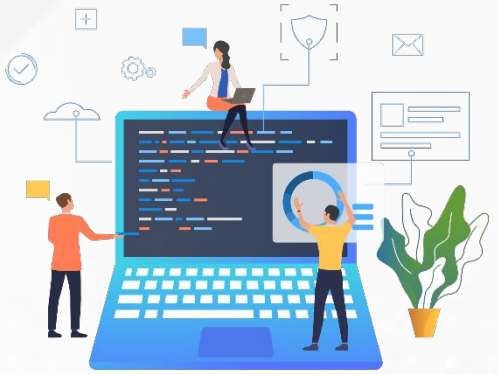
Microprocessor and Interfacing
(MPI)
GTU # 3160712

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

# Unit-3: 8085 Microprocessor




**Prof. Swati R Sharma**
Computer Engineering Department
Darshan Institute of Engineering & Technology, Rajkot

swati.sharma@darshan.ac.in
(O) 9727747317

# Subject Overview

| Sr. No. | Unit | % Weightage |
|---|---|---|
| 1 | Introduction to Microprocessor | 8% |
| 2 | Microprocessor Architecture and Operations | 7% |
| **3** | **8085 Microprocessor** | **12%** |
| 4 | Assembly Language Programming Basics | 13% |
| 5 | 8085 Assembly Language Programs | 12% |
| 6 | Stack & Subroutines | 13% |
| 7 | I/O Interfacing | 20% |
| 8 | Advanced Microprocessors | 15% |

## Topics to be covered

- Introduction to 8085
- 8085 Programming Model
- Bus Organization of 8085
- 8085 pin diagram
- 8085 Architecture/Block Diagram
- T-States, Machine and Instruction Cycle
- Demultiplexing Address and Data Bus AD0-AD7
- Timing Diagram
- Memory Interfacing
- Generating Control Signals

# Introduction to 8085

# Introduction to 8085

- 8085 is pronounced as "eighty-eighty-five" microprocessor.
- It is an 8-bit microprocessor designed by Intel in 1977 using NMOS technology.

# Introduction to 8085

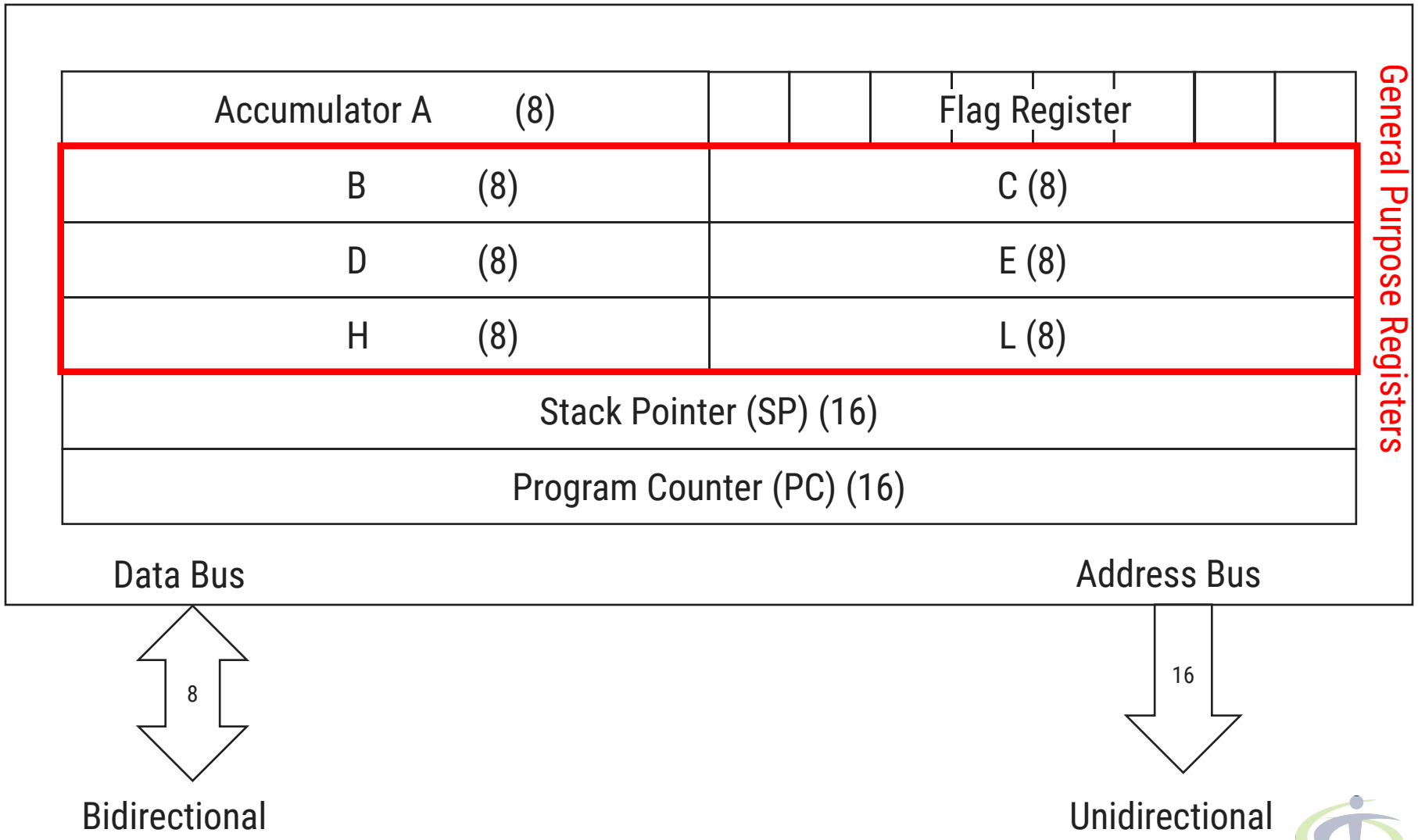
- 8 bit General purpose microprocessor (i.e. 8 bit data bus).
- It is a single chip N MOS device with 40 pins.
- It has multiplexed address and data bus.($AD_0$ - $AD_7$).
- It works on 5-Volt DC power supply.
- The maximum clock frequency is 3MHz while minimum frequency is 500kHz.
- It provides 16 address lines, therefore capable of addressing $2^{16}$= 64K of memory.
- It supports external interrupt request.
- It has two 16 bit registers named program counters (PC) and stack pointer (SP).

# Introduction to 8085

- It generates 8 bit I/O address so it can access $2^8 = 256$ input ports.
- It provides 5 hardware interrupts:
  1. TRAP
  2. RST 5.5
  3. RST 6.5
  4. RST 7.5
  5. INTR
- It provides accumulator, 5 flag register, 6 general purpose registers and 2 special purpose registers (SP,PC).

# 8085 Programming Model

# 8085 Programming Model

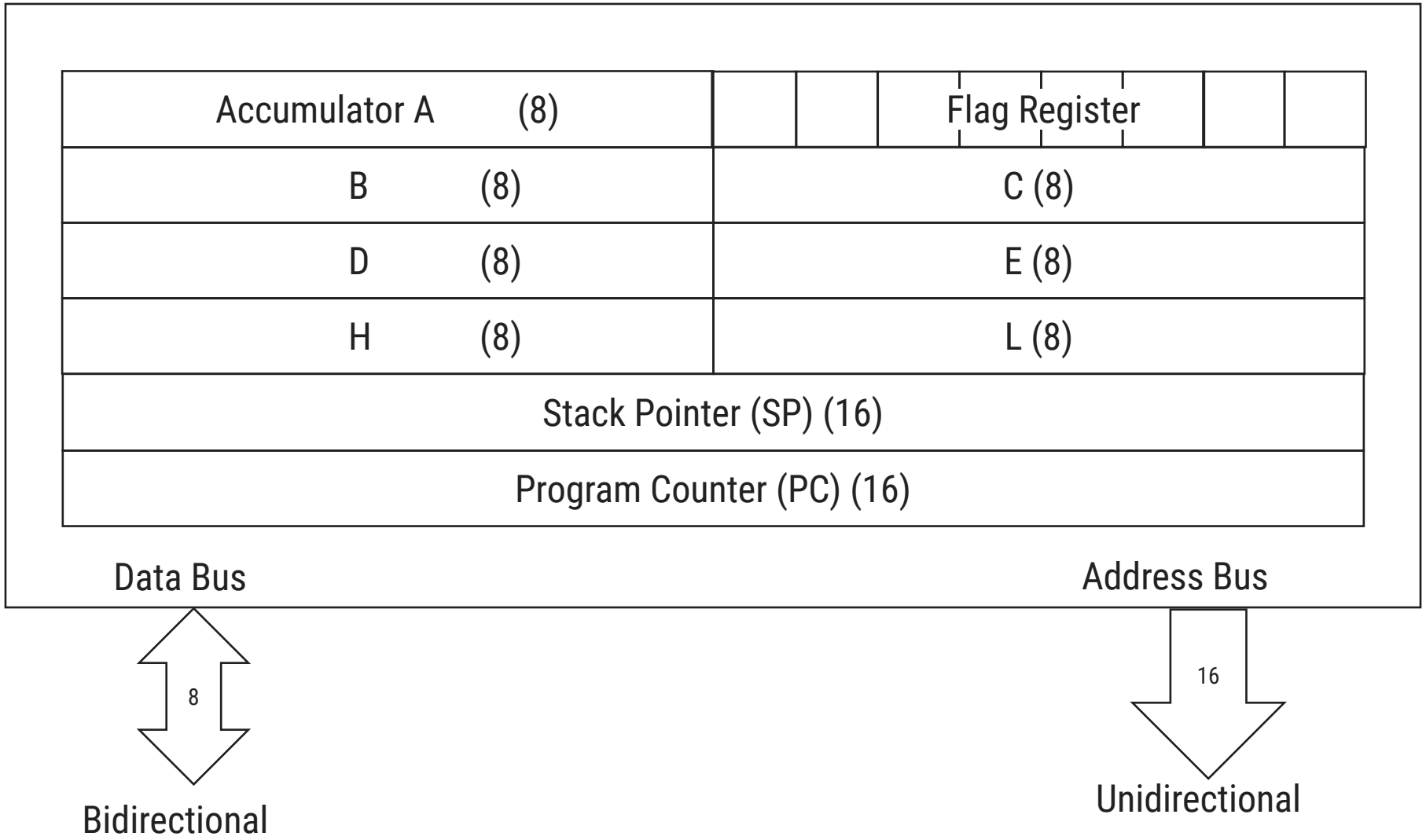


# General Purpose Registers

- 6 general purpose registers to store 8-bit data B, C, D, E, H & L.
- Can be combined as fixed register pairs – BC, DE, HL to perform 16 bit operations.
- Used to store or copy data using data copy instructions.

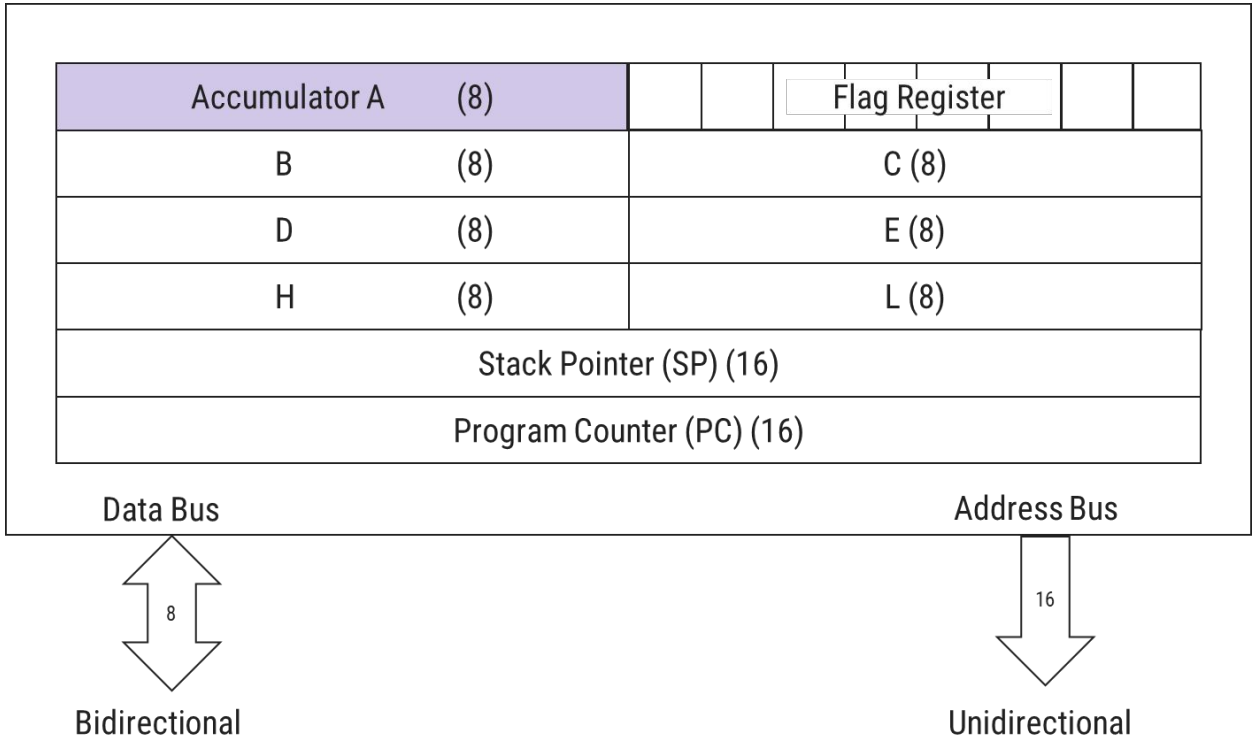
| | |
|---|---|
| B (8) | C (8) |
| D (8) | E (8) |
| H (8) | L (8) |

# 8085 Programming Model

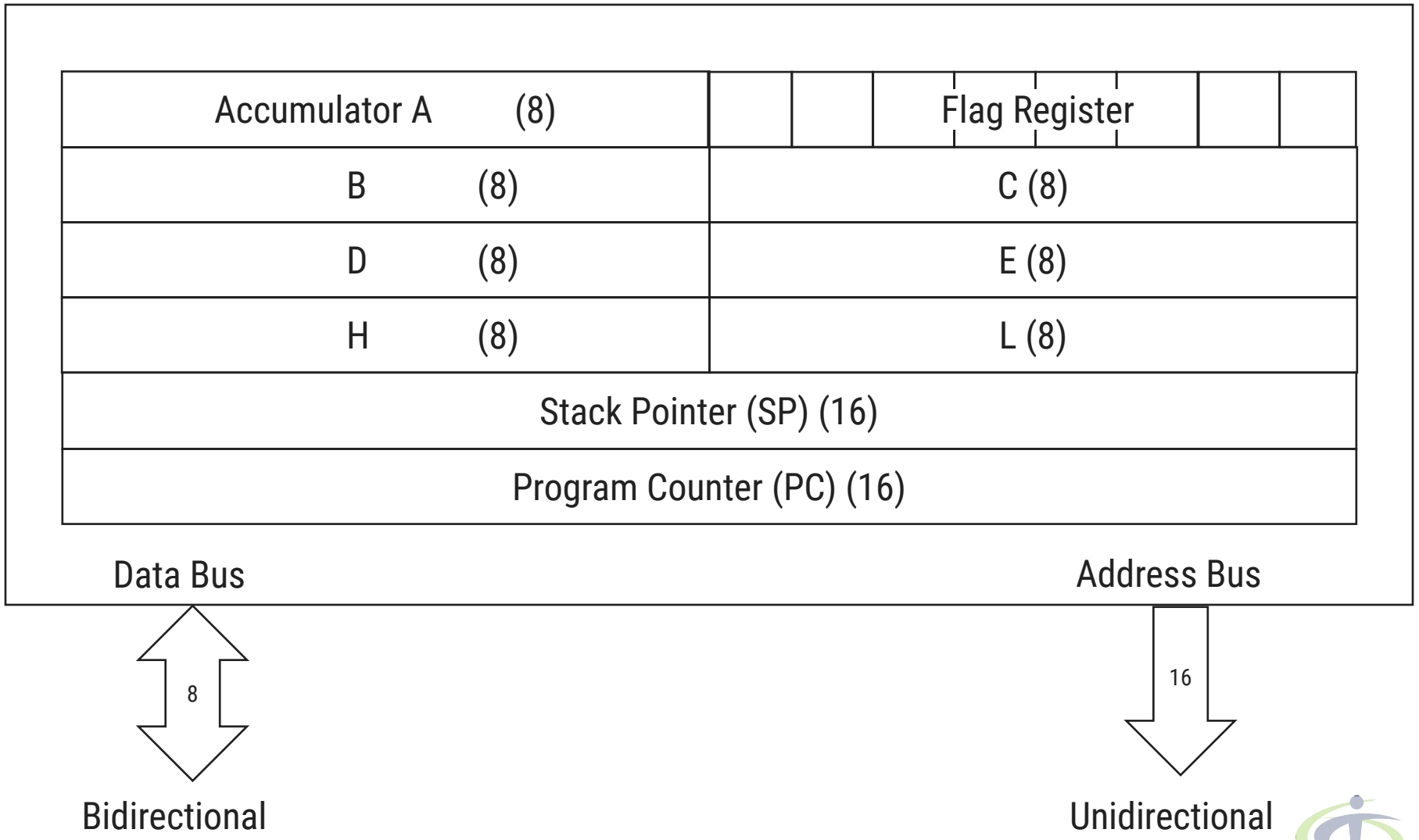


# Accumulator

- 8 - bit register, identified as **A**
- Part of ALU
- Used to store 8-bit data to perform **arithmetic** & **logical** operations.
- Result of operation is stored in Accumulator.



# 8085 Programming Model



# Flag Register

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| S | Z | X | AC | X | P | X | CY |

**X**:Undefined

**S -Sign Flag**
Set (1) if 7th bit of result is 1; otherwise reset (0)

1 0 1 0 1 0 1 0
*S = 1*

**Z -Zero Flag**
Set (1) when result is zero; otherwise reset(0)

```
  1 0 1 0
- 1 0 1 0
---------
  0 0 0 0
```
*Z = 1*

**P -Parity Flag**
Set (1) if result has even no. of 1's & Reset(0) if result has odd no. of 1's

1 0 0 1 0 0 1 1
*P = 1*

**AC -Auxiliary Carry Flag**
Set (1) when carry bit is generated by 3rd bit & passed to bit 4th bit.

```
    1 1   1
  0 0 1 0   1 0 1 0
+ 0 1 1 0   1 0 0 1
-----------------
  1 0 0 1   0 0 1 1
```
*AC = 1*

**CY -Carry Flag**
Set (1) if arithmetic operation results in carry; otherwise reset(0)

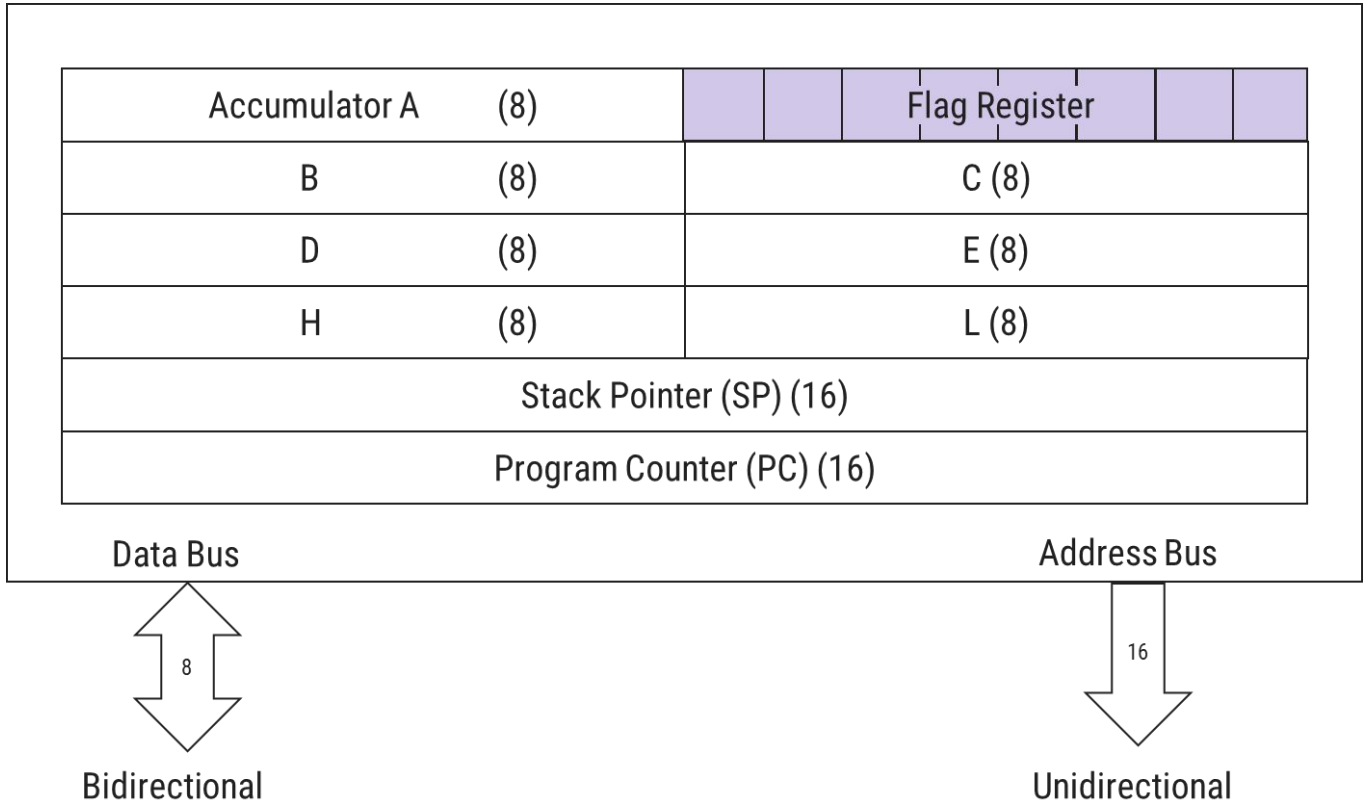
```
    1 1   1
  1 0 1 0   1 0 1 0
+ 0 1 1 0   1 0 0 1
-----------------
1 0 0 0 1   0 0 1 1
```
*CY = 1*

# Flag Register

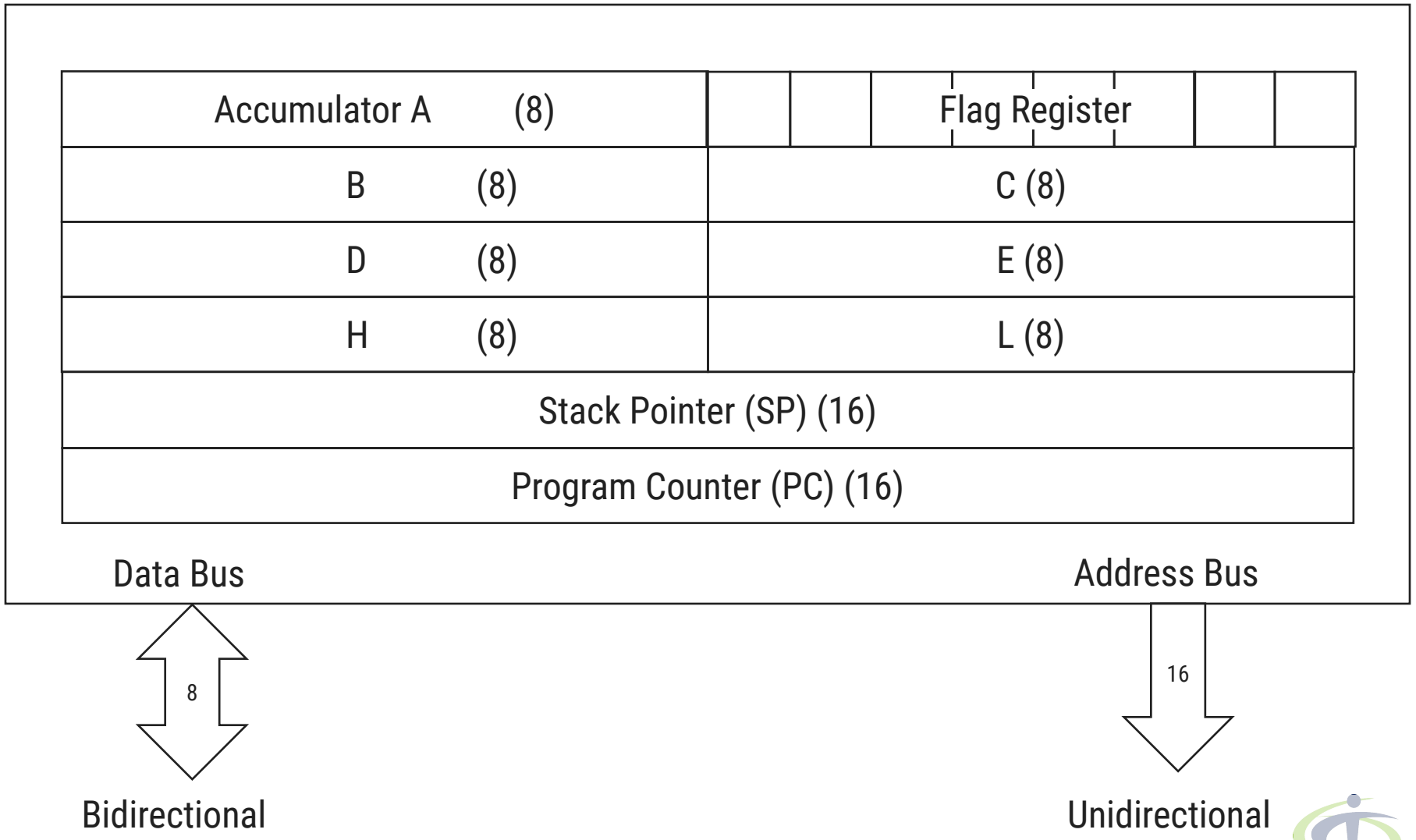
- ALU has 5 Flag Register that set/reset after an operation according to data conditions of the result in accumulator & other registers.
- Helpful in decision making process of microprocessor.
- Conditions are tested through software instructions.

  For e.g.

  JC (Jump On Carry) is implemented to change the sequence of program when CY(Carry Flag) is set(1).



# 8085 Programming Model



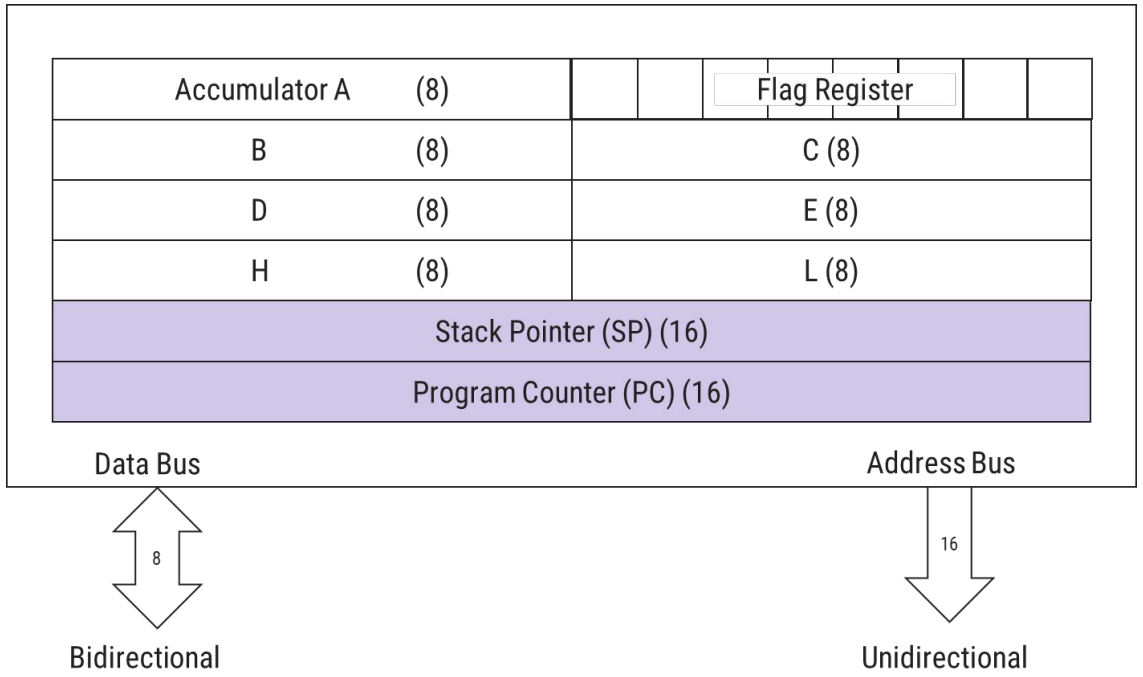
# Stack Pointer & Program Counter

## Stack Pointer(SP)

- Used as memory pointer.
- Points to the memory location in R/W memory, called Stack.
- Beginning of stack is defined by loading a 16-bit address in the stack pointer.

