CHAPTER 1
INTRODUCTION TO MICROPROCESSORS

MICROPROCESSORS

- A Microprocessor is a multipurpose, Programmable clock-driven, register based electronic device that read binary instruction from a storage device called memory, accepts binary data as input and processes data according to those instructions and provides results as outputs.
- A Microprocessor is a clock driven semiconductor device consisting of electronic circuits manufactured by using either a LSI or VLSI technique.
- A typical programmable machine can be represented with three components : MPU, Memory and I/O as shown in Figure

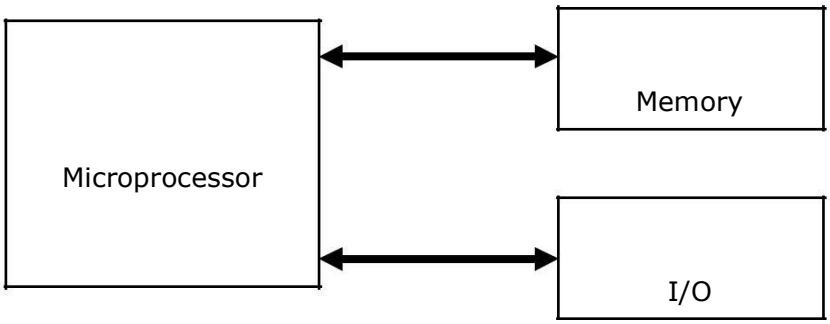


Figure: A Programmable Machine

- These three components work together or interact with each other to perform a given task; thus they comprise a system
- The machine (system) represented in above figure can be programmed to turn traffic lights on and off, compute mathematical functions, or keep trace of guidance system.
- This system may be simple or sophisticated, depending on its applications.
- The MPU applications are classified primarily in two categories : reprogrammable systems and embedded systems
- In reprogrammable systems, such as Microcomputers, the MPU is used for computing and data processing.
- In embedded systems, the microprocessor is a part of a final product and is not available for reprogramming to end user.

MICROCOMPUTER

- As the name implies, Microcomputers are small computers
- They range from small controllers that work directly with 4-bit words to larger units that work directly with 32-bit words

- Some of the more powerful Microcomputers have all or most of the features of earlier minicomputers.
- Examples of Microcomputers are Intel 8051 controller-a single board computer, IBM PC and Apple Macintosh computer.

MICRO CONTROLLER

- Single-chip Microcomputers are also known as Microcontrollers.
- They are used primarily to perform dedicated functions.
- They are used primarily to perform dedicated functions or as slaves in distributed processing.
- Generally they include all the essential elements of a computer on a single chip: MPU, R/W memory, ROM and I/O lines.
- Typical examples of the single-chip microcomputers are the Intel 8051, AT89C51, AT89C52 and Zilog Z8.
- Most of the micro controllers have an 8-bit word size, at least 64 bytes of R/W memory, and 1K byte of ROM
- I/O lines varies from 16 to 40

APPLICATIONS OF MICROPROCESSOR

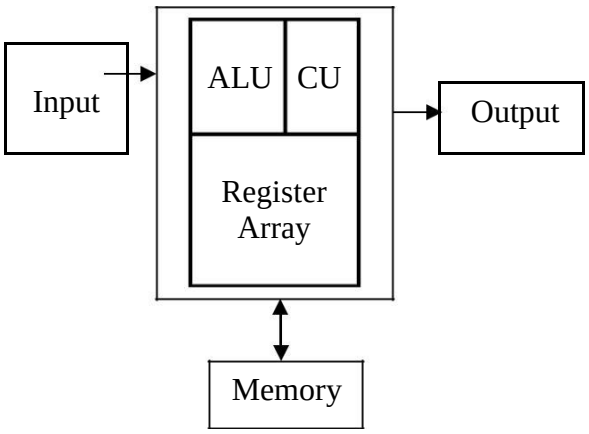
- Microcomputers
- Industrial Control
- Robotics
- Traffic Lights
- Washing Machines
- Microwave Oven
- Security Systems
- On Board Systems

Difference between Microprocessors and Microcontrollers

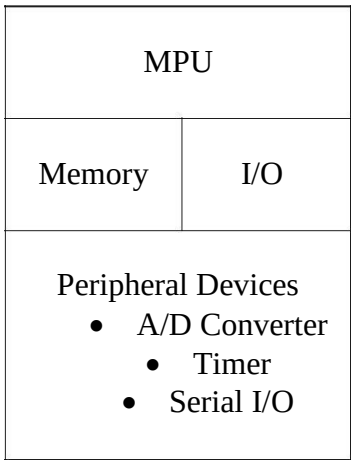
Although both microprocessor and microcontrollers have been designed for real time applications and they share many common features, they have significant differences which are as follows:

Microprocessor	Microcontroller
1. Microprocessor is a silicon chip which includes ALU, register circuit and control circuits. 2. The general block diagram to show microprocessor is as shown below:	1. Microcontroller is a silicon chip which includes microprocessor, memory and I/O in a single package. 2. The general block diagram of microcontroller is as shown below:

MPU



3. Normally used for general purpose computers as CPU.
4. The performance speed, i.e. clock speed of microprocessor is higher ranging frequency from MHz to GHz.
5. Addition of external RAM, ROM and I/O ports makes these systems bulkier and much more expensive.
6. Microprocessors are more versatile than microcontrollers as the designers can decide on the amount of RAM, ROM and I/O ports needed to fit the task at hand. E.gs. Intel 8085, 8086, Motorola 68000, Intel Core i7, etc.



3. Normally microcontrollers are used for specific purposes (embedded system) e.g. traffic light controller, printer, etc.
4. The performance speed of microcontroller is relatively slower than that of microprocessors, with clock speed from 3-33 MHz.
5. Has fixed memory and all peripherals are embedded together on a single chip, so are not bulkier and are cheaper than microprocessors.
6. As microcontrollers have already fixed amount of RAM, ROM and I/O ports, so are not versatile as the user cannot change the amount of memory and I/O ports. E.gs. AT89C51, ATmega32, AT89S52, etc.

Evolution of Intel Microprocessors

4 bit Microprocessors

4004

- Introduced in 1971
- First microprocessor by Intel
- It was a 4-bit microprocessor
- Its clock speed was 740 KHz
- It had 2,300 transistors
- It could execute around 60,000 instructions per seconds
- Used in calculators

4040

Introduced in 1974

- 4-bit microprocessor
- 3,000 transistors were used
- Clock speed was 740 KHz
- Interrupt features were available

8 Bit Microprocessors

8008

- Introduced in 1972 it was first 8 bit microprocessor
- Its clock speed was 500 KHz
- Could execute 50,000 instruction per second
- Used in: Computer terminals, Calculator, Bottling Machines, industrial Robots

8080

- Introduced in 1974
- It was also 8-bit microprocessor
- Its clock speed was 2 MHz
- It has 6,000 transistors
- 10 times faster than 8008
- Could execute 500,000 instructions per second
- Used In: Calculators, Industrial Robots

8085

- Introduced in 1976
- It was also 8-bit microprocessor
- Its clock speed was 3 MHz
- Its data bus is 8 bit and address bus is 16 bit
- It has 6,500 transistors
- It could execute 769,230 instructions per second
- It could access 64KB of memory
- It has 246 instructions
- Used In: early PC, On-Board Instrument Data Processors

