## What is a Microprocessor?

- The word comes from the combination micro and processor.
- Processor means a device that processes data. In this context processor means a device that processes numbers, specifically binary numbers, 0’s and 1’s.
- In the late 1960’s, processors were built using discrete elements. These devices performed the required operation, but were too large and too slow.
- In the early 1970’s a new technology was developed in which all of the components that made up the processor were now placed on a single piece of silicon. So, the size became several thousand times smaller and the speed became several hundred times faster.
- The “Micro” Processor was born.

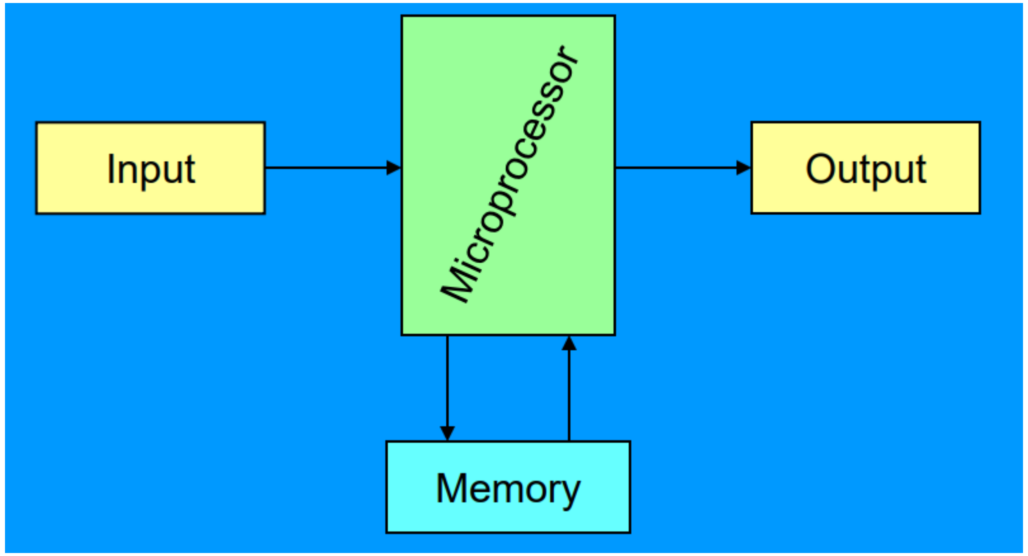
A microprocessor is a <u>multipurpose</u>, <u>programmable</u>, <u>clock-driven</u>, <u>register-based</u> electronic device that reads <u>binary instructions from a storage device called memory</u>, accepts binary data as input and <u>processes data</u> according to those instructions, and <u>provides result as output.</u>

## Multipuropse

- Consumer Electronics
- House Hold applications
- Home automation and security system
- Automotive industry
- Telecommunication
- Computer peripherals
- Computer networking system
- Healthcare
- Measurement and Instrumentation
- Banking and Retail