## Program Counter(PC)

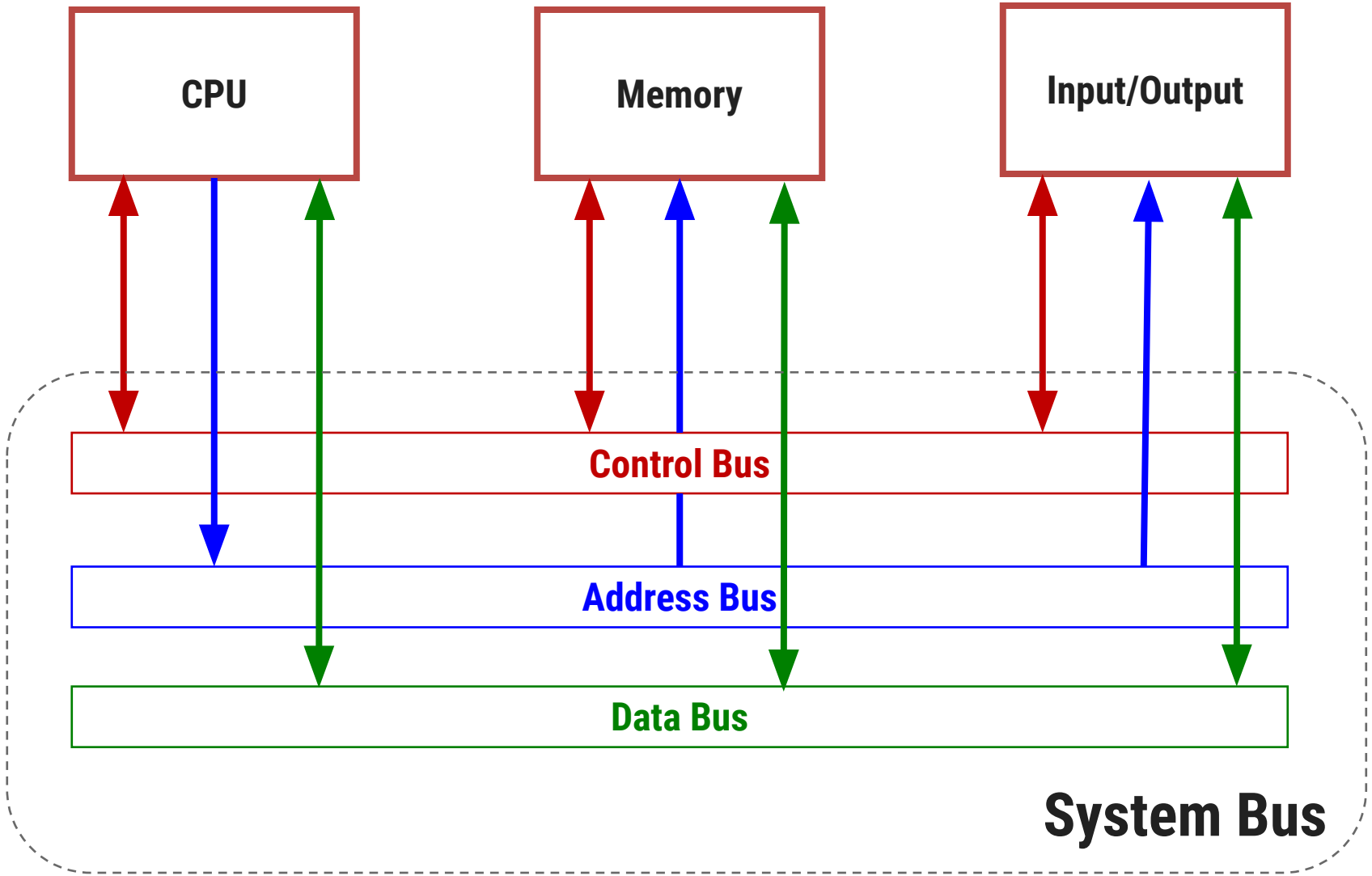
- Microprocessor uses PC register to sequence the execution of instructions.
- Its function is to point to memory address from which next byte is to be fetched.
- When a byte is being fetched, PC is incremented by 1 to point next memory location.



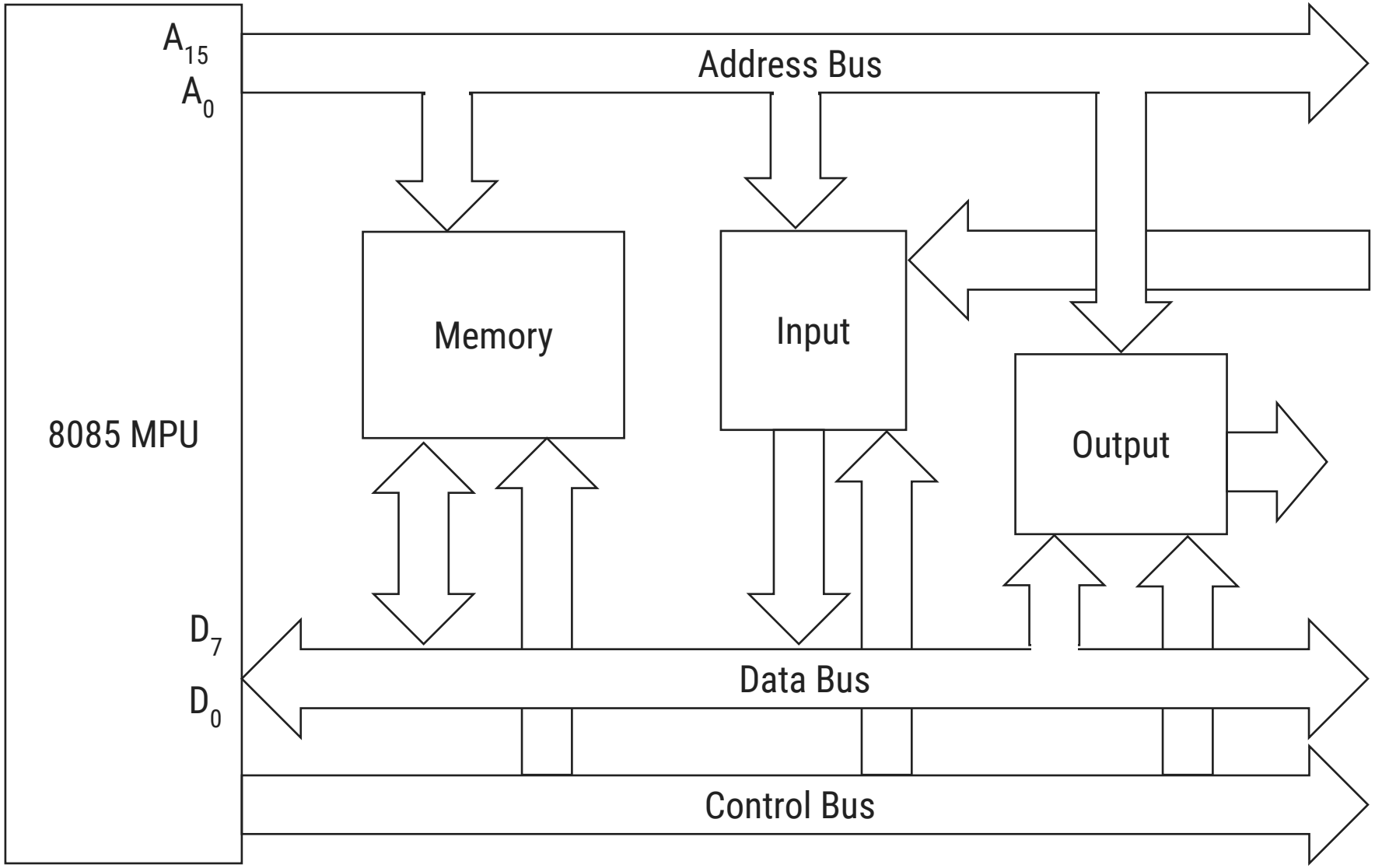

*GTU Que: Explain 8085 Programming Model and flag register.[7m] (W'18,W'19)*

# Bus Organization of 8085

# System bus

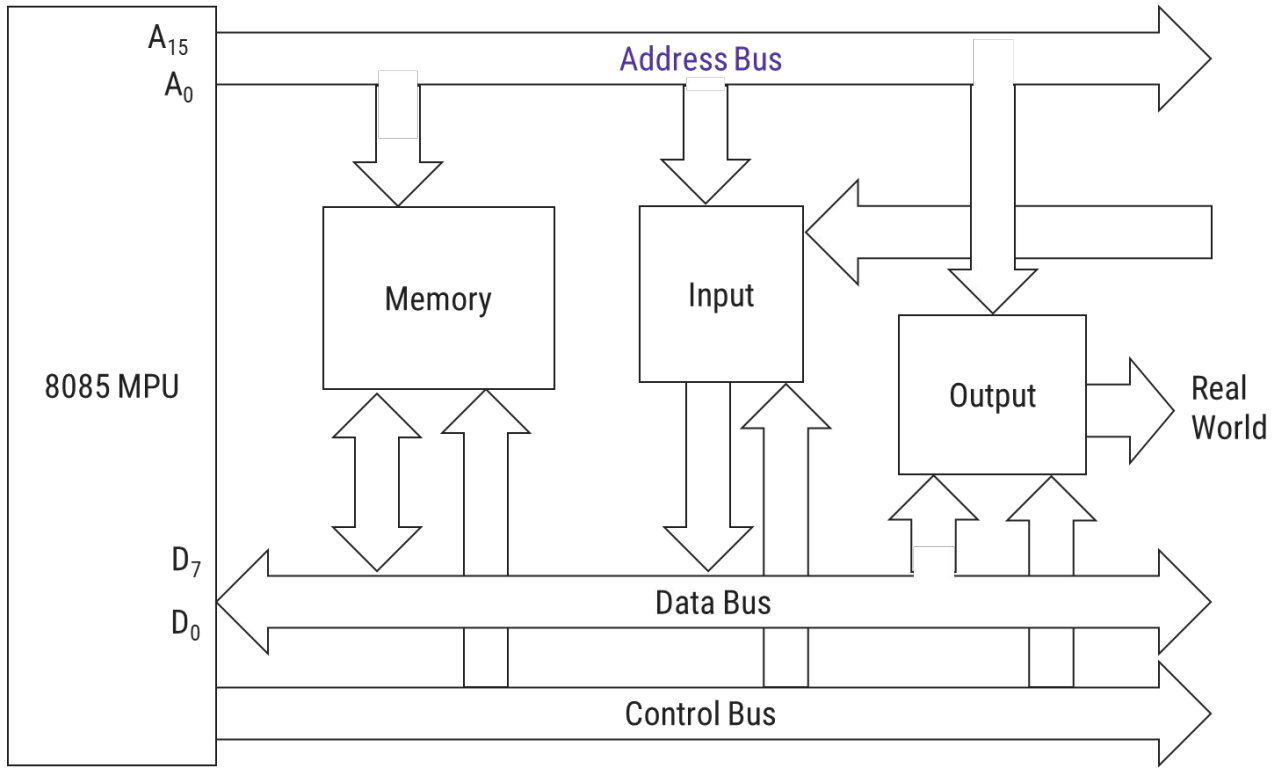


# Bus Organization of 8085



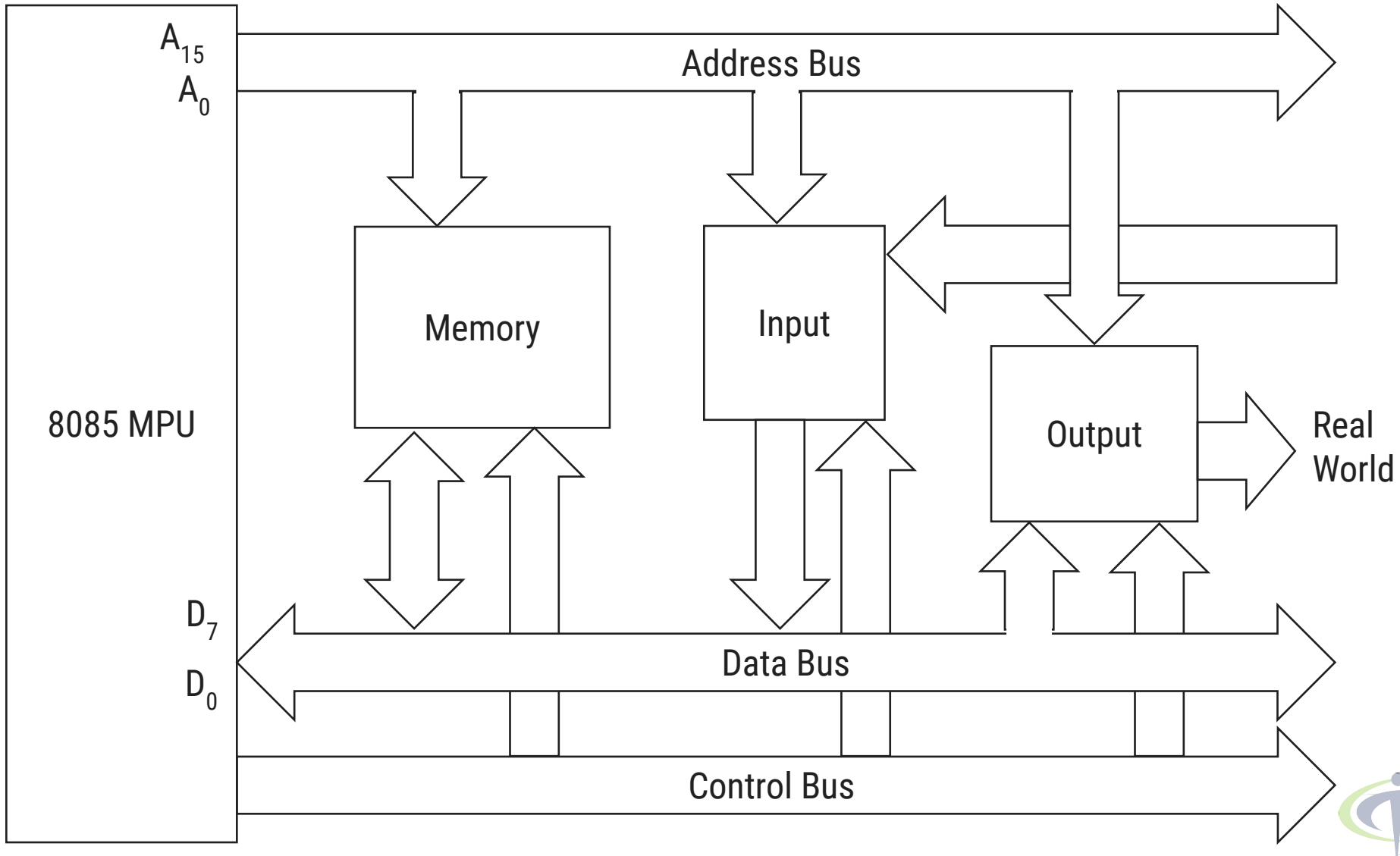
# Address bus

- Group of 16 unidirectional lines generally identified as $A_0$ to $A_{15}$.
  i.e. bits flow from microprocessor to peripheral devices.
- 16 address lines are capable of addressing 65536 memory locations. So, 8085 has 64K memory locations.



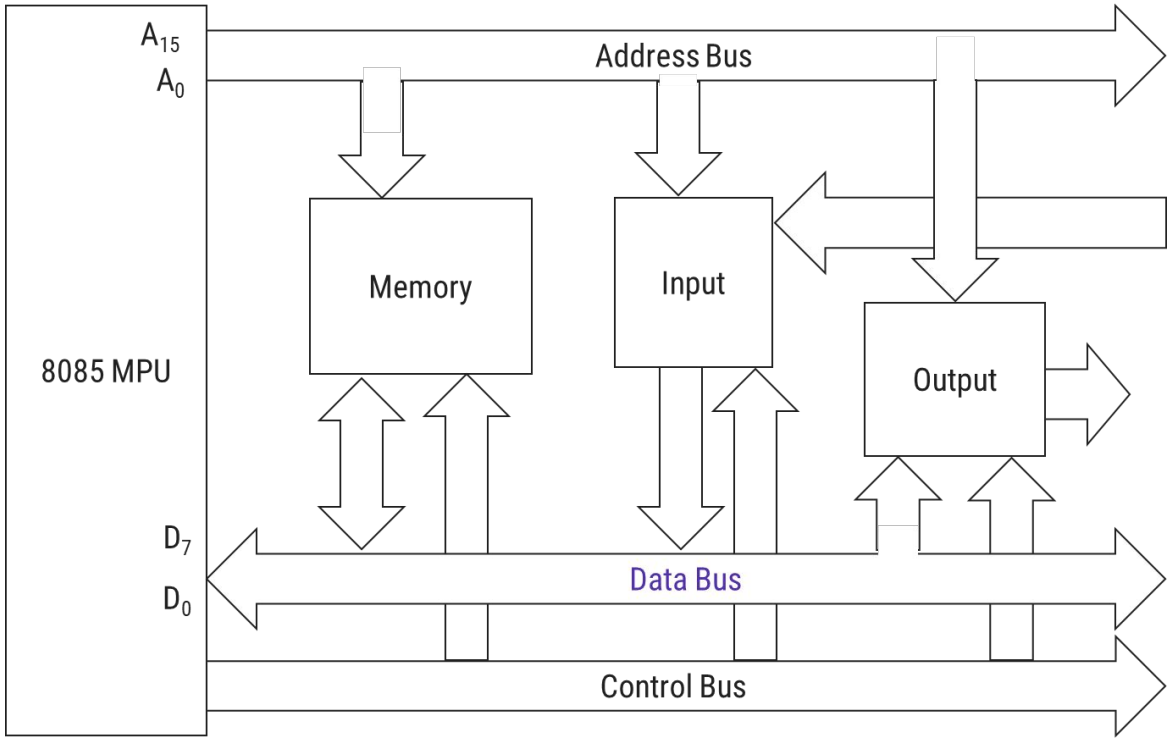

| | |
|---|---|
| HEX | FFFF |
| DEC | 65,535 |
| OCT | 177 777 |
| BIN | 1111 1111 1111 1111 |

# Bus Organization of 8085

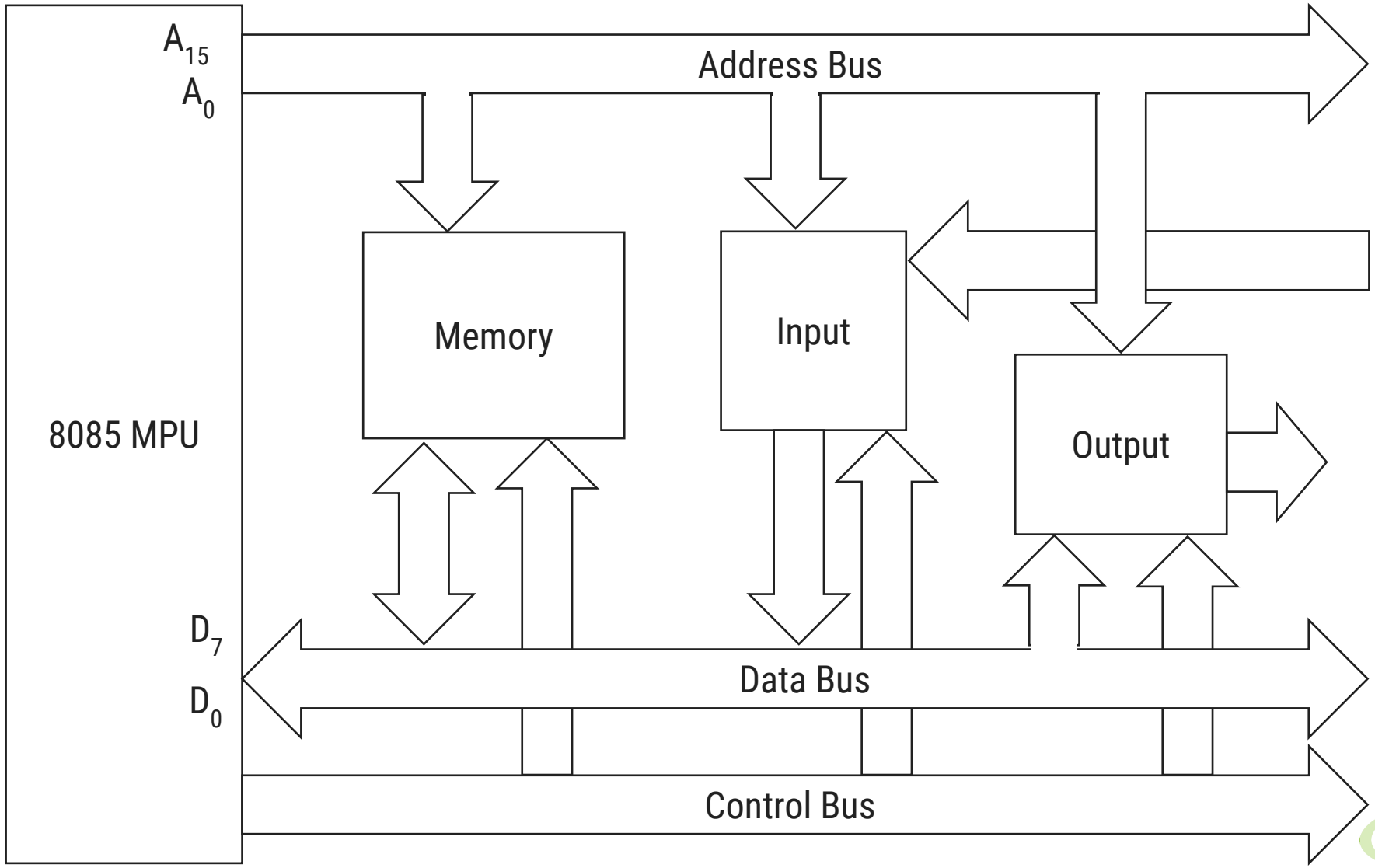


# Data bus

- Group of 8 lines identified as $D_0$ to $D_7$.
- They are bidirectional i.e. data flow in both directions between microprocessor, memory & peripheral.
- 8 data lines enable microprocessor to manipulate data ranging from 00 H to FF H ($2^8$=256 numbers).
- Largest number appear on data bus is 1111 1111 => $(255)_{10}$.
- As Data bus is of 8-bit, 8085 is known as 8-bit Microprocessor.

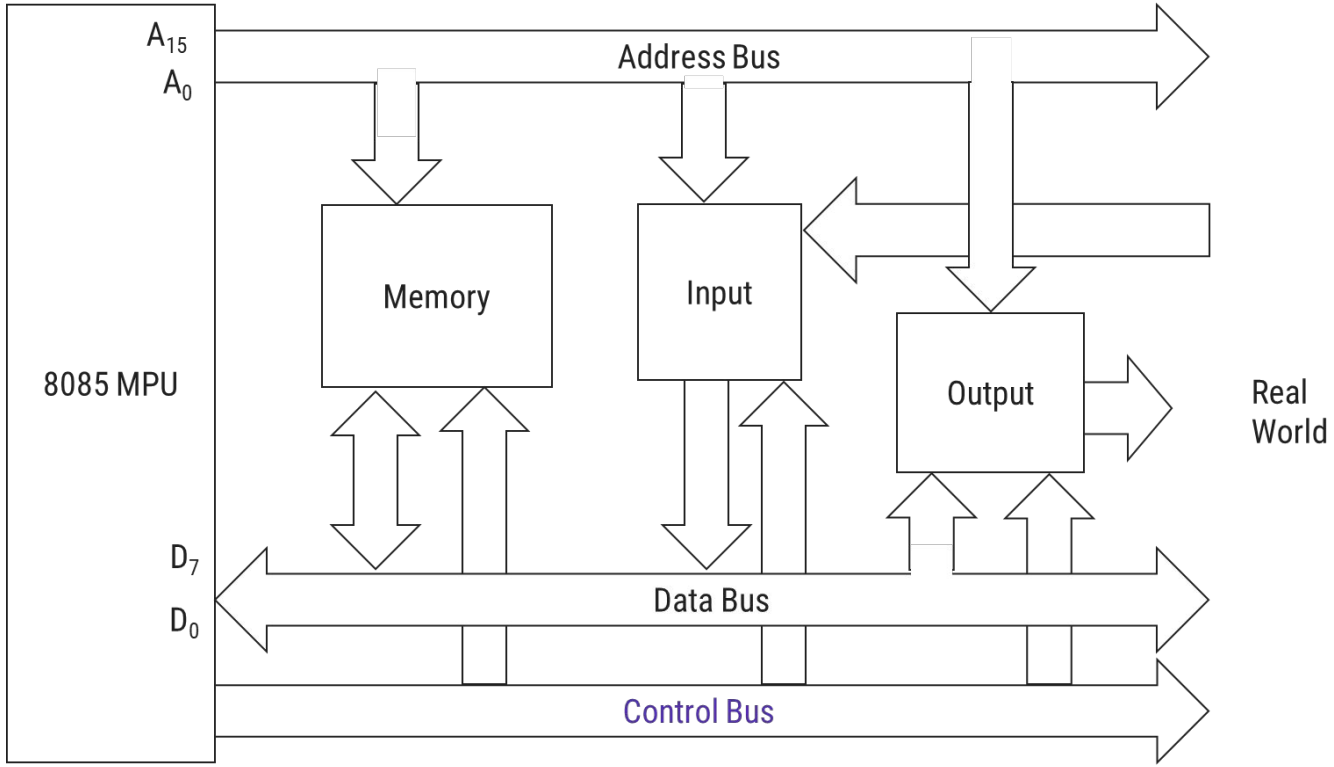


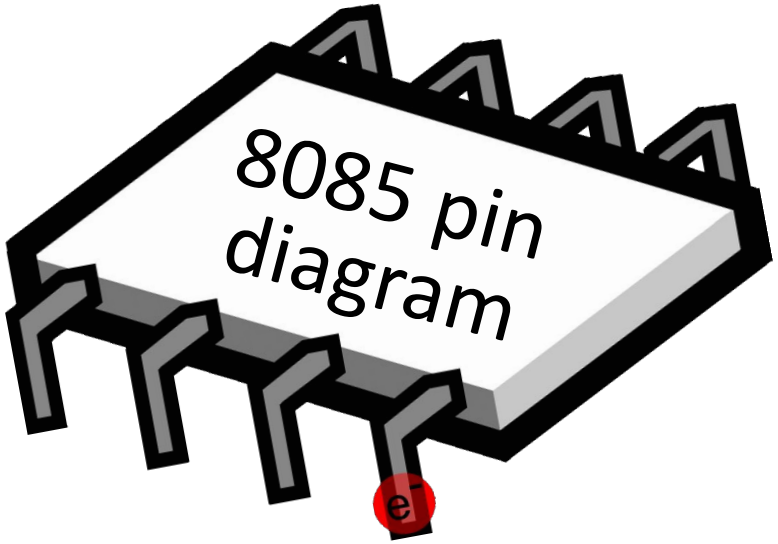
# Bus Organization of 8085



# Control bus

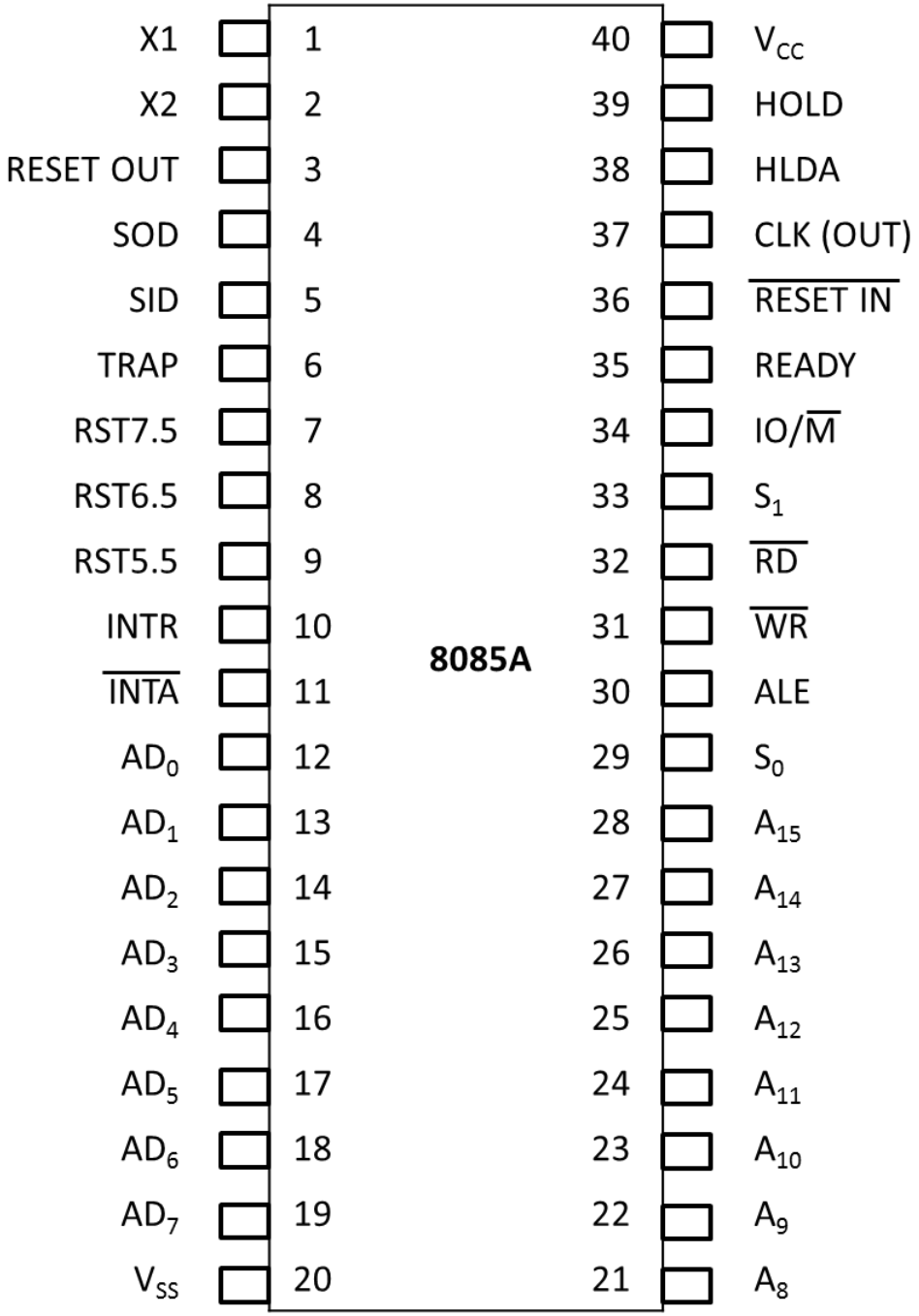
- It comprises of various single lines that carry synchronization, timing & control signals.
- These signals are used to identify a device type with which MPU intends to communicate.
- Some control signals are **Read**, **Write** and **Opcode fetch** etc.






