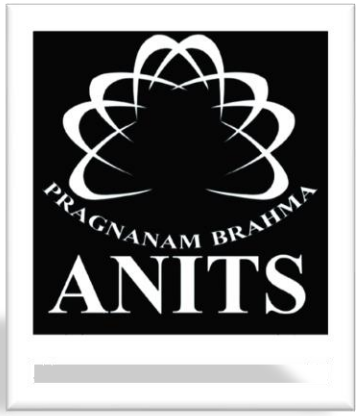
# Course Name: Micro Processors and Micro Controllers

## Topic Name: 8085 Microprocessor




Dr. Appalabathula Venkatesh M.Tech, Ph.D
Assistant Professor,
Department of EEE, ANITS

# Cos

Course Objectives:

At the end of the course, students will be able to

- **CO1** (BL-2) Describe the architecture and various addressing modes of a typical 8085 microprocessor
- **CO2** (BL-4) Classify different memory devices to Discuss the interfacing between memory and 8085 microprocessor
- **CO3** (BL-3) Describe the architecture of a typical 8086 microprocessor to illustrate the general bus operations
- **CO4** (BL-3) Describe the various peripheral devices and show how the peripherals (8259,8251 & 8253) are interfaced with Microprocessor.
- **CO5** (BL-4) Use the architecture of 8051 microcontroller and illustrate how 8051 is interfaced with advanced applications.

# Syllabus

**UNIT 1**

**8085 MICROPROCESSOR:**

Introduction and evolution of microprocessors, Internal architecture and functional pin description of 8085, Addressing modes, Types of instruction and instruction set, Timing diagrams, Types of interrupt.

Dr. A. Venkatesh

# Introduction and evolution of microprocessors

What is Microprocessor?

Microprocessor is an integrated electronic circuit that serves as a central processing unit (CPU) of a computer or other digital devices that performs arithmetic and logic operations which controls the input and output devices by executing the instructions from the memory.

Dr. A. Venkatesh

The microprocessor is often considered the "brain" of the computer because it handles all the computational tasks and manages the flow of data within the system.

# Applications of microprocessors

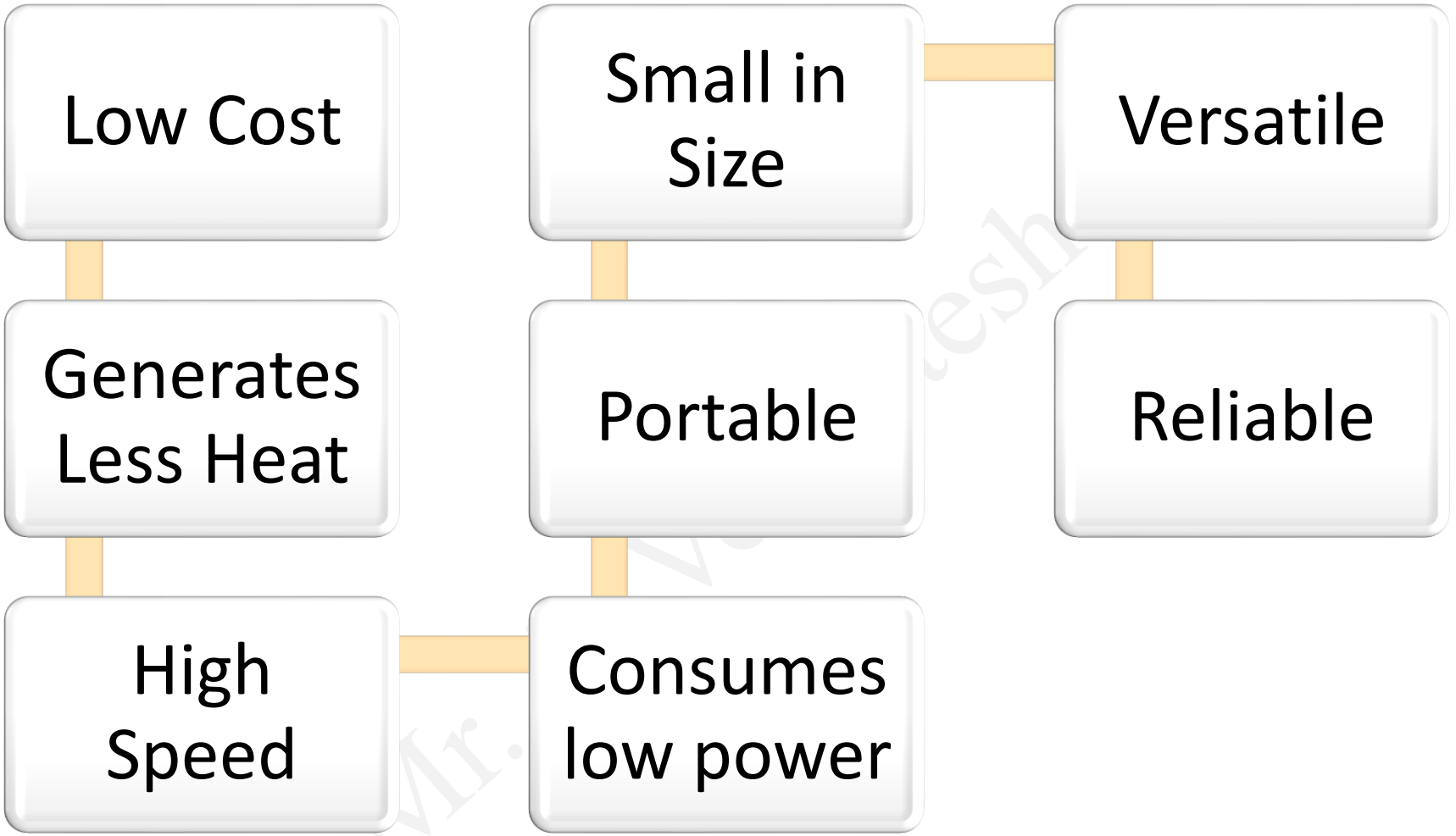
- ✓ **Computers and Laptops:**
- ✓ **Servers:** High-performance microprocessors power servers used for web hosting, cloud computing, and data centers.
- ✓ **Smartphones and Tablets:**
- ✓ **Embedded Systems:** automotive control systems, medical devices, industrial automation, home appliances, and consumer electronics.
- ✓ **Gaming Consoles:** PlayStation, Xbox, and Nintendo Switch use powerful microprocessors
- ✓ **Networking Equipment:** Routers, switches, and network appliances employ microprocessors
- ✓ **Telecommunications:** Including cell towers and base stations, uses microprocessors for call routing, signal processing, and network management.
- ✓ **Digital Cameras:** handle image processing, autofocus, exposure control, and other functions that contribute to high-quality photography.
- ✓ **Home Automation:** Smart home devices, such as thermostats, security cameras, and voice assistants, use microprocessors
- ✓ **Wearable Devices:** Fitness trackers, smartwatches, and health monitoring devices

- ✓ **Automotive Systems:** various systems, including engine control units (ECUs), infotainment systems, and advanced driver-assistance systems (ADAS).
- ✓ **Medical Devices:** Medical equipment, such as MRI machines, patient monitors, and insulin pumps, relies on microprocessors
- ✓ **Aerospace and Defense:** Avionics, navigation systems, missile guidance, radar systems, and military equipment for real-time data processing and control.
- ✓ **Consumer Electronics:** TVs, DVD players, home audio systems, and microwave ovens.
- ✓ **Robotics:** To control motors, sensors, and actuators, enabling precise movements and task execution.
- ✓ **Instrumentation and Test Equipment:** Scientific instruments and test equipment leverage microprocessors for data acquisition, analysis, and control.
- ✓ **Gaming Machines:** Slot machines, arcade games, and gaming kiosks rely on microprocessors to provide entertainment and manage game logic.
- ✓ **Point-of-Sale (POS) Systems:** Retail and restaurant POS systems use microprocessors to process transactions, manage inventory, and generate receipts.
- ✓ **Educational Devices:** Educational toys, calculators, and learning aids incorporate microprocessors to provide interactive learning experiences.
- ✓ **Traffic Control Systems:** Traffic lights, toll collection systems, and smart transportation infrastructure for traffic management and control

# Processor vs Micro-Processor

| Processor | Micro-processor |
| --- | --- |
| • A processor, often referred to as a **central processing unit (CPU),** is the primary hardware component responsible for executing instructions and performing **arithmetic and logic operations** in a computer.<br><br>• **General-purpose CPUs** used in **desktop and laptop** computers to **Specialized processors** used in **servers, gaming consoles, and embedded systems.**<br><br>• **Modern processors** are highly complex and can execute a **wide range of instructions at high speeds.**<br><br>• Processors are typically designed to execute a broad spectrum of applications and tasks, including **running operating systems, executing software programs, and managing system resources.** | • A microprocessor is a specific type of processor that is designed to be **small, compact, and integrated into a single chip (integrated circuit).**<br><br>• Microprocessors are commonly used in **embedded systems**, such as **home appliances, automotive control systems, digital cameras**, and countless other devices where computational capabilities are required.<br><br>• Unlike general-purpose CPUs, microprocessors are often designed for specific tasks or applications. For example, a microprocessor in a **microwave oven** may be dedicated to **controlling cooking functions**, while a microprocessor in a **digital camera** manages **image processing**.<br><br>• Microprocessors are typically **optimized for low power consumption and may not have the same level of computational power as high-end desktop CPUs**.<br><br>• They are often **paired with other hardware components to create a complete system**, such as microcontrollers, which include a microprocessor, memory, and input/output interfaces on a single chip |

# Features of microprocessors



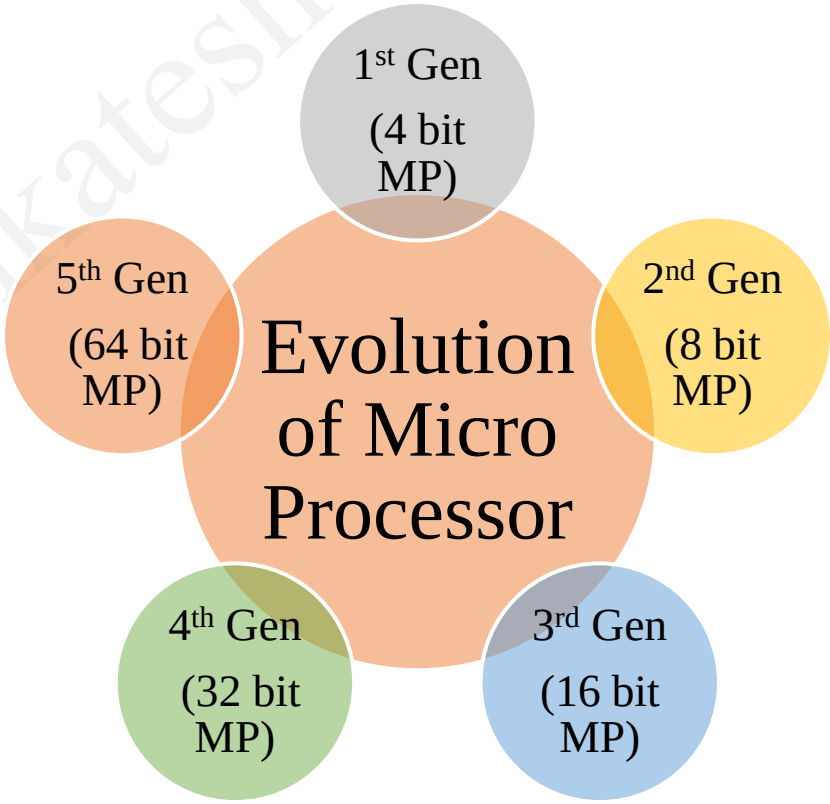
# Introduction and evolution of microprocessors

## First Generation (4 - bit Microprocessors)

- **1971-1972** by Intel Corporation. It was named **Intel 4004**
- It was a processor on a single chip. It could perform simple arithmetic and logical operations such as addition, subtraction, Boolean OR and Boolean AND.
- It had a control unit capable of performing control functions like fetching an instruction from storage memory, decoding it, and then generating control pulses to execute it.

## Second Generation (8 - bit Microprocessor)

- **1973** again by Intel.
- It could perform arithmetic and logic operations on 8-bit words. It was Intel 8008, and another improved version was Intel 8088.




Dr. A. Venkatesh

# Introduction and evolution of microprocessors

## Third Generation (16 - bit Microprocessor)

- Introduced in **1978** were represented by **Intel's 8086, Zilog Z800 and 80286**
- Performance like minicomputers.