16 Bit Microprocessors

8086

- Introduced in 1978
- First 16-bit microprocessor
- Clock speed is 5 to 10 MHz
- Data bus is 16-bit and address bus is 20-bit
- It had 29,000 transistors
- It could execute 2.5 million instructions per second
- Could access 1MB of memory
- It had 22,000 instructions
- Used In: CPU of Microcomputers

8088

- Introduced in 1979
- It was also 16-bit microprocessor
- It was creates as cheaper version of Intel's 8086
- 16-bit processor with an 8-bit data bus
- Could execute 2.5 million instructions per second
- The chip become the most popular in the computer industry when IBM used it for its first PC

80286

- Introduced in 1982
- It was 16-bit microprocessor
- Its clock speed was 8 MHz
- Data bus is 16-bit and address bus is 24-bit
- Could address 16 MB of memory
- It has 134,000 transistors
- Could execute 4-million instructions per second

32 Bit Microprocessors

80386

- Introduced in 1986
- First 32-bit microprocessor
- Data bus is 32 bit and address bus is 32-bit
- It could address 4GB of memory
- It has 275,000 transistors
- Clock speed varied from 16 MHz to 33 MHz depending upon different versions
- Different Versions
    - 80386DX
    - 80386SX
    - 80386SL

80486

- Introduced in 1989
- 32-bit microprocessor
- Had 1.2 million transistors
- Clock speed varied from 16 MHz to 100 MHz depending upon the various versions
- It had five different versions
    - 80486DX
    - 80486SX
    - 80486DX2
    - 80486SL
    - 80486DX4
- 8KB of cache memory was introduced

Pentium

- Introduced in 1993
- It was also 32-bit microprocessor
- Clock speed was 66 MHz
- Data bus is 32-bit and address bus is 32-bit
- Could address 4GB of memory
- Could execute 110 million instructions per second
- Cache memory
  - 8KB for Instruction
  - 8KB for data
- Upgraded Version: Pentium Pro

Pentium II

- Introduced in 1997
- 32-bit microprocessor
- Clock speed was 233 to 450 MHz
- MMX technology was supported
- L2 cache and processor were on one circuit
- Upgraded Version: Pentium II Xenon

Pentium III

- Introduced in 1999
- It was 32-bit microprocessor
- Clock speed varied from 500 MHz to 1.4 GHz
- It had 9.5 million transistors

Pentium IV

- Introduced in 2000
- 32-bit microprocessor
- Clock speed was from 1.3 GHz to 3.8 GHz
- L1 cache was 32 KB and L2 cache was 256 KB
- It had 42 million transistors

Intel Dual Core

- Introduced in 2006
- It is 32-bit or 64 bit Microprocessor
- It has 2-cores
- Both cores have their own internal bus and L1 cache but share the external bus and L2 cache
- Support SMT (Simultaneously Multithreading Technology)

64 Bit Microprocessors

Intel Core 2

- Introduced in 2006
- 64-bit microprocessor
- Clock speed is from 1.2 GHz to 3GHz
- It has 291 million transistors
- L1 cache- 64 KB per core
- L2 cache- 4 MB
- Versions:
  - Intel Core 2 Duo
  - Intel Core 2 Quad
  - Intel Core 2 Extreme

Intel Core i7

- Introduced in 2008
- 64-bit microprocessor
- It has 4 physical cores
- Clock speed is from 2.66 GHz to 3.33 GHz
- It has 781 million transistors
- L1 cache- 64 KB per core
- L2 cache- 256 KB
- L3 cache- 4 MB

Intel Core i5

- Introduced in 2009
- It is a 64-bit microprocessor
- It has 4 physical cores
- Its clock speed is from 2.40 GHz to 3.60 GHz
- It has 781 million transistors
- L1 cache- 64 KB per core
- L2 cache- 256 KB
- L3 cache- 8 MB

Intel Core i3

- Introduced in 2010
- 64-bit microprocessor
- It has 2 physical cores
- Clock speed is from 2.93 GHz to 3.33 GHz
- It has 781 million transistors
- L1 cache- 64 KB per core
- L2 cache- 512 KB
- L3 cache- 4 MB

Intel Core i9

- Introduced in 2017
- 64-bit microprocessor
- It has 10 physical cores
- Clock speed is from 3.3 GHz to 4.5 GHz
- It has around 7 billion transistors
- L3 smart cache- 13.75MB

Advantages of Microprocessor based system

 Computational/processing speed is high.
 Intelligence has been brought to systems.
 Automation of industrial processes and office administration.
 Since the devices are programmable, there is flexibility to alter the system by changing the software alone.
 Less number of components, compact in size and cost less. Also it is more reliable.
 Operation and maintenance are easier.

Disadvantages of Microprocessor based System

7. It has limitations on the size of data.
8. The applications are limited by the physical address space.
9. The analog signals cannot be processed directly and digitizing the analog signals introduces errors.
10. The speed of execution is slow and so real time applications are not possible.
11. Most of the microprocessors does not support floating point operations.

Harvard vs. Von Neumann Architecture

Two dominant computer architectures exist for designing microprocessors and microcontrollers. These two architectures include Harvard and von Neumann. The Harvard and von Neumann architectures consist of four major subsystems: memory, input/output (I/O), arithmetic/logic unit (ALU), and control unit (diagrammed in Figures 1a and 1b). The ALU and control unit operate together to form the central processing unit (CPU). Instructions and data are stored in high-speed memories called registers within the CPU. Each of these components interact together to complete the execution of instructions.

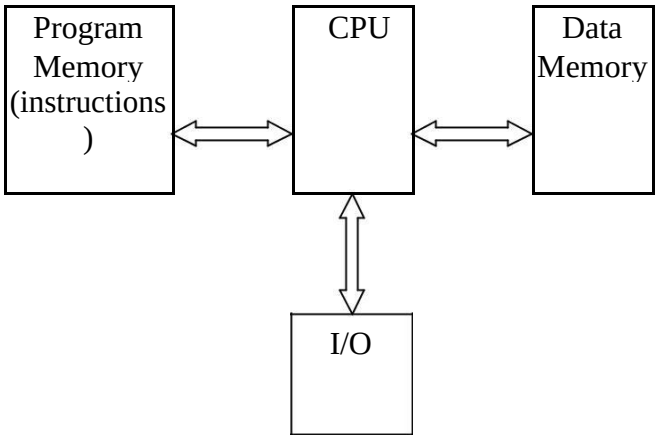


Figure 1a: Harvard Architecture

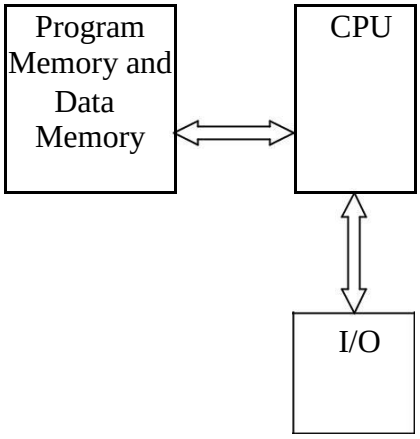


Figure 1b: von Neumann Architecture

The von Neumann architecture allows for one instruction to be read from memory or data to be read/written from/to memory at a time. In other words, an instruction fetch and data operation cannot be performed at the same time. Instructions and data are stored in the same memory subsystem and share a communication pathway or bus to the CPU. This constraint is referred to as the von Neumann bottleneck and directly impacts the performance of the system.

