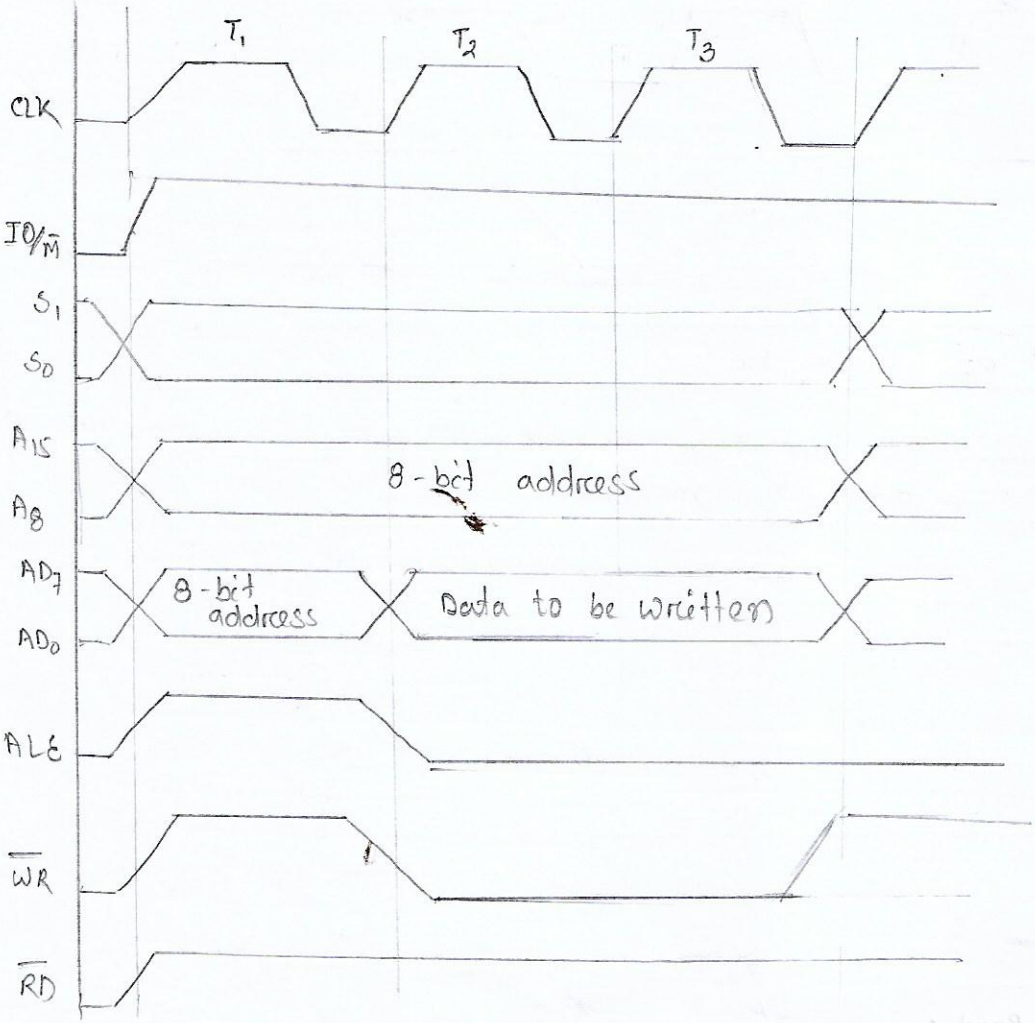
Timing Diagram for MW :



Timing Diagram for I/O Read :



Timing Diagram for I/O write :



→ The I/O write machine cycle are 3T states in length. In I/O write write data in I/O port. The following steps are used in I/O write operation.

step-1 : In T_1 state, 8085 MP places the address on address bus and activate ALE signal high to latch low order byte address and sends status signal $IO/\bar{M} = 1$, $S_1 = 0$ and $S_1 = 1$ for I/O write machine cycle.

step-2 : In T_2 state, 8085 MP places data bus and sends $\overline{WR}$ signal for writing into the selected I/O location.