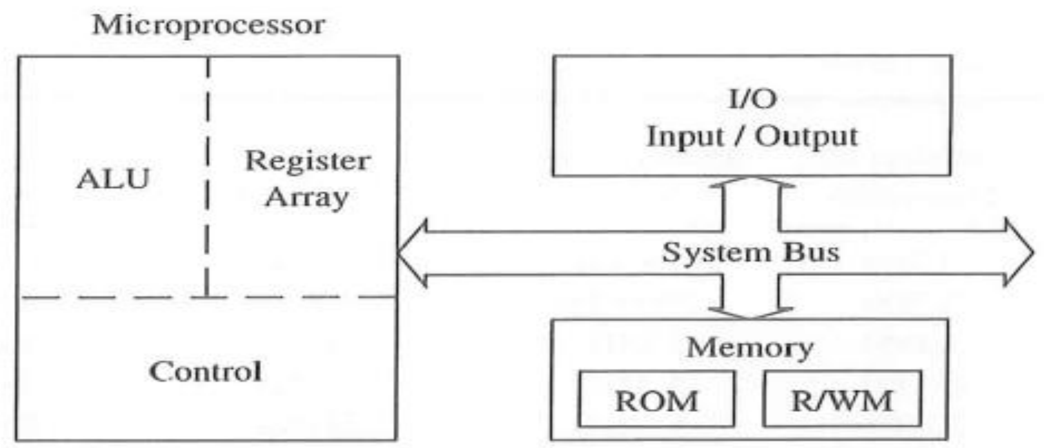
## Programmable

- A typical programmable machine can be represented with four components: microprocessor, memory, input and output as shown in figure. These four components work together or interact with each other to perform a task, thus they comprise a system.
- The physical components of this system called hardware.
- A set of instructions written for the microprocessor to perform a task is called a program (stored in memory), and a group of program is called software.

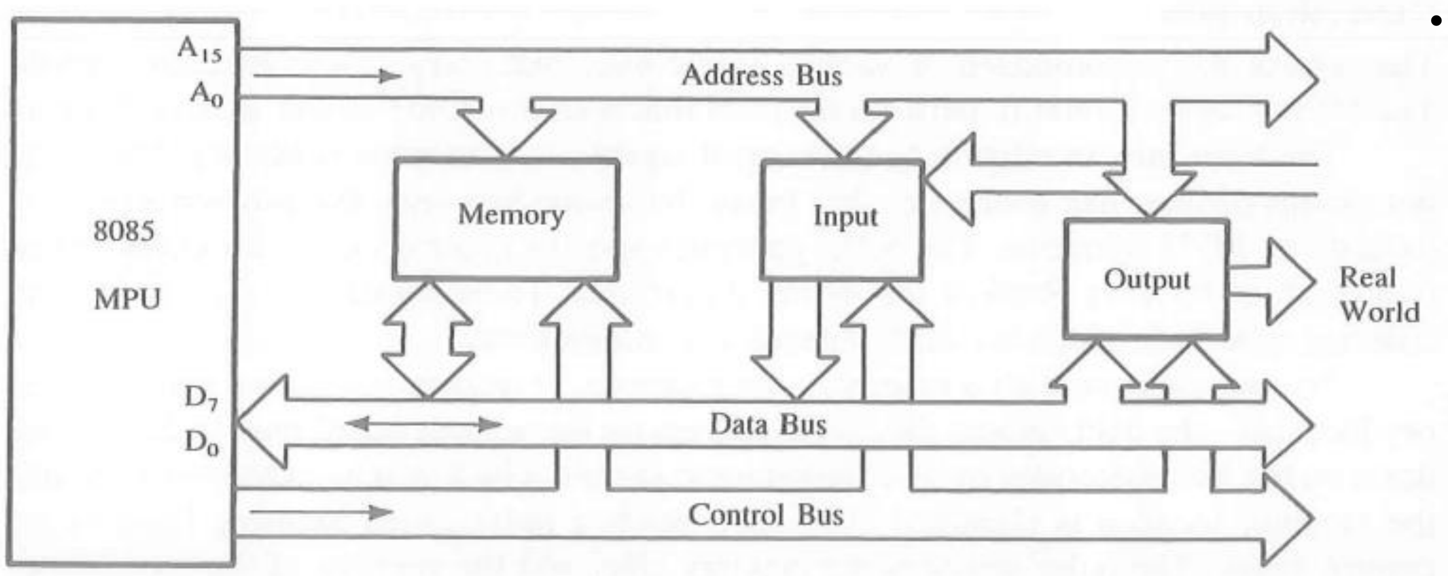


# Organization of a Microprocessor-Based System.

- Internally, the microprocessor is made up of some main units.
- An array of registers used to store data temporarily during execution of a program and also used by the user.
- The Arithmetic/Logic Unit (ALU) to perform arithmetic and logical operations.
- Instruction decoder and the control unit.
- CPU **reads the instruction from the memory and perform the task specified**. It communicates with **input/output devices either to accept or to send data.** These devices are also known as peripherals. The CPU is primary and central player in communicating with devices such as memory, input and output. However the **timing of the communication process is controlled by the group of circuits called the control unit**.