The von Neumann bottleneck limits the throughput or data transfer rate between the CPU and memory. In order to complete most operations, the CPU is required to wait for data. The bottleneck becomes more of a burden on high-performance systems as memory and processing speed increases. The Harvard architecture addresses some of these limitations.

The Harvard architecture alternatively consists of separate pathways or buses for interaction between the CPU and memory. The separation allows for instructions and data to be accessed concurrently. Also, a new instruction may be fetched from memory at the same time another one is finishing execution, allowing for a primitive form of pipelining. Pipelining decreases the execution time of one instruction, but main memory access time, in many cases, is a major bottleneck in the overall performance of the system

Bus Structure of microprocessor:

A bus is a path or a collection of wires or lines that carries data, address and control signals.

There are three buses in Microprocessor:

1.Address Bus
2.Data Bus
3.Control Bus

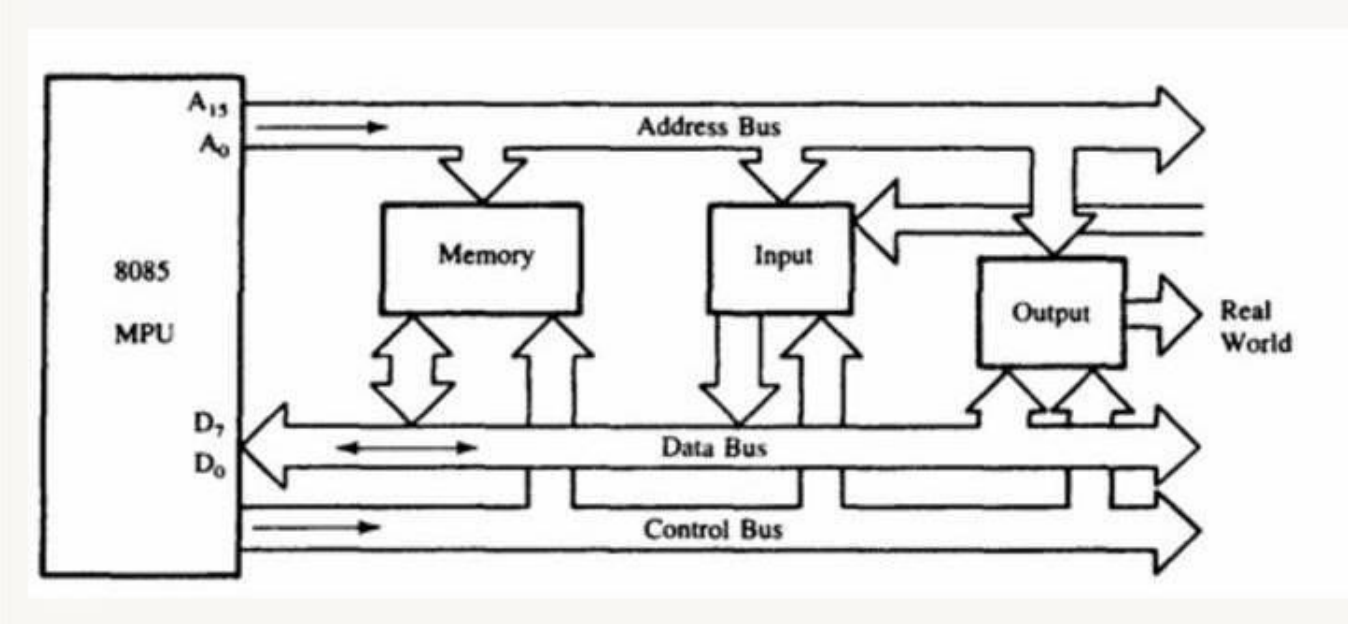


Fig: Bus Structure of 8085 microprocessor

1. **Address Bus**:- The bus over which the CPU sends out the address of the memory location is known as Address bus. The address bus carries the address of memory location to be written or to be read from. The address bus is unidirectional. It means bits flowing occur only in one direction, only from microprocessor to peripheral devices. We can find that how much memory location it can use the formula2^ N. where N is the number of bits used for address lines.
8085 Microprocessor has 16 bit address bus.
here, 2^16 = 65536bytes or 64Kb
So we can say that it can access up to 64 kb memory location.

2.**Data Bus**:-8085 Microprocessor has 8 bit data bus. So it can be used to carry the 8 bit data starting from 00000000H (00H) to 11111111H (FFH). Here 'H' tells the Hexadecimal Number. It is bidirectional. These lines are used for data flowing in both direction means data can be transferred or can be received through these lines. The data bus also connects the I/O ports and CPU. The largest number that can appear on the data bus is 11111111.
It has 8 parallel lines of data bus. So it can access upto 2^8 = 256 data bus lines.

3.**Control Bus**:-The control bus is used for sending control signals to the memory and I/O devices. The CPU sends control signal on the control bus to enable the outputs of addressed memory devices or I/O port devices.

Some of the control bus signals are as follows:
1.Memory read
2.Memory write
3.I/O read
4.I/O write.

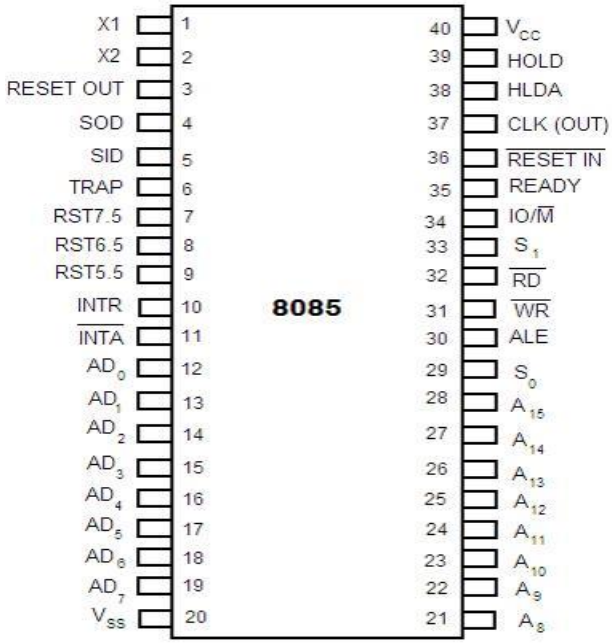
CHAPTER 2
8085 Microprocessors and ALP

8085 INTRODUCTION

The features of INTEL 8085 are:

- It is an 8 bit processor.
- It is a single chip N-MOS device with 40 pins.
- It works on 5 Volt dc power supply.
- The maximum clock frequency is 3 MHz while minimum frequency is 500 KHz.
- It provides 74 instructions with 5 different addressing modes.
- It has multiplexed address and data bus (AD0-AD7).
- It provides 16 address lines so it can access 2^{16} =64K bytes of memory.
- It generates 8 bit I/O address so it can access 2^{8}=256 input ports.

8085 Pin Diagram



Some important pins are:

AD0-AD7: Multiplexed Address and data lines.

A8-A15: Tri-stated higher order address lines.

ALE: Address latch enable is an output signal. It goes high when operation is started by processor.

S0, S1: These are the status signals used to indicate type of operation.
RD: Read is active low input signal used to read data from I/O device or memory.
WR: Write is an active low output signal used write data on memory or an I/O device.