# 8085 pin diagram

# 8085 pin diagram

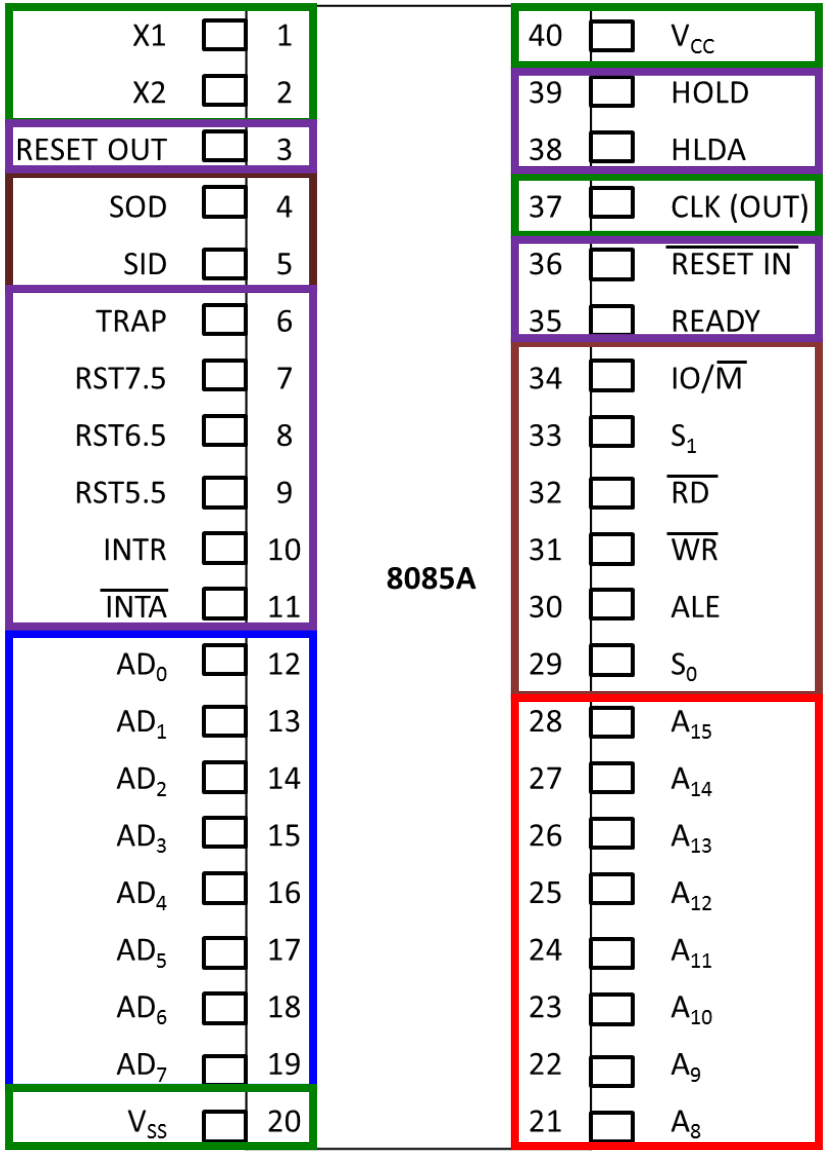



- 8-bit general purpose microprocessor.
- Capable of addressing 64K of memory.
- It has 40 pins.
- Requires +5V single power supply.

# 8085 pin diagram

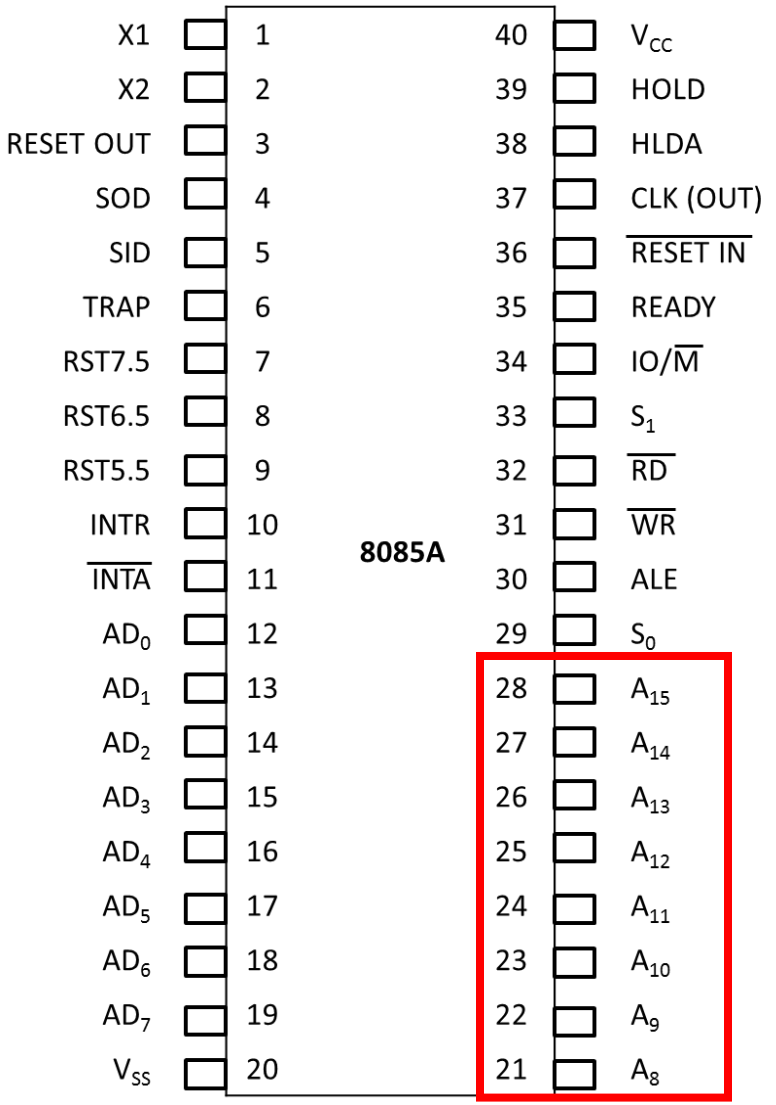
**Signals are classified into 6 groups:**

1. Address bus
2. Multiplexed address/data bus
3. Control & status signals
4. Power supply & frequency signals
5. Externally initiated signals
6. Serial I/O ports



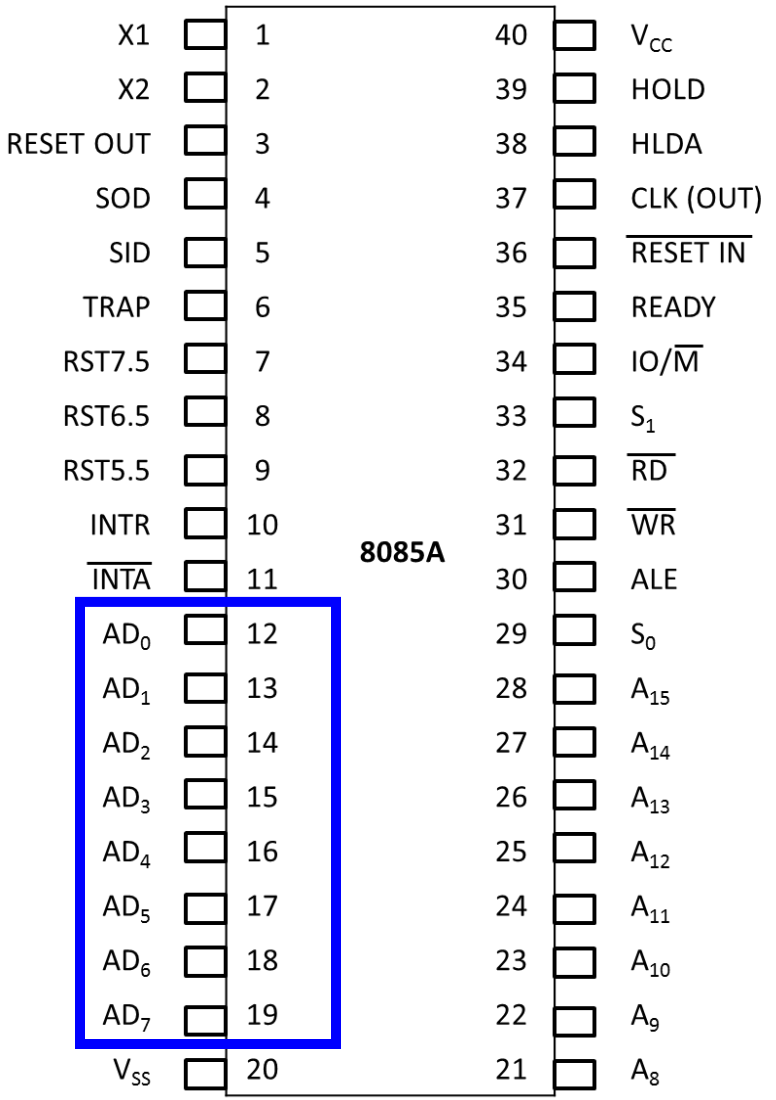
# 8085 pin diagram: Address Bus

- 16 signal lines are used as address bus.
- However these lines are split into two segments: $A_{15}$ - $A_8$ and $AD_7$ - $AD_0$
- $A_{15}$ - $A_8$ are unidirectional and used to carry high-order address of 16-bit address.
- $AD_7$ - $AD_0$ are used for dual purpose.



# 8085 pin diagram: Multiplexed Address/Data Bus

- Signal lines $AD_7$-$AD_0$ are bidirectional and serve dual purpose.
- They are used as low-order address bus as well as data bus.
- The low-order address bus can be separate from these signals by using a latch (ALE).



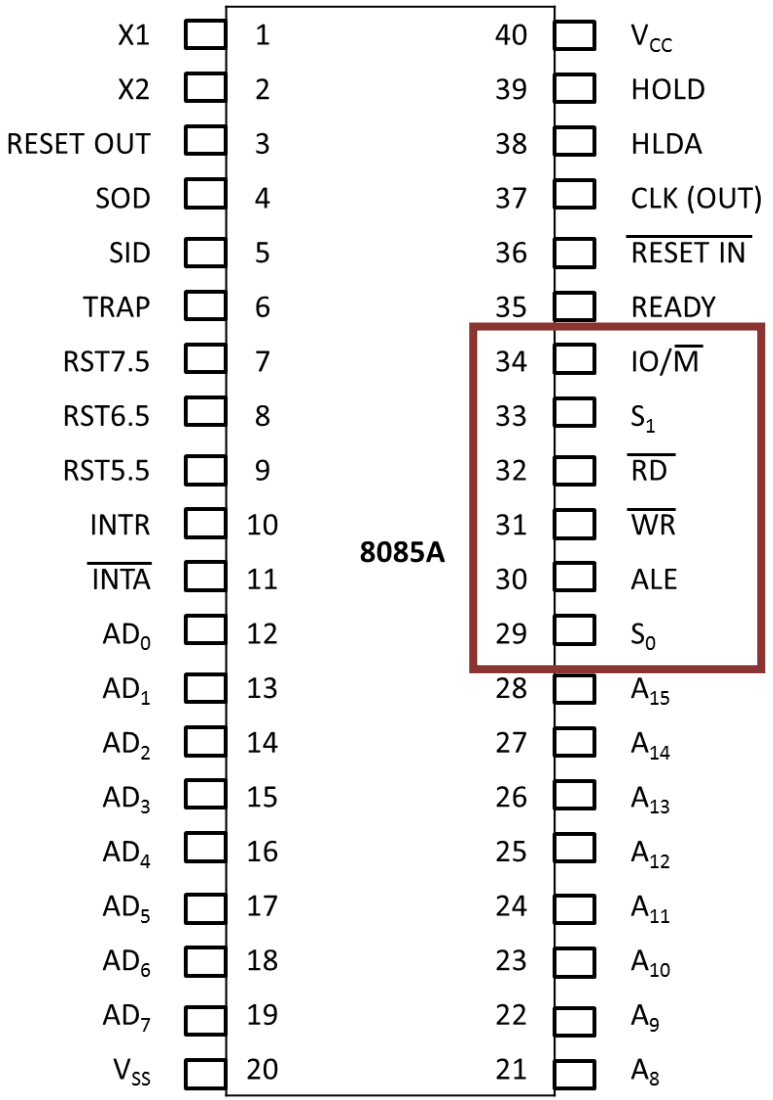
# 8085 pin diagram: Control & Status Signals

To identify nature of operation

- Two Control Signals
  1. $\overline{RD}$(Read)
  2. $\overline{WR}$(Write)
- Three Status Signals
  1. $S_1$
  2. $S_0$
  3. IO/$\overline{M}$
- To indicate beginning of operation
  1. ALE(Address Latch Enable)

     ALE ← 1, then Address bus

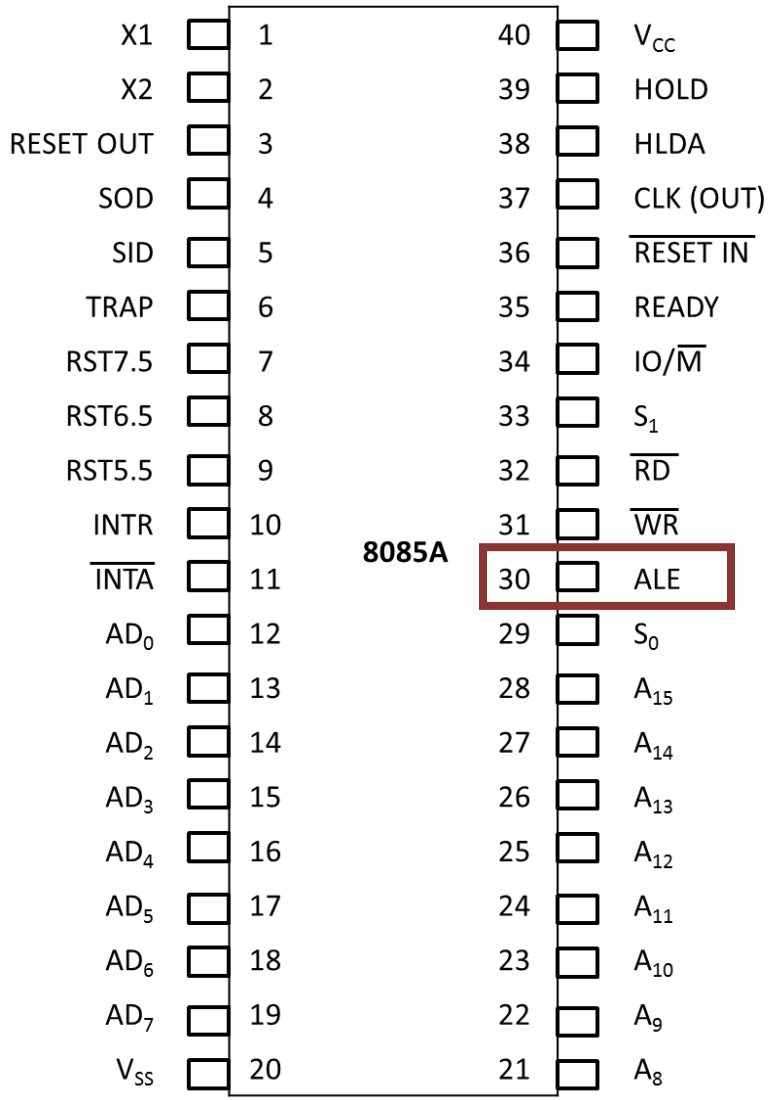
     ALE ← 0, then Data bus



# 8085 pin diagram: Control & Status Signals

## ALE: Pin 30

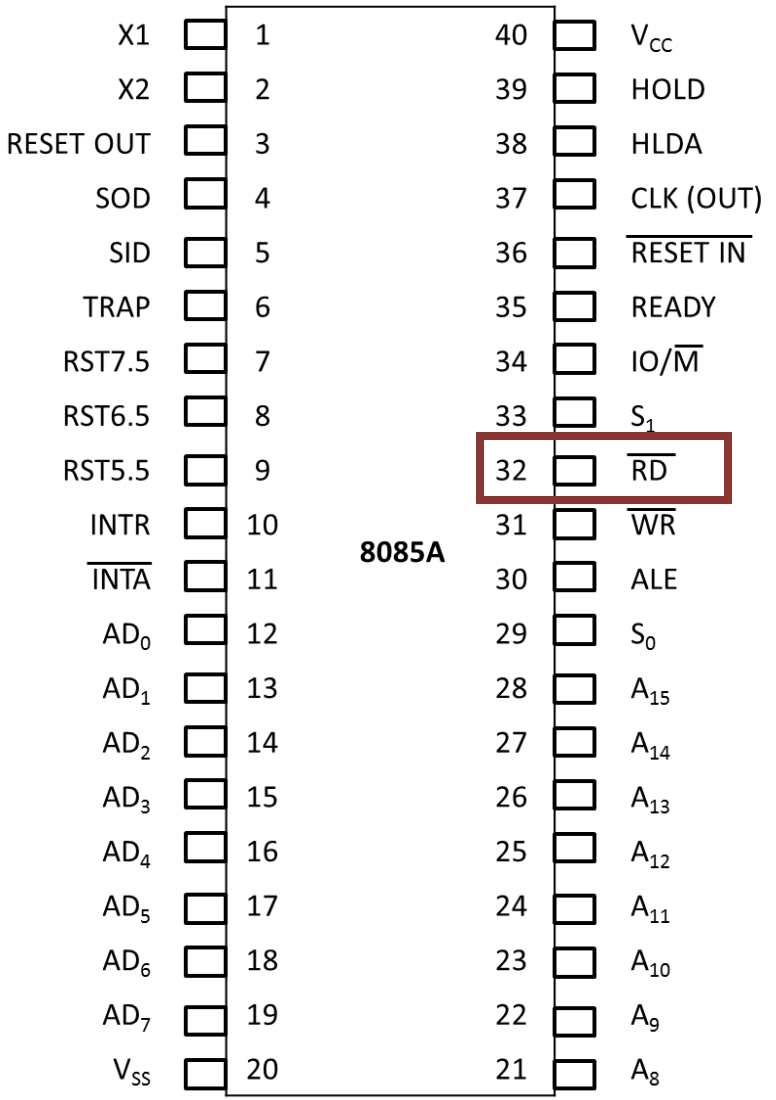
- This is positive going pulse generated every time the 8085 begins an operation (machine cycle).
- It indicates that the bits on $AD_7$-$AD_0$ are address bits.
- This signal is used primarily to latch the low-address from multiplexed bus & generate a separate set of address lines $A_7$-$A_0$.



# 8085 pin diagram: Control & Status Signals

## $\overline{RD}$(Read): Pin 32

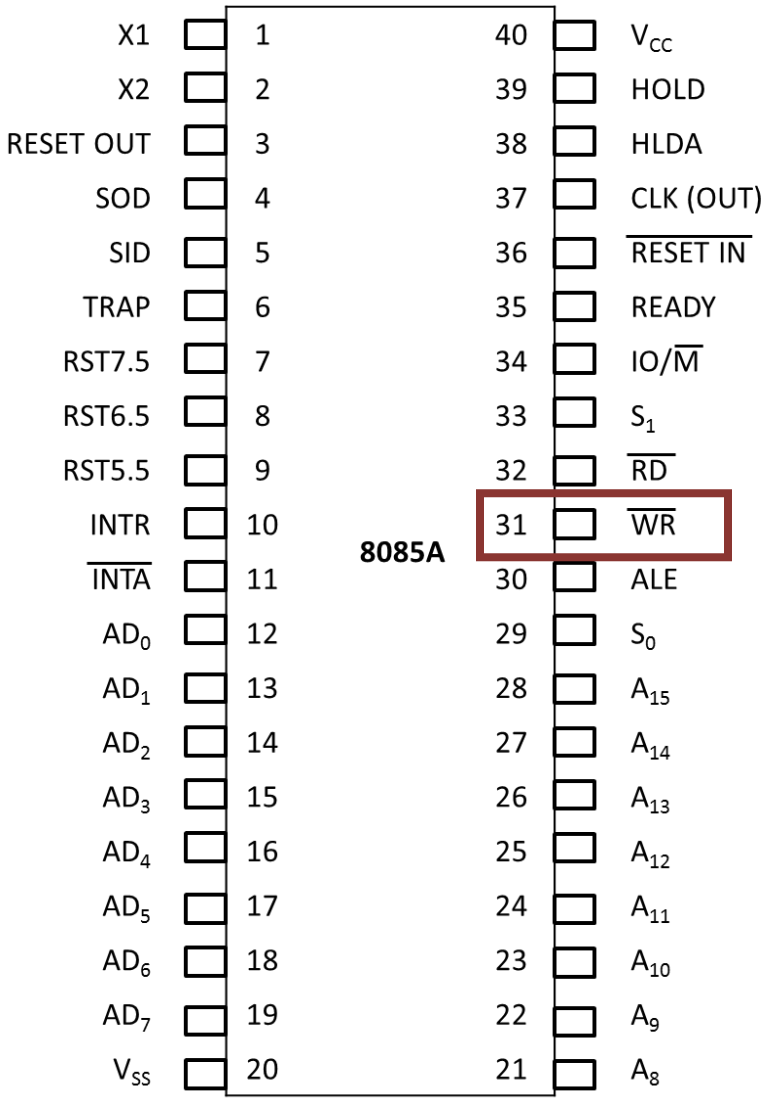
- This is a read control signal (active low)
- This signal indicates that the selected I/O or Memory device is to be read & data is available on data bus.



# 8085 pin diagram: Control & Status Signals

## $\overline{WR}$ (Write): Pin 31

- This is a write control signal (active low)
- This signal indicates that the selected I/O or Memory device is to be write & data is available on data bus.



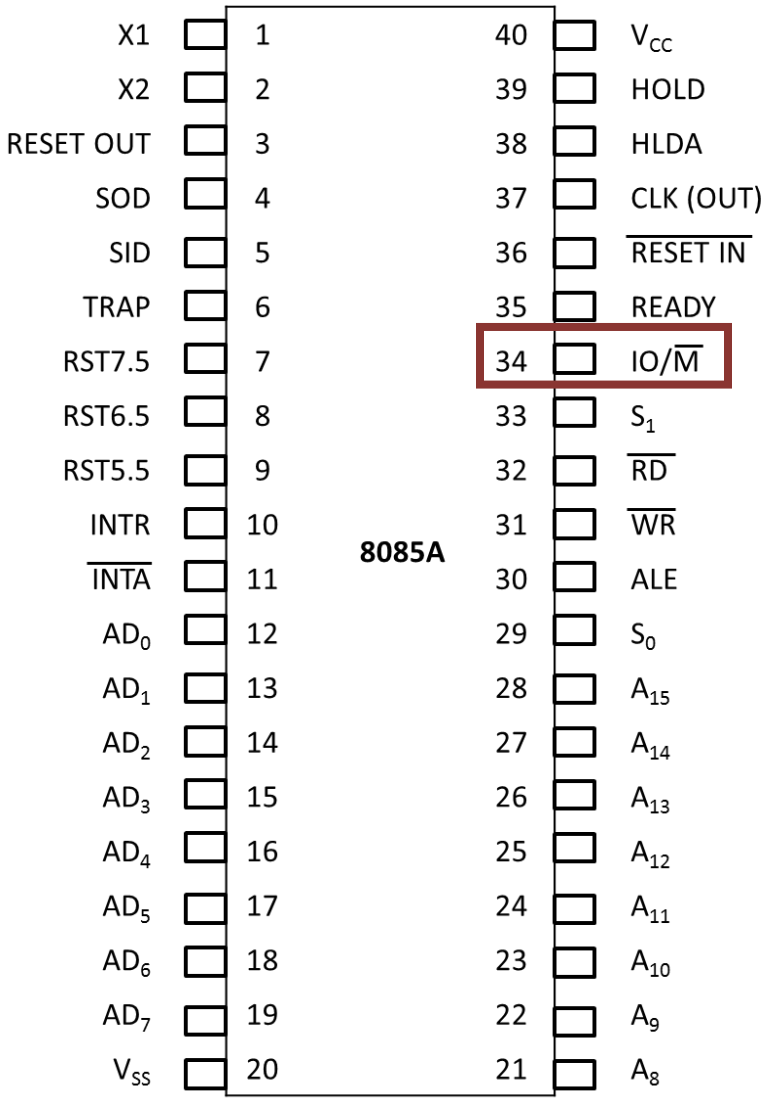
# 8085 pin diagram: Control & Status Signals

## IO/$\overline{M}$: Pin 34

- This is a status signal used to differentiate I/O and memory operation.
- When signal is

  high → I/O operation

  low → Memory operation
- This signal is combined with $\overline{RD}$ and $\overline{WR}$ to generate I/O & memory control signals.

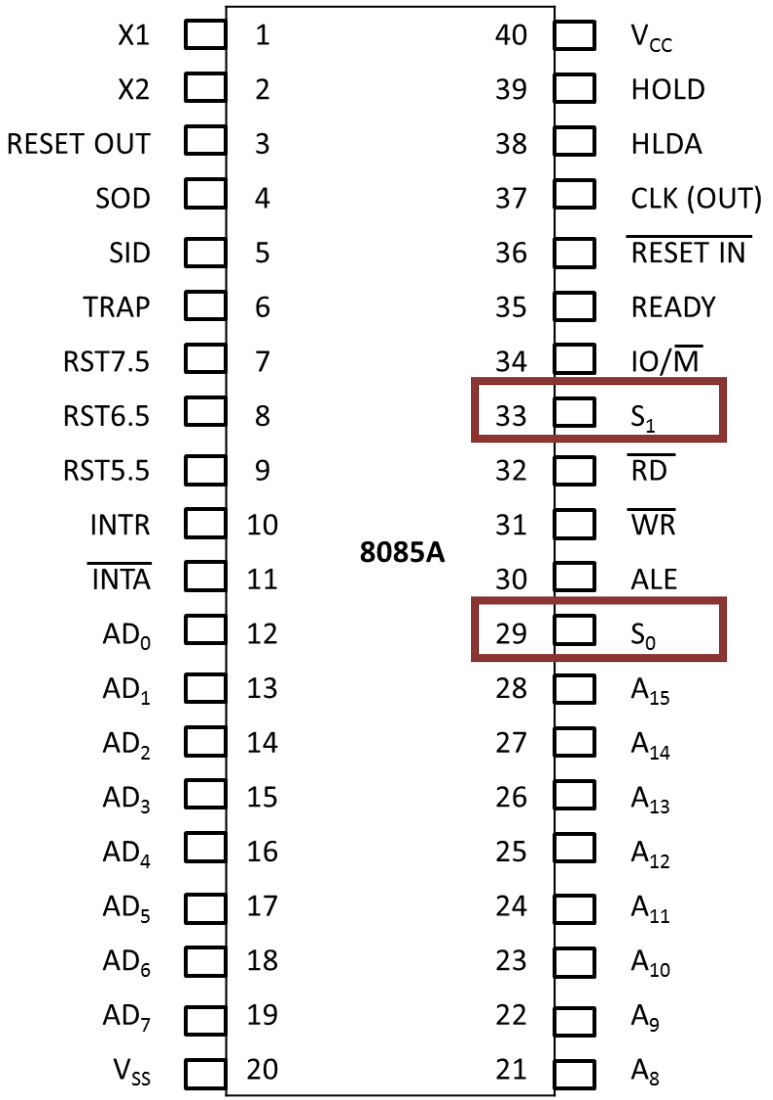


# 8085 pin diagram: Control & Status Signals

## $S_1$ (Pin 33) & $S_0$ (Pin 29)

- These status signals can identify various operations.

| $S_1$ | $S_0$ | Mode |
|---|---|---|
| 0 | 0 | HLT |
| 0 | 1 | WRITE |
| 1 | 0 | READ |
| 1 | 1 | OPCODE FETCH |

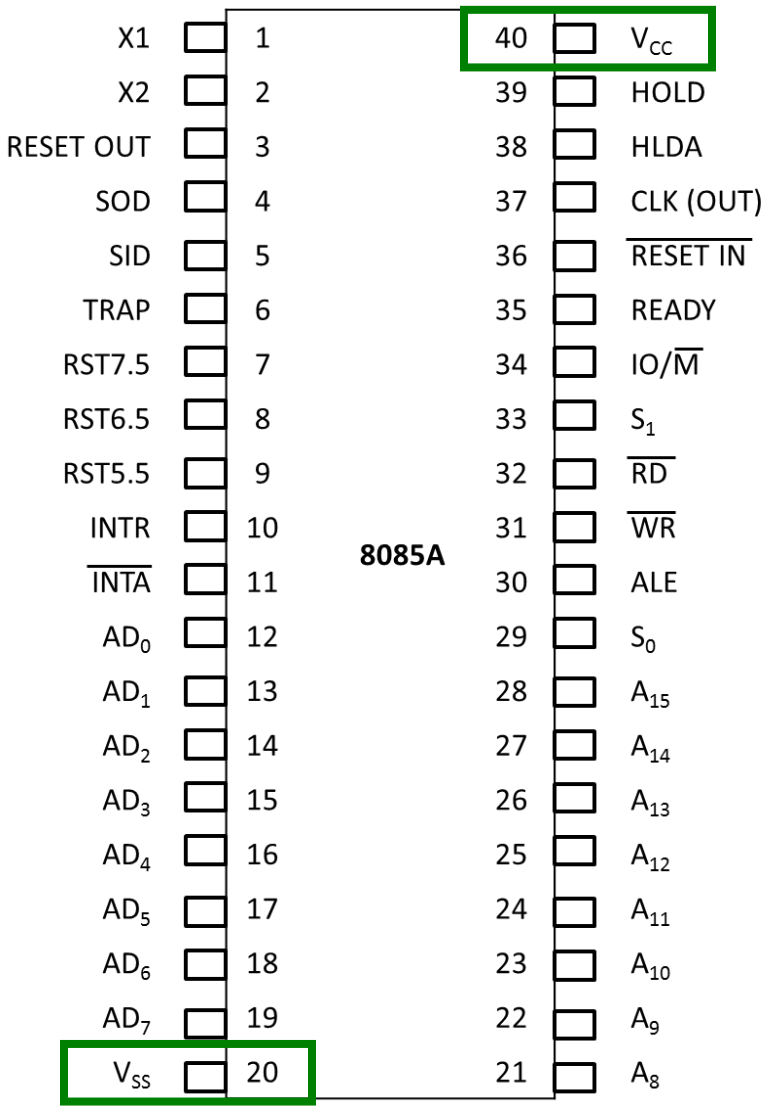


# 8085 pin diagram

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 0 | HLT |
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | Opcode Fetch |
| 1 | 0 | 0 | HLT |
| 1 | 0 | 1 | $\overline{IOR}$ |
| 1 | 1 | 0 | $\overline{IOW}$ |
| 1 | 1 | 1 | NOP |