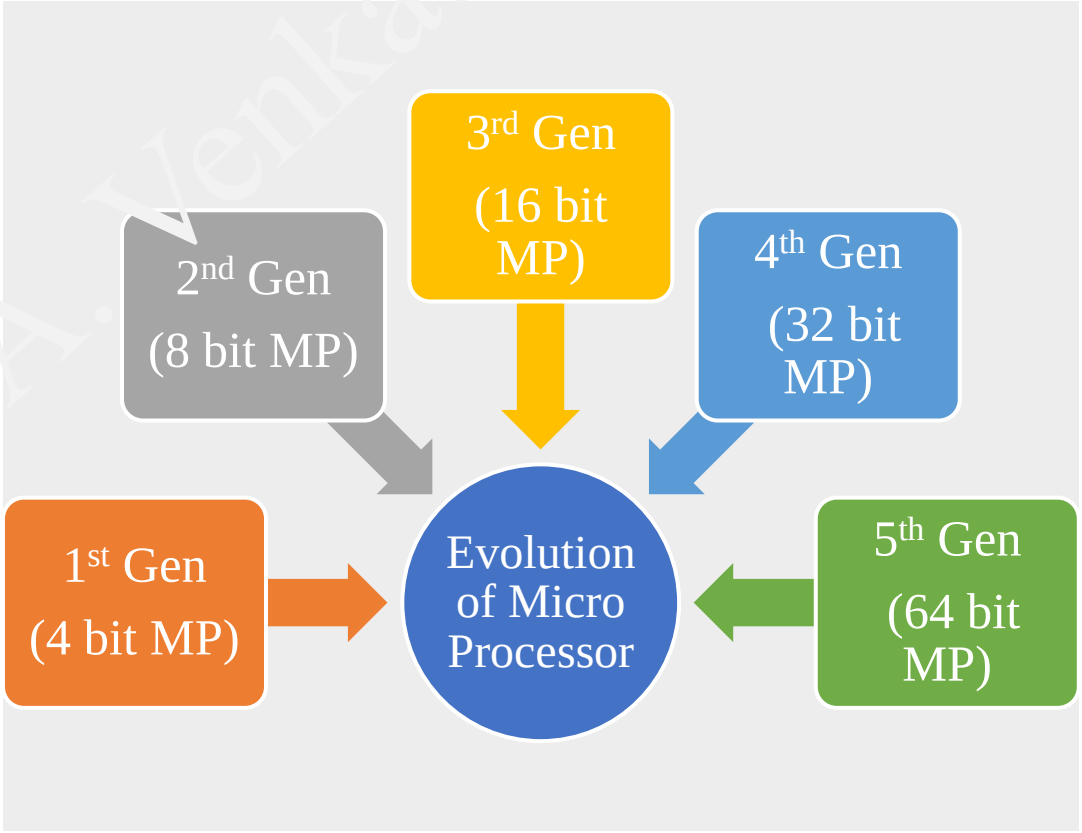
## Fourth Generation (32 - bit Microprocessors)

- Several different companies introduced the 32-bit microprocessors, but the most popular one is the **Intel 80386**.

## Fifth Generation (64 - bit Microprocessors)

- From **1995 to Now**.
- After 80856, Intel came out with a new processor namely Pentium processor followed by **Pentium Pro CPU**, which allows multiple CPUs in a single system to achieve multiprocessing.

Other improved 64-bit processors are **Celeron, Dual, Quad, Octa Core processors**.




Dr. A. Venkatesh

# Introduction and evolution of microprocessors

| Microprocessor | Year of Invention | Word Length | Memory addressing Capacity | Pins | Clock | Remarks |
|---|---|---|---|---|---|---|
| 4004 | 1971 | 4-bit | 1 KB | 16 | 750 KHz | First Microprocessor |
| 8085 | 1976 | 8-bit | 64 KB | 40 | 3-6 MHz | Popular 8-bit Microprocessor |
| 8086 | 1978 | 16-bit | 1MB | 40 | 5-8 MHz | Widely used in PC/XT |
| 80286 | 1982 | 16-bit | 16MB real, 4 GB virtual | 68 | 6-12.5 MHz | Widely used in PC/AT |
| 80386 | 1985 | 32-bit | 4GB real, 64TB virtual | 132 14X14 PGA | 20-33 MHz | Contains MMU on chip |
| 80486 | 1989 | 32-bit | 4GB real, 64TB virtual | 168 17X17 PGA | 25-100 MHz | Contains MMU, cache and FPU, 1.2 million transistors |
| Pentium | 1993 | 32-bit | 4GB real,32-bit address,64-bit data bus | 237 PGA | 60-200 | Contains 2 ALUs,2 Caches, FPU, 3.3 Million transistors, 3.3 V, 7.5 million transistors |
| Pentium Pro | 1995 | 32-bit | 64GB real, 36-bit address bus | 387 PGA | 150-200 MHz | It is a data flow processor. It contains second level cache also,3.3 V |
| Pentium II | 1997 | 32-bit | - | - | 233-400 MHz | All features Pentium pro plus MMX technology,3.3 V, 7.5 million transistors |
| Pentium III | 1999 | 32-bit | 64GB | 370 PGA | 600-1.3 MHz | Improved version of Pentium II; 70 new SIMD instructions |
| Pentium 4 | 2000 | 32-bit | 64GB | 423 PGA | 600-1.3 GHz | Improved version of Pentium III |
| Itanium | 2001 | 64-bit | 64 address lines | 423 PGA | 733 MHz-1.3 GHz | 64-bit EPIC Processor |

Dr. A. Venkatesh

- **PGA** - Pin Grid Array
- **MMX** - MultiMedia eXtensions
- **EPIC** - Explicitly Parallel Instruction Computing

- **SIMD** - Single Instruction Multiple Data
- **ALU** - Arithmetic and Logic Unit
- **MMU** - Memory Management Unit
- **FPU** - Floating Point Unit

# Working of microprocessor

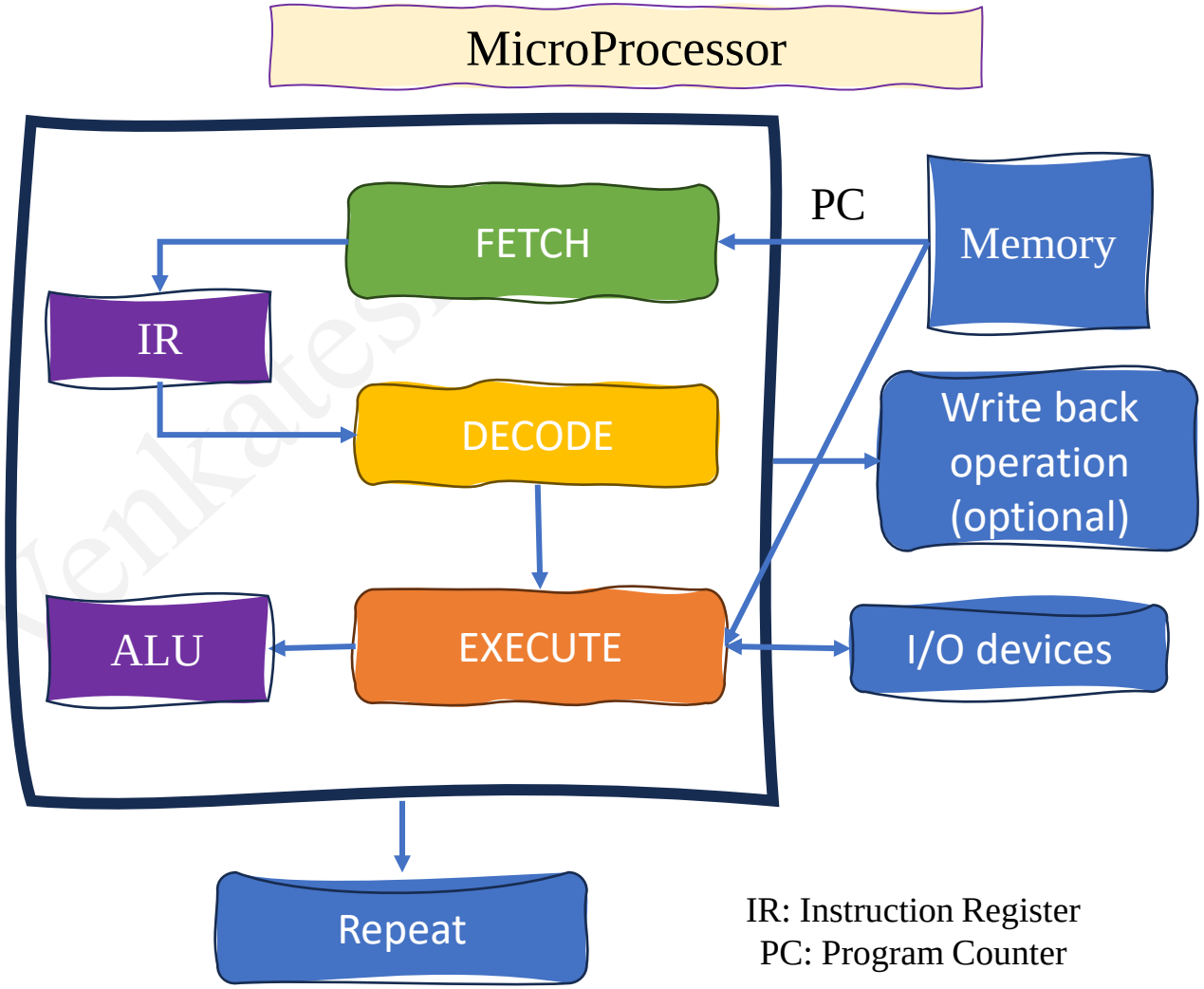
Working of a Microprocessor

There are three steps that a microprocessor follows –

1. **Fetch** – The instructions are in storage from where the processor fetches them.

2. **Decode** – It then decodes the instruction to assign the task further. During this, the arithmetic and logic unit also performs to register the data temporarily.

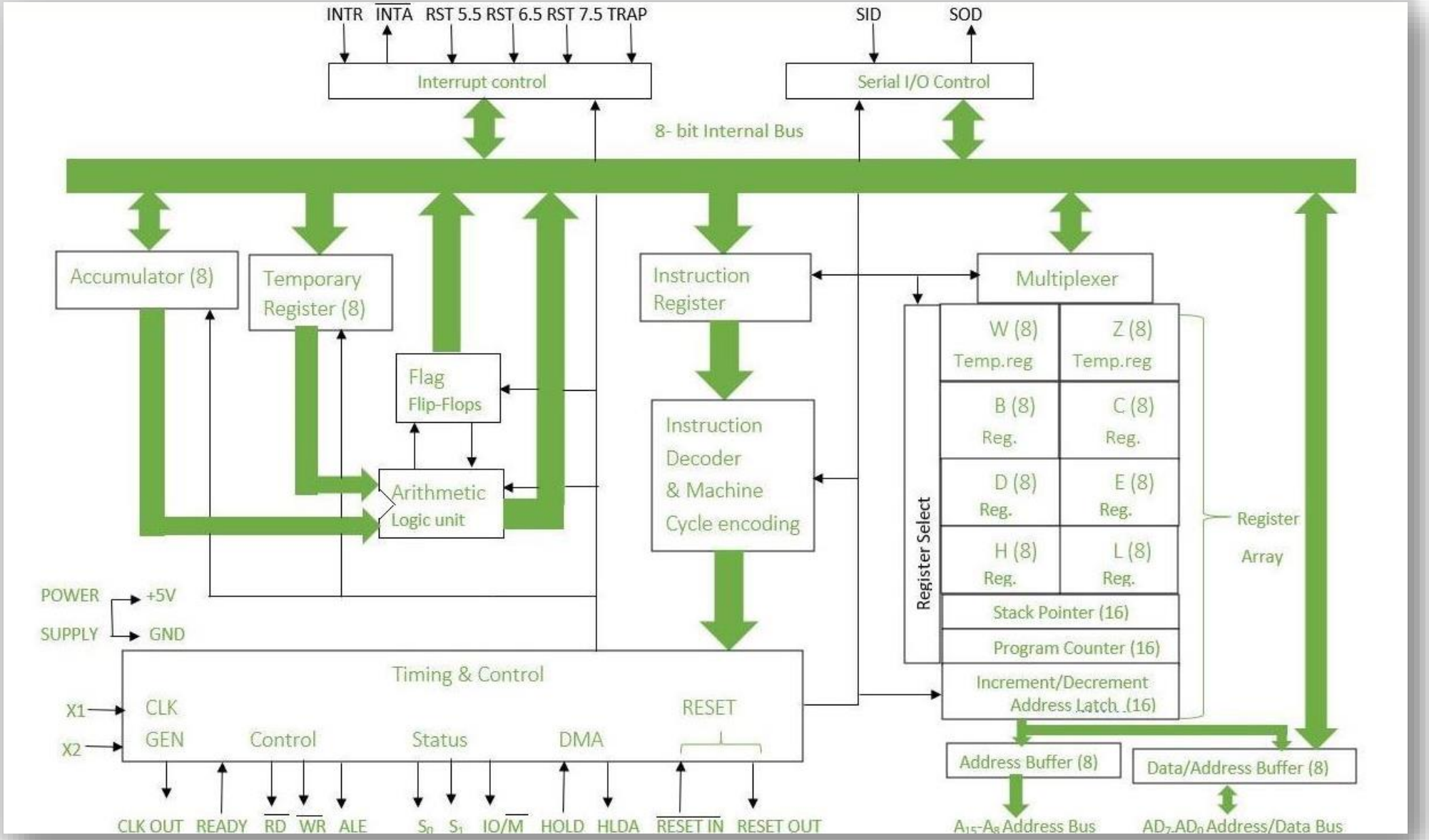
3. **Execute** – The assigned tasks undergo execution and reach the output port in binary form.