READY: This an output signal used to check the status of output device. If it is low, µP will WAIT until it is high.

TRAP: It is an Edge triggered highest priority, non-maskable interrupt. After TRAP, restart occurs and execution starts from address 0024H.

RST 5.5, 6.5, 7.5: These are maskable interrupts and have low priority than TRAP.

INTR & $\overline{\text{INTA}}$: INTR is an interrupt request signal after which µP generates $\overline{\text{INTA}}$ or interrupt acknowledge signal.

IO/$\overline{\text{M}}$: This is output pin or signal used to indicate whether 8085 is working in I/O mode (IO/$\overline{\text{M}}$=1) or Memory mode (IO/$\overline{\text{M}}$=0).

HOLD & HLDA: HOLD is an input signal .When µP receives HOLD signal it completes current machine cycle and stops executing next instruction. In response to HOLD µP generates HLDA that is HOLD Acknowledge signal.

$\overline{\text{RESET IN}}$: This is input signal. When $\overline{\text{RESET IN}}$ is low µp restarts and starts executing from location 0000H.

SID: Serial input data is input pin used to accept serial 1 bit data.

SOD: Serial output data is output pin used to send serial 1 bit data.

X1, X2: These are clock input signals and are connected to external LC or RC circuit. These are divide by two so if 6 MHz is connected to X1X2, the operating frequency becomes 3 MHz.

VCC & VSS: Power supply VCC=+ -5Volt& VSS=-GND reference.

RESTART INTERRUPTS;

These three inputs have the same timing as INTR except they cause an

internal RESTART to be automatically inserted.

RST 7.5 - Highest Priority

RST 6.5

RST 5.5- Lowest Priority

The priority of these interrupts is ordered as shown above. These interrupts have a higher priority than the INTR.

RST5.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 2CH (hexadecimal) address.

RST6.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 34H (hexadecimal) address.

RST7.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 3CH (hexadecimal) address.

TABLE 1.1 : 8085 SIGNAL DESCRIPTION SUMMARY

Pin Name	Description	Type
AD_0 - AD_7	Address / Data Bus	Bidirectional, Tristate
A_8 - A_{15}	Address Bus	Output, Tristate
ALE	Address Latch Enable	Output, Tristate
$\overline{RD}$	Read Control	Output, Tristate
$\overline{WR}$	Write Control	Output, Tristate
IO/$\overline{M}$	I/O or memory Indicator	Output, Tristate
S_0, S_1	Bus State Indicator	Output
READY	Wait state request	Input
SID	Serial Input Data	Input
SOD	Serial Output Data	Output
HOLD	HOLD request	Input
HLDA	HOLD acknowledge	Output
INTR	Interrupt request	Input
TRAP	Nonmaskable interrupt request	Input
RST 5.5	Hardware vectored Interrupt request	Input
RST 6.5	Hardware vectored Interrupt request	Input
RST 7.5	Hardware vectored Interrupt request	Input
$\overline{INTA}$	Interrupt acknowledge	Output
$\overline{RESET\ IN}$	System reset	Input
RESET OUT	Peripherals reset	Output
X_1, X_2	Crystal or RC Connection	Input
CLK (OUT)	Clock Signal	Output
V_{cc}, V_{ss}	Power, ground	.....

***Note :** A overbar on the signal, indicates that it is active low. (i.e., the signal is normally high and when the signal is activated it is low).*

IO/$\overline{M}$	S_1	S_0	Operation performed by the 8085
0	0	1	Memory WRITE
0	1	0	Memory READ
1	0	1	I/O WRITE
1	1	0	I/O READ
0	1	1	Opcode fetch
1	1	1	Interrupt acknowledge

8085 ARCHITECTURE

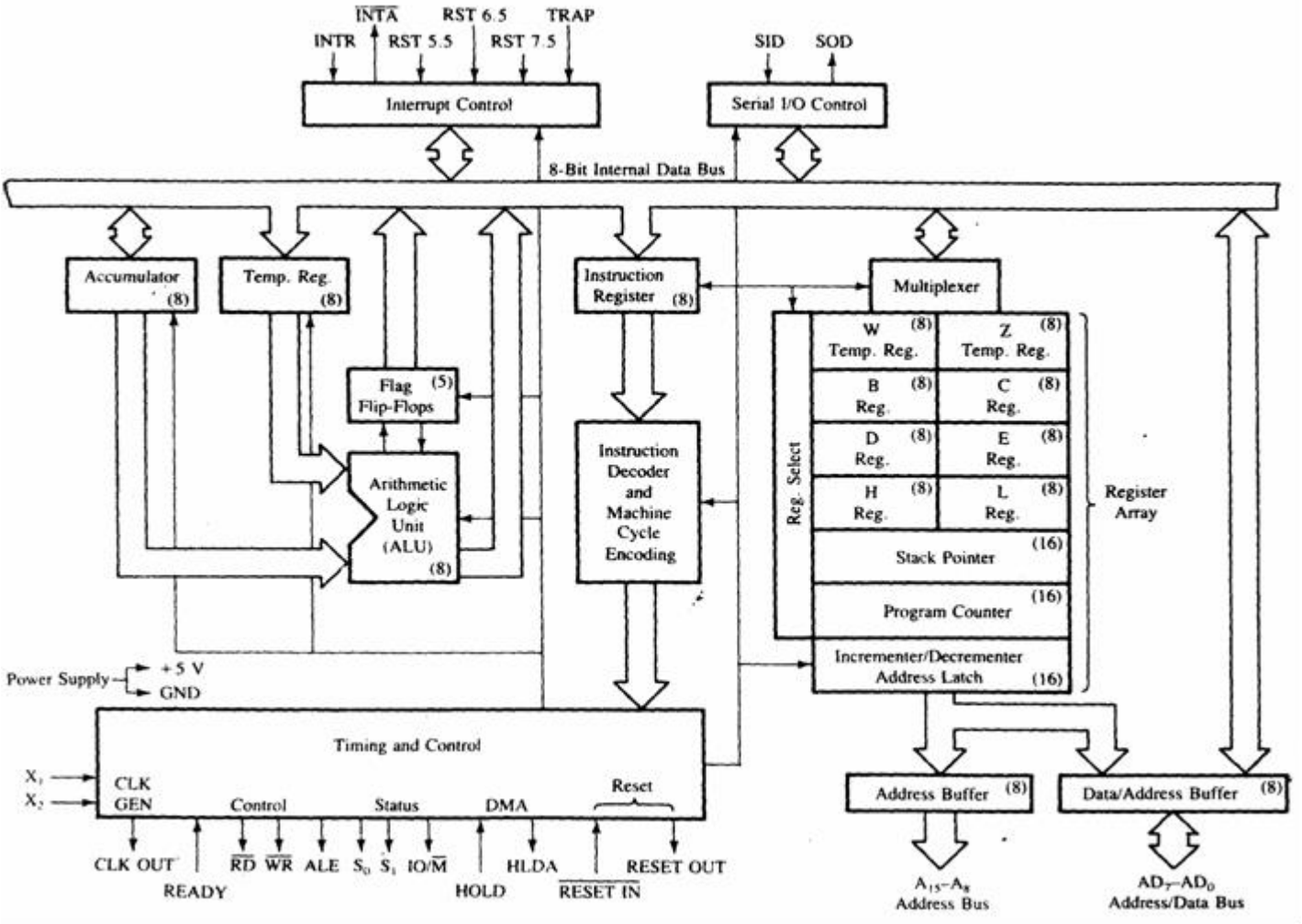


Figure: Internal Architecture of 8085 microprocessor.