# 8085 Pin Diagram: Power Supply & Frequency Signal

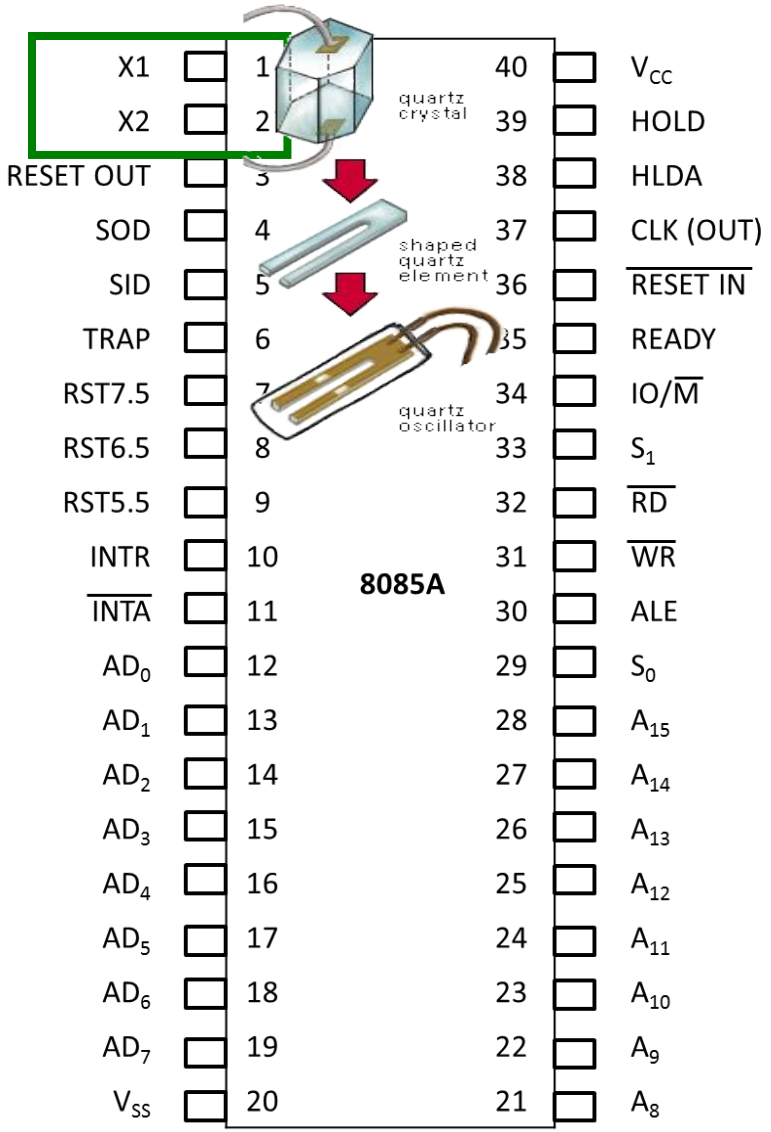
- $V_{cc}$ → Pin 40, +5V Supply.
- $V_{ss}$ → Pin 20, Ground Reference



# 8085 Pin Diagram: Power Supply & Frequency Signal

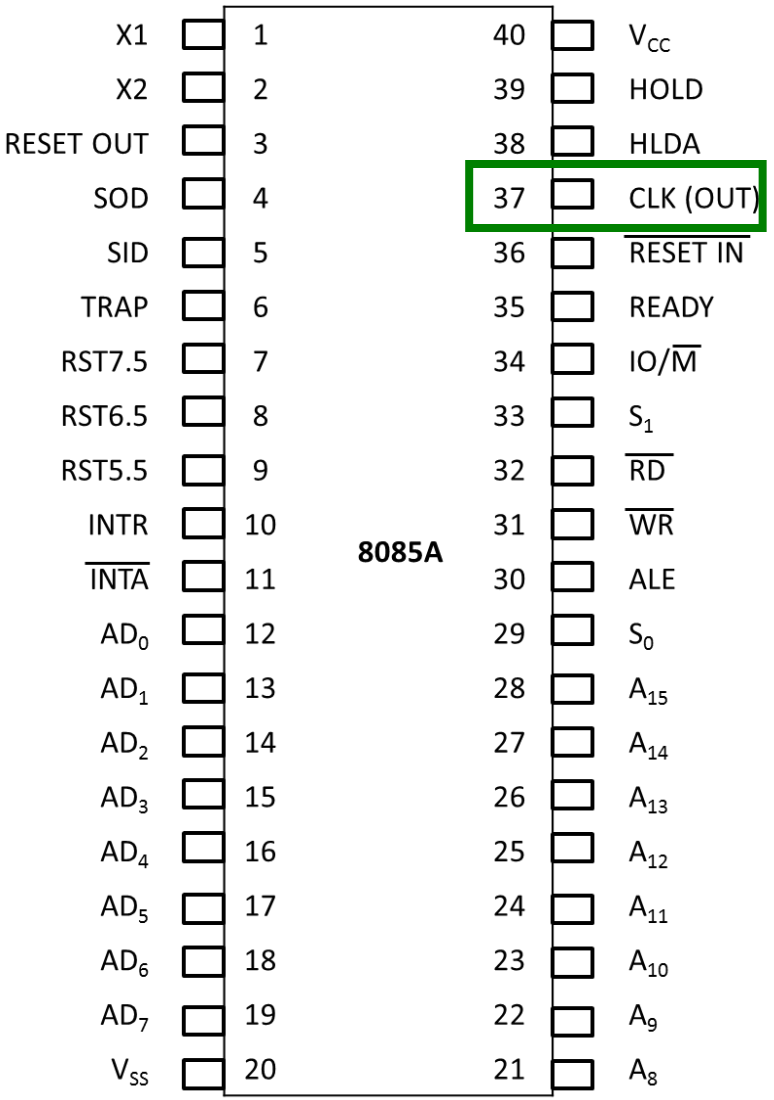
- **X1, X2** → Pin 1 & 2, Crystal Oscillator is connected at these two pins.
- The frequency is internally divided by two;

  therefore, to operate a system at 3MHz, the crystal should have a frequency of 6MHz.



# 8085 Pin Diagram: Power Supply & Frequency Signal

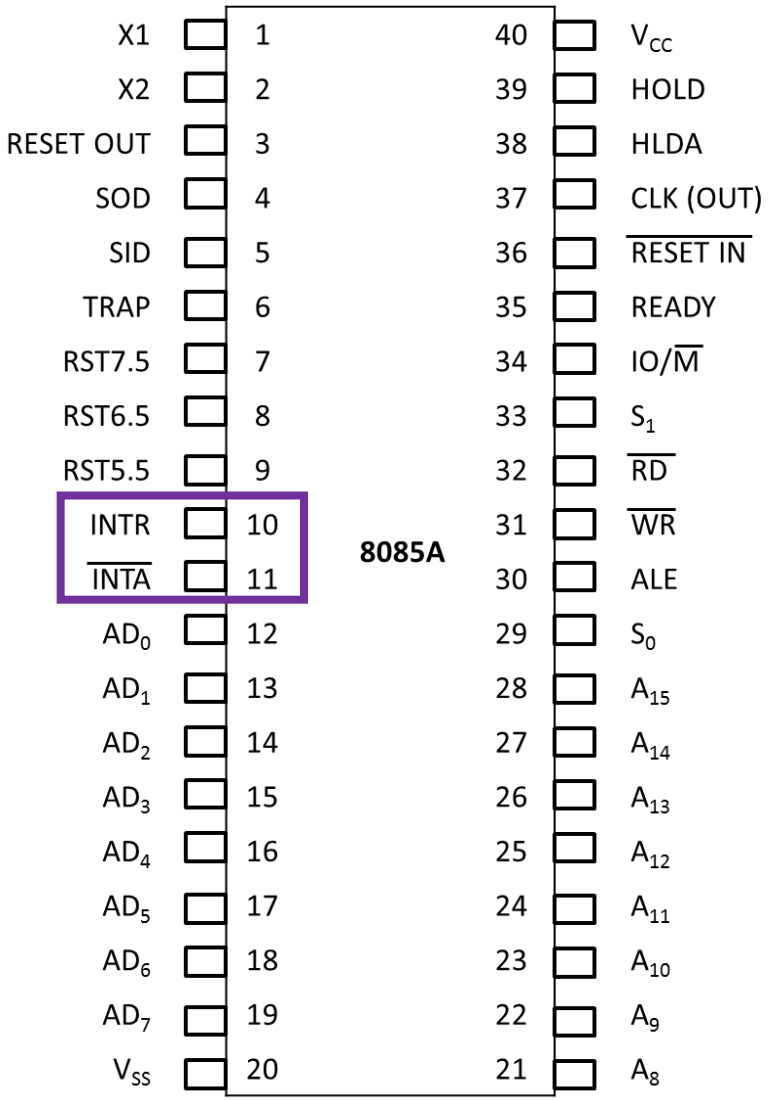
- **CLK (OUT)** → Clock output
- Pin 37: This signal is used as system clock for other I/O devices for synchronization with Microprocessor.



# 8085 Pin Diagram: Externally Initiated Signals

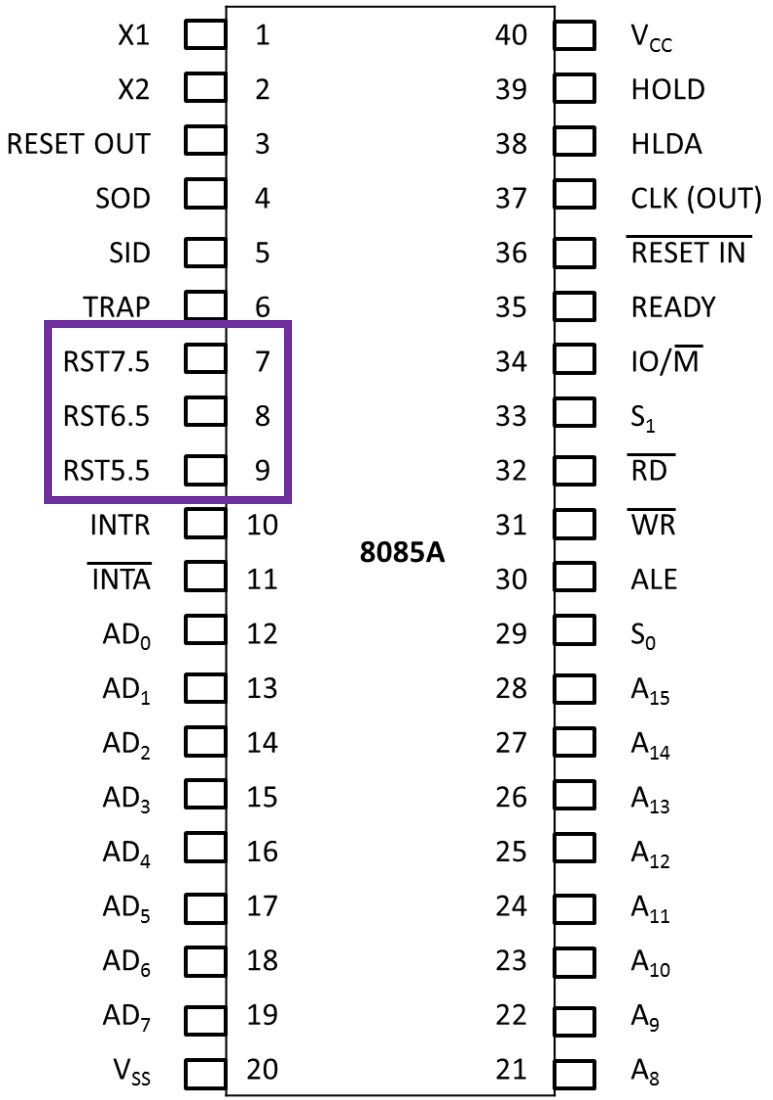
- **INTR**(Input) → Interrupt Request

  It is used for general purpose interrupt.
- **$\overline{\text{INTA}}$**(Output) → Interrupt Acknowledge.



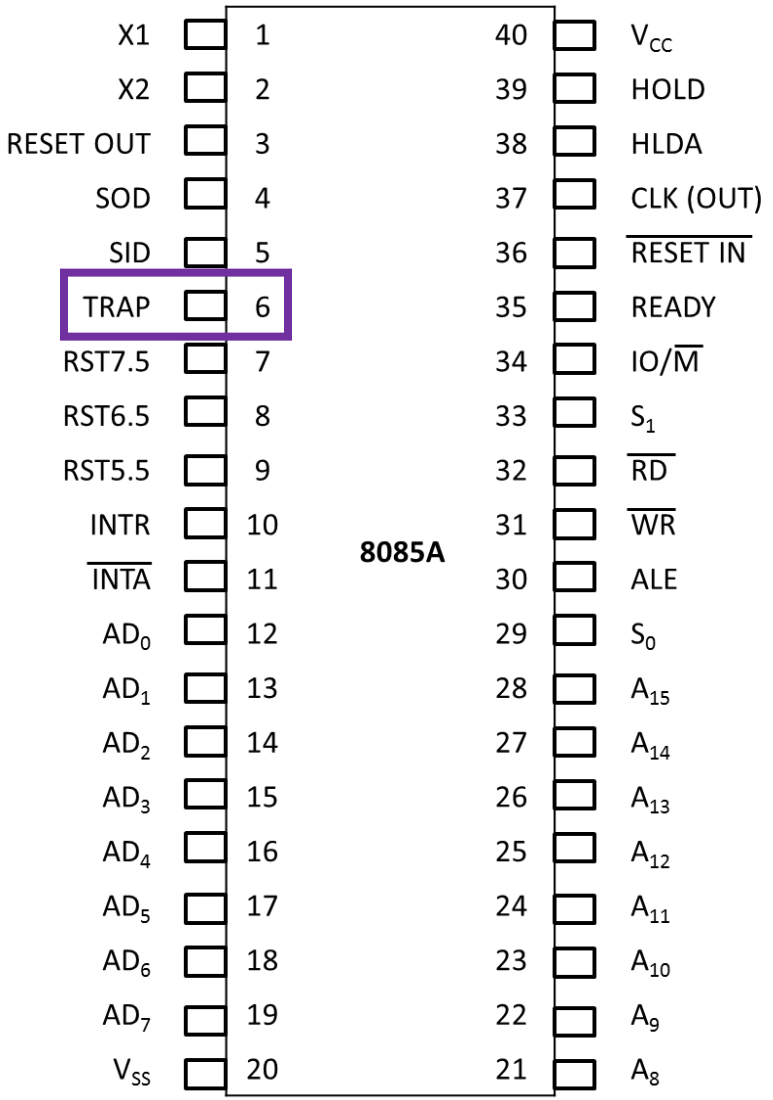
# 8085 Pin Diagram: Externally Initiated Signals

- **RST7.5, RST6.5, RST5.5** (Input) → Restart Interrupts.
- These are vector interrupts that transfer the program control to specific memory locations.
- RST7.5, RST6.5, RST5.5 have higher priorities than INTR interrupt.
- Among these 3 interrupts, the priority order (higher to lower) is RST7.5, RST6.5, RST5.5 respectively.



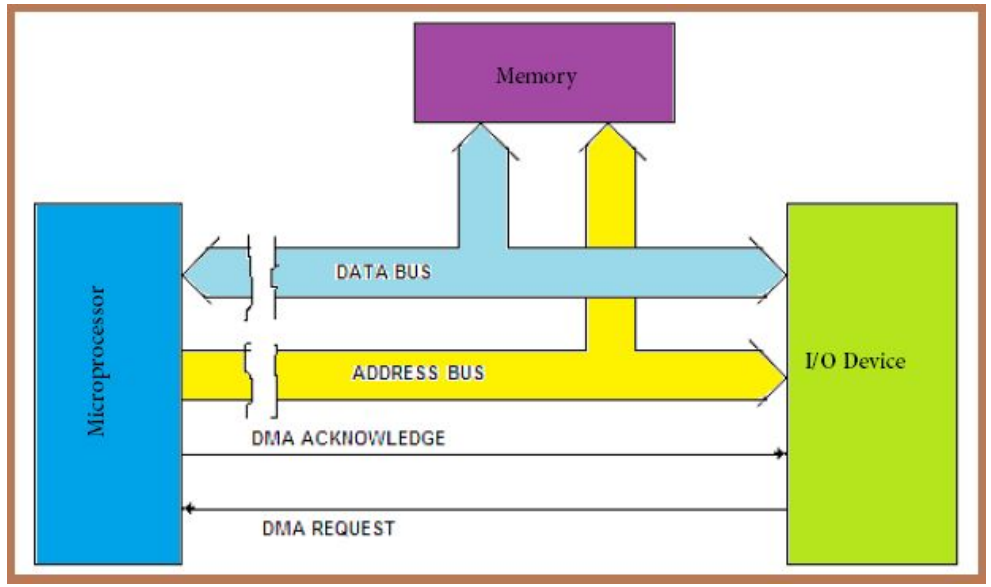
# 8085 Pin Diagram: Externally Initiated Signals

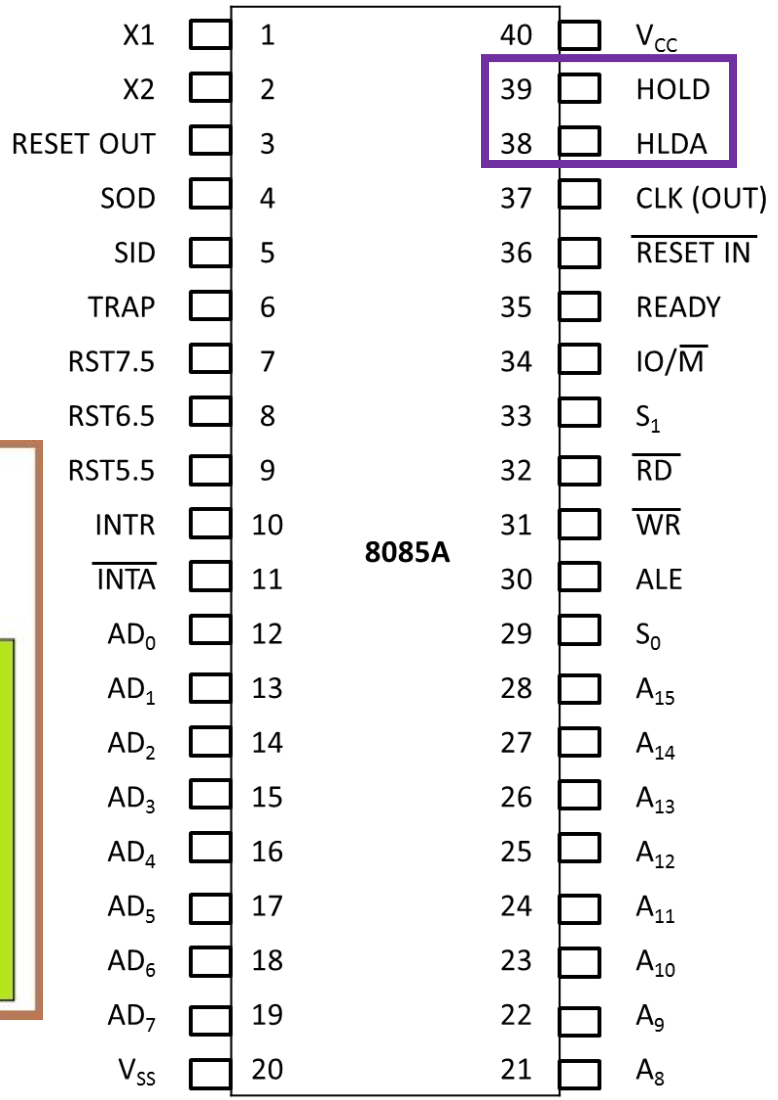
- **TRAP(Input)** → This is a non maskable interrupt & has the **highest priority**.



# 8085 Pin Diagram: Externally Initiated Signals

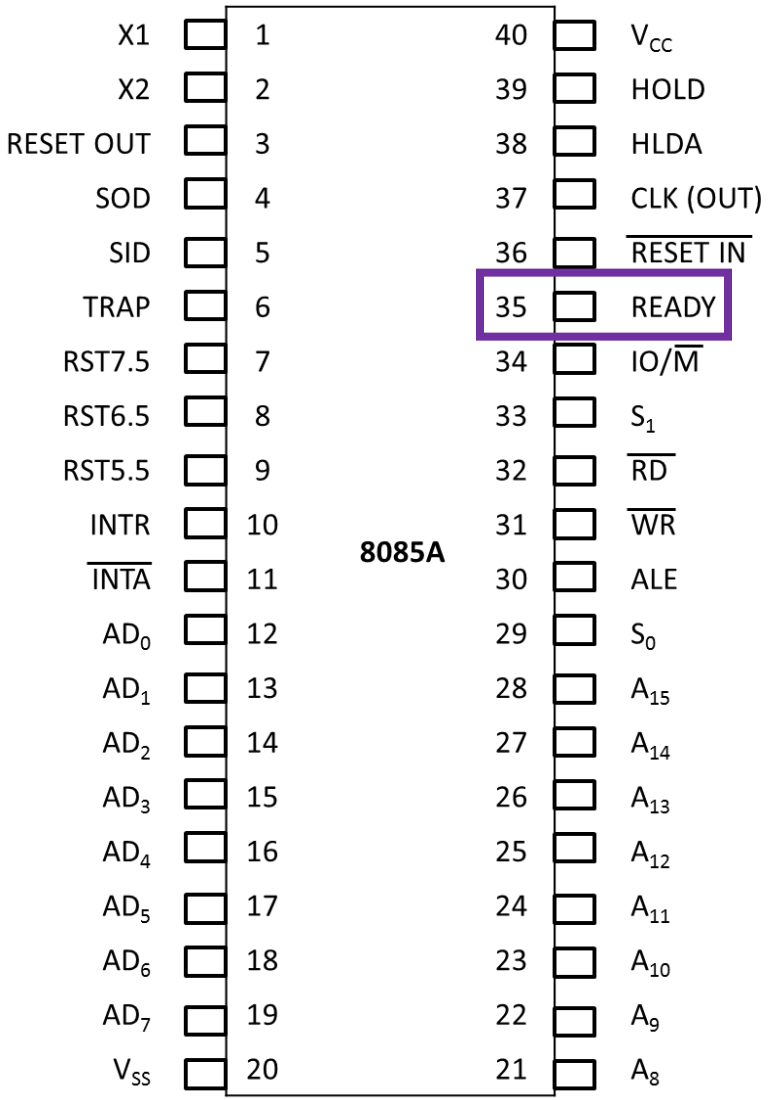
- **HOLD(Input)** → This signal indicates that a peripheral such as DMA Controller is requesting the use of address & data buses.
- **HLDA(Output)** → Hold Acknowledge. This signal acknowledges the HOLD request.



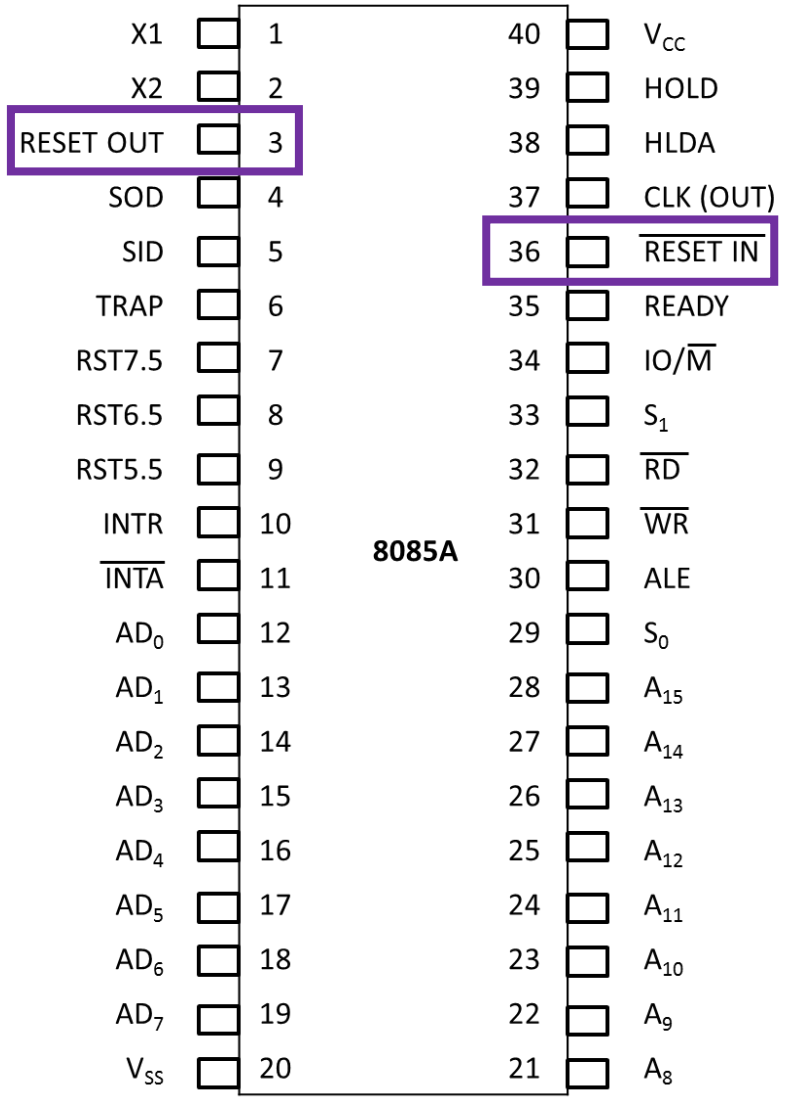
# 8085 Pin Diagram: Externally Initiated Signals

- **READY**(Input) → This signal is used to delay the microprocessor read or write cycles until low-responding peripheral is ready to send or accept data.
- When the signal goes low, the microprocessor waits for an integral no. of clock cycles until **READY** signal goes high.



# 8085 Pin Diagram: Externally Initiated Signals

- **$\overline{\text{RESET IN}}$** (Input) → When the signal on this pin goes low, the Program Counter is set to zero, the buses are tri-stated & microprocessor is reset.
- **RESET OUT** (Output) → This signal indicates that microprocessor is being reset. The signal is also used to reset other devices.

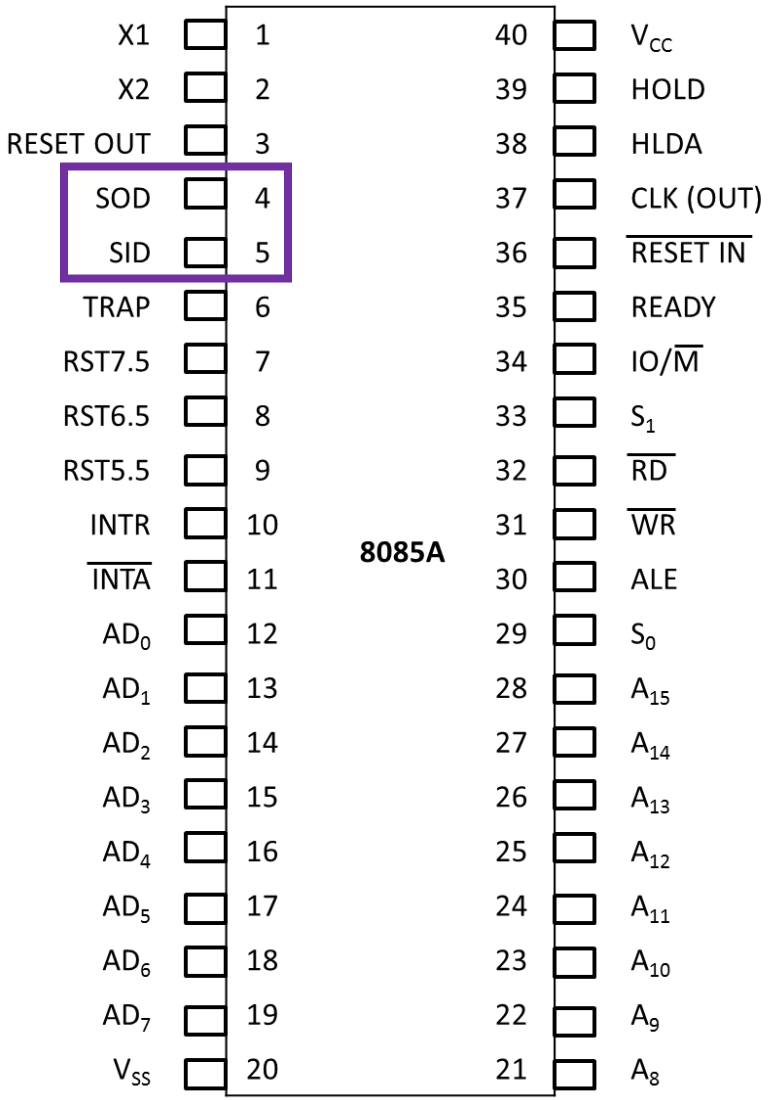


# 8085 Pin Diagram: Serial I/O Ports

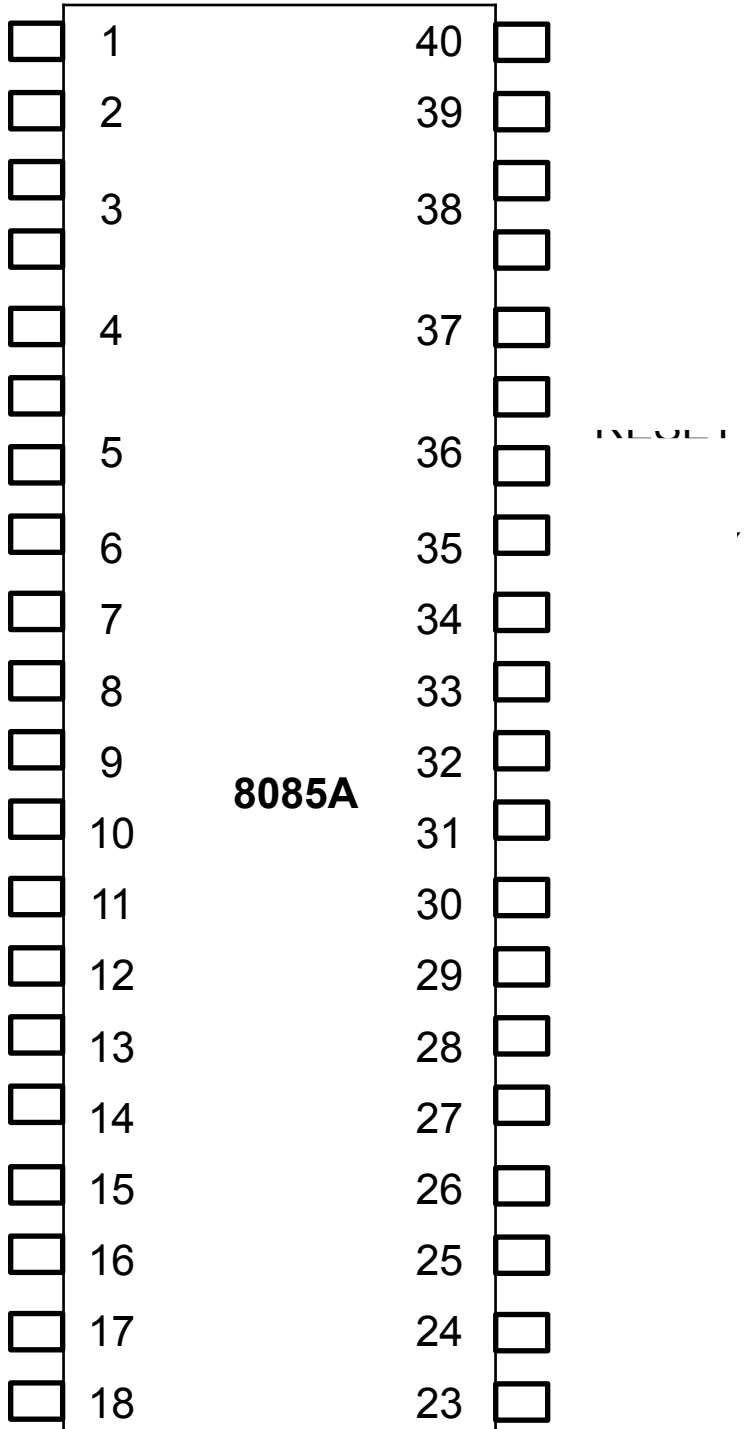
Two pins for serial transmission

1. **SID** (Serial Input Data)
2. **SOD** (Serial Output Data)