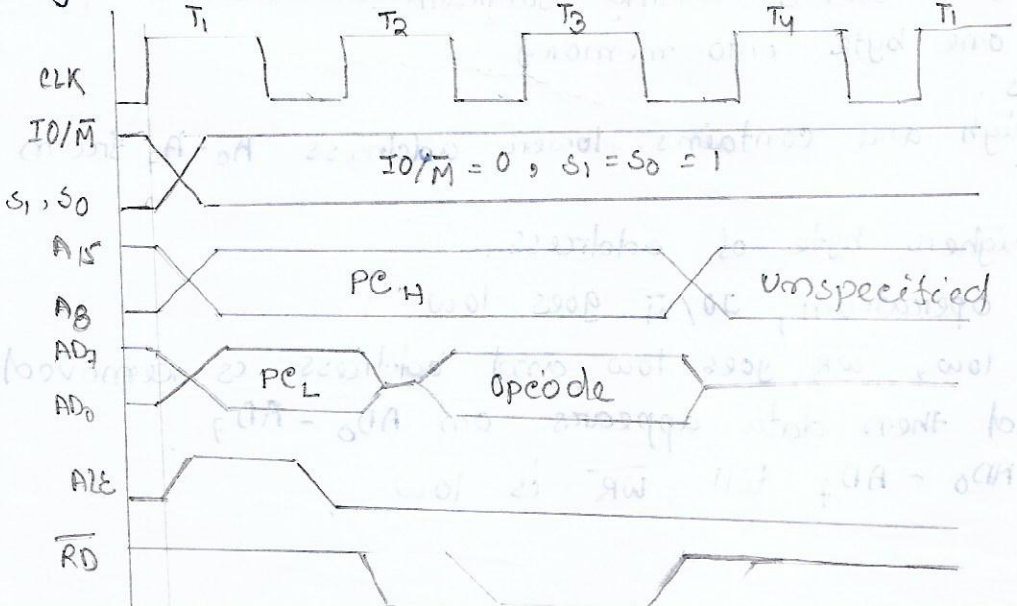
step-3 : During T_3 state, $\overline{WR}$ goes high, which disables the memory device and terminates the write operation.

Rules of identify number of machine cycles in an instruction :

1. If an addressing mode is direct, immediate or implicit then No. of machine cycles = No. of bytes.
2. If the addressing mode is indirect then No. of machine cycles = No. of bytes + 1. Add + 1 to the No. of machine cycles if it is memory read/write operation.
3. If the operand is 8-bit or 16-bit address then. No. of machine cycles = No. of bytes + 1