ARITHMETIC AND LOGIC UNIT (ALU)

Accumulator:

It is 8 bit general purpose register. It is connected to ALU. So, most of the operations are done in Accumulator (A).

Temporary register:

It is not available for user. All the arithmetic and logical operations are done in the temporary register but user can't access it.

Flag Register: It is an 8-bit register which consists of 5 flip flops used to know status of various operations done.

S	Z	-	AC	-	P	-	CY

Fig. *8085 Flag Register*

S: Sign flag is set when result of an operation is negative.
Z: Zero flag is set when result of an operation is 0.
AC: Auxiliary carry flag is set when there is a carry out of lower nibble or lower four bits of the operation.
CY: Carry flag is set when there is carry generated by an operation.
P: Parity flag is set when result contains even number of 1's.
Rest are don't care flip flops and reserved for future use.

REGISTER ARRAY

Temporary registers (W, Z): These are not available for user. These are loaded only when there is an operation being performed.

General purpose: There are six 8-bit general purposes register in 8085 namely B, C, D, E, H and L. These are used for various data manipulations. They can be used in pairs as 16-bit registers. The register pairs are: BC pair, DE pair and HL pair. **Special purpose:** There are two special purpose registers in 8085:

SP (Stack Pointer): It is a 16-bit register used to hold the address of stack during stack operation i.e PUSH and POP operations.

PC (Program Counter): It is a 16-bit register which holds the address of next instruction to be fetched. When a single byte instruction is executed PC is automatically incremented by 1. Upon reset PC contents are set to 0000H.

TIMIMG AND CONTROL UNIT

This unit synchronizes all the microprocessor operations with the clock and generates the control signals necessary for communication between the microprocessor and peripherals. The $\overline{\text{}}$ and $\overline{\text{}}$ signals are sync pulse indicating the availability of data on the data bus.

INSTRUCTION REGISTER AND DECODER

The instruction register and decoder are part of ALU. When an instruction is fetched from memory, it is loaded in the instruction register. The decoder decodes the instruction and establishes the sequence of events to flow. The instruction register is not programmable and cannot be accessed through any instructions.

INTERRUPT CONTROL

It accepts different interrupts like TRAP, RST 5.5, RST 6.5, RST 7.5 and INTR. INTA is interrupt acknowledgement signal.

SERIAL IO CONTROL

It is used to accept and send the serial 1 bit data by using SID and SOD signals and it can be performed by using SIM & RIM instructions.

8085 INSTRUCTION SETS

a) **Data Transfer Instructions**
b) **Arithmetic Instructions**
c) **Logical Instructions**
d) **Rotate Instructions**
e) **Branching Instructions**
f) **Control Instructions**

a) Data Transfer instructions

Mnemonics	Description	Example
MOV Rd, Rs	• Copies the content of source register Rs into destination register Rd • Rs and Rd can be A, B, C, D, E, H, L	MOV A, B
MOV Rd, M	• Copies the content of memory location M into the destination register Rd • Rd can be A, B, C, D, E, H, L • The memory location M is specified by HL pair	MOV A, M
MOV M, Rs	• Copies the content of register Rs into memory location M • Rs can be A, B, C, D, E, H, L • The memory location M is specified by HL pair	MOV M, A
MVI Rd, 8-bit	• The 8-bit data is stored in the destination register Rd • Rd can be A, B, C, D, E, H, L	MVI A, 32H
MVI M, 8-bit	• The 8-it data is stored in memory location M • M is specified by HL pair	MVI M, 32H
LDA 16-bit	• Copies the content of memory	LDA 2015H

(Load Accumulator Direct)	location specified by 16-bit address into A	A ←[2015]
STA 16-bit (Store Accumulator Direct)	• Copies the content of A into 16-bit memory address	STA 2015H A→[2015]
LDAX Rp (Load Accumulator Inirect)	• Copies the content of memory location specified by register pair Rp into A • Rp can be B or D i.e. BC pair or DE pair	LDAX B
STAX Rp (Store Accumulator Indirect)	• Copies the content of A into 16-bit memory address specified by register pair Rp • Rp can be B or D i.e. BC pair or DE pair	STAX B
LXI Rp, 16-bit (Load Register Pair)	• Loads 16-bit data into register pair • Rp can be B, D or H i.e. BC pair, DE pair or HL pair	LXI H, 2015H L←15 H←20
IN 8-bit	• The data from i/p port specified by 8-bit address is transferred into A	IN 40H A← [40] 40H is address of input port
OUT 8-bit	• The data of A is transferred into output port specified by 8-bit address	OUT 10H A→ [10] 10H is address of output port
XCHG	• Exchange the content of HL pair with DE pair i.e. the content of H and D are exchanged whereas content of L and E are exchanged	XCHG H←→D L←→E

b) Arithmetic Instructions

Mnemonics	Description	Example
ADD R/M (add register/memory)	• The content of register /memory (R/M) is added to the A and result is stored in A • The memory M is specified by HL pair	ADD B A←A+B ADD M A←A+M
ADC R/M (add with carry)	• The content of register /memory (R/M) is added to the A along with carry flag CF and result is stored in A • The memory M is specified by HL pair	ADC B A←A+B+CF ADC M A←A+M+CF
ADI 8-bit (add immediate)	• The 8-bit data is added to A and result is stored in A	ADI 32H A←A+32
ACI 8-bit	• The 8-bit data is added to A	ACI 32H

(add immediate with carry)	along with carry flag CF and result is stored in A	A←A+32+CF
SUB R/M (subtract register/memory)	• The content of register /memory (R/M) is subtracted from A and result is stored in A • The memory M is specified by HL pair	SUB B A←A-B SUB M A←A-M
SBB R/M (subtract with borrow)	• The content of register /memory (R/M) is subtracted from A along with borrow flag BF and result is stored in A • The memory M is specified by HL pair	SBB B A←A-B-BF SBB M A←A-M-CF
SUI 8-bit (subtract immediate)	• The 8-bit data is subtracted from A and result is stored in A	SUI 32H A←A-32
INR R/M (Increment Register/Memory)	• Increment the content of register/memory by 1 • Memory is specified by HL pair	INR B B←B+1 INR M M←M+1
DCR R/M (Decrement Register/Memory)	• Decrement the content of register/memory by 1 • Memory is specified by HL pair	DCR B B←B-1 DCR M M←M-1
INX Rp (Increment Register Pair)	• Increment the content of register pair Rp by 1	INX H HL←HL+1
DCX Rp (Decrement Register Pair)	• Decrement the content of register pair Rp by 1	DCX H HL←HL-1

c) Logical Instructions

Mnemonics	Description	Example
CMP R/M (Compare Register/Memory)	• Compares the content of register/memory with A • The result of comparison is: If A< R/M : Carry Flag CY=1 If A= R/M : Zero Flag Z=1 If A> R/M : Carry Flag CY=0	CMP B CMP M
CPI 8-bit (Compare Immediate)	• Compares 8-bit data with A • The result of comparison is: If A< 8-bit : Carry Flag CY=1 If A= 8-bit : Zero Flag Z=1 If A> 8-bit : Carry Flag CY=0	 CPI 55H