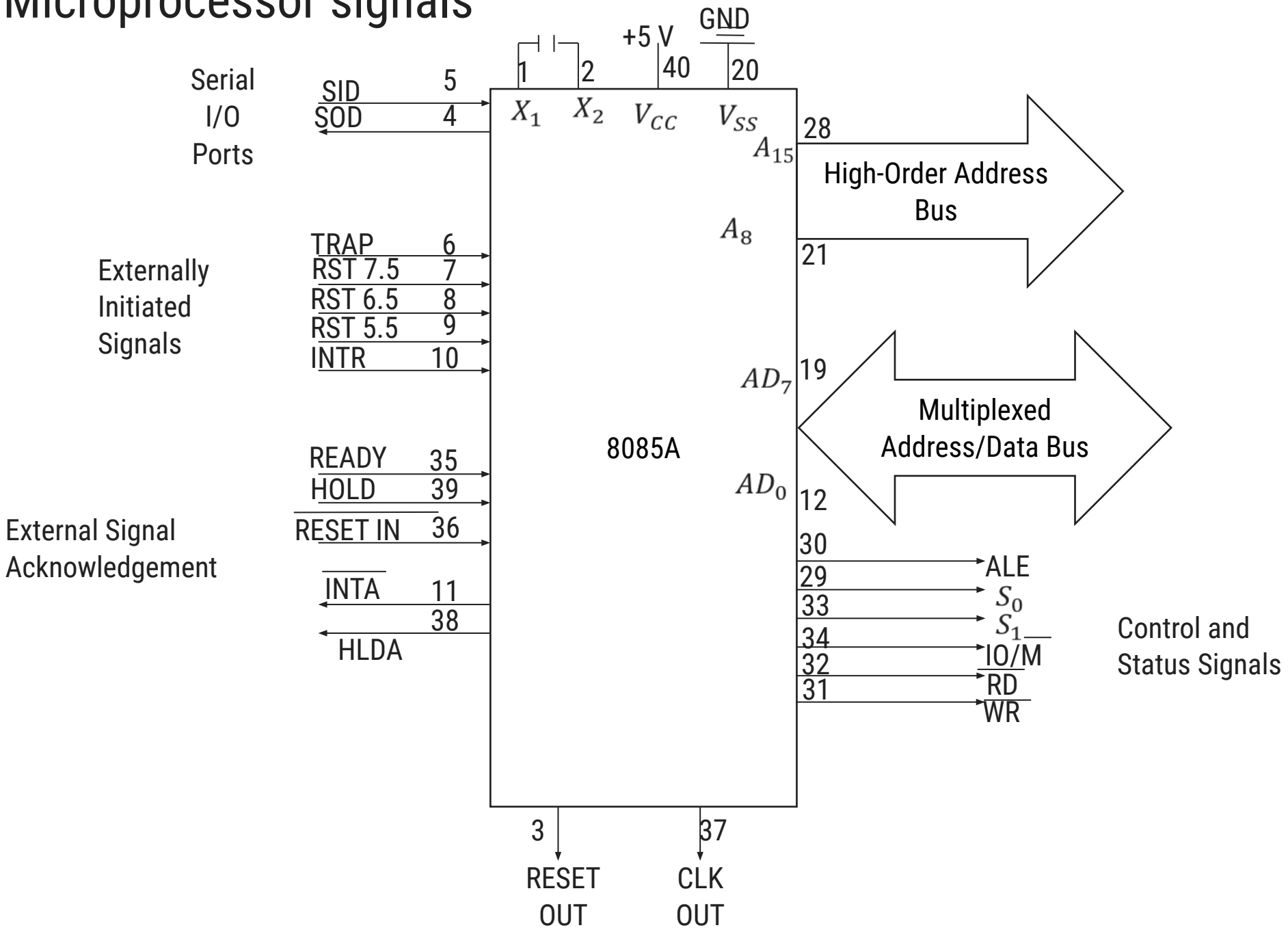
- In serial transmission, data bits are sent over a single line, one bit at a time.

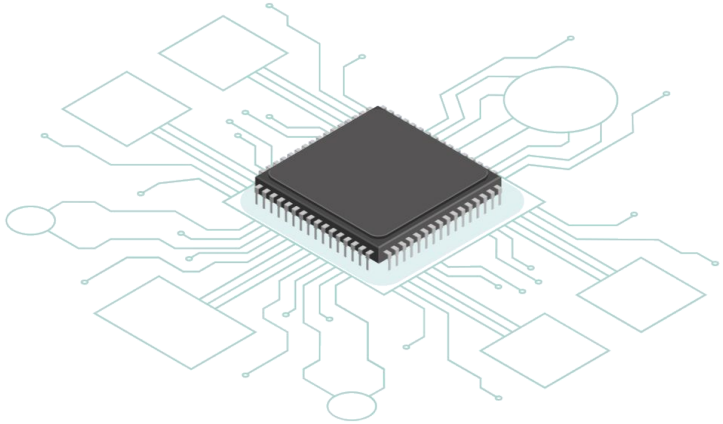


| Sr. | GTU Questions | Marks | Year |
|---|---|---|---|
| 1. | Draw and Explain the pin diagram of 8085 microprocessor. | 7 | Win-2019 |
| 2. | What is the use of ALE pin in 8085? | 1 | Sum-2019 |
| 3. | Explain the following pins of the 8085 microprocessor: IO/M , INTR, RESETIN | 3 | Win-2017 |



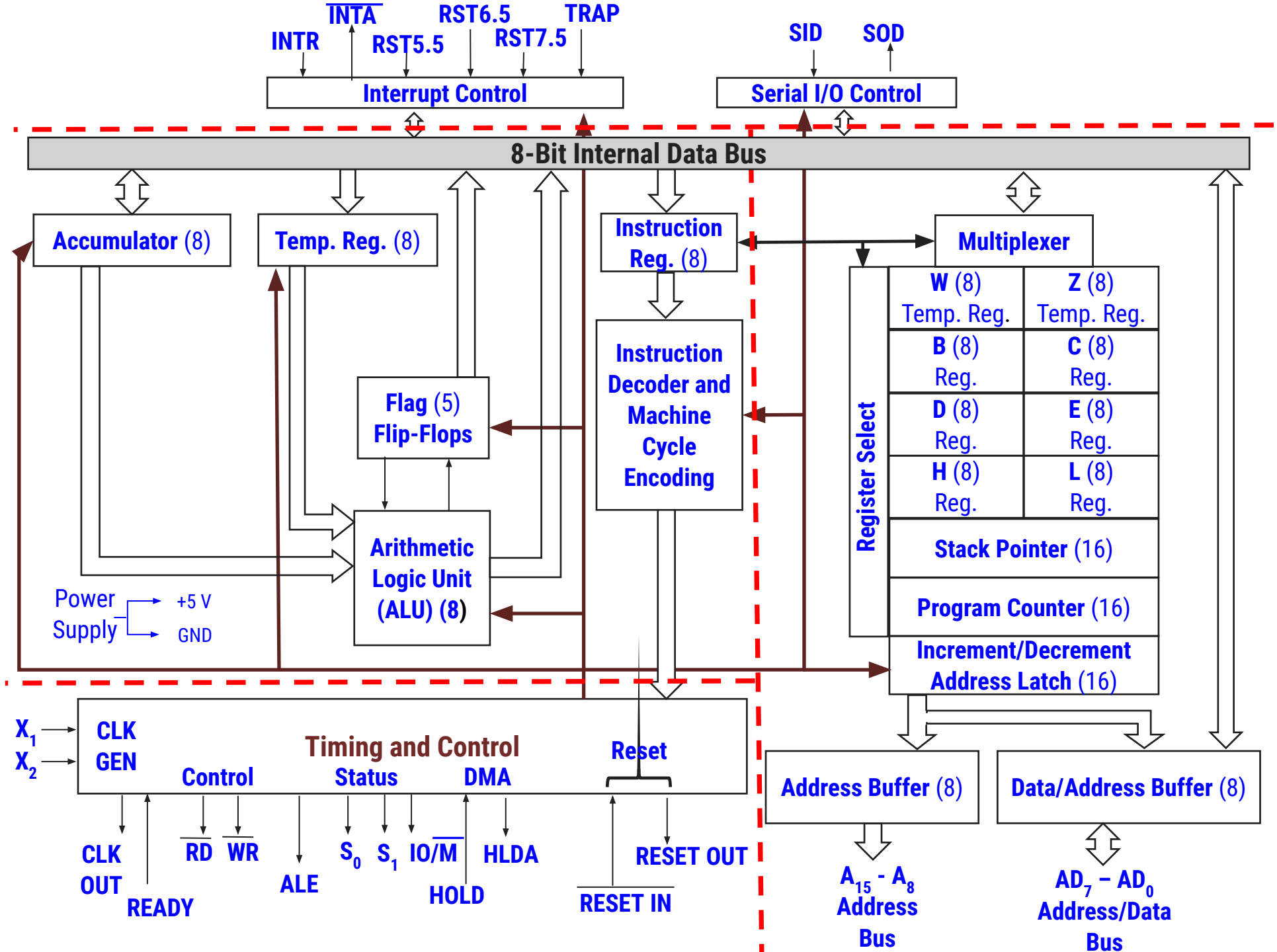
# 8085 Microprocessor signals



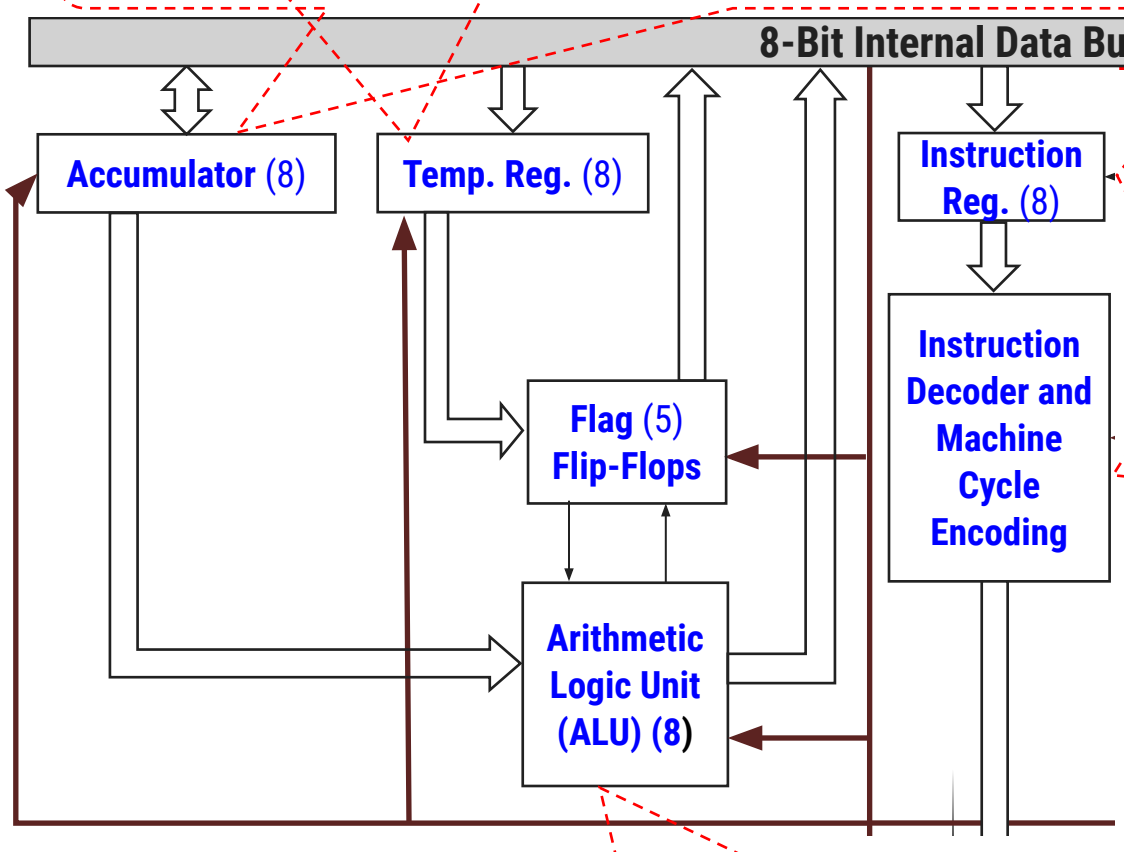


# 8085 Architecture/Block Diagram

# 8085 Block Diagram



- Used to hold data (i.e. temporary data) during ALU operation.
- Used to store 8-bit data to perform arithmetic & logical operations.
- Result of operation is stored in Accumulator.



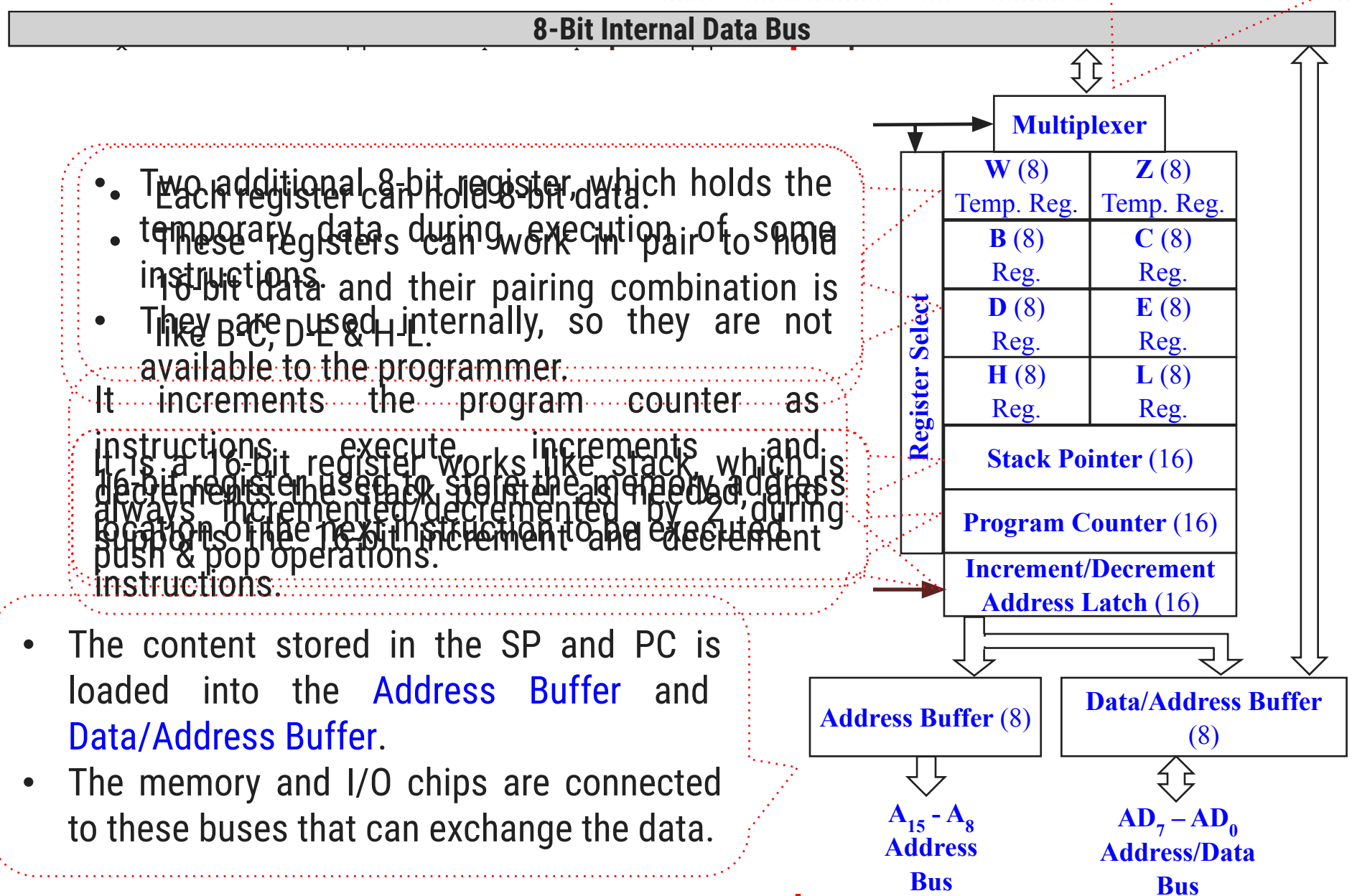

When Instruction is fetched from memory, it is loaded in the Instruction Registor (IR).

S Z AC P CY

Instruction decoder decodes the information present in the Instruction register.

- Performs Computing Functions.
- Accumulator, Temporary Register and Flag Registers are part of ALU.

A multiplexer pulls out the right group of bits, depending on the instruction.

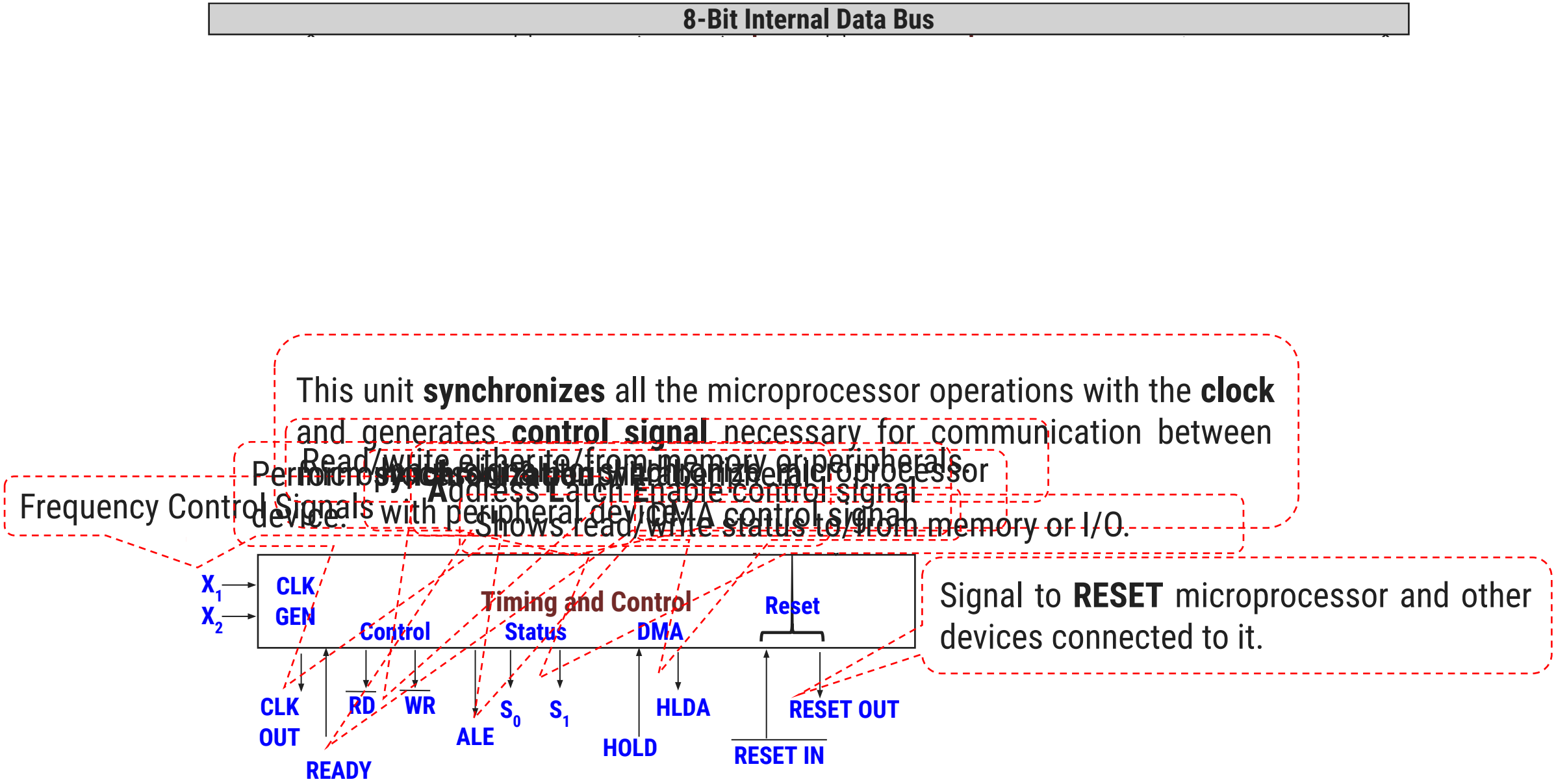



- Each register can hold 8-bit data.
- These registers can work in pair to hold 16-bit data and their pairing combination is like B-C, D-E & H-L.
- They are used internally, so they are not available to the programmer.

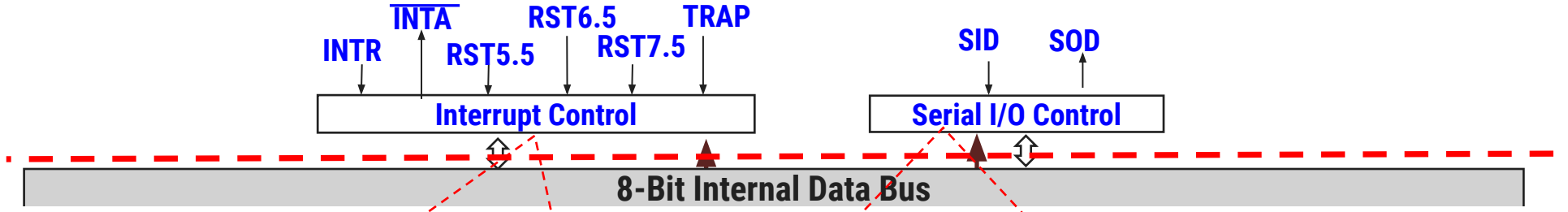
It increments the program counter as instructions execute increments and decrements the stack pointer as needed, and supports the 16-bit increment and decrement instructions.

It is a 16-bit register works like stack, which is always incremented/decremented by 2 during push & pop operations.

- The content stored in the SP and PC is loaded into the Address Buffer and Data/Address Buffer.
- The memory and I/O chips are connected to these buses that can exchange the data.


8-Bit Internal Data Bus
This unit synchronizes all the microprocessor operations with the clock and generates control signal necessary for communication between
Read/write either to/from memory or peripherals
Performs synchronization between internal microprocessor
Address Latch Enable control signal
device.
Frequency Control Signals
with peripheral devices
DMA control signal
Shows read/write status to/from memory or I/O.
X1
X2
CLK GEN
Timing and Control
Control
Status
DMA
Reset
Signal to RESET microprocessor and other devices connected to it.
CLK OUT
READY
RD
WR
ALE
S0
S1
HOLD
HLDA
RESET IN
RESET OUT

8-Bit Internal Data Bus

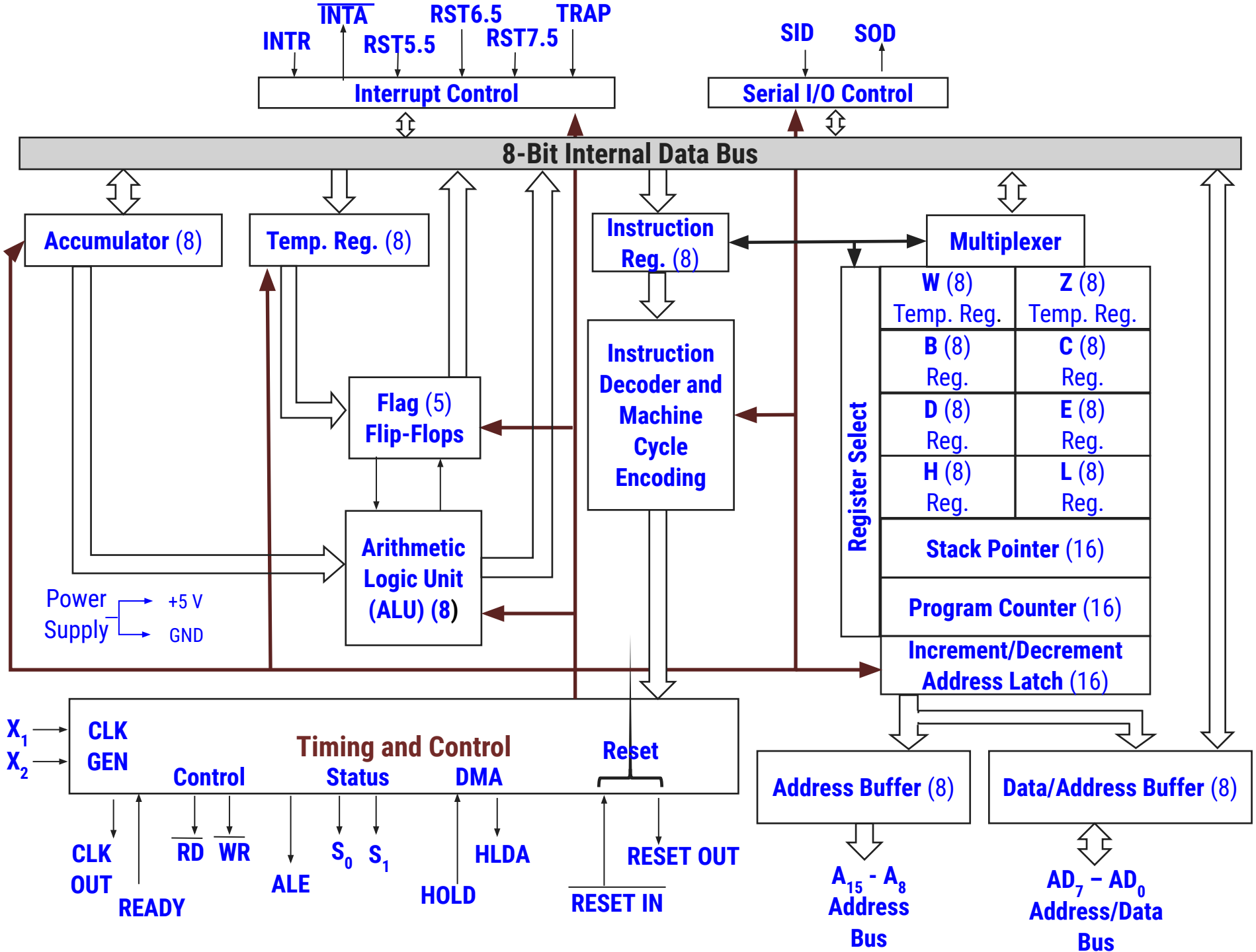



- It controls the interrupts during a process.
- There are 5 interrupt signals in 8085 microprocessor: TRAP, RST 7.5, RST 6.5, RST 5.5, INTR.

It controls the serial data communication by using two instructions:

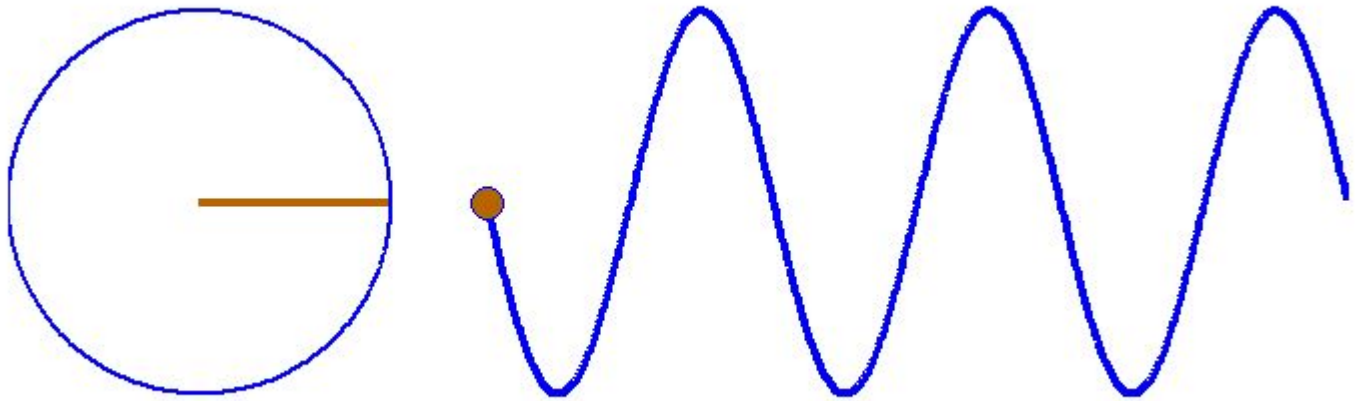
i. SID (Serial input data)

ii. SOD (Serial output data)


INTR
INTA
RST5.5
RST6.5
RST7.5
TRAP
Interrupt Control
SID
SOD
Serial I/O Control
8-Bit Internal Data Bus
Accumulator (8)
Temp. Reg. (8)
Instruction Reg. (8)
Multiplexer
Flag (5) Flip-Flops
Instruction Decoder and Machine Cycle Encoding
Arithmetic Logic Unit (ALU) (8)
Register Select
W (8) Temp. Reg.
Z (8) Temp. Reg.
B (8) Reg.
C (8) Reg.
D (8) Reg.
E (8) Reg.
H (8) Reg.
L (8) Reg.
Stack Pointer (16)
Program Counter (16)
Increment/Decrement Address Latch (16)
Power Supply
+5 V
GND
X1
X2
CLK GEN
Timing and Control
Control
Status
DMA
Reset
CLK OUT
READY
RD
WR
ALE
S0
S1
HOLD
HLDA
RESET IN
RESET OUT
Address Buffer (8)
Data/Address Buffer (8)
A15 - A8 Address Bus
AD7 – AD0 Address/Data Bus

# GTU Exam Questions

| Sr. | GTU Questions | Marks | Year |
|---|---|---|---|
| 1. | Draw and explain the internal block diagram of 8085 microprocessor. | 7 | Sum-2019<br>Win-2018<br>Sum-2018 |



# T-States, Machine and Instruction Cycle

# T-States, Machine and Instruction Cycle

**T-States**

Operation performed in one **clock period**.