IR: Instruction Register
PC: Program Counter

Dr. Venkatesh

# Internal Architecture of 8085 microprocessor



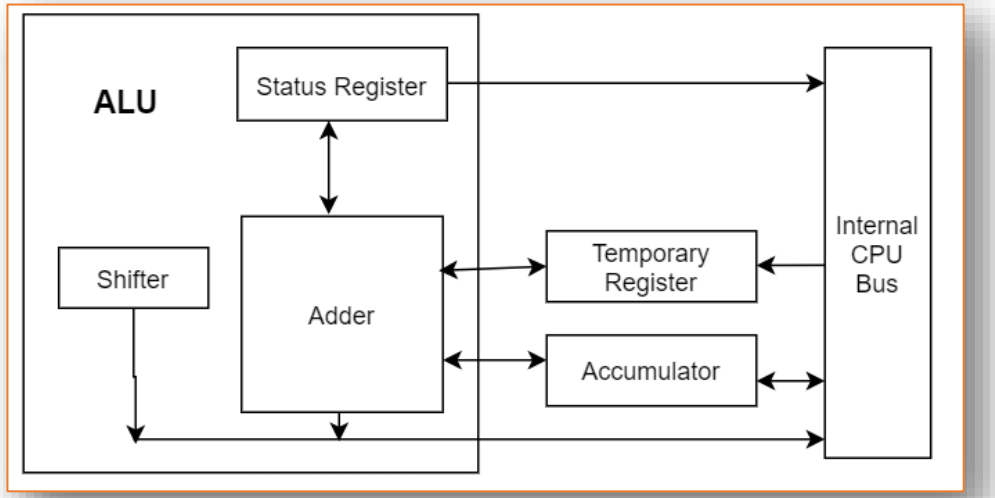
# Internal Architecture of 8085 microprocessor

**Functional blocks in internal architecture of 8085**

1. **Arithmetic and Logic Unit (ALU):** The ALU performs arithmetic and logical operations such as addition, subtraction, AND, OR, XOR, and more. It operates on 8-bit data at a time.
2. **Registers:**
   - Accumulator (A): The primary accumulator for arithmetic and logic operations.
   - B and C registers: Used for data storage and transfer operations.
   - D and E registers: Also used for data storage and transfer.
   - H and L registers: Often used as a pair for indirect addressing and for 16-bit operations.
   - Stack Pointer (SP): Used to manage the stack in memory.
   - Program Counter (PC): Keeps track of the address of the next instruction to be executed.
3. **Flags Register (F):** The flags register contains various status flags, including the Sign (S), Zero (Z), Auxiliary Carry (AC), Parity (P), Carry (CY), and Overflow (V) flags, which are set or cleared based on the result of arithmetic and logical operations.
4. **Instruction Decoder&Control Unit:** This unit fetches and decodes instructions, controlling the flow of data within the microprocessor.
5. **Timing&Control Unit:**It generates control signals for various operations and synchronizes the internal operations of the microprocessor.
6. **Interrupt Control Unit:** Manages interrupts from external devices and prioritizes interrupt requests.
7. **Series I/O control unit:**

Dr A Venkatesh

# 1. ALU




It is multi operational combinational logic circuit, same as IC 74181(ALU).It performs arithmetic and logical operations like ANDing, ORing, EX-ORing, ADDITON, SUBTRACTION, etc.

- It is not accessible by user.
- The word length of ALU depends upon of an internal data bus.
- IT is 8 bit. It is always controlled by timing and control circuits.
- It provides status or result of flag register.
- Figure shown below shows some functional sections of ALU:

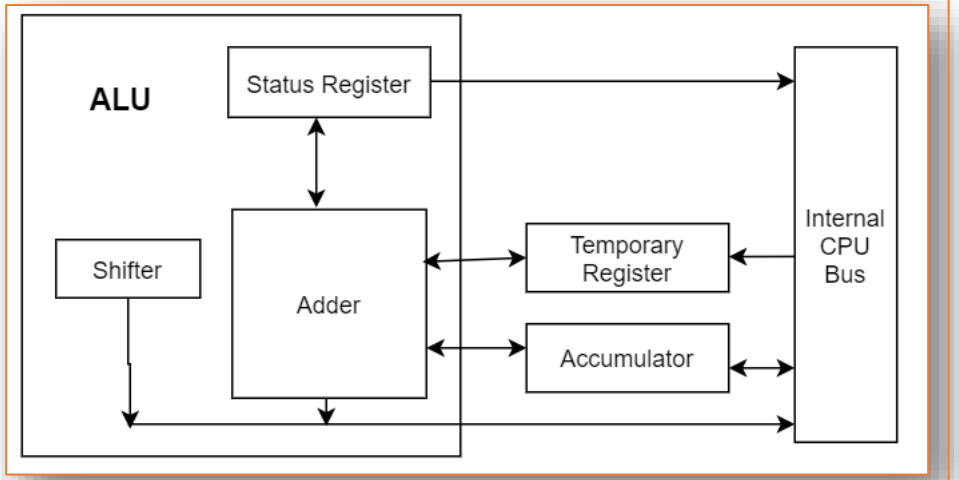
**Adder:** It performs arithmetic operations like addition, subtraction, increment, decrement, etc. The result of operation is stored into accumulator.

**Shifter:** It performs logical operations like rotate left, rotate right, etc. The result of operation is again stored into accumulator.

**Status Register:** Also known as flag register. It contains a no. of flags either to indicate conditions arising after last ALU operation or to control certain operations.

Dr. A. Venkatesh

# 2. Registers

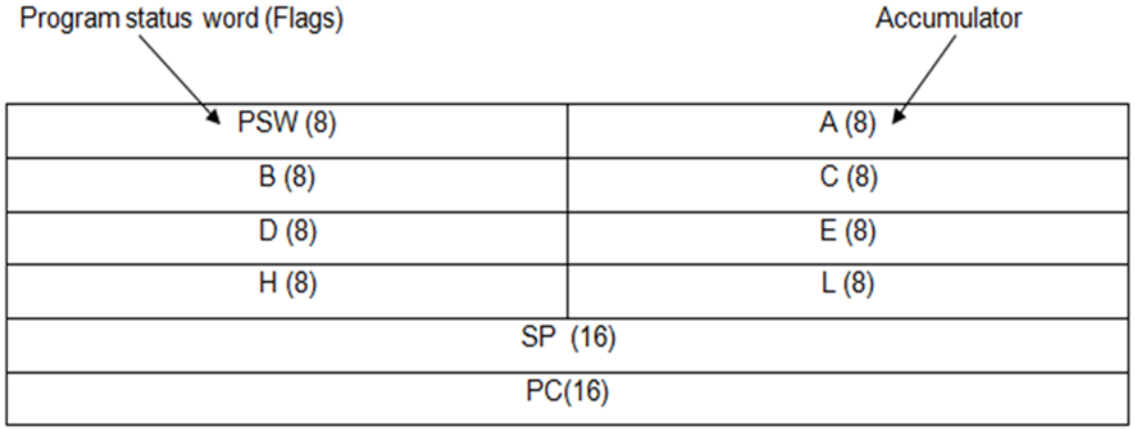



## Accumulator:

- It is one of the general purpose register of microprocessor also called as A register.
- The accumulator is an 8-bit register that is a part of arithmetic/logic unit (ALU).
- This register is used to store 8-bit data and to perform arithmetic&logical operations.
- The result of an operation is stored in the accumulator.
- The user can access this register by giving appropriate instructions (commands).

## Temporary Register:

- It is also called as operand register (8 bit).
- It provides operands to ALU. ALU can store immediate result in temporary register.
- It is not accessible by user.




Dr. Venkatesh

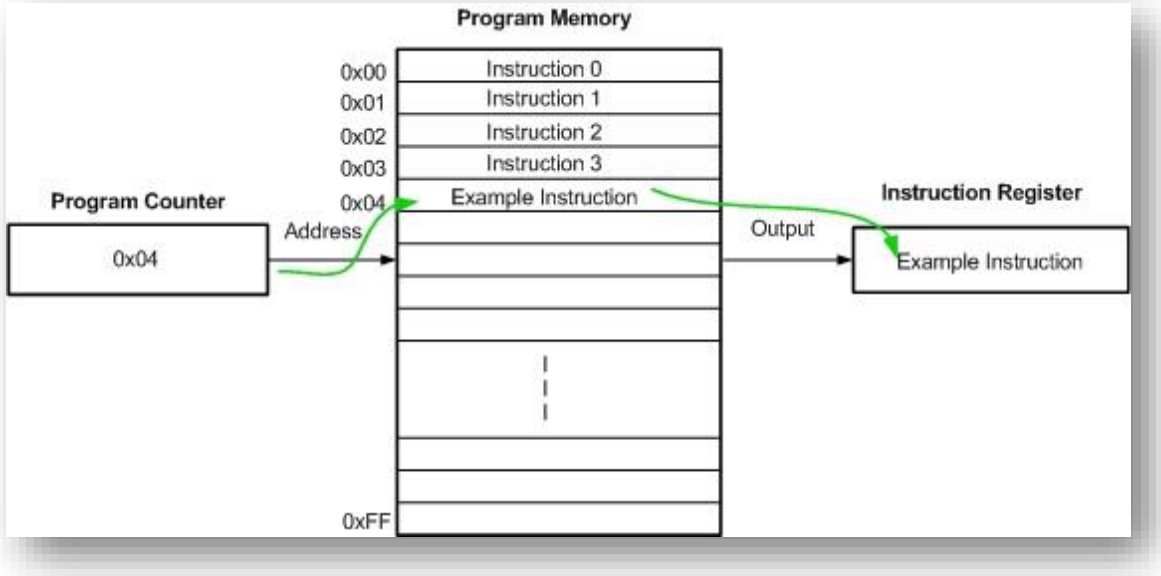
## General Purpose Register

- **Apart from accumulator** 8085 consists of **6 special types of registers** called **General Purpose Registers.**
- These general purpose registers are used to hold data like any other registers.
- The **general purpose registers** in 8085 processors are **B, C, D, E, H and L**. Each register can hold 8-bit data.
- Apart from the above function these registers can also be used to work in pairs to hold 16-bit data.
- They can work in **pairs** such as **B-C, D-E and H-L to store 16-bit data**.
- The **H-L pair(Active Register pair)** and data in HL pair presents in **memory pointer**.
- They can store 16-bit address as they work in pair.

Dr. A. Venkatesh

## Special Purpose Register
## i. Program counter:

- Program counter is a special purpose register.
- Consider that an instruction is being executed by processor. As soon as the ALU finished executing the instruction, the processor looks for the next instruction to be executed.
- So, there is a necessity for holding the address of the next instruction to be executed in order to save time. This is taken care by the program counter.
- A **program counter** stores the **address of the next instruction** to be executed.
- Microprocessor increments the program whenever an instruction is being executed, so that the program counter points to the memory address of the next instruction that is going to be executed. **Program counter is a 16-bit register**.

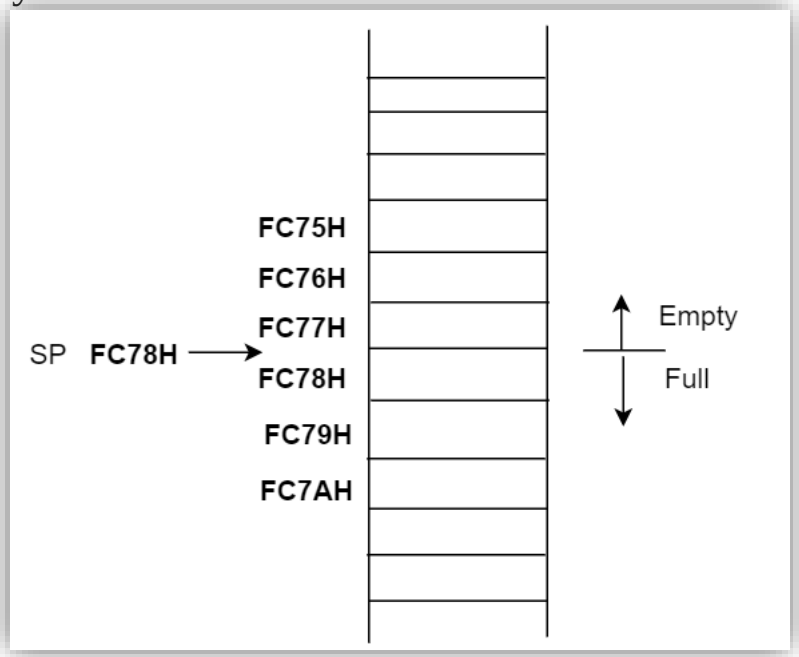



Dr. A. Venkatesh

## Special Purpose Register
## ii. Stack pointer

- Stack pointer is also a 16-bit register which is used as a memory pointer.
- A stack is nothing but the portion of RAM (Random access memory).
- Stack pointer maintains the **address of the last byte that is entered into stack**.
- Each time when the data is loaded into stack, Stack pointer gets decremented.
- Conversely it is incremented when data is retrieved from stack

- The memory locations FC78H, FC79H, FC80H…… FFFFH holds useful data and these memory locations are considered as filled locations.