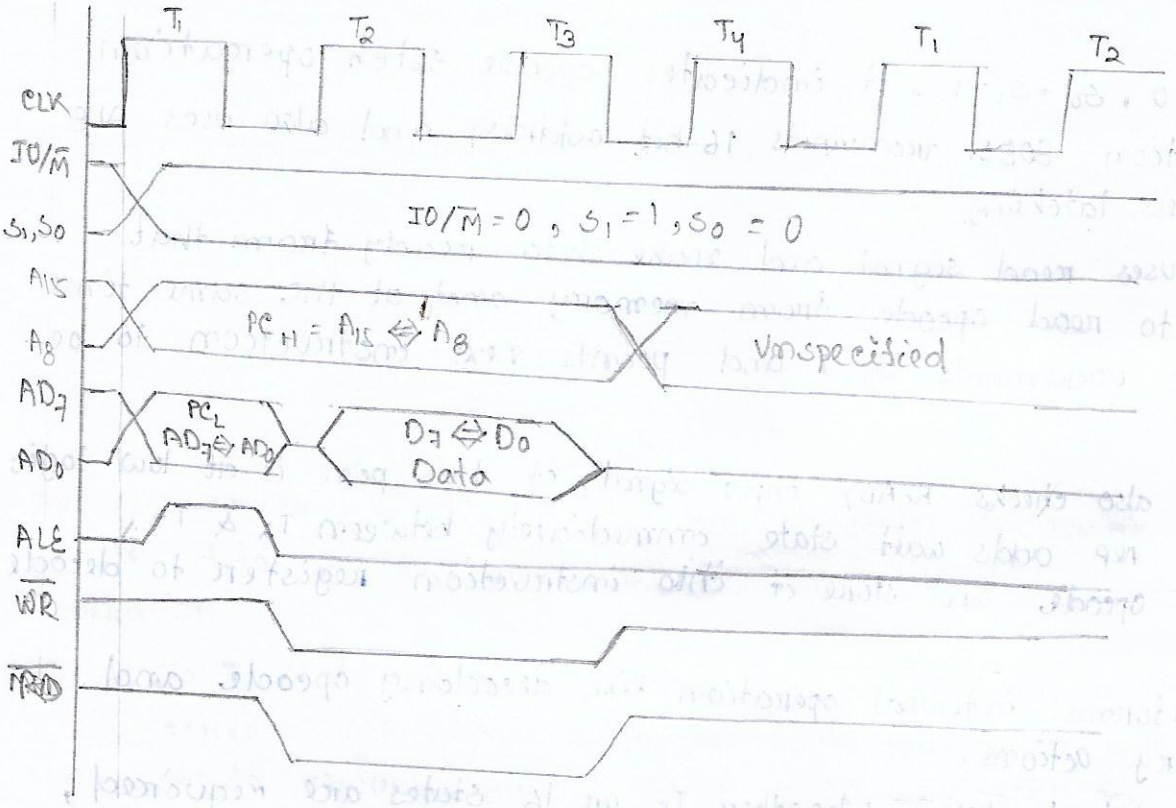
Operation for opcode fetch timing diagram :

- During T_1 state, MP uses $IO/\overline{M}$, S_0, S_1 signals are used to instruct MP to fetch opcode.
- Thus when $IO/\overline{M} = 0$, $S_0 = S_1 = 1$, it indicates opcode fetch operation.
- During this operation 8085 transmits 16-bit address and also uses ALE signal for address latching.
- AT T_2 state MP uses read signal and make data ready from that memory location to read opcode from memory and at the same time program counter increments by 1 and points next instruction to be fetched.
- On this state MP also checks READY input signal, if this pin is at low logic level i.e. '0' then MP adds wait state immediately between T_2 & T_3.
- AT T_3, MP reads opcode and store it into instruction register to decode it further.
- During T_4 MP performs internal operation like decoding opcode and providing necessary actions.
- The opcode is decoded to know wheather T_5 or T_6 states are required, if they are not required then MP performs next operation.