**Machine Cycle**

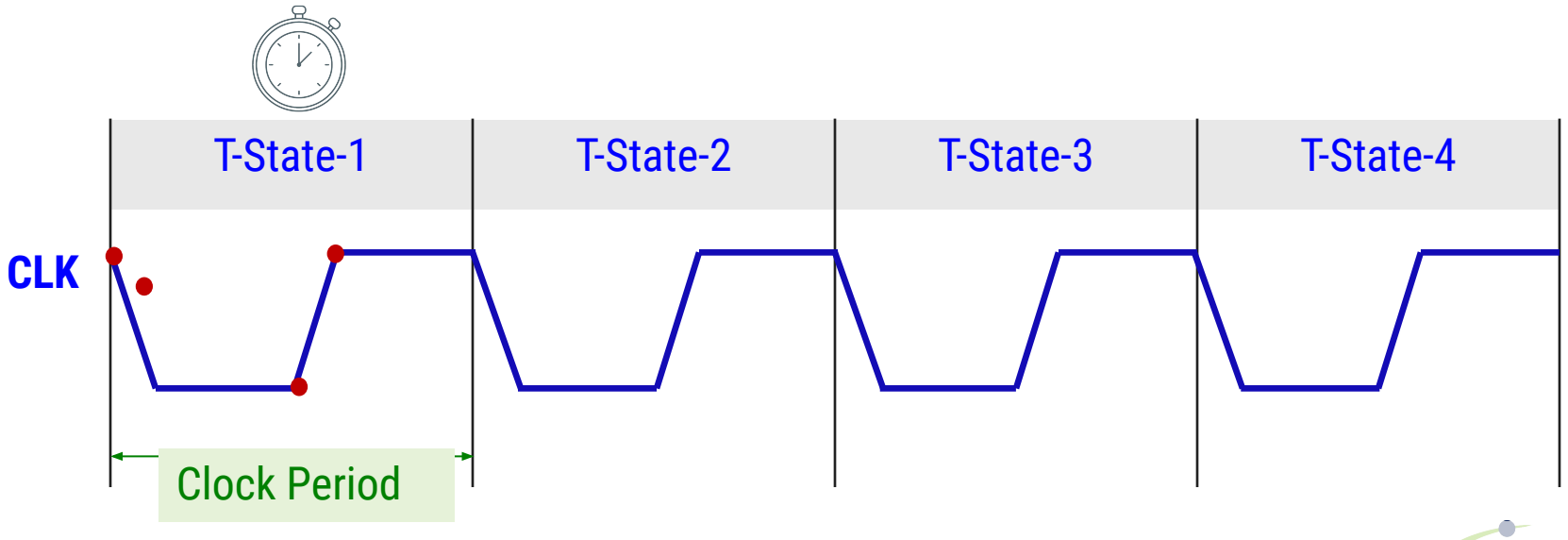
Time required by the microprocessor to complete an **operation**.

**Instruction Cycle**

Time required to complete **execution** of an instruction.

# T-States

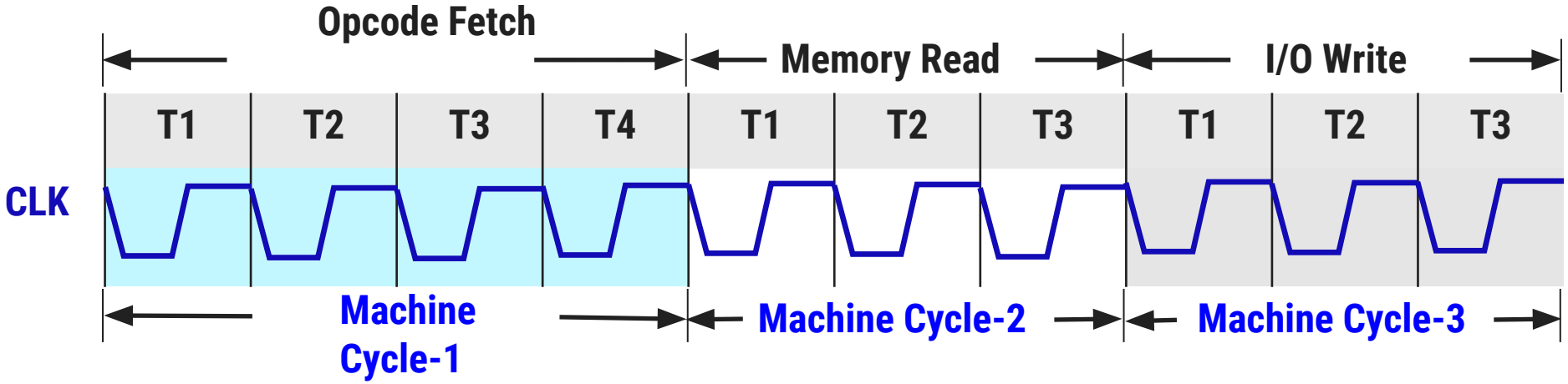
- *"T-States are defined as operation performed in one clock period."*
- These sub-divisions are internal states synchronized with system clock & each T-state is precisely equal to **one clock period**.



# Machine Cycle

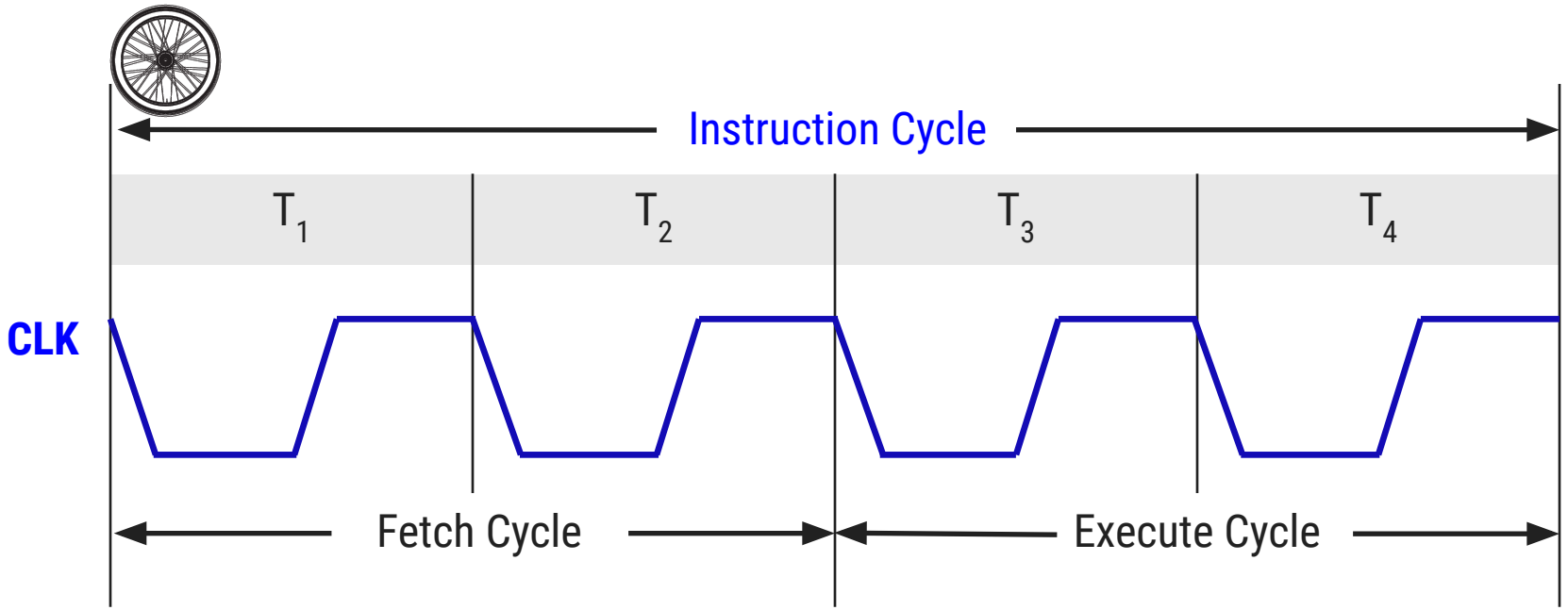
- *"Machine Cycle is defined as time required by the microprocessor to complete an operation."*
- This cycle may consist 3 to 6 T-states.

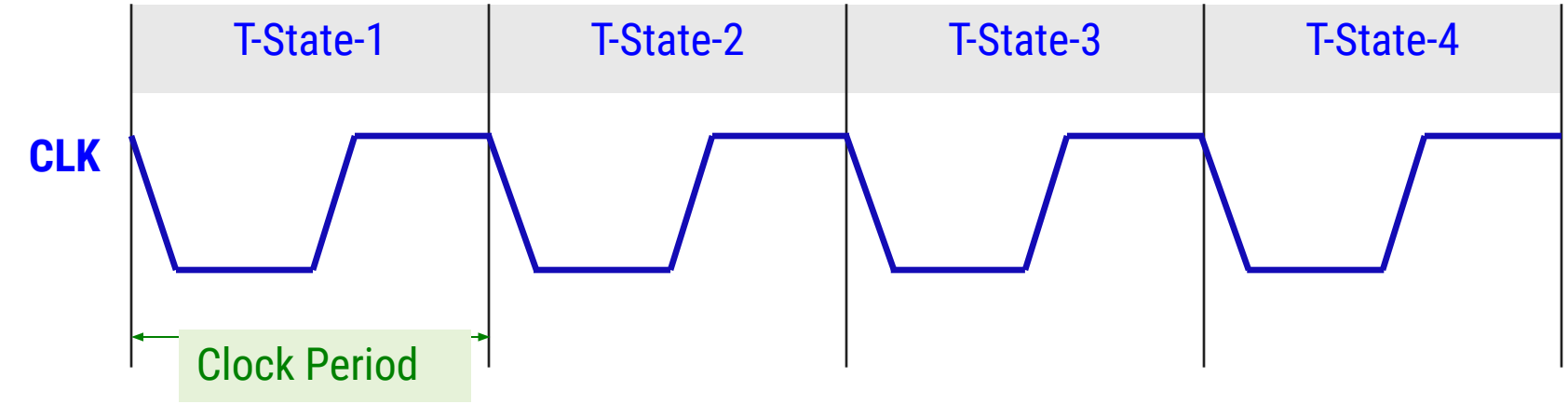

- The basic microprocessor operation such as reading a byte from **I/O port** or writing a byte to **memory**.



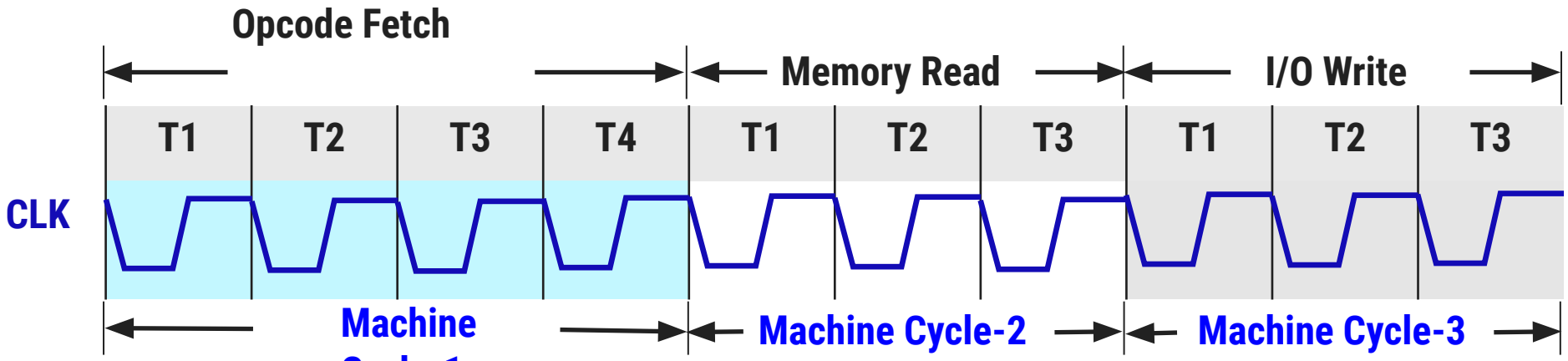
# Instruction Cycle

- *"Instruction Cycle is defined as time required to complete execution of an instruction."*
- In 8085 microprocessor instruction cycle consists of 1 to 6 Machine Cycles or 1 to 6 operations.

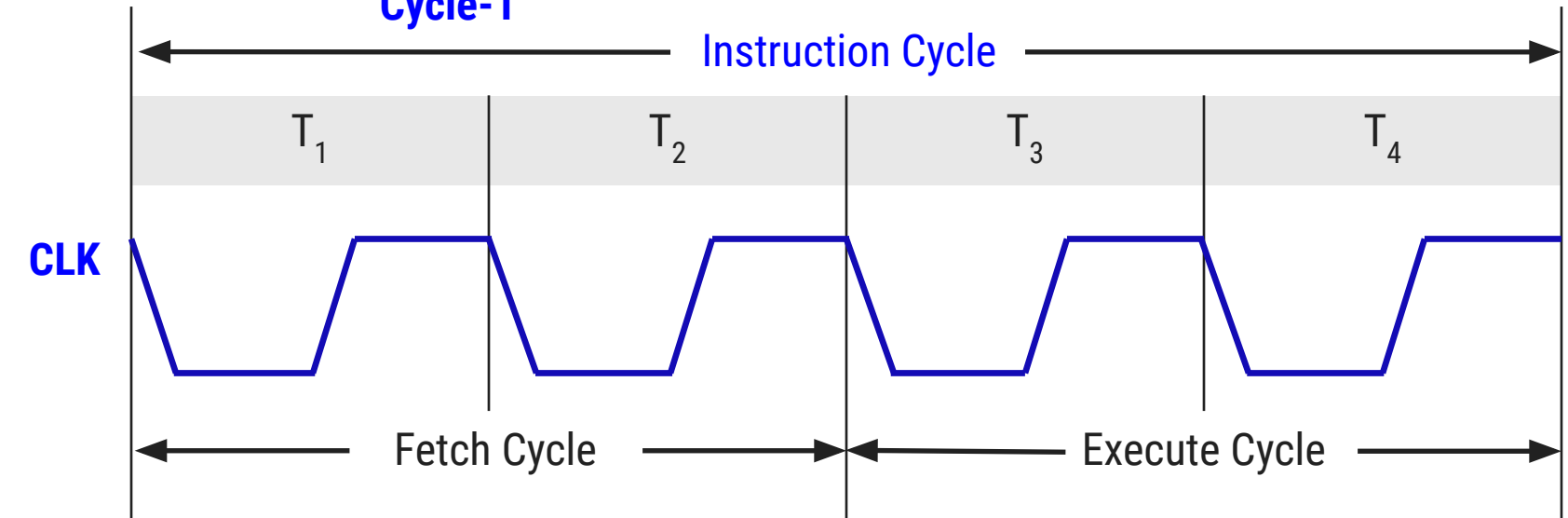




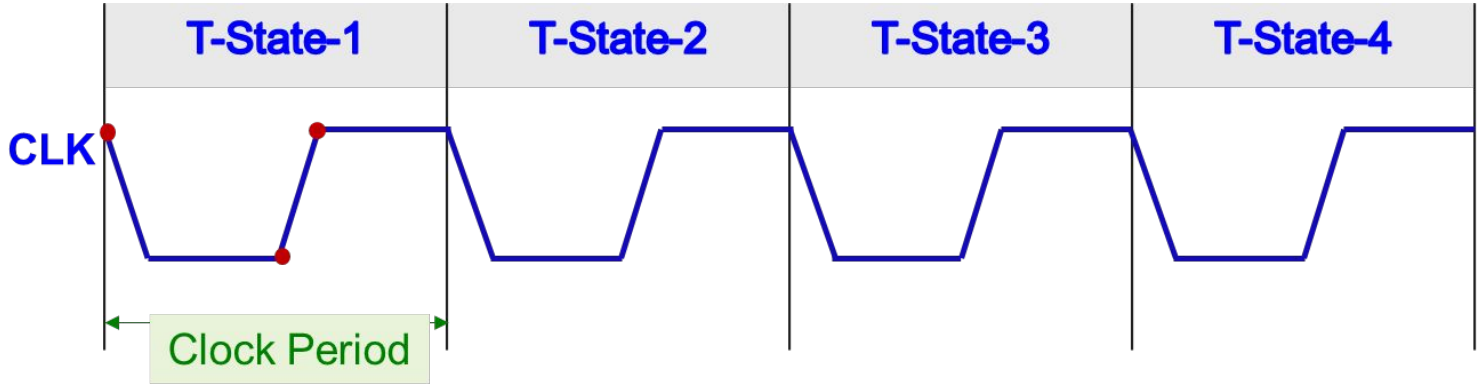

## T-States



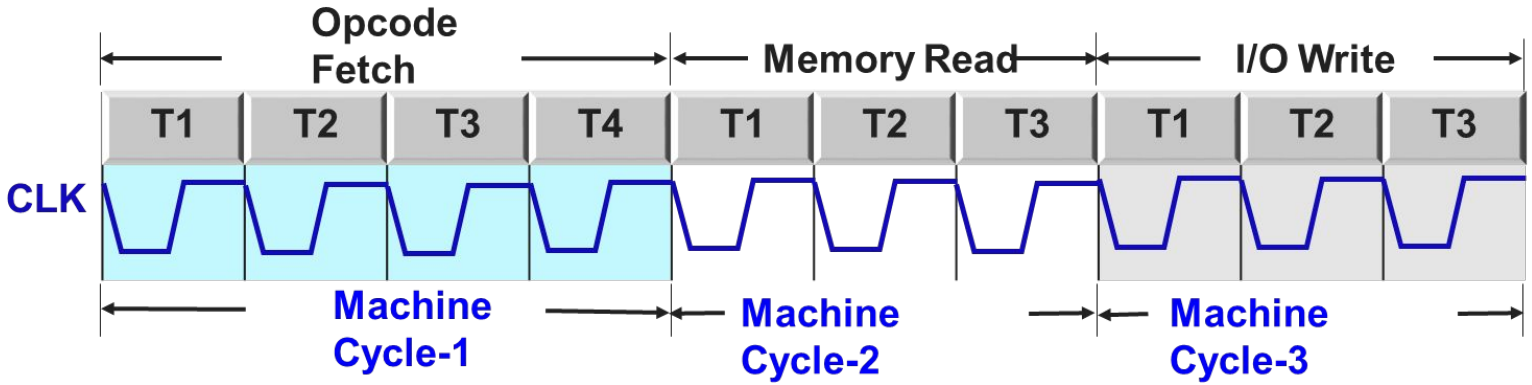

## Machine Cycle



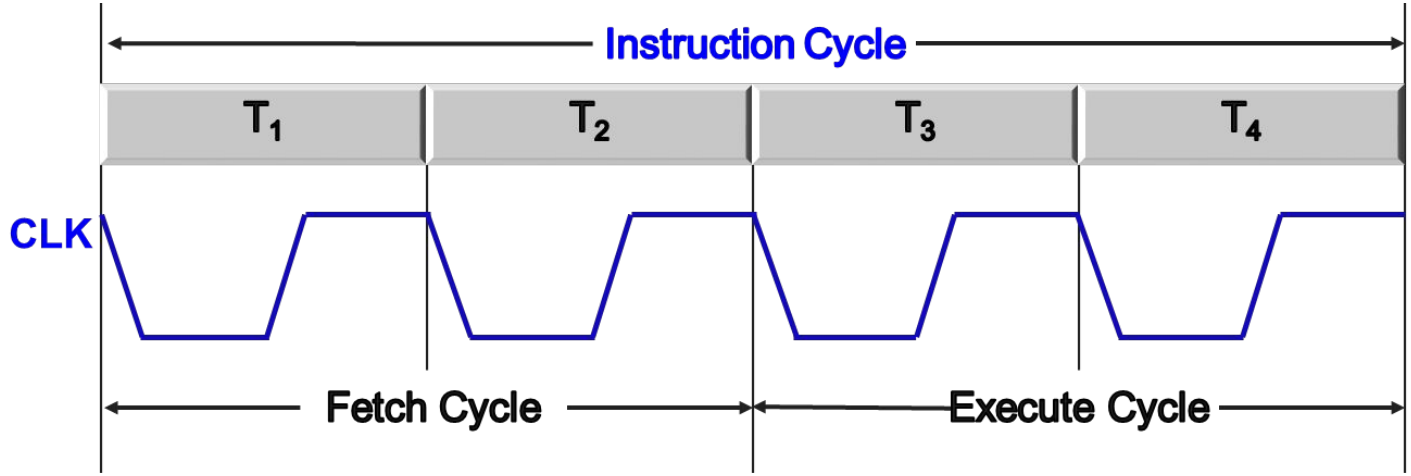

## Instruction Cycle



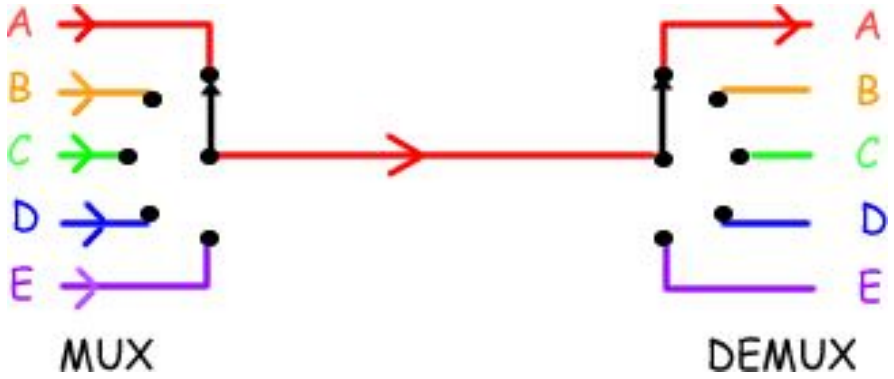

T-States




Machine Cycle

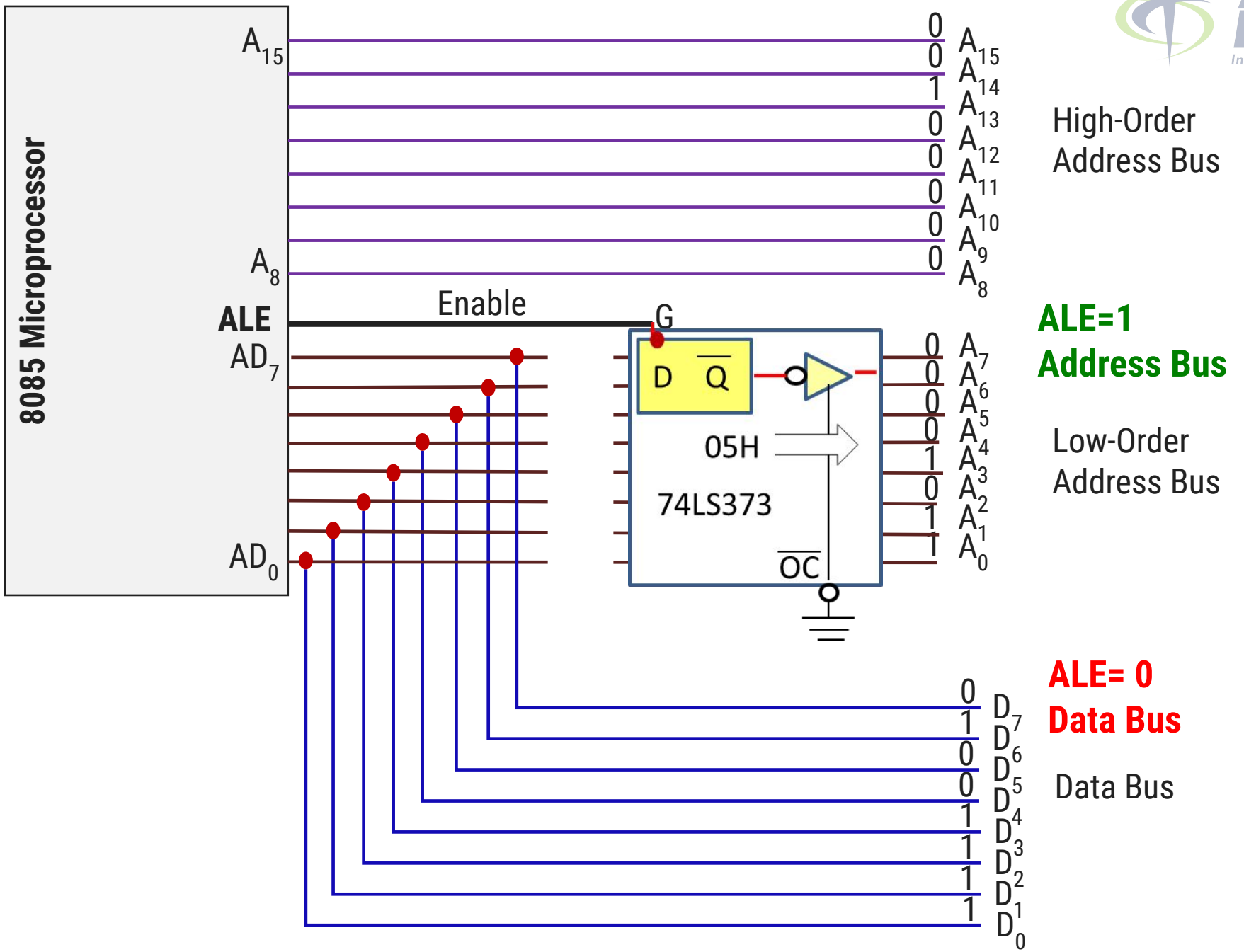



Instruction Cycle




# Demultiplexing Address and Data Bus $AD_0$-$AD_7$

# Demultiplexing $AD_0$-$AD_7$




High-Order Address Bus

**ALE=1**
**Address Bus**

Low-Order Address Bus

**ALE= 0**
**Data Bus**

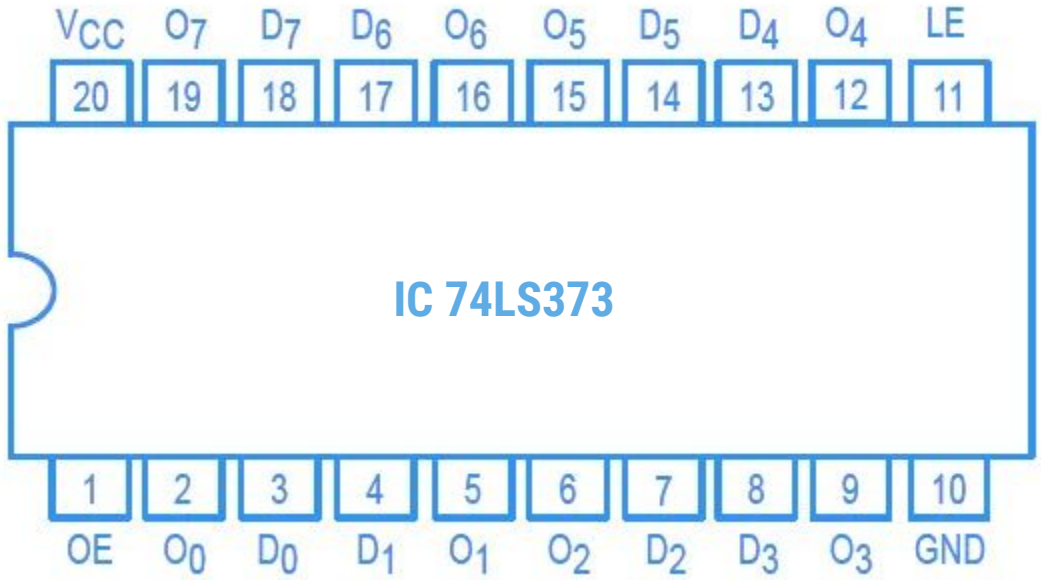
Data Bus

# Demultiplexing $AD_0$-$AD_7$

- The dual-purpose of the $AD_0$-$AD_7$ pins is achieved through multiplexing.
- In simple words, multiplexing allows us to use the pins of a microprocessor for more than one function.
- **Advantage**: Since each pin can serve multiple purposes, the total number of pins can be reduced.

## IC 74LS373

- IC 74LS373 is an IC with 20 pins.
- it is a memory unit to hold one bit of data.