- Whereas FC77H, FC76H….0000H does not have any data and those are considered empty locations

  - So, SP contents indicate the uppermost locations in the stack

Dr. A. Venkatesh

## Special Purpose Register
## iii. Increment/decrement register

- The 8-bit contents of a register or a memory location can be incremented or decremented by 1.
- This 16-bit register is used to increment or decrement the content of program counter and stack pointer register by 1.
- Increment or decrement can be performed on any register or a memory location.

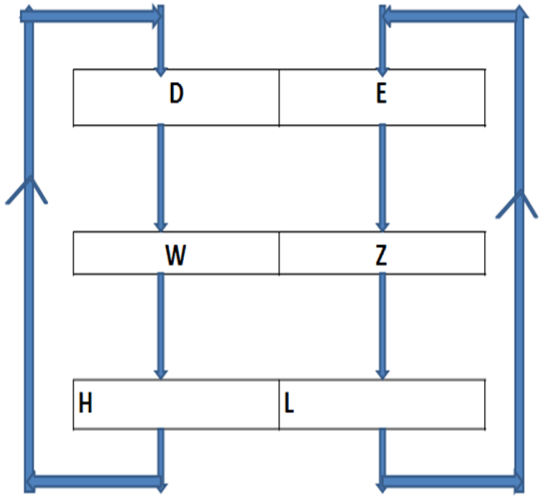
## Special Purpose Register
## iv. Address /data buffer and address buffer

- **The contents of the stack pointer and program counter are loaded into the address buffer and address-data buffer**.
- These buffers are then used to drive the external address bus and address-data bus.
- As the memory and I/O chips are connected to these buses, the CPU can exchange desired data to the memory and I/O chips.
- The address-data buffer is not only connected to the external data bus but also to the internal data bus which consists of 8-bits.
- The address data buffer can both send and receive data from internal data bus.

Dr. A. Venkatesh

## Temporary Register (W and Z)

- ✓ This temporary register can only be accessed by the microprocessor and it is completely inaccessible to programmers.
- ✓ Temporary register is an 8-bit register.
- ✓ This register is used by control systems to hold operand, intermediate operand, and address of memory and I/O devices temporarily.




Dr. A. Venkatesh

# 3. Flag Registers

- Flag register is a **group of flip flops used to give status of different operations result**.
- The flag register is connected to ALU.
- Once an operation is performed by ALU the result is transferred on internal data bus and status of result will be stored in flip flops.
- They are called Zero (Z), Carry (CY), Sign (S), Parity (P), and Auxiliary Carry (AC) flags.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| S | Z | X | AC | X | P | X | CY |

1. **Carry flag(CY):** If an operation performed in ALU generates the carry from D7 to next stage then CY flag is set, else it is reset.
2. **Auxiliary carry(AC):** If an operation performed in ALU generates the carry from lower nibble (D0 to D3) to upper nibble (D4 to D7) AC flag is set, else it resets.
3. **Zero flag(Z):** If an operation performed in ALU results 0 value of entire 8-bits then zero flag is set, else it resets.
4. **Sign flag(S):** If MSB = 0 then the number is positive, else it is negative.
5. **Parity flag(P):** If the result contains even no. of ones this flag is set and for odd no. of ones this flag is reset.

Dr. A. Venkatesh

# 4. Instruction register, decoder and Control Unit:

## Instruction register, decoder :

- Instruction register is 8-bit register just like every other register of microprocessor.
- **The instruction may be anything like adding two data's, moving a data, copying a data etc**.
- When such an instruction is fetched from memory, it is directed to Instruction register. So the instruction registers are specifically to store the instructions that are fetched from memory.
- There is an **Instruction decoder which decodes the information present in the Instruction register for further processing**.

## Control Unit:

- The Control Unit **generates control signals that govern the microprocessor's various components**, such as the arithmetic logic unit (ALU), registers, memory interfaces, and input/output (I/O) operations.
- The Control Unit also manages the fetching of subsequent instructions by updating the program counter (PC) or instruction pointer.
- It handles control flow operations, such as branching and conditional execution, based on the results of previous instructions.

## Instruction register, decoder and Control Unit:

Together, the Instruction Decoder and Control Unit work in tandem to execute instructions in a microprocessor. The **Instruction Decoder interprets the specifics of each instruction**, while the **Control Unit manages the overall operation and coordination of various microprocessor components to ensure that instructions are executed accurately and efficiently**

Dr. A. Venkatesh

# 5. Timing and Control Unit:

**Timing and Control Unit**

- Timing and control unit is a very important unit as **it synchronizes the registers and flow of data through various registers and other units.**
- This unit consists of an oscillator and controller sequencer which sends **control signals needed for internal and external control of data and other units**.
- The oscillator generates two-phase clock signals which aids in synchronizing all the registers of 8085 microprocessor.
- Signals that are **associated with Timing and control unit are:**

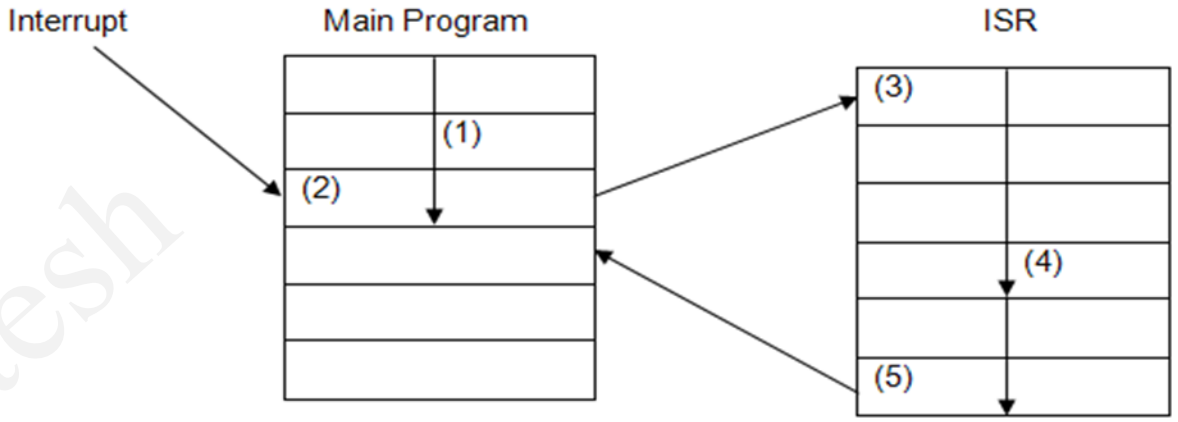
1) **Control Signals:** READY, RD(bar), WR(bar), ALE(Address Latch Enable)
2) **Status Signals:** S0, S1, IO/M(bar)
3) **DMA Signals:** HOLD, HLDA
4) **RESET Signals:** RESET IN(bar), RESET OUT.

Dr. A. Venkatesh

# 6. Interrupt Control Unit

## Interrupt Control Section

- As the name suggests this control **interrupts a process**.
- Consider that a microprocessor is executing the main program. Now whenever the interrupt signal is enabled or requested the microprocessor shifts the control from main program to process the incoming request and after the completion of request, the control goes back to the main program.
- For example an Input/output device may send an interrupt signal to notify that the data is ready for input.
- The microprocessor temporarily stops the execution of main program and transfers control to specific special routine known as "**Interrupt Service Routine"(ISR).**
- After ISR control is transferred back to main program.
- Figure below shows the idea of communication between microprocessor and device with the help of interrupt




Interrupt signals present in 8085 are:
1. INTR
2. RST 7.5
3. RST 6.5
4. RST 5.5
5. TRAP

Dr. A. Venkatesh

## Interrupt Control Section

- Of the above four interrupts **TRAP** is a **Non-maskable** interrupt control and other **three** are **maskable** interrupts.
- A **non-maskable** interrupt is an interrupt which is given the **highest priority** in the order of interrupts.
- Suppose you want an instruction to be processed immediately, then you can give the instruction as a non-maskable interrupt.
- Further the non-maskable interrupt cannot be disabled by programmer at any point of time.
- Whereas the maskable interrupts can be disabled and enabled using EI and DI instructions.
- Among the **maskable** interrupts **RST 7.5** is given the **highest priority** above **RST 6.5** and **least priority** is given to **INTR.**

Dr. A. Venkatesh

# 7. Serial I/O Control Section

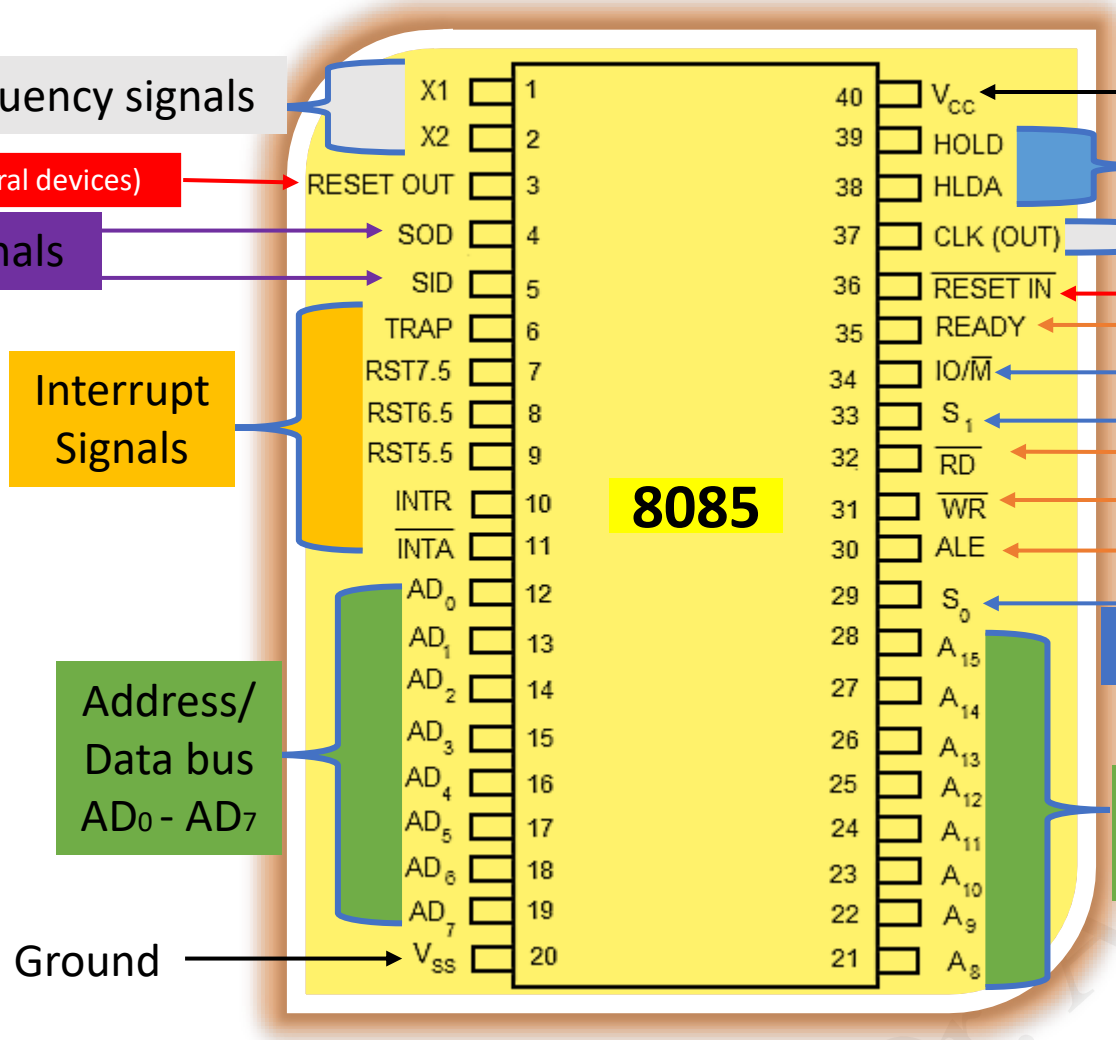
**Serial I/O control section:**

- The input and output of serial data can be carried out using 2 instructions in 8085:
  1. SID-Serial Input Data
  2. SOD-Serial Output Data

- Two more instructions are used to perform serial-parallel conversion needed for serial I/O devices.