ANA R/M (logical AND register/memory)	• The content of A are logically ANDed with the content of register/memory and result is stored in A • Memory M must be specified by HL pair	ANA B A←A.B ANA M A←A.M
ANI 8-bit (AND immediate)	• The content of A are logically ANDed with the 8-bit data and result is stored in A	ANI 32H A←A.32H
ORA R/M (logical OR register/memory)	• The content of A are logically ORed with the content of register/memory and result is stored in A • Memory M must be specified by HL pair	ORA B ORA M
ORI 8-bit (OR immediate)	• The content of A are logically ORed with the 8-bit data and result is stored in A	ORI 32H
XRA R/M (logical XOR register/memory)	• The content of A are logically XORed with the content of register/memory and result is stored in A • Memory M must be specified by HL pair	XRA B XRA M
XRI 8-bit (XOR immediate)	• The content of A are logically XORed with the 8-bit data and result is stored in A	XRI 32H

d) Rotate Instructions

Mnemonics	Description	Example
RLC (Rotate Accumulator Left)	• Each bit of A is rotated left by one bit position. • Bit D7 is placed in the position of D0.	RLC
RRC (Rotate Accumulator Right)	• Each bit of A is rotated right by one bit position. • Bit D0 is placed in the position of D7.	RRC
RAL (Rotate Accumulator Left with Carry)	• Each bit of A is rotated left by one bit position along with carry flag CY • Bit D7 is placed in CY and CY in the position of D0.	RAL
RAR (Rotate Accumulator Right with Carry)	• Each bit of A is rotated right by one bit position along with carry flag CY. • Bit D7 is placed in CY and CY in the position of D0.	RLC

e) Branching Instructions

Mnemonics	Description	Example
JMP 16-bit (Unconditional Jump)	The program sequence is transferred to the memory location specified by 16-bit address	JMP C000H
JC	Jump on Carry (CY=1)	
JNC	Jump No Carry (CY=0)	
JP	Jump on Positive (S=0)	
JM	Jump on Negative (S=1)	
JZ	Jump on Zero (Z=1)	
JNZ	Jump No Zero (Z=0)	
JPE	Jump on Parity Even (P=1)	
JPO	Jump on Parity Odd (P=0)	
CALL 16-bit	The program sequence is transferred to the subroutine at memory location specified by the 16-bit address	CALL C000H
RET	The program sequence is transferred from the subroutine program to calling program	RET

f) Control Instructions

Mnemonics	Description	Example
NOP	No operation is performed	NOP
HLT	The CPU finishes executing the current instruction and stops any further execution	HLT

8085 ADDRESSING MODES

The ways by which operands are specified in an instruction are called addressing modes. The different addressing modes of 8085 are:

1. Immediate Addressing Mode

In immediate addressing mode, the data is specified in the instruction itself. The data will be apart of the program instruction. All instructions that have 'I' in their mnemonics are of Immediate addressing type.

Examples: MVI A, 05H
ADI 55H
LXI H, C000H

2. Register Addressing Mode

If the data is present in the register and the register are specified in an instruction, than it is called register addressing mode.

Example: MOV A, B
ADD B
ANA C

3. Direct Addressing Mode

In direct addressing mode, the address of the data is specified in the instruction. The data will be in memory. In this addressing mode, the program instructions and data can be stored in different memory blocks. This type of addressing can be identified by 16-bit address present in the instruction.

Example: LDA 2000H
STA 2000H
IN 10H
OUT 01H

4. Register Indirect Addressing Mode

If the register pair which contains the address of the data is specified in the instruction, than it is called register indirect addressing mode.

Example: LDAX B
STAX D
MOV M, A
MOV B, M

5. Implied Addressing Mode

If the opcode in an instruction tells about the operand, than it is called implied addressing mode.

Example: RAL
RRC

TIMING DIAGRAMS OF 8085 INSTRUCTIONS

a) Related Terms

Instruction Cycle: The time taken to complete the execution of an instruction is called instruction cycle. It is the combinations of machine cycles.

Machine Cycle: The time taken by the processor to access memory location, IO ports or to acknowledge an interrupt once is called as machine cycles. It is the combinations of T-states.

T-States: It is the sub-division of operation performed in one clock cycle of processor's clock.

b) Status Signals for Various Machine Cycles

Machine Cycles	Status Signals		
		S_1	S_0
Opcode Fetch	0	1	1
Memory Read	0	1	0
Memory Write	0	0	1
IO Read	1	1	0
IO Write	1	0	1

c) Types of Instruction On the Basis of Size One

Byte Instructions

These Instruction use a total memory of one-byte

These Instructions include the opcode and operand in the same byte.

Examples: MOV C, A

ADD B RLC

Two Byte Instructions

These instructions use a total memory of two bytes.

The first byte specifies the opcode and second byte specifies the operand

Examples: MVI A, 32H

ADI 30H IN

40H

Three Byte Instructions

These instructions use a total memory of three bytes.

The first byte specifies the opcode and the remaining two bytes specifies the 16-bit address i.e. second byte specifies the lower order address and third byte specifies the higher order address.

Example: LDA 2070H STA

2050H LXI H,

2070H

Machine Cycle of 8085:

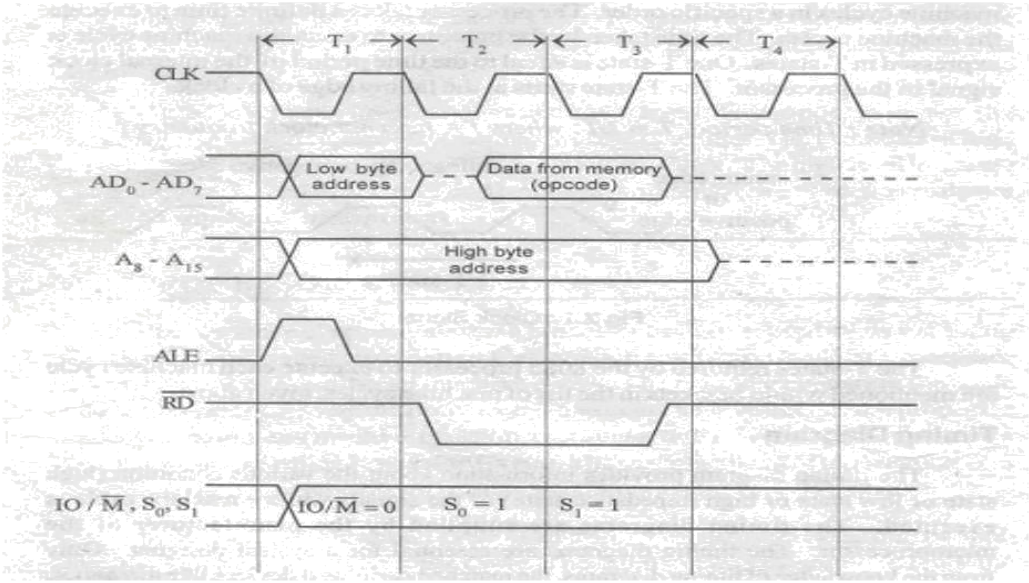
The various machine cycles are

1. Opcode fetch …….. - 4 / 6 T
2. Memory Read ……. - 3 T
3. Memory Write ……. - 3 T
4. I/O Read …….. - 3 T
5. I/O Write ……. - 3 T
6. Interrupt Acknowledge …… - 6 / 12 T
7. Bus Idle …… - 2 / 3 T

Opcode fetch machine cycle of 8085

Each instruction of the processor has one byte opcode. The opcodes are stored in memory. The opcode fetch machine cycle is executed by the processor to fetch the opcode from memory. Hence, every instruction starts with opcode fetch machine cycle.