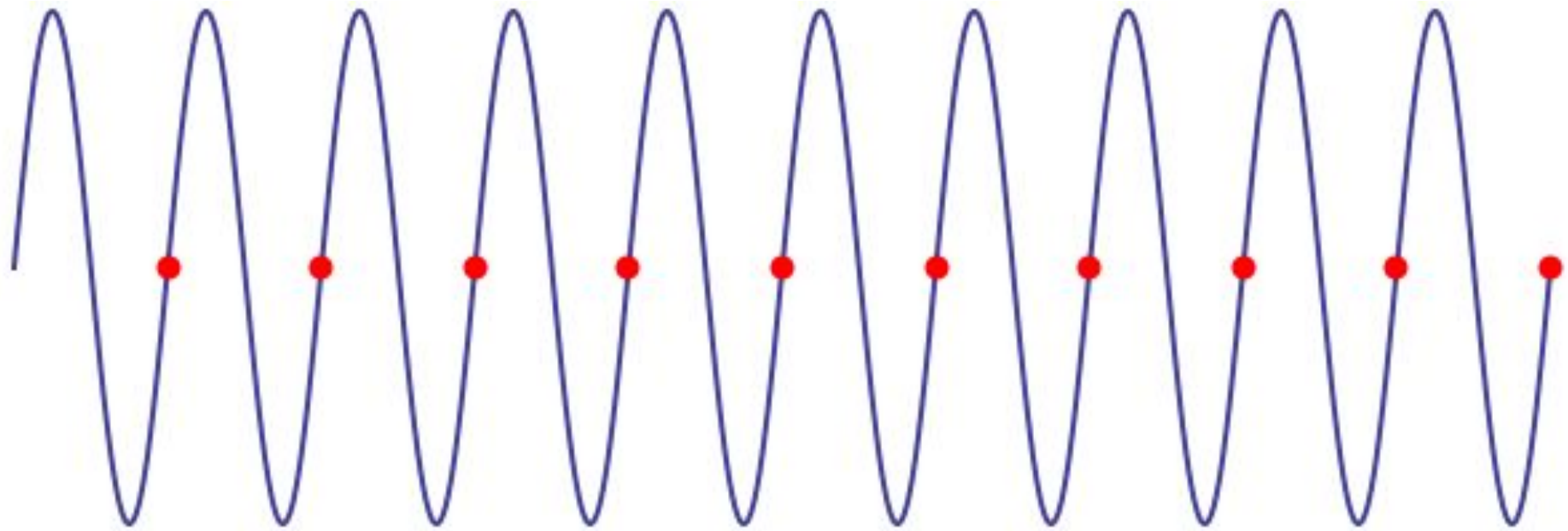


# GTU Questions

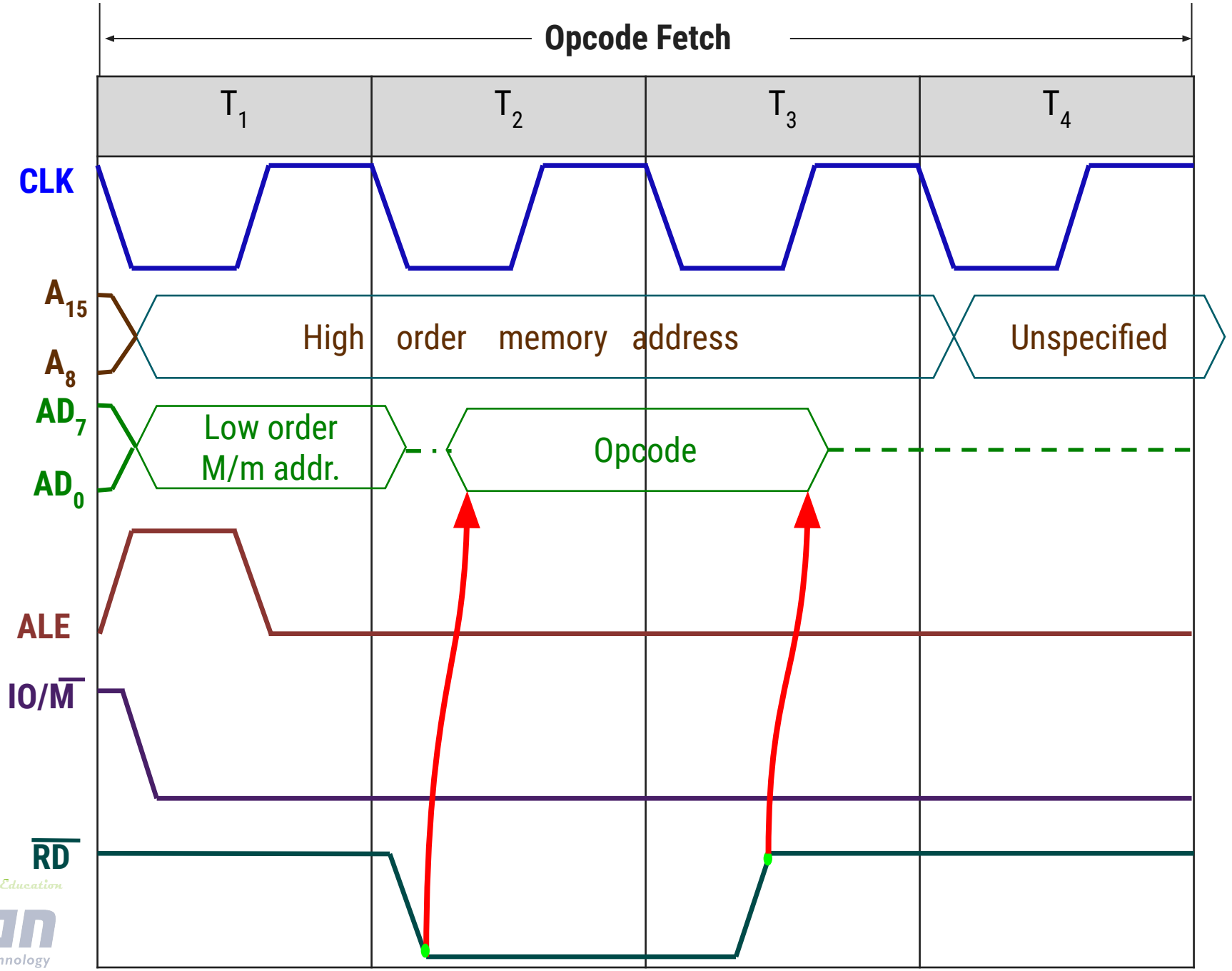
| Sr. | GTU Questions | Marks | Year |
|---|---|---|---|
| 1. | How will the multiplexed address/data bus ($AD_0$-$AD_7$) of the 8085 microprocessor be demultiplexed? | 4 | Win-2017 |

# Timing Diagram

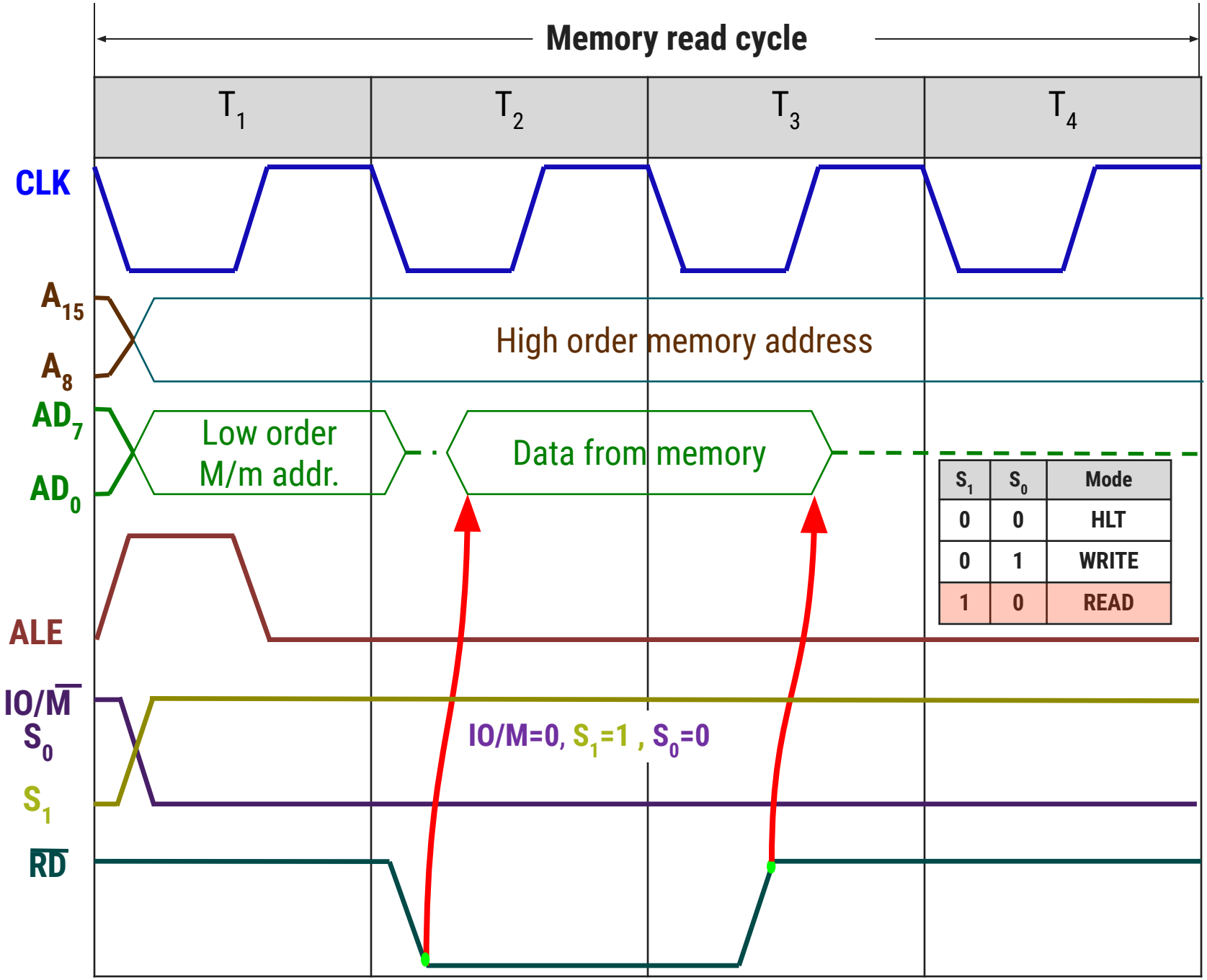
# Timing Diagram



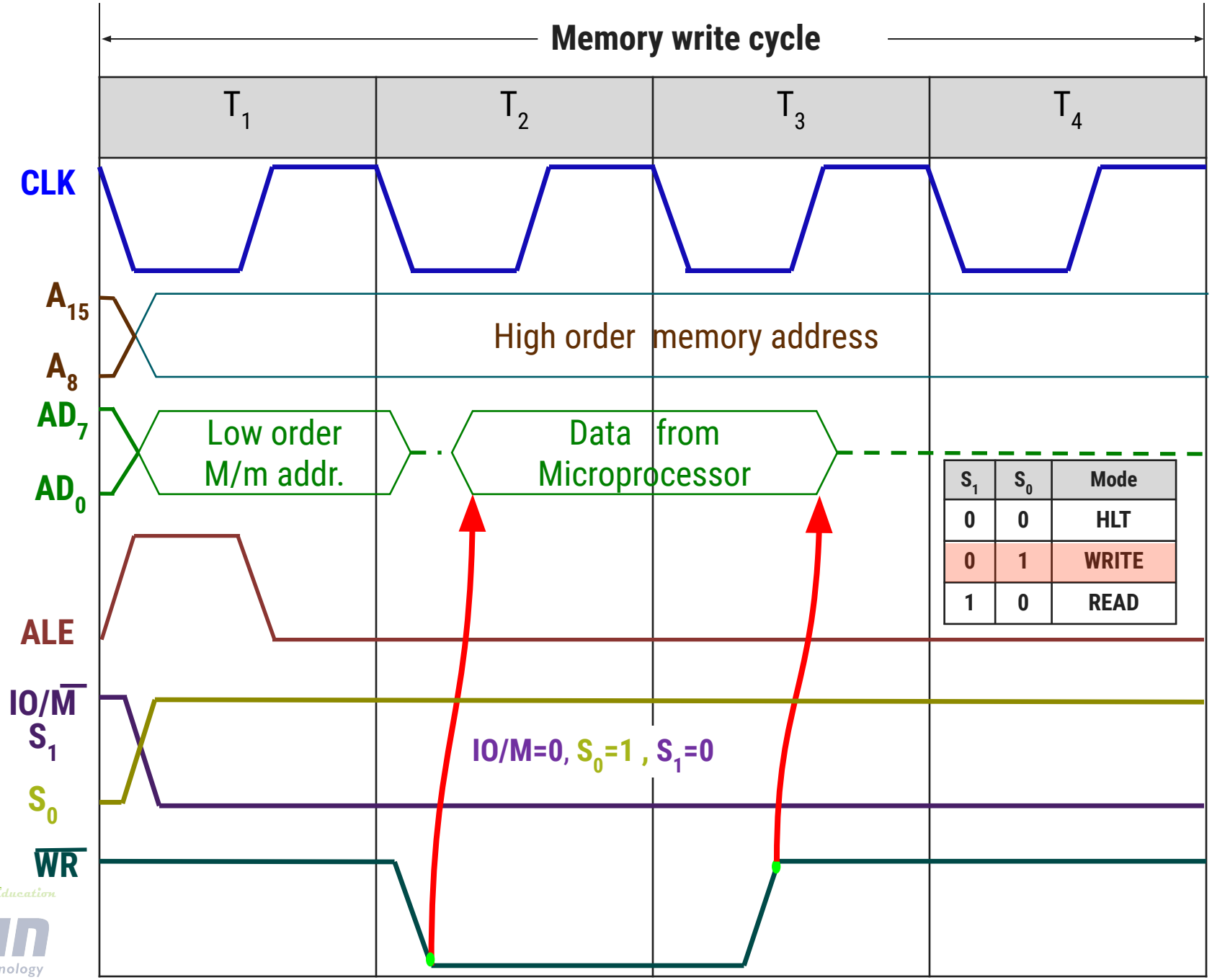
# Timing Diagram: Opcode Fetch



# Timing Diagram: Memory Read Cycle



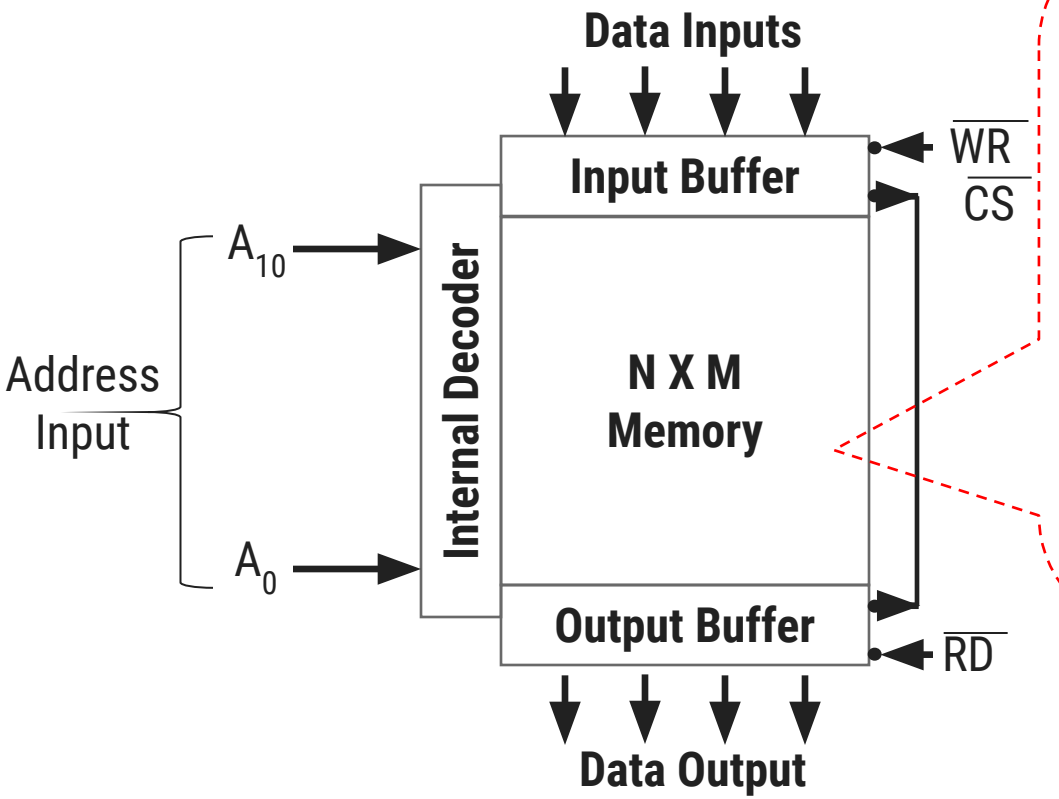
# Timing Diagram: Memory Write Cycle



# Memory Interfacing

# Memory structure

**Block diagram of Memory Device: RAM**




N=Number of Register
M=Word Length
E.g.
If a memory is having
address lines=13
data lines=8
then

1. Number of memory locations = $N=2^{13}$ = **8192**
2. Word length M= **8 bit**

Therefore, N X M= 8192 X 8

# Memory structure

| No. of Lines | Memory size(bytes) | No. of Lines | Memory size(bytes) |
|---|---|---|---|
| 1 | 2 | 9 | 512 |
| 2 | 4 | 10 | 1024=1k |
| 3 | 8 | 11 | 2048=2k |
| 4 | 16 | 12 | 4096=4k |
| 5 | 32 | 13 | 8192=8k |
| 6 | 64 | 14 | 16384=16k |
| 7 | 128 | 15 | 32768=32k |
| 8 | 256 | 16 | 65536=64k |


# Memory Interfacing

- 8085 can access 64K of memory, thus address bus is of 16-bit.
- It is not always necessary to use full 64K address space. The total memory size depends upon the application.
- Generally EPROM is used as a program memory and RAM is used as data memory.
- When both are used then total 64K address will be shared by both.
- The capacity of program memory and data memory depends on the application.

# Memory Interfacing

- It is not always necessary to select 1 EPROM and 1 RAM. We can have multiple EPROMs and multiple RAMs as per the requirement of application.
- We can place EPROM / RAM anywhere in full 64 Kbytes address space.
- Program memory (EPROM) should be located from address 0000H, since reset address of 8085 microprocessor is 0000H.
- It is not always necessary to locate EPROM and RAM in consecutive memory addresses.

# Memory Interfacing: Example 1

Interface 4kB of EPROM with starting address from 0000H and 2kB of RAM with starting address followed by EPROM

**Step-1: To calculate no. of EPROM and RAM chip required**

Total **EPROM** required = 4kB

Chip size available = 4kB

No. of chips required = 4kB/4kB=**1**

Total **RAM** required = 2kB

Chip Size Available = 2kB

No. of Chips required = 2kB/2kB=**1**

# Memory Interfacing: Example 1

**Step-2: To calculate starting and ending address of EPROM**

**EPROM Chip-1:**

Starting Address = **0000H**

Chip Size = 4kB (i.e. address Line=12, m/m = $2^{12}$ = 4096B = 4k)

Ending Address = **0FFFH**

**Step-3: To calculate starting and ending address of RAM**

**RAM Chip-1:**

Starting address = Ending address of EPROM +1

= 0FFFH+1

= **1000H**

Chip Size = 2kB = 07FFH

Ending address = 1000H+07FFH

= **17FFH**

4,095

HEX FFF
DEC 4,095
OCT 7 777
BIN 1111 1111 1111

| No. of Lines | Memory size(bytes) |
|---|---|
| 9 | 512 |
| 10 | 1024=1k |
| 11 | 2048=2k |
| 12 | 4096=4k |
| 13 | 8192=8k |
| 14 | 16384=16k |
| 15 | 32768=32k |
| 16 | 65536=64k |

# Memory Interfacing: Example 1

| Step-4: Memory Map | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPROM 4k | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 0FFF H | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RAM 2k | Start Address 1000 H | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 17FF H | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

EPROM chip size = 4kB & RAM chip size = 2kB

∴ smaller chip size RAM = 2kB = $2^{11}$

Thus neglect lower 11 address lines ($A_0$ to $A_{10}$), and consider $A_{11}$ to $A_{15}$ for Decoding.

# Memory Interfacing: Example 1

| | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EPROM** | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 0FFF H | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**EPROM**

- Required Address Lines: $A_{11}$ to $A_{15}$.
- Now, EPROM has two Possibilities, either **00000** b or **00001** b.
- Therefore, it requires $Y_0$ and $Y_1$ outputs of decoder.

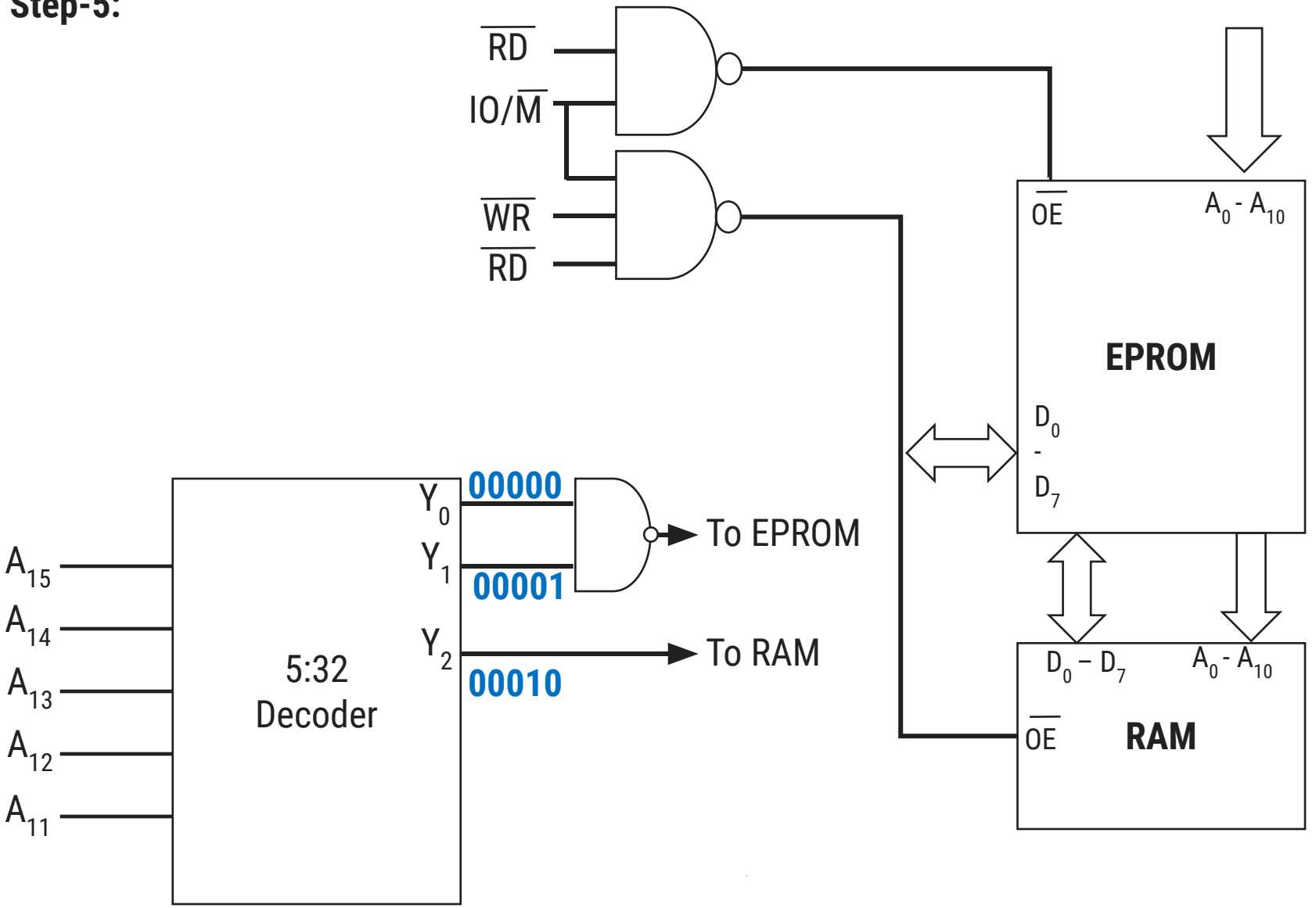
# Memory Interfacing: Example 1

| | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EPROM** | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 0FFF H | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RAM | Start Address 1000 H | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 17FF H | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

RAM has **00010** b, hence it requires $Y_2$ output of decoder.

# Memory Interfacing: Example 1 Implementation

**Step-5:**



# Memory Interfacing: Example 2

Interface 16kB of EPROM with **chip size** of 8kB and starting address from 0000H and 8kB of RAM with starting address followed by EPROM.

**Step-1: To calculate no. of EPROM and RAM chip required**

Total **EPROM** required =16kB

Chip size available = 8kB

No. of chips required = 16kB/8kB = 2

Total **RAM** required = 8kB

Chip size available = 8kB

No. of chips required = 8kB/8kB = 1

# Memory Interfacing: Example 2

## Step-2: To calculate starting and ending address of EPROM

### EPROM Chip-1:

Starting Address = **0000H**

Chip Size = 8kB (i.e. address Line=13, m/m=$2^{13}$ =8192B =8k)

Ending Address = **1FFFH**


### EPROM Chip-2:

Starting Address =1FFFH + **1 = 2000H**

Chip Size = 1FFFH

Ending Address = 2000+1FFFH = **3FFFH**

| No. of Lines | Memory size(bytes) |
|---|---|
| 9 | 512 |
| 10 | 1024=1k |
| 11 | 2048=2k |
| 12 | 4096=4k |
| 13 | 8192=8k |
| 14 | 16384=16k |
| 15 | 32768=32k |
| 16 | 65536=64k |

# Memory Interfacing: Example 2

**Step-3:RAM Chip-1:**

Starting Address = EPROM ending address + 1

Starting Address = 3FFFH+1 = **4000H**

Chip Size = 8kB (i.e. address Line =13, m/m=$2^{13}$ = 8192B =8k)

Chip Size = 1FFFH

Ending Address = 4000h+1FFFH = **5FFFH**

# Memory Interfacing: Example 2

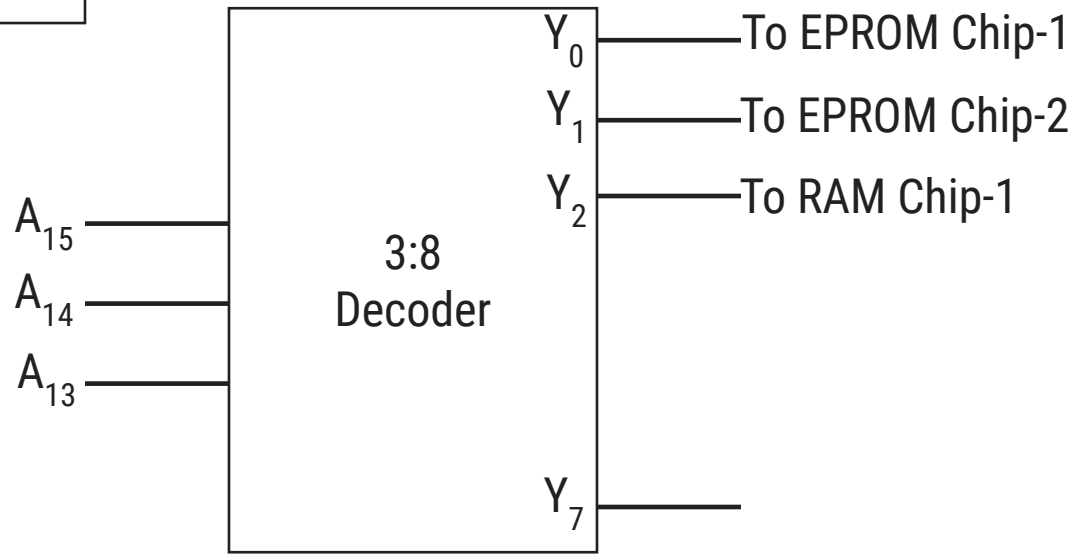
| Step-4: Memory Map | | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EPROM1** | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 1FFF H | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **EPROM2** | Start Address 2000 H | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 3FFF H | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **RAM** | Start Address 4000 H | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 5FFF H | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

# Memory Interfacing: Example 2

## Step-5:Interface Implementation

- EPROM chip size = RAM chip size = 8kB = $2^{13}$
- Therefore, neglect lower 13 address lines (i.e. $A_0$ to $A_{12}$) and consider only $A_{13}$ to $A_{15}$ for decoding.

| | |
|---|---|
| EPROM CHIP-01 | **000** |
| EPROM CHIP-02 | **001** |
| RAM CHIP-01 | **010** |

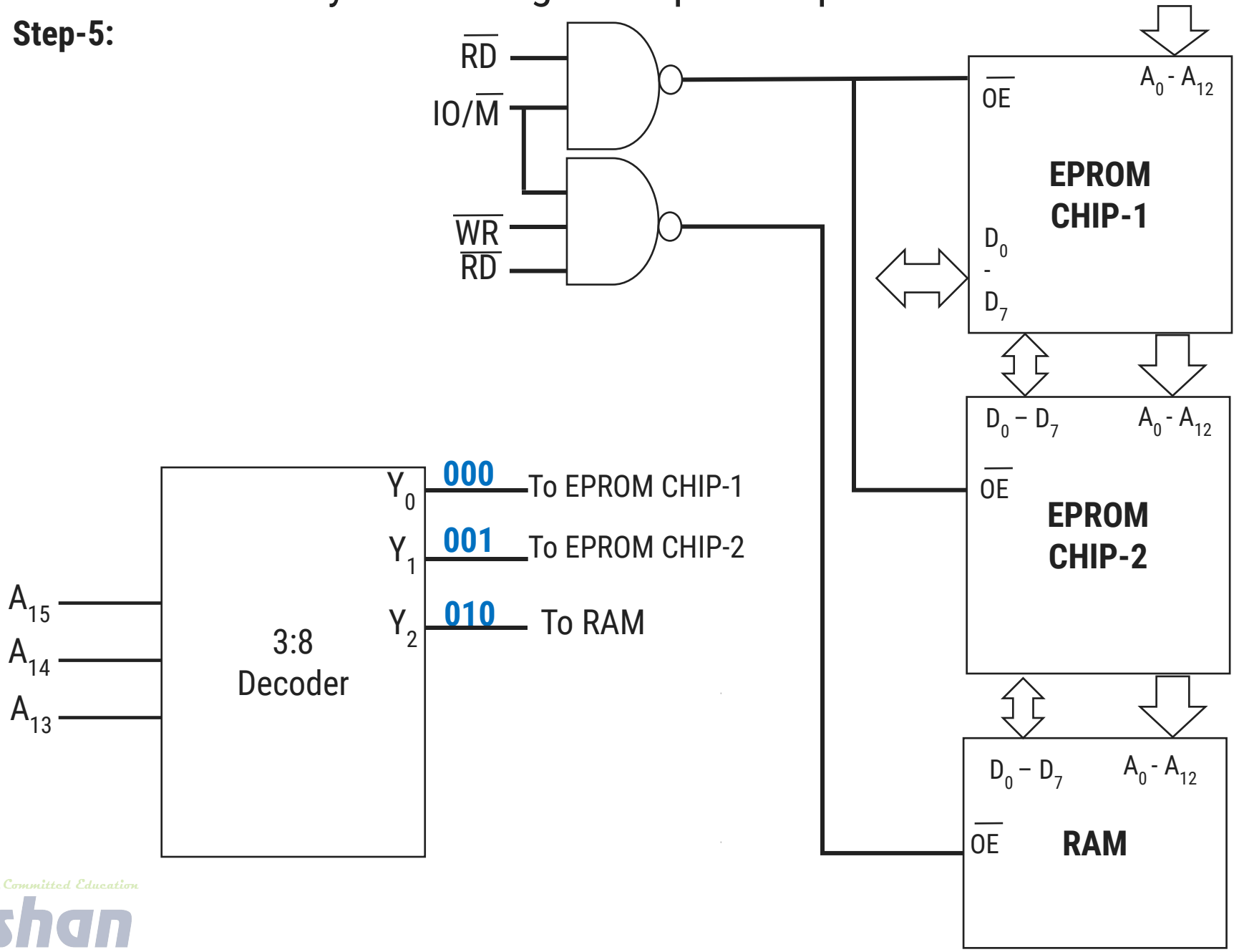


# Memory Interfacing: Example 2

| Step-4: Memory Map | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EPROM1** | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 1FFF H | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **EPROM2** | Start Address 2000 H | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 3FFF H | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **RAM** | Start Address 4000 H | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 5FFF H | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