Dr. A. Venkatesh

# Pin description of 8085 microprocessor




Input Clock frequency signals

Reset Signal(Peripheral devices)

Serial I/O signals

Interrupt Signals

Address/ Data bus $AD_0$ - $AD_7$

Ground

Power Supply

Direct Memory Access Signals

Output Clock frequency signal

Reset Signal

Control Signals RD,WR, ALE, READY

Status Signals $S_0$,$S_1$, IO/M'

Address bus $A_8$ - $A_{15}$

| IO/ M' | S1 | S0 | Data Bus Status |
|---|---|---|---|
| 0 | 1 | 1 | Opcode fetch |
| 0 | 1 | 0 | Memory read |
| 0 | 0 | 1 | Memory write |
| 1 | 1 | 0 | I/O read |
| 1 | 0 | 1 | I/O write |
| 1 | 1 | 1 | Interrupt acknowledge |
| 0 | 0 | 0 | Halt |

Dr. Venkatesh

# Internal architecture of 8085 microprocessor

**d8253c-2 Timer/Counter** Generating-precise timing intervals and interrupt signals

26-pin FRC (Flexible Ribbon Cable)-- Address

26-pin FRC DATA

universal synchronous/asynchronous receiver-transmitter **(USART)**--serial communication in embedded systems, including communication with peripherals, external sensors, and other microcontroller-based devices

256K x 4-bit **DRAM** chip,

**KC8279P Programmable-keyboard chip**

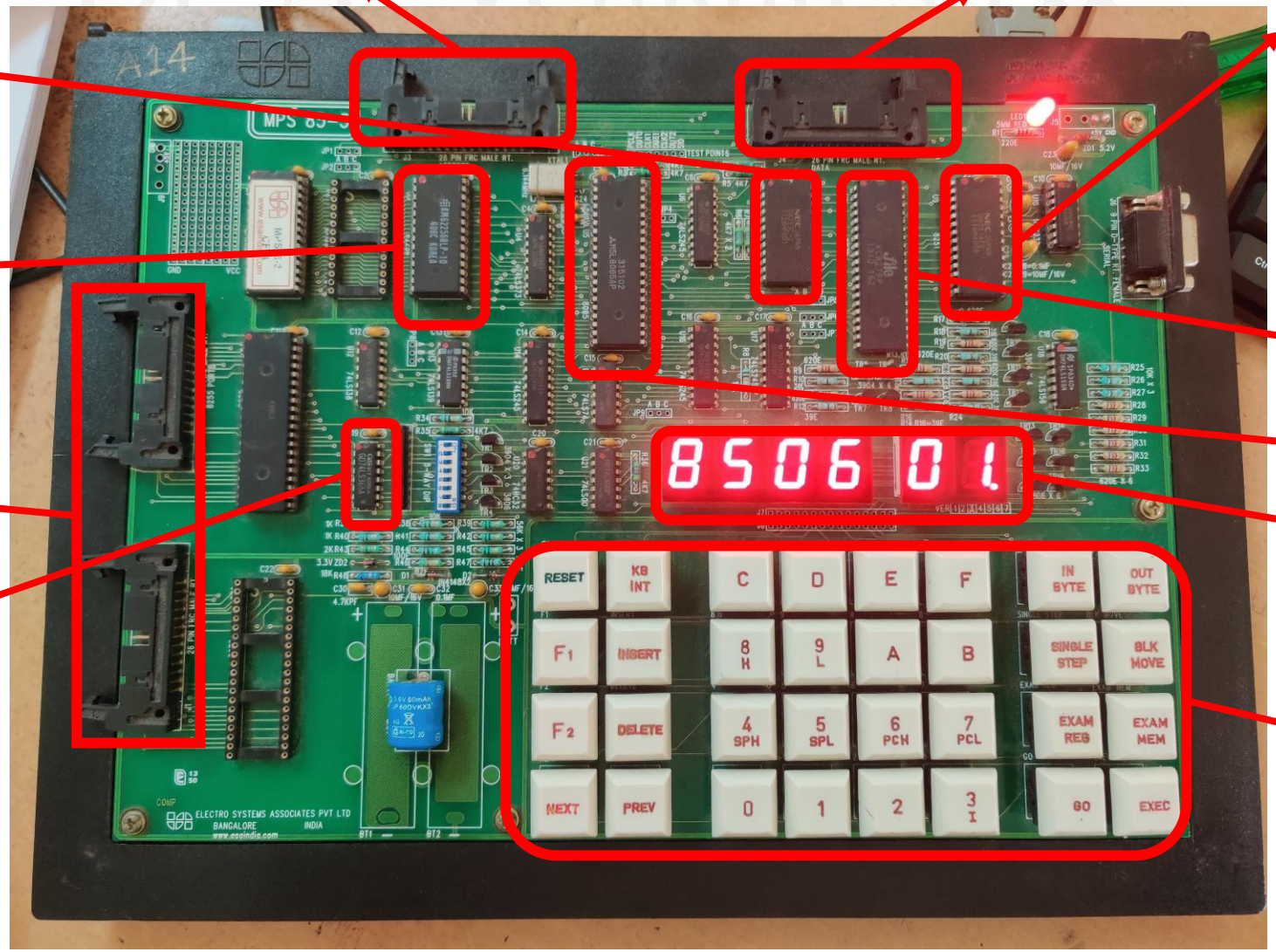
8255 PPI(Programmable peripheral interface port)

M5L 8085AP

LCD

**GD74LS365A** hex buffer/line driver **digital signal interfacing and level shifting**

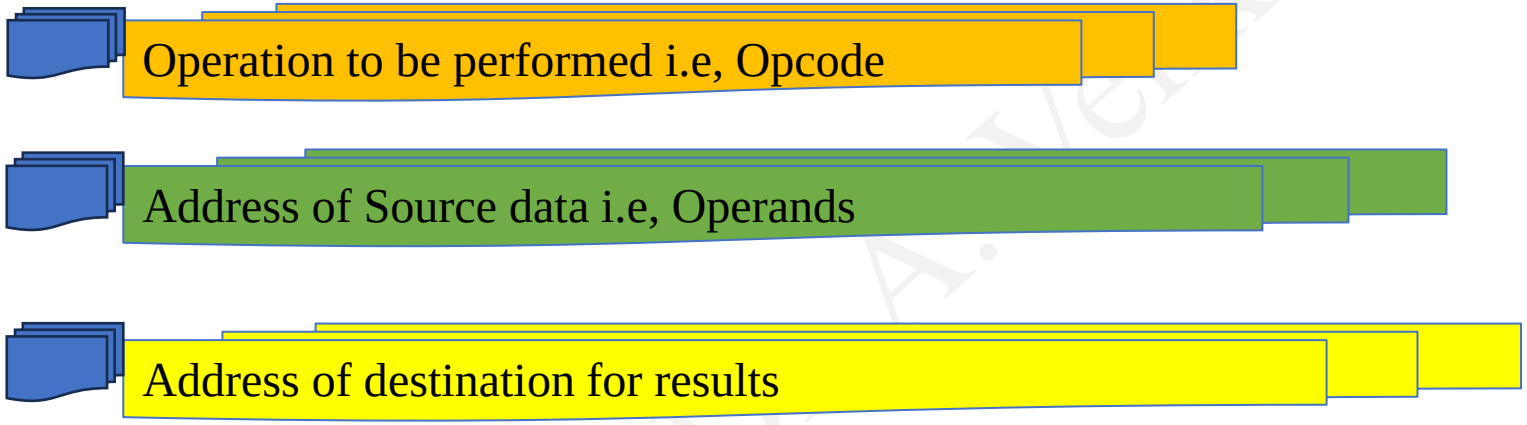
Keyboard



# Addressing modes of 8085 microprocessor

To perform any operation, Instructions are to be given to microprocessor.

Instruction should consists of following features.




The Instructions are to be given to microprocessor, The source and destination operands can be a

- ✓ Register
- ✓ Memory location
- ✓ 8-bit number

Dr. A. Venkatesh

# Addressing modes of 8085 microprocessor

Dr. A. Venkatesh

| Addressing modes of 8085 microprocessor | | |
|---|---|---|
| Immediate Addressing Mode | 1 | The data is specified in the instruction itself. |
| Register Addressing Mode | 2 | The source and destination operands are general-purpose registers |
| Direct Addressing Mode | 3 | The 16-bit address of the operand is given within the instruction itself. |
| Indirect Addressing Mode | 4 | The instruction references the memory through a register pair. |
| Implied Addressing Mode | 5 | It does not require any operand, The data is specified within OPCODE itself. |

# Addressing modes of 8085 microprocessor

**Addressing modes of 8085 microprocessor**

**Immediate Addressing Mode**

- In immediate addressing mode, the data (8 / 16 bit) is specified in the instruction itself.
- The immediate addressing instructions are either 2 bytes or 3 bytes long.
- In 2-byte instruction, the first byte is OPCODE, and the second byte is the 8-bit data.
- In 3-byte instruction, the first byte is OPCODE, second and third bytes are 16-bit data.
- The instruction containing the letter "I" indicate immediate addressing mode.

Dr. A. Venkatesh

**Examples of Immediate Addressing Mode:**

**MVI B, 0A H**: This instruction transfers immediate data (0A H ) to B register.

**LXI H, E101 H**: This instruction transfer 16-bit immediate data E101 to HL register pair. Lower order data(01H) to L register and high order data (E1 H) to H register.

**JMP address**: Jump to operand address immediately.

# Addressing modes of 8085 microprocessor

**Addressing modes of 8085 microprocessor**

**Register Addressing Mode**

- In register addressing mode, the source and destination operands are general-purpose registers.
- The register addressing instructions are generally of 1 byte i.e. OPCODE only.
- The OPCODE specifies the operation and registers to be used to perform the operation.

Dr. A. Venkatesh

**Examples of Register Addressing Mode:**

**MOV D, B:** This instruction copies the contents of register B to the D register. The source and destination operands are both registers.

**ADD B:** This instruction adds the content of the B register and A register, The data is present in both B and A registers. The result is stored in the accumulator.

**PCHL:** This instruction will transfer the content of register pair HL to the PC ( Program Counter).

**INR A:** Increment the contents of the register A by one.

# Addressing modes of 8085 microprocessor

**Addressing modes of 8085 microprocessor**

**Direct Addressing Mode**

- In direct addressing mode, the 16-bit address of the operand is given within the instruction itself.
- The instruction in the direct addressing mode is 3-byte instructions. The first byte is OPCODE, the second slower order address mode, and the third is the higher-order address mode.
- For I/O instruction that uses direct addressing mode is 2-byte as the address if I/O is one byte

Dr. A. Venkatesh

**Examples of Direct Addressing Mode:**

**LDA A201 H**: Load accumulator directly from the memory location. In this instruction, the contents of the A201 memory location are transferred to the accumulator.

**LHLD 82C1 H**: Load contents of 16-bit memory location(82C1) into H-L register pair.

**STA 1B20 H**: Store accumulator directly to memory location. In this instruction, the content of the accumulator is stored at memory location 1B20 H.

# Addressing modes of 8085 microprocessor

**Addressing modes of 8085 microprocessor**

**Indirect Addressing Mode**

- In indirect addressing mode, the instruction references the memory through a register pair.
- i.e. the memory address where the operand is located is specified by the content of a register pair

Dr. A. Venkatesh

**Examples of Indirect Addressing Mode:**

**MOV A, M:** In this case, M is a memory pointer specifying the HL register pair where the address is stored. The contents of the HL pair are used as addresses and the content of that memory location is transferred to the accumulator.

**LDAX B:** In this case, the BC register pair is used as an address and the content of the memory location specified by the BC register pair is copied to the accumulator

# Addressing modes of 8085 microprocessor

| Addressing modes of 8085 microprocessor | • The implied mode of addressing does not require any operand.<br>• The data is specified within OPCODE itself.<br>• Generally, the implied addressing mode instruction is a 1-byte instruction.<br>• The data is supposed to be present generally in the accumulator |
|---|---|
| **Implied or Implicit Addressing Mode** | |

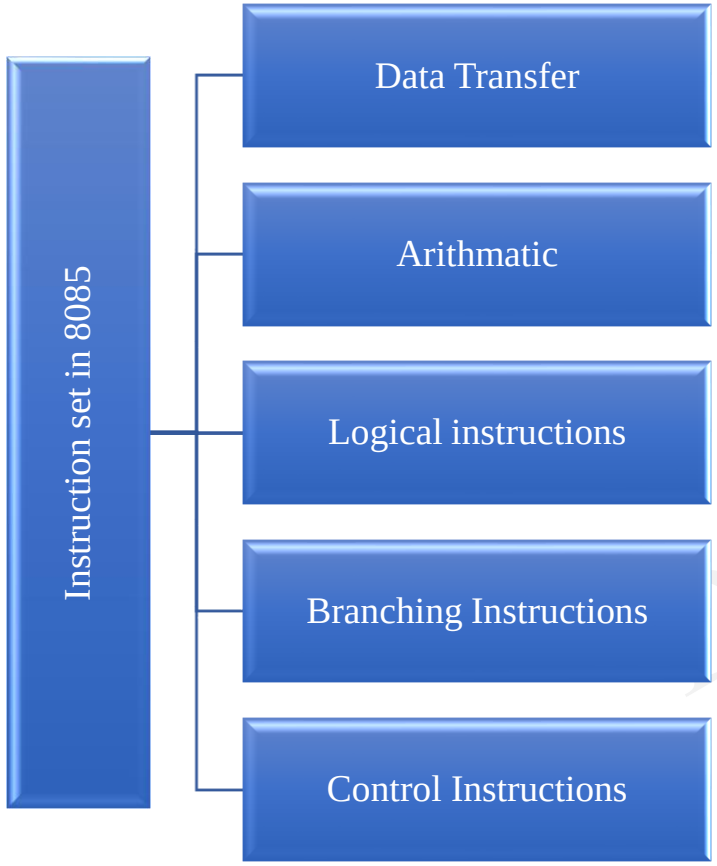
Dr. A. Venkatesh

**Examples of Implicit Addressing Mode:**

- **RAL**: Rotate the accumulator left, it operates on the data in the accumulator only. So whenever RAL is used it is implied that the data to be operated on is available in the accumulator only.
- **CMC**: Complement carry flag
- **CMA**(Complement Accumulator)
- **RRC:** Rotate accumulator A right by one bit
- **RLC:** Rotate accumulator A left by one bit

# Types of instruction and Instruction set in 8085 microprocessor

A binary command which is used to perform a specific task in a microprocessor with the available data is called as “Instruction”. A group or set of instructions collectively called as “Instruction Set”




Dr. A. Venkatesh

# Instruction set Format in 8085 microprocessor

The Instruction Format of 8085 set consists of one, two and three byte instructions.