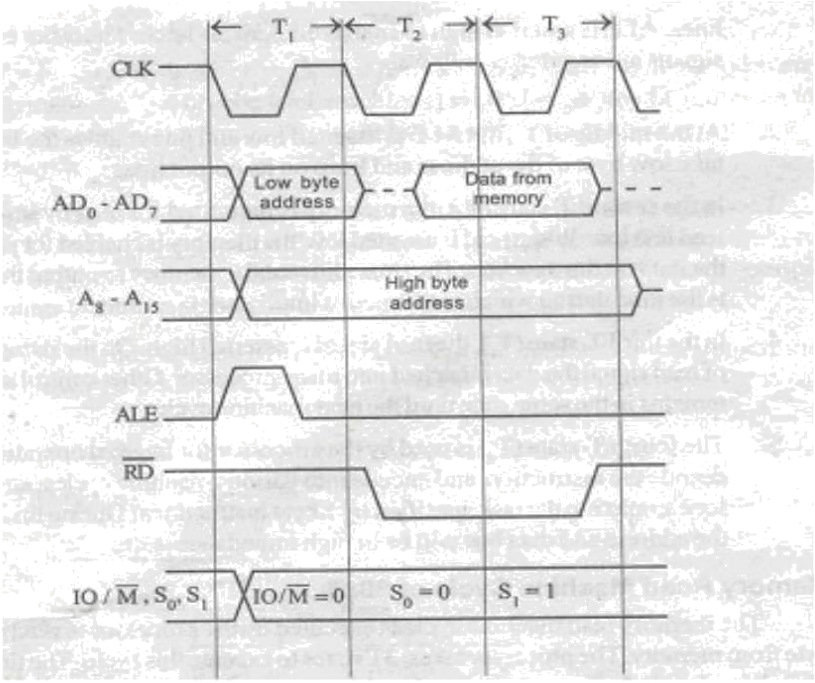
The time taken by the processor to execute the opcode fetch cycle is either 4 T or 6 T . In this time, the first, 3 T-states are used for fetching the opcode from memory and the remaining T-states are used for internal operations by the processor. The timings of various signals during opcode fetch cycle is shown in fig 2.2.



1. At the falling edge of first T-state (T_1), the microprocessor outputs the low byte address on AD_0-AD_7 lines and high byte address on A_8 to A_{15} lines. ALE is asserted high to enable the address latch. The other control signals are asserted as follows.
   $IO/\overline{M} = 0$, $S_0 = 1$, $S_1 = 1$.
2. At the middle of T_1, the ALE is asserted low and this enables the latch to take low byte of the address and keep on its output lines.
3. In the second T-state (T_2), the memory is requested for read by asserting read line low. When read is asserted low, the memory is enabled for placing the data on the data bus. The time allowed for memory to output the data is the time during which read remains low.
4. In the third T-state (T_3), the read signal is asserted high. On the rising edge of read signal the data is latched into microprocessor. Other control signals remains in the same state until the next machine cycle.
5. The fourth T-state (T_4) is used by the processor for internal operations to decode the instruction and encode into various machine cycles, and also for completing the task specified by 1 byte instructions. During this cycle the address and data bus will be in high impedance state.

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 timings of various signals during memory read cycle are shown in fig 2.3.

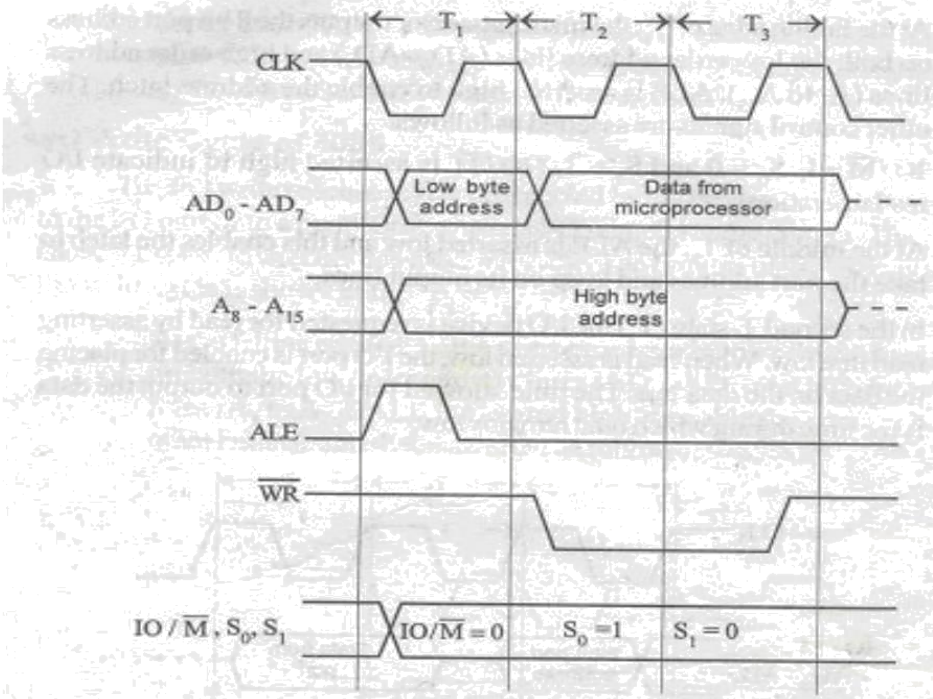


1. At the falling edge of T_1, the microprocessor outputs the low byte address on AD_0 - AD_7 lines and high byte address on A_8 to A_{15} lines. ALE is asserted high to enable the address latch. The other control signals are asserted as follows.

   $IO/\overline{M} = 0, S_0 = 0, S_1 = 1.$
2. At the middle of T_1, the ALE is asserted low and this enables the latch to take low byte of address and keep on its output lines.
3. In the second T-state (T_2) the memory is requested for read by asserting read line low. When read is asserted low, the memory is enabled for placing the data on the data bus. The time allowed for memory to output the data is the time during which read remains low.
4. At the end of T_3, the read signal is asserted high. On the rising edge of read signal the data is latched into microprocessor. Other control signals remains in the same state until the next 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. The timings of various signals during memory write cycle are shown in fig 2.4.