# Memory Interfacing: Example 2 Implementation

**Step-5:**



# Exercise: Memory Interface Example

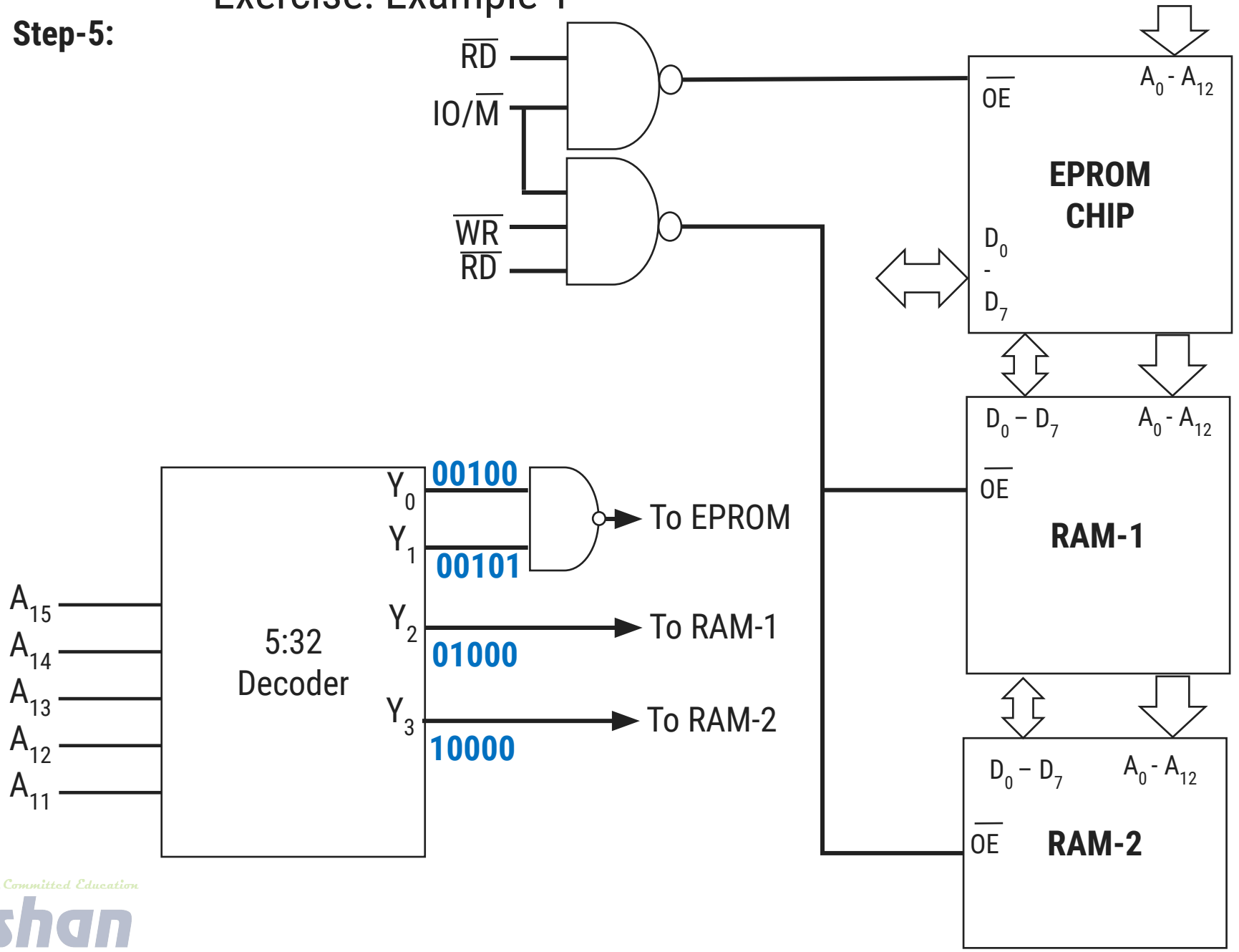
1. Draw the interfacing of a 4KB EPROM having a starting address 2000h and two 2KB static RAMs having starting addresses 4000h and 8000h, respectively, with 8085 microprocessor.
2. Design an 8085 microprocessor system such that it should contain 16KB of EPROM and 4KB of RAM with starting addresses 0000H and 4000H respectively. Use two 8KB of EPROMs (2764) and two 2KB of RAMs (6116) for this system.

# Exercise: Example 1

| Step-4: Memory Map | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **EPROM1** | Start Address 2000 H | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 2FFF H | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **RAM** 1 | Start Address 4000 H | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 47FF H | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **RAM** 2 | Start Address 8000 H | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 87FF H | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

# Exercise: Example 1

**Step-5:**

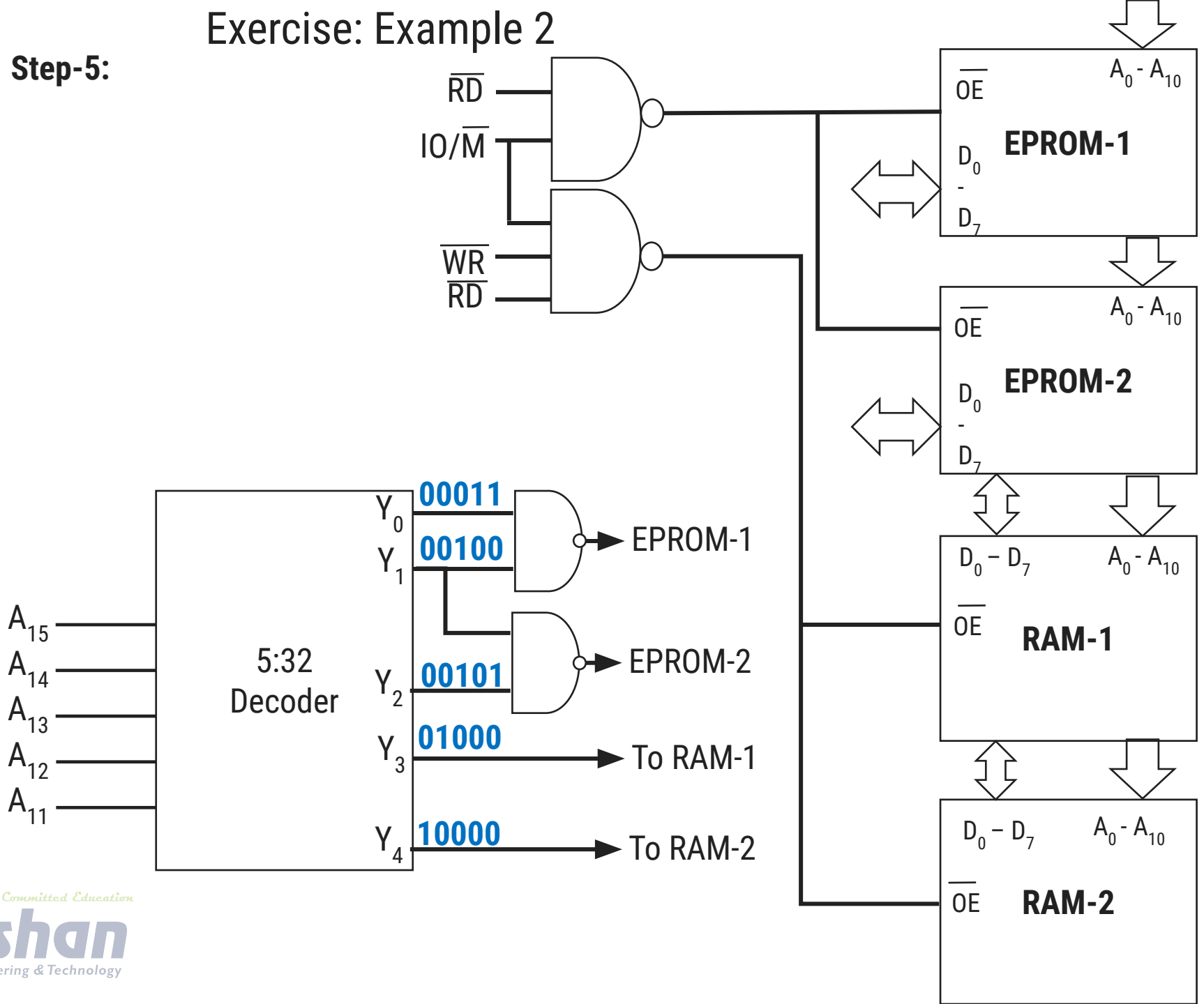


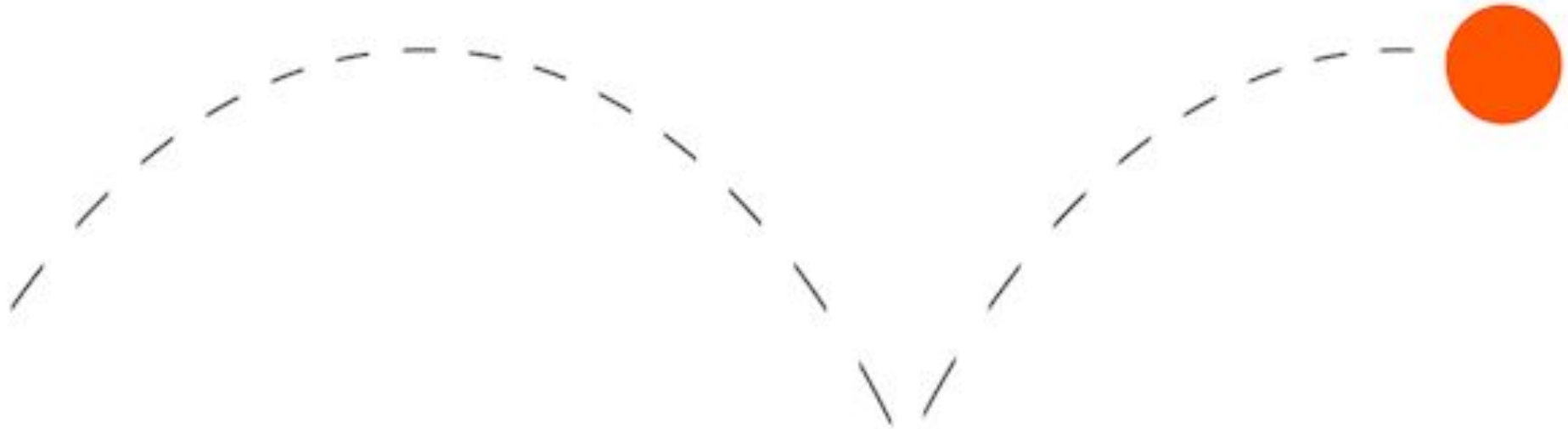
Exercise: Example 2

| Step-4: Memory Map | | A 15 | A 14 | A 13 | A 12 | A 11 | A 10 | A 9 | A 8 | A 7 | A 6 | A 5 | A 4 | A 3 | A 2 | A 1 | A 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPROM1 | Start Address 0000 H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 1FFF H | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| EPROM2 | Start Address 2000 H | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 3FFF H | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RAM 1 | Start Address 4000 H | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 47FF H | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RAM 2 | Start Address 4800 H | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | End Address 4FFF H | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

# Exercise: Example 2

**Step-5:**

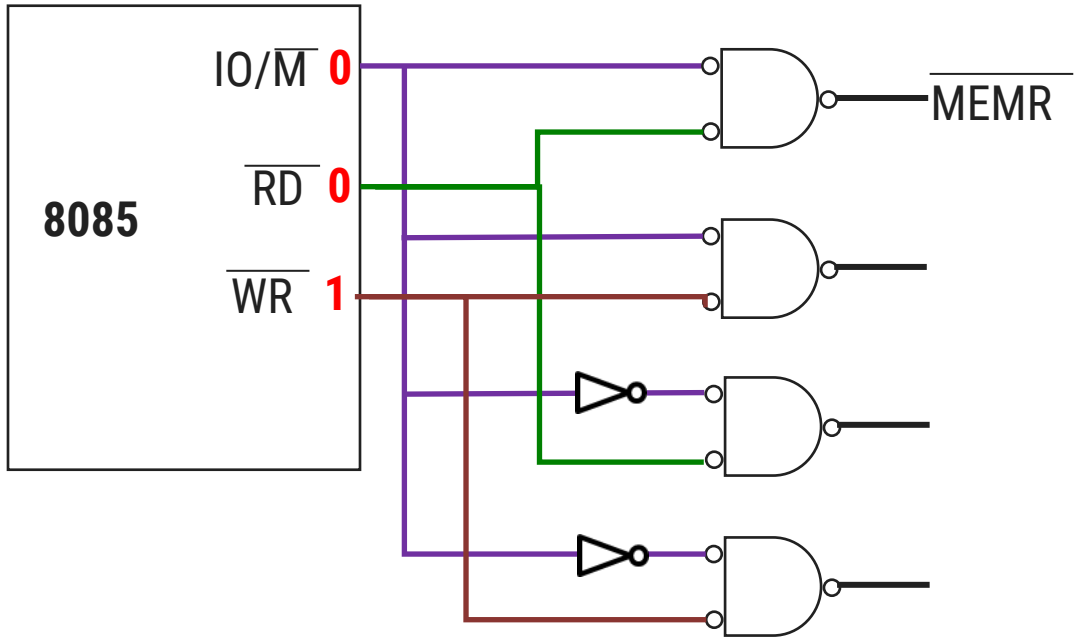




# Generating Control Signals

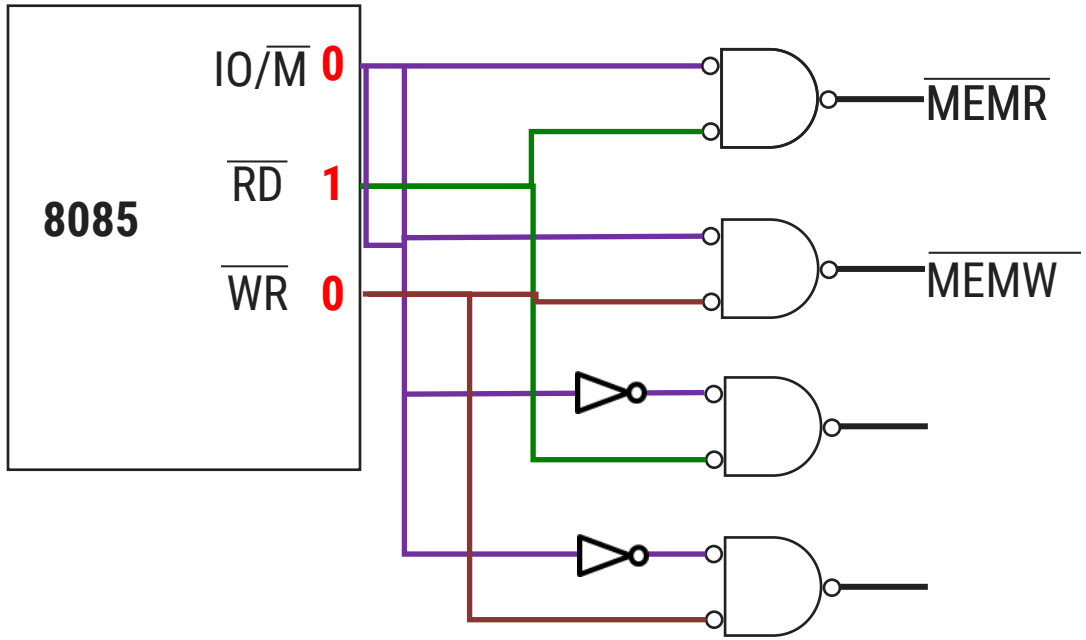
# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 0 | Invalid |
| 0 | 0 | 1 | $\overline{MEMR}$ |



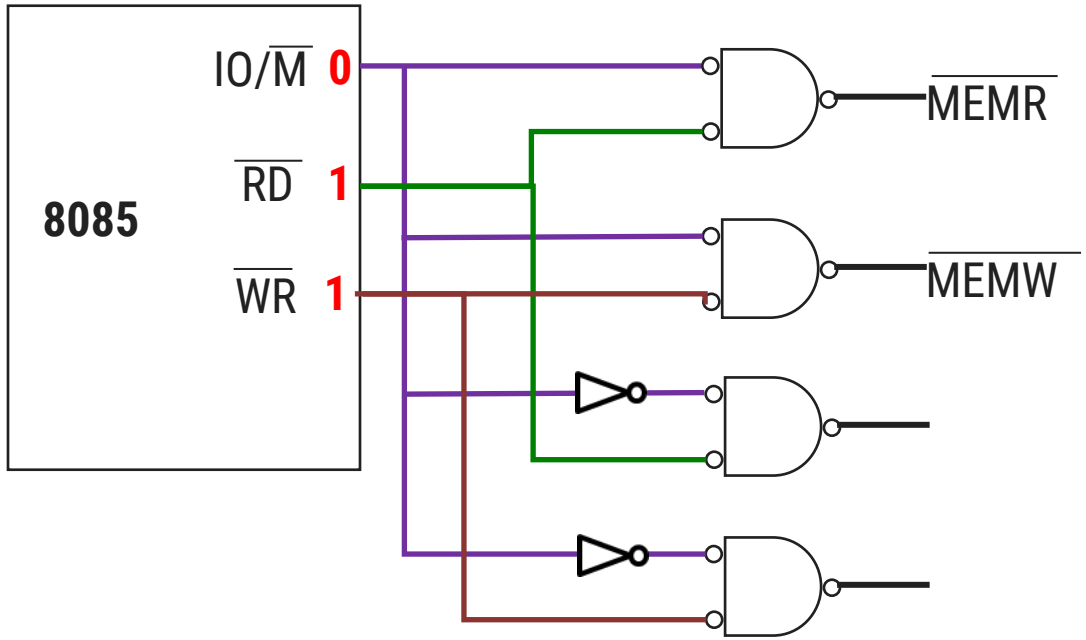
# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 0 | Invalid |
| 0 | 0 | 1 | $\overline{MEMR}$ |
| **0** | **1** | **0** | $\overline{MEMW}$ |



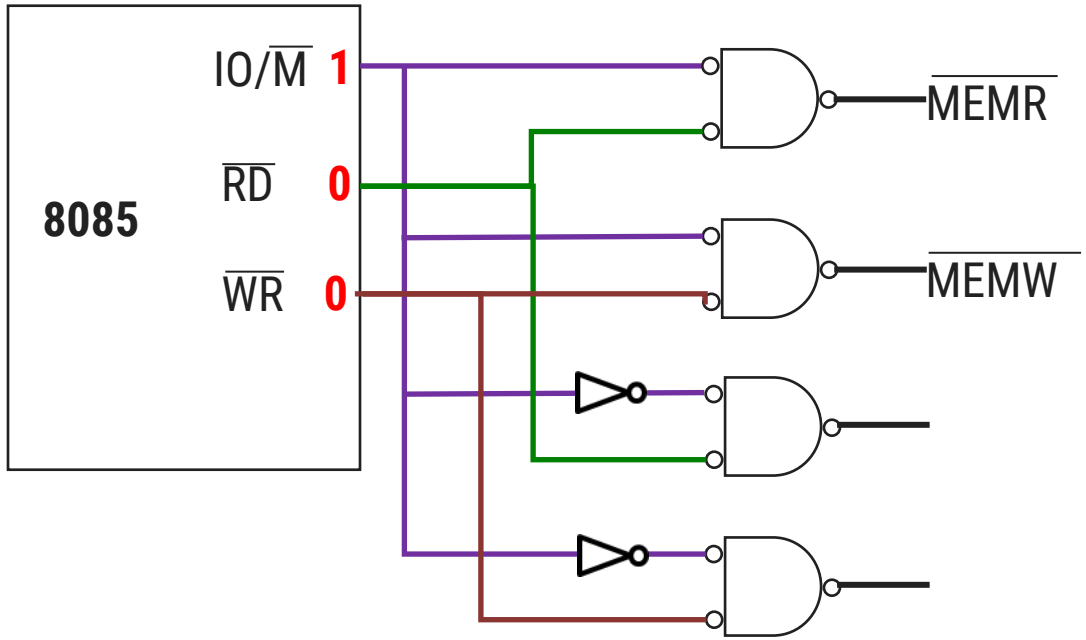
# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 0 | Invalid |
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| **0** | **1** | **1** | **NOP** |



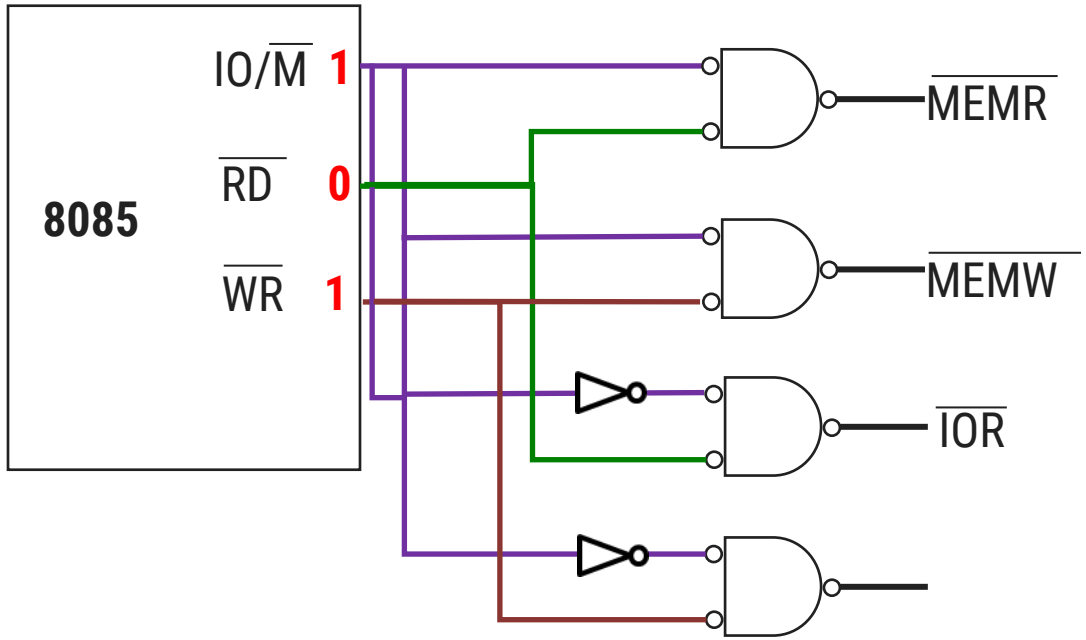
# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | NOP |
| **1** | **0** | **0** | **Invalid** |



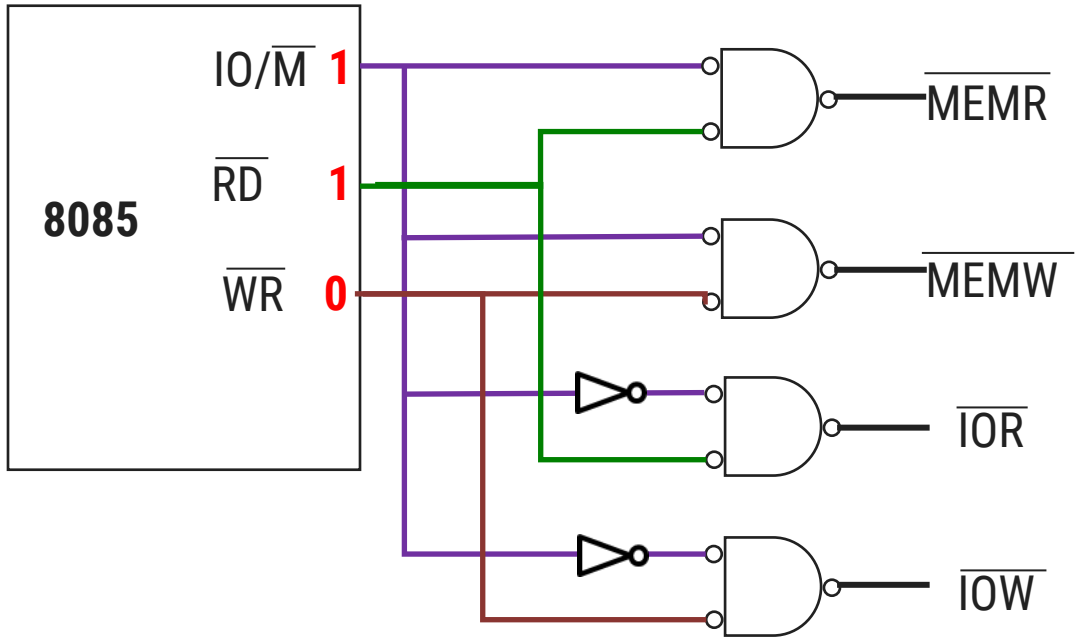
# Generating Control Signals

| $IO/\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | NOP |
| **1** | **0** | **1** | $\overline{IOR}$ |



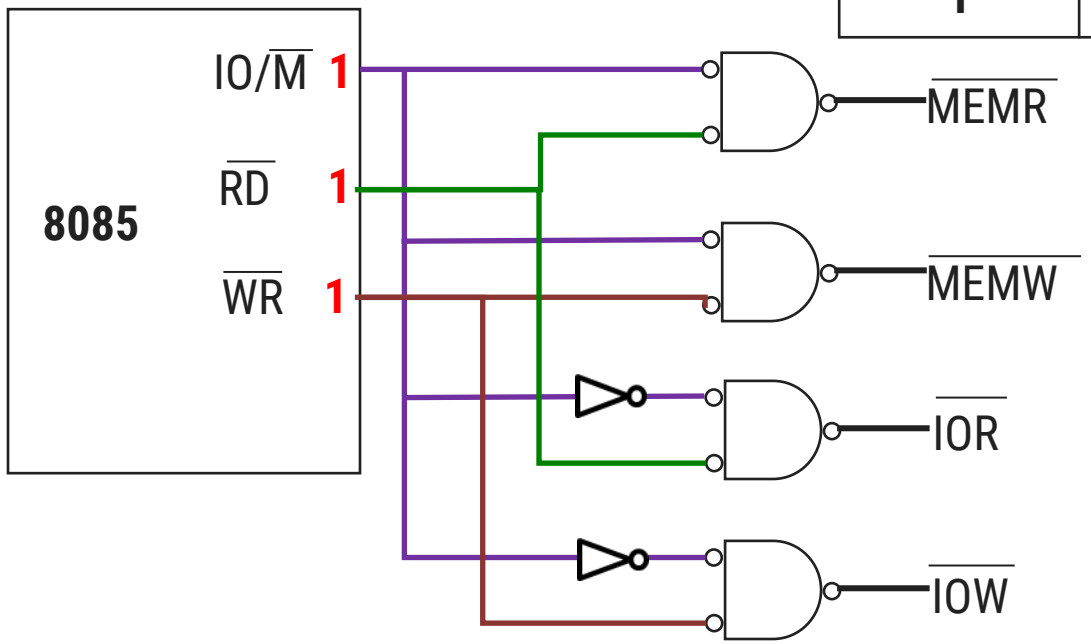
# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | NOP |
| 1 | 0 | 1 | $\overline{IOR}$ |
| **1** | **1** | **0** | **$\overline{IOW}$** |



# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | NOP |
| 1 | 0 | 1 | $\overline{IOR}$ |
| 1 | 1 | 0 | $\overline{IOW}$ |
| **1** | **1** | **1** | **NOP** |



# Generating Control Signals

| IO/$\overline{M}$ | $\overline{RD}$ | $\overline{WR}$ | Operation |
|---|---|---|---|
| 0 | 0 | 0 | HLT |
| 0 | 0 | 1 | $\overline{MEMR}$ |
| 0 | 1 | 0 | $\overline{MEMW}$ |
| 0 | 1 | 1 | NOP |
| 1 | 0 | 0 | HLT |
| 1 | 0 | 1 | $\overline{IOR}$ |
| 1 | 1 | 0 | $\overline{IOW}$ |
| 1 | 1 | 1 | NOP |

# GTU Exam Questions

| Sr. | GTU Questions |
|---|---|
| 1. | How will the multiplexed address/data bus (AD0-AD7) of the 8085 microprocessor be demultiplexed? |
| 2. | Explain the flag register of the 8085 microprocessor with examples. |
| 3. | Draw the interfacing of a 4KB EPROM having a starting address 2000h and two 2KB static RAMs having starting addresses 4000h and 8000h, respectively, with 8085 microprocessor. Use demultiplexed address/data lines and use 3-to-8 decoder (74LS138). |
| 4. | Draw and explain the internal block diagram of 8085 microprocessor. |
| 5. | List the sequence of events that occurs when the 8085 MPU reads from memory. |
| 6. | Explain 8085 Programming Model and Flag Register. |
| 7. | Discuss the programming model of 8085 µP with the help of suitable diagram. |
| 8. | Draw and Explain the pin diagram of 8085 microprocessor. |

# References

Book: Microprocessor Architecture, Programming, and Applications with the 8085, Ramesh S. Gaonkar Pub: Penram International

# *Thank You*