The first byte is always the opcode; in two-byte instructions the second byte is usually data; in three byte instructions the last two bytes present address or 16-bit data.

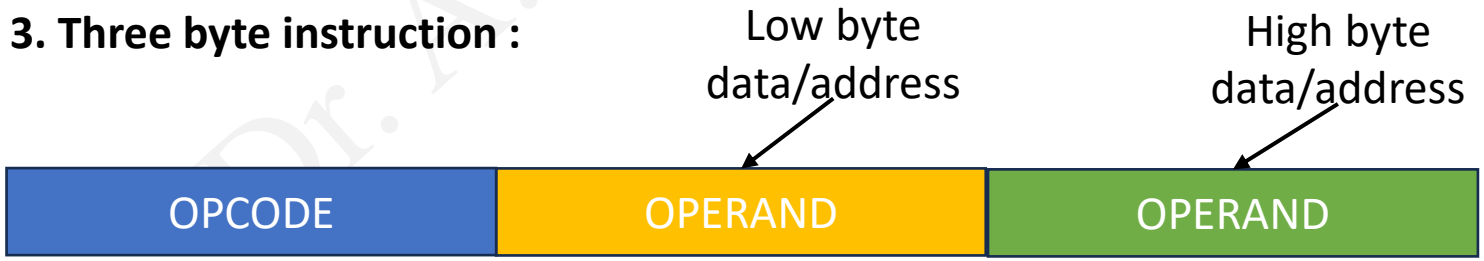
OPCODE → The operation to be performed.

OPERAND→ Data to be operated(The data or memory location used to execute that OPCODE instruction)

**1. One byte instruction :**


**2. Two byte instruction :**


**3. Three byte instruction :**




Dr. A. Venkatesh

# Types of instruction and Instruction set in 8085 microprocessor

**Instruction sets of 8085 microprocessor**

**Data Transfer Instruction**

**Examples of Data Transfer Instructions:**

The data transfer instructions move data between registers or between memory and registers.

| | |
|---|---|
| MOV | Move |
| MVI | Move Immediate |
| LDA | Load Accumulator Directly from Memory |
| STA | Store Accumulator Directly in Memory |
| LHLD | Load H & L Registers Directly from Memory |
| SHLD | Store H & L Registers Directly in Memory |

An 'X' in the name of a data transfer instruction implies that it deals with a register pair (16-bits);

| | |
|---|---|
| LXI | Load Register Pair with Immediate data |
| LDAX | Load Accumulator from Address in Register Pair |
| STAX | Store Accumulator in Address in Register Pair |
| XCHG | Exchange H & L with D & E |
| XTHL | Exchange Top of Stack with H & L |

# Types of instruction and Instruction set in 8085 microprocessor

**Instruction sets of 8085 microprocessor**

**Arithmetic Instruction**

Dr. A. Venkatesh

**Examples of Arithmetic Instructions:**

The arithmetic instructions add, subtract, increment, or decrement data in registers or memory.

ADD → Add to Accumulator

ADI → Add Immediate Data to Accumulator

ADC → Add to Accumulator Using Carry Flag

ACI → Add immediate data to Accumulator Using Carry

SUB → Subtract from Accumulator

SUI → Subtract Immediate Data from Accumulator

SBB → Subtract from Accumulator Using Borrow (Carry) Flag

SBI → Subtract Immediate from Accumulator Using Borrow (Carry) Flag

INR → Increment Specified Byte by One

DCR → Decrement Specified Byte by One

INX → Increment Register Pair by One

DCX → Decrement Register Pair by One

DAD → Double Register Add; Add Content of Register

DAA → Decimal adjustment of accumelator

Pair to H & L Register Pair

# Types of instruction and Instruction set in 8085 microprocessor

**Instruction sets of 8085 microprocessor**

**Logical Instruction**

Dr. A. Venkatesh

**Examples of Logical Instructions:**

This group performs logical (Boolean) operations on data in registers and memory and on condition flags. The logical AND, OR, and Exclusive OR instructions enable you to set specific bits in the accumulator ON or OFF.

ANA → Logical AND with Accumulator

ANI → Logical AND with Accumulator Using Immediate Data

ORA → Logical OR with Accumulator

OR → Logical OR with Accumulator Using Immediate Data

XRA → Exclusive Logical OR with Accumulator

XRI → Exclusive OR Using Immediate Data

The Compare instructions compare the content of an 8-bit value with the contents of the accumulator;

CMP → Compare

CPI → Compare Using Immediate Data

The rotate instructions shift the contents of the accumulator one bit position to the left or right:

RLC → Rotate Accumulator Left

RRC → Rotate Accumulator Right

RAL → Rotate Left Through Carry

RAR → Rotate Right Through Carry

Complement and carry flag instructions:

CMA → Complement Accumulator

CMC → Complement Carry Flag

STC → Set Carry Flag

# Types of instruction and Instruction set in 8085 microprocessor

Dr. A. Venkatesh

**Instruction sets of 8085 microprocessor**

**Branching Instruction**

**Examples of Branching Instructions:**

The branching instructions alter normal sequential program flow, either unconditionally or conditionally. The **unconditional branching instructions** are as follows:

JMP → Jump

CALL → Call

RET → Return

Un Conditional branching Instructions

Conditional branching Instructions

**Conditional branching instructions** examine the status of one of four condition flags to determine whether the specified branch is to be executed. The conditions that may be specified are as follows:

NZ → Not Zero (Z = 0)

Z → Zero (Z = 1)

NC → No Carry (C = 0)

C → Carry (C = 1)

PO → Parity Odd (P = 0)

PE → Parity Even (P = 1)

P → Plus (S = 0)

M → Minus (S = 1)

Thus, the **conditional branching instructions** are specified as follows:

| Jumps | Calls | Returns | |
|---|---|---|---|
| INC | CNC | RNC | (No Carry) |
| JNZ | CNZ | RNZ | (Not Zero) |
| JM | CM | RM | (Minus) |
| JP0 | CPO | RPO | (Parity Odd) |
| JM | CM | RM | (Minus) |
| JPE | CPE | RPE | (Parity Even) |
| JP0 | CPO | RPO | (Parity Odd) |

Two other instructions can affect a branch by replacing the contents or the program counter:

PCHL → Move H & L to Program Counter

RST → Special Restart Instruction Used with Interrupts

# Types of instruction and Instruction set in 8085 microprocessor

**Instruction sets of 8085 microprocessor**

**Control Instruction**

Dr. A. Venkatesh

**Examples of Control Instructions:**

**5 Stack Instructions**

The following instructions affect the Stack and/or Stack Pointer

| | |
|---|---|
| PUSH | Push Two bytes of Data onto the Stack |
| POP | Pop Two Bytes of Data off the Stack |
| XTHL | Exchange Top of Stack with H & L |
| SPHL | Move content of H & L to Stack Pointer |

**6 I/0 instructions**

| | |
|---|---|
| IN | Initiate Input Operation |
| OUT | Initiate Output Operation |

**7 Machine Control instructions**

| | |
|---|---|
| EI | Enable Interrupt System |
| DI | Disable Interrupt System |
| HLT | Halt |
| NOP | No Operation |
| RIM | Reset Interrupt Maskable |
| RIM | Set Interrupt Maskable |

# Timing Diagrams of 8085 microprocessor

**Timing diagrams of 8085 microprocessor**

**Timing Diagram definition and purpose**

Dr. A. Venkatesh

## Timing diagram of 8085 Microprocessor

- ✓ It is a graphical representation that illustrates the timing relationship between various control signals, memory operations(read/write) and instruction execution stages.

## Purpose of Timing diagram for 8085 Microprocessor

- ✓ To provide a **visual reference for understanding how the microprocessor executes instructions and manages its internal timing during operation**. These timing diagrams are essential for **Trouble Shooting**(designing and debugging) systems that use the 8085 microprocessor.

# Timing Diagrams of 8085 microprocessor

**Timing diagrams of 8085 microprocessor**

**Machine Cycle, T State and Instruction Cycle**

## Machine Cycle

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

- ✓ Machine cycles of 8085:
  - Opcode fetch cycle (4T)
  - Memory read cycle (3 T)
  - Memory write cycle (3 T)
  - I/O read cycle (3 T)
  - I/O write cycle (3 T)

Dr. A. Venkatesh

## T state

- ✓ 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.
- ✓ A portion of an operation carried out in one system clock period is called as T-state.
- ✓ 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.**

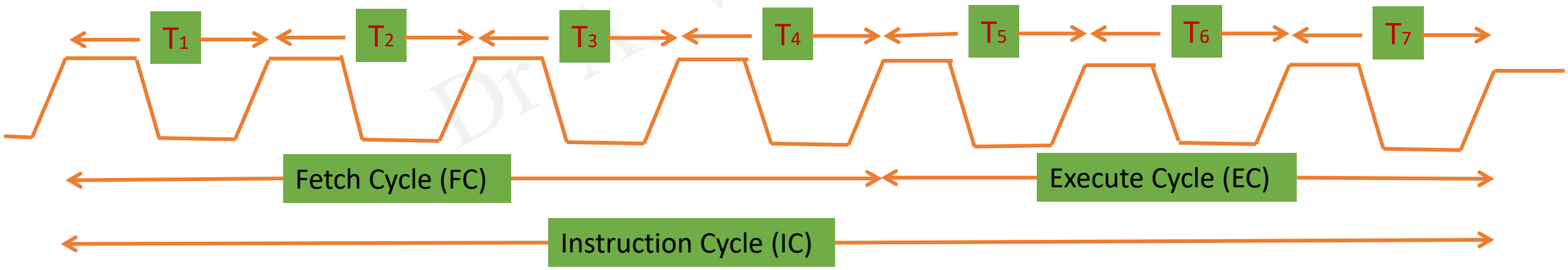
# Timing Diagrams of 8085 microprocessor

**Timing diagrams of 8085 microprocessor**

**Machine Cycle, T State and Instruction Cycle**

## Instruction Cycle

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




Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

**Timing diagrams of 8085 microprocessor**

**Pre requisites**

## Machine cycles

- ✓ Machine cycles of 8085:
  - Opcode fetch cycle (4T)
  - Memory read cycle (3T)
  - Memory write cycle (3T)
  - I/O read cycle (3T)
  - I/O write cycle (3T)

| Address/Data bus |
|---|
| $AD_0$ - $AD_7$ |
| Data bus |
| $A_8$ - $A_{15}$ |

| IO/M' | S1 | S0 | Data Bus Status |
|---|---|---|---|
| 0 | 1 | 1 | Opcode fetch |
| 0 | 1 | 0 | Memory read |
| 0 | 0 | 1 | Memory write |
| 1 | 1 | 0 | I/O read |
| 1 | 0 | 1 | I/O write |
| 1 | 1 | 1 | Interrupt acknowledge |
| 0 | 0 | 0 | Halt |

## Timing and control unit Signals

1) **Control Signals:** READY, RD(bar), WR(bar), ALE(Address Latch Enable)
2) **Status Signals:** S0, S1, IO/M(bar)

Control Signals(ALE)

- ALE(Address Latch Enable)
  - ➢ ALE=1→ Address Bus
  - ➢ ALE=0→ Data Bus

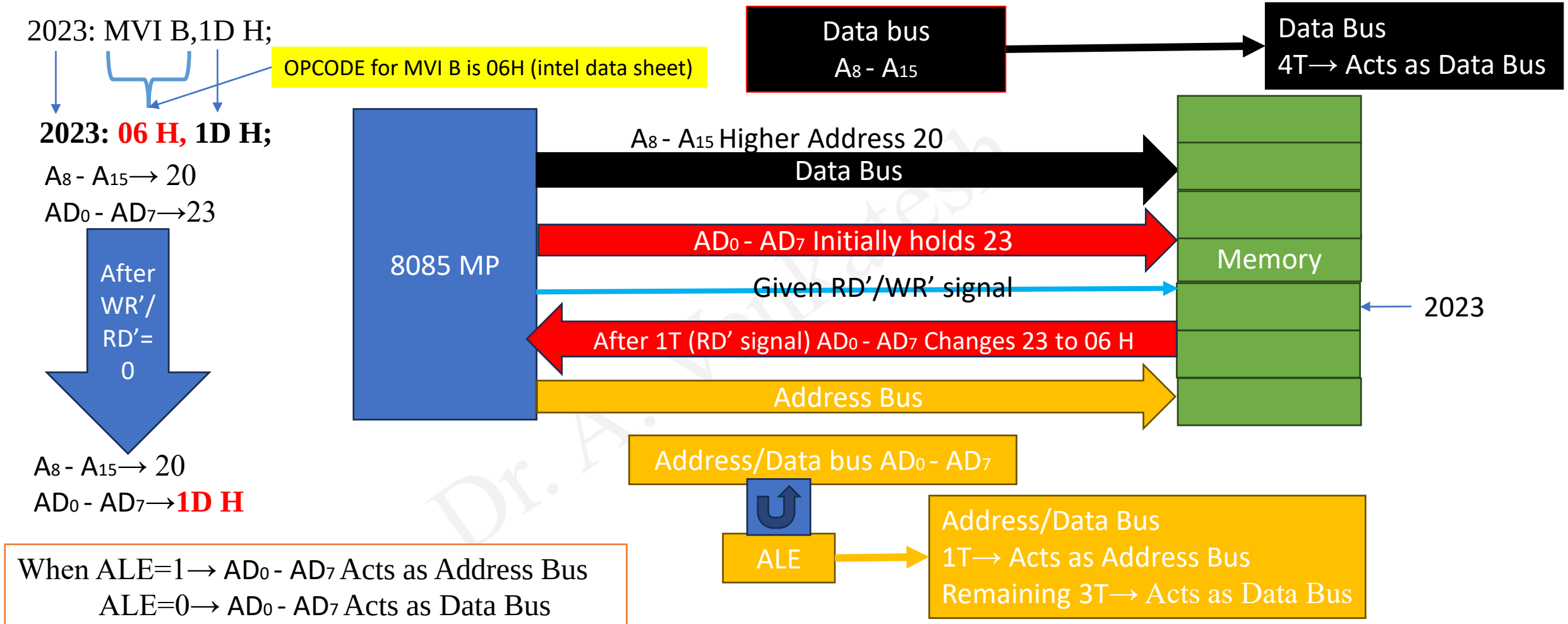
Status Signals(S1,S0 and IO/M')

Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

Example MVI B,1EH ← Chosen Immediate Data Addressing Mode where 1D H is the immediate data transferring to B register.