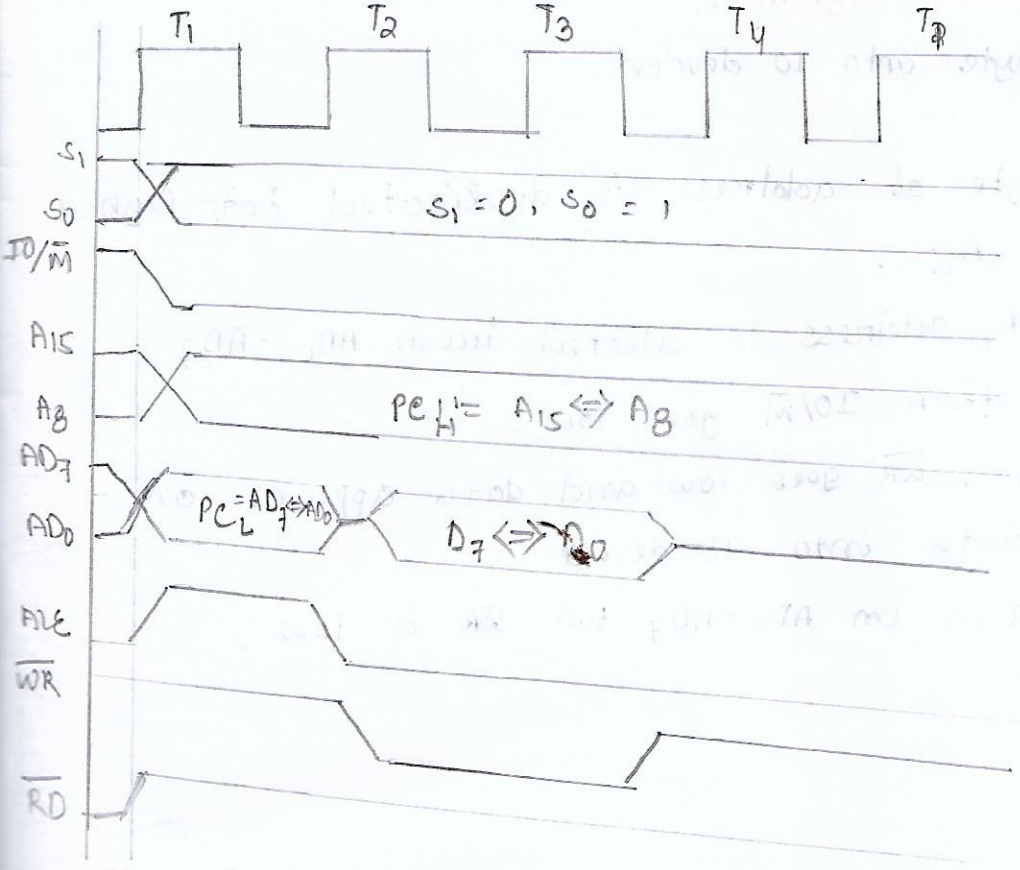
Operation for Memory Read Timing Diagram :

- It is used to fetch one byte from the memory.
- It requires 3 T-states.
- It can be used to fetch operand or data from the memory
- During T_1, $A_8 - A_{15}$ contains higher byte of address. At the same time ALE is high. Therefore lower byte of address $A_0 - A_7$ is selected from $AD_0 - AD_7$.
- since it is memory read operation, $IO/\overline{M}$ goes low.
- During T_2, ALE goes low, $\overline{RD}$ goes low. Address is removed from $AD_0 - AD_7$ and $D_0 - D_7$ appears on $AD_0 - AD_7$.
- During T_3, Data remains on $AD_0 - AD_7$ till $\overline{RD}$ is at low signal.



Operation for Memory write Timing Diagram :

- It is used to send one byte into memory.
- It requires 3 T-states.
- During T_1, ALE is high and contains lower address $A_0 - A_7$ from $AD_0 - AD_7$.
- $A_8 - A_{15}$ contains higher byte of address.
- As it is memory operation, $IO/\overline{M}$ goes low
- During T_2, ALE goes low, $\overline{WR}$ goes low and address is removed from $AD_0 - AD_7$ and then data appears on $AD_0 - AD_7$.
- Data remains on $AD_0 - AD_7$ till $\overline{WR}$ is low



Operation for I/O Read Timing Diagram :

- It is used to fetch one byte from an IO port.
- It requires 3 T-states.
- During T_1, the lower byte of IO address is duplicated into higher order address bus $A_8 - A_{15}$.
- ALE is high and $AD_0 - AD_7$ contains address IO device.
- IO/$\overline{M}$ goes high as it is an IO operation.
- During T_2, ALE goes low, $\overline{RD}$ goes low and data appears on $AD_0 - AD_7$ as input from IO device.
- During T_3 Data remains on $AD_0 - AD_7$ till $\overline{RD}$ is low.