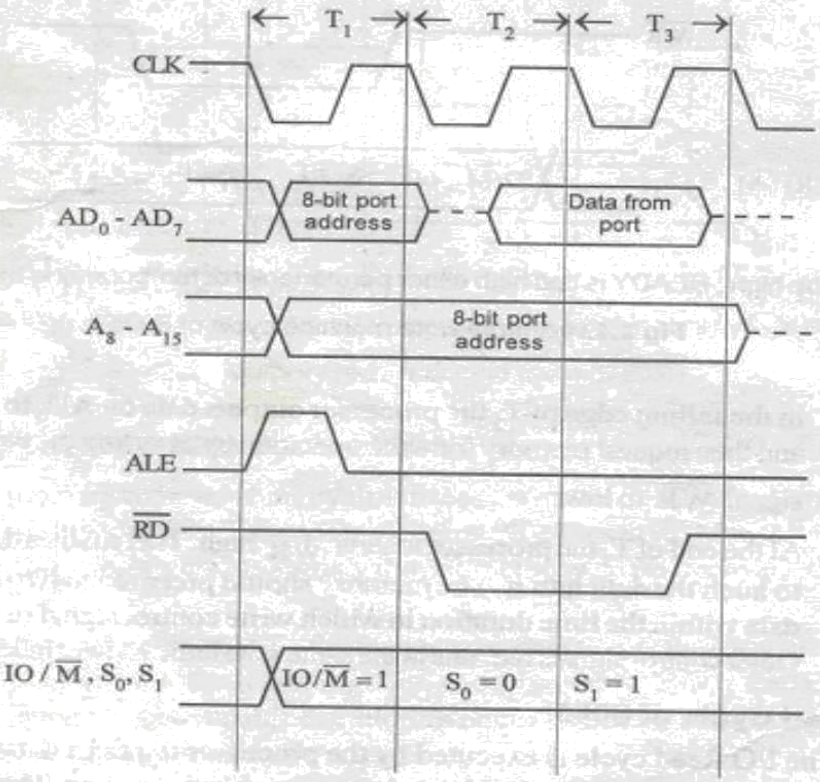
($\overline{RD}$ will be high ; READY is tied high either permanently or temporarily in the system.)

Fig 2.4 : Memory write machine cycle of 8085

1. At the falling edge of T_1, the microprocessor outputs the low byte address on AD_0 - AD_7 lines and high byte address on A_8 to A_{15} lines. ALE is asserted high to enable the address latch. The other control signals are asserted as follows.
   $IO / \overline{M} = 0, S_0 = 1, S_1 = 0.$
2. At the middle of T_1, the ALE is asserted low and this enables the D latch for latching the low byte address into its output lines.
3. In the falling edge of T_2 the processor outputs data on AD_0 to AD_7 lines and then request memory for write operation by asserting the write control signal $\overline{WR}$ to low.
4. At the end of T_3 the processor asserts $\overline{WR}$ high. This enables the memory to latch the data into it. The memory should prepare itself to accept the data within the time duration in which write control signal remains low. Other control signals remains in the same state until the next 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 timings of various signals during this machine cycle are shown in fig 2.5.



($\overline{WR}$ will be high ; READY is tied high either permanently or temporarily in the system.)

Fig 2.5 : I/O read machine cycle of 8085

1. At the falling edge of T_1, the microprocessor outputs the 8 bit port address on both the low order address lines (AD_0 - AD_7) and high order address lines (A_8 to A_{15}). ALE is asserted high to enable the address latch. The other control signals are asserted as follows.

   $IO/\overline{M} = 1$, $S_0 = 0$ and $S_1 = 1$. ($IO/\overline{M}$ is asserted high to indicate I/O read operation).

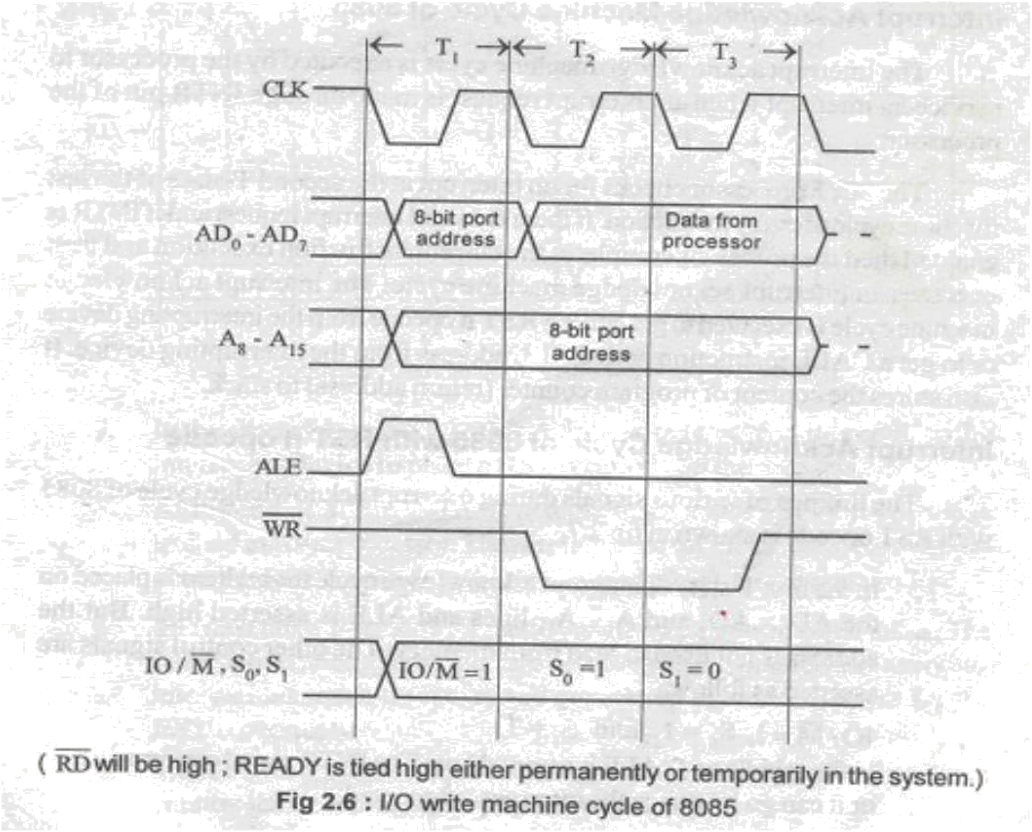
2. At the middle of T_1, the ALE is asserted low and this enables the latch to take the port address and keep on its output lines.

3. In the second T-state (T_2) the I/O device is requested for read by asserting read line low. When read is asserted low, the I/O port is enabled for placing the data on the data bus. The time allowed for I/O port to output the data is the time during which read remains low.

4. At the end of T_3, the read signal is asserted high. On the rising edge of read signal the data is latched into microprocessor. Other control signals remains in the same state until the next 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. The timings of the various signals of I/O write cycle is shown in fig 2.6.



($\overline{RD}$ will be high ; READY is tied high either permanently or temporarily in the system.)

Fig 2.6 : I/O write machine cycle of 8085

1. At the falling edge of T_1, the microprocessor outputs the 8 bit port address on both the low order address lines (AD_0 - AD_7) and high order address lines (A_8 to A_{15}). ALE is asserted high to enable the address latch. The other control signals are asserted as follows.

   $IO/\overline{M} = 1$, $S_0 = 1$ and $S_1 = 0$. ($IO/\overline{M}$ is asserted high to indicate I/O read operation).

2. At the middle of T_1, the ALE is asserted low and this enables the D-latch for latching the port address into its output lines.

3. In the falling edge of T_2 the processor outputs data on AD_0 - AD_7 lines and then request I/O port for write operation by asserting the write control signal $\overline{WR}$ to low.

4. At the end of T_3 the processor asserts $\overline{WR}$ high. This enables the I/O port to latch the data into it. The I/O port should prepare itself to accept the data within the time duration in which write control signal remains low. Other control signals remains in the same state until the next machine cycle.