**OPCODE Fetching process explanation (3T used for OPCODE Fetch; 1T used to decode and execute)**

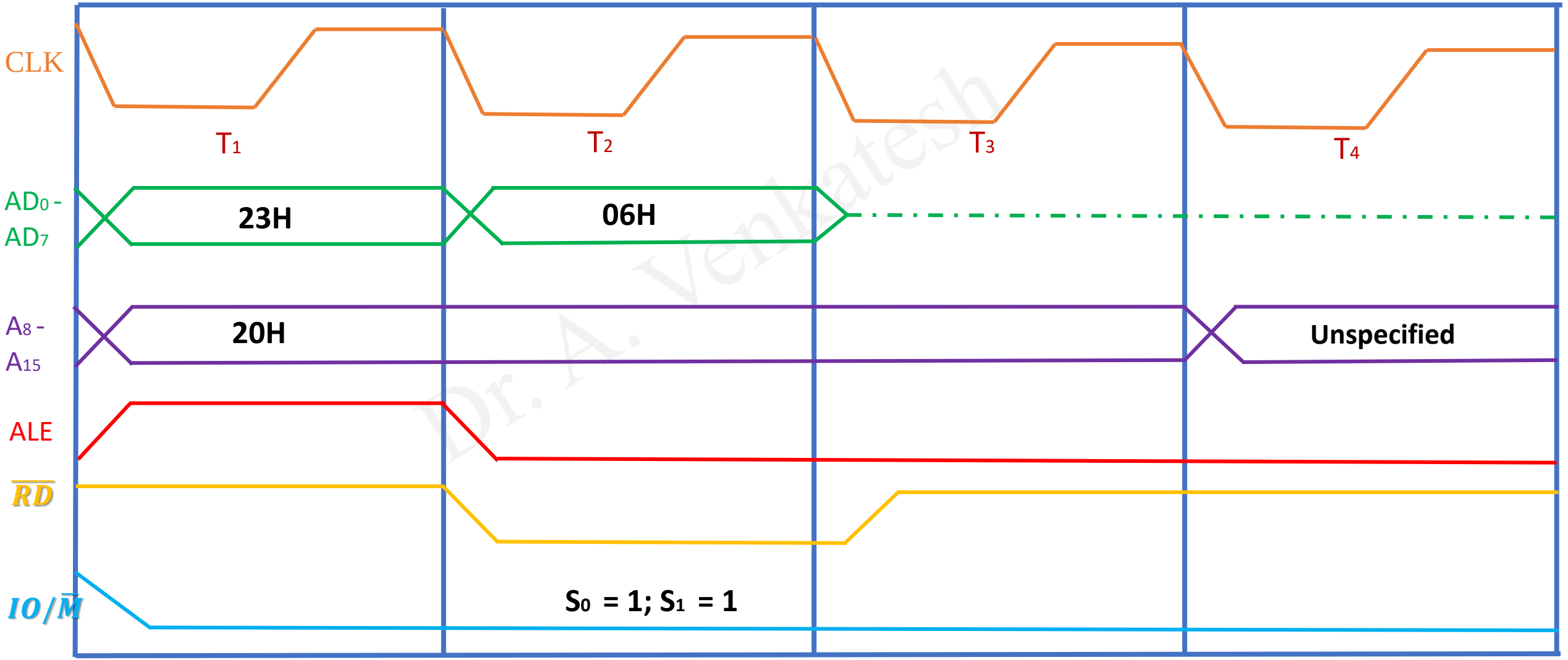



When ALE=1→ AD0 - AD7 Acts as Address Bus
ALE=0→ AD0 - AD7 Acts as Data Bus

Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

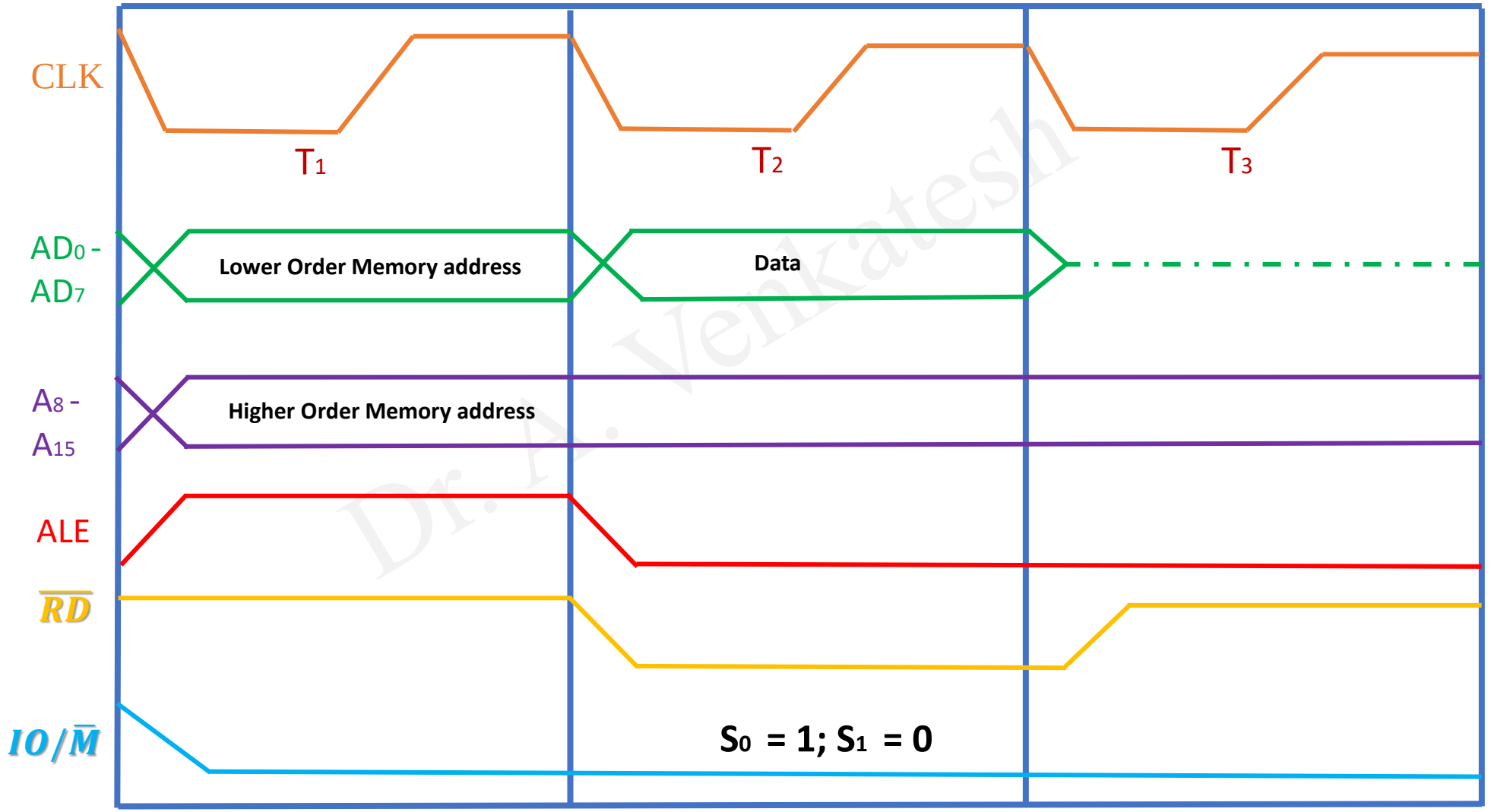
**Opcode Fetch machine Cycle(4T)** **(3T used for OPCODE Fetch; 1T used to decode and execute)**




Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

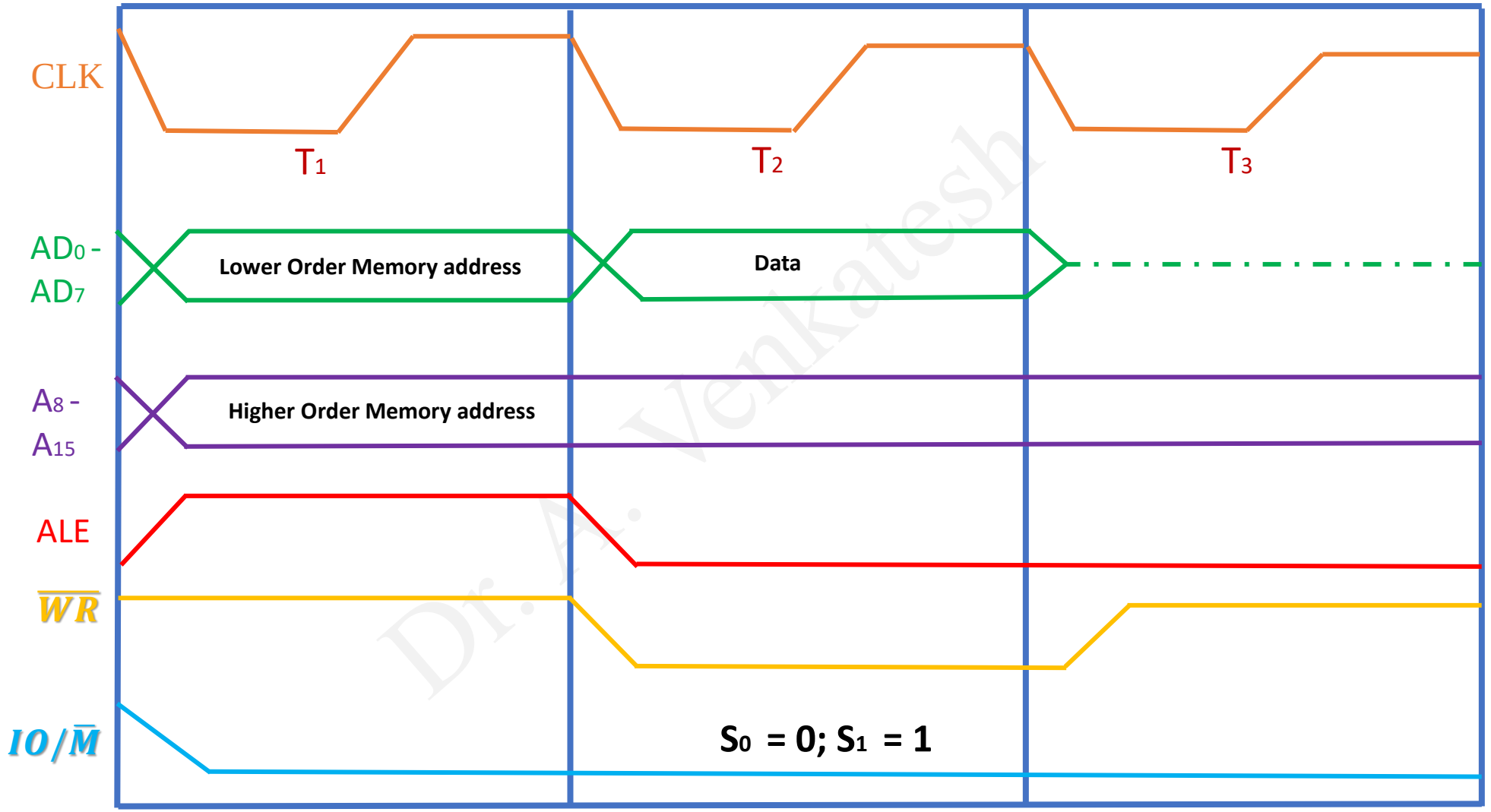
## Memory Read machine Cycle(3T) (3T states sufficient)




Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

## Memory Write machine Cycle(3T) (3T states sufficient)

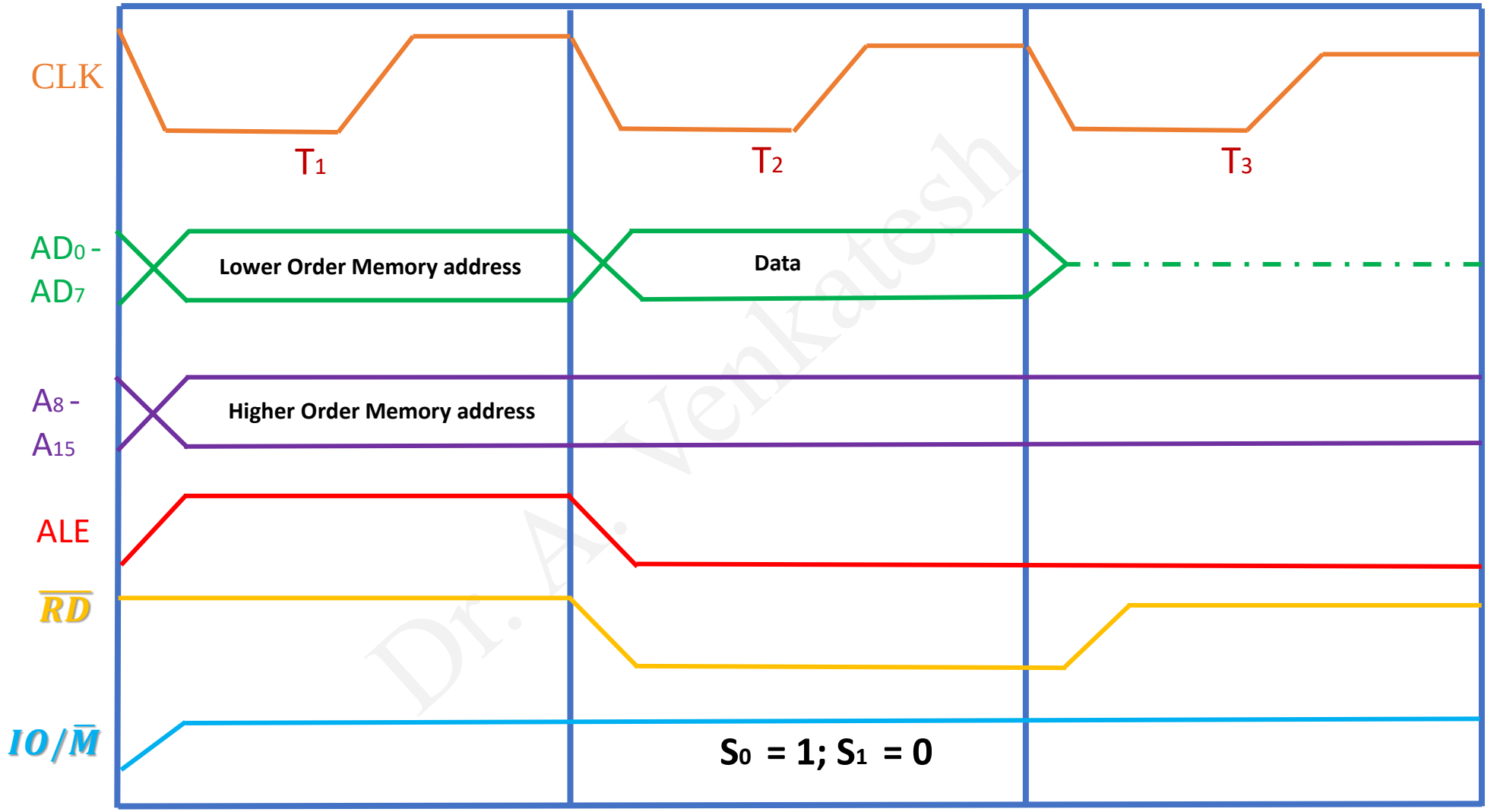



Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

## I/O Read machine Cycle(3T)

**(3T states sufficient)**

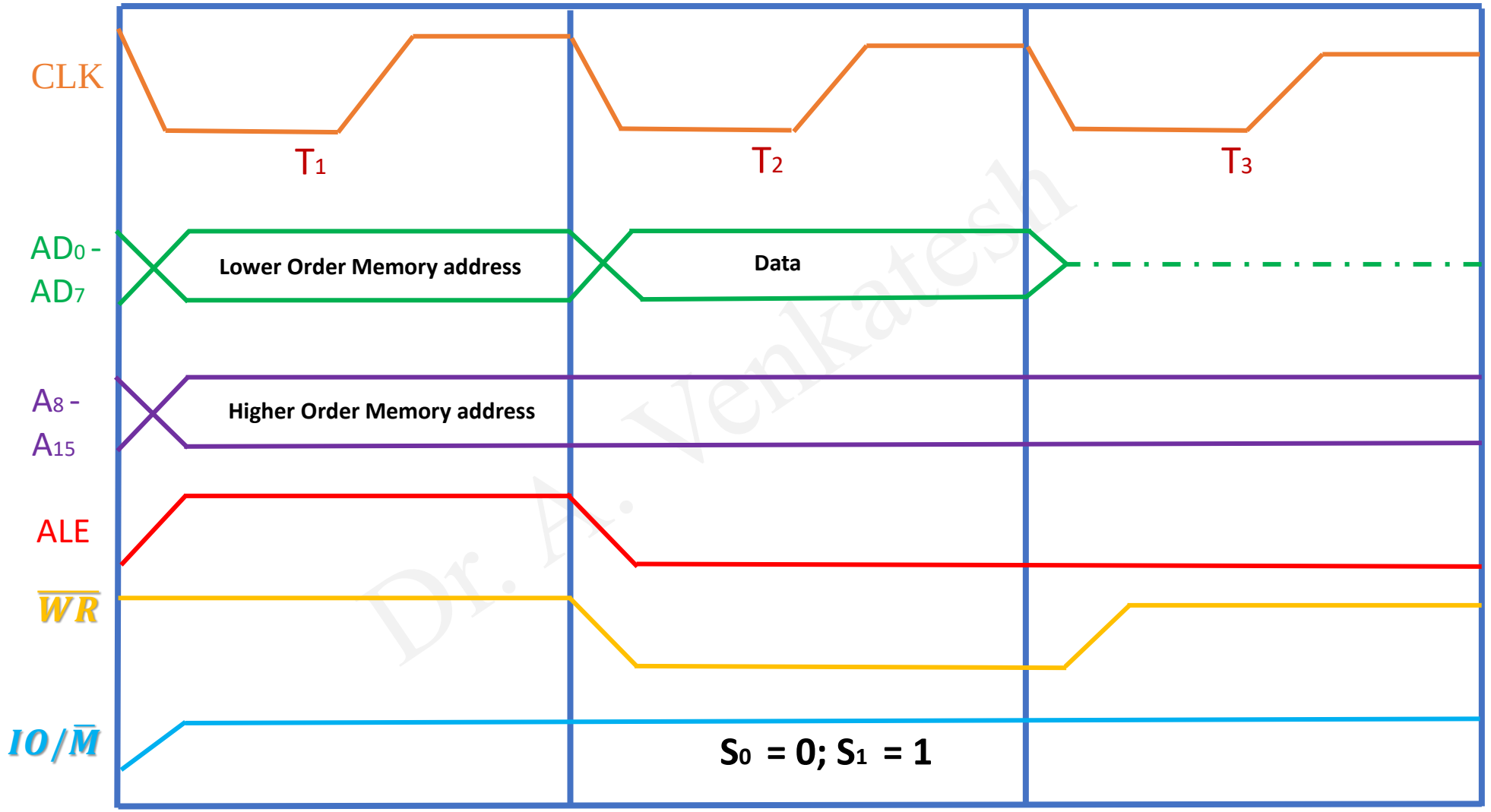



Dr. A. Venkatesh

# Timing Diagrams of 8085 microprocessor

## I/O Write machine Cycle(3T)

**(3T states sufficient)**




Dr. A. Venkatesh

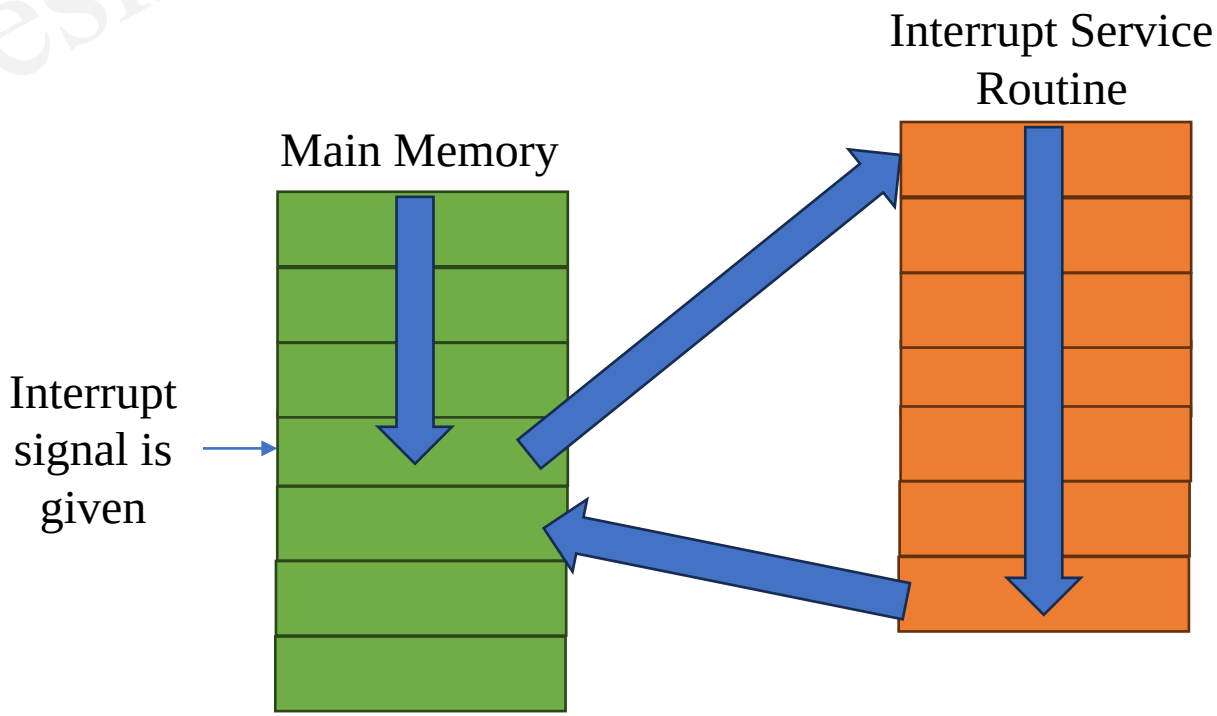
# Type of Interrupts

## Interrupts working mechanism in Microprocessor

- For example an Input/output device may send an interrupt signal to notify that the data is ready for input.
- The microprocessor temporarily stops the execution of main program and transfers control to specific special routine known as "**Interrupt Service Routine"(ISR).**
- After ISR control is transferred back to main program.
- Figure below shows the idea of communication between microprocessor and device with the help of interrupt

## What is an Interrupt in Microprocessor

It is a mechanism by which an I/O device ( hardware interrupts) or an instruction (software interrupts) can suspend the normal execution of the processor and get itself serviced.




Dr. A. Venkatesh

# Type of Interrupts

**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 RST 7

Dr. A. Venkatesh

**Hardware Interrupts**

- 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 8085 has five hardware interrupts

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

## Maskable and Non-Makable Interrupts

- Of the above four interrupts **TRAP** is a **Non-maskable** interrupt control and other **three** are **maskable** interrupts.
- A **non-maskable** interrupt is an interrupt which is given the **highest priority** in the order of interrupts.
- Suppose you want an instruction to be processed immediately, then you can give the instruction as a non-maskable interrupt.
- Further the non-maskable interrupt cannot be disabled by programmer at any point of time.
- Whereas the maskable interrupts can be disabled and enabled using EI and DI instructions.
- Among the **maskable** interrupts **RST 7.5** is given the **highest priority** above **RST 6.5** and **least priority** is given to **INTR.**

Dr. A. Venkatesh

## Comparison between Software and Hardware Interrupts

| Sr. No | Software Interrupt | Hardware Interrupt |
|---|---|---|
| 1. | It is a synchronous event. | It is an asynchronous event. |
| 2. | This interrupt is requested by executing instruction. | This interrupt is requested by an external device on a pin. |
| 3. | PC in incremented. | PC is not incremented. |
| 4. | The microprocessor does not execute any interrupt acknowledge cycle to acknowledge this interrupt. The microprocessor executes a normal instruction cycle. | The microprocessor executes either interrupt acknowledge cycle bus or idle machine cycle to acknowledge this interrupt. |
| 5. | It cannot be ignored or masked. | It can be masked except for TRAP. |
| 6. | It has the highest priority among all interrupts. | The priority is lower than that of a software interrupt. |
| 7. | It does not affect interrupt control logic. | It affects interrupt control logic. |
| 8. | It is not used to interface peripherals that means it does not improve the throughput of the system. It is used in program debugging. | It is used to interface peripherals in interrupt-driven I/O. It improves the throughput of the system. |