# System Bus

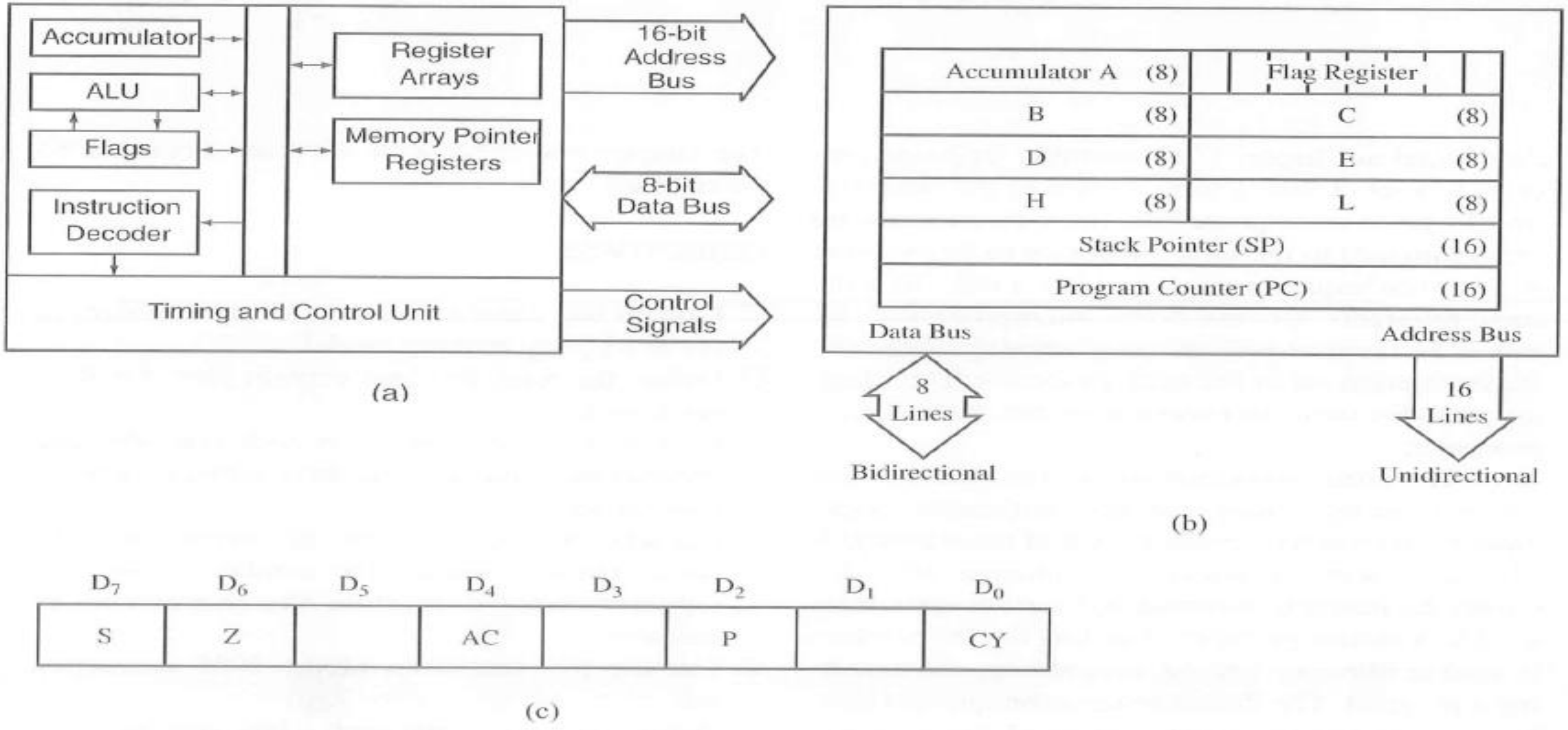



- **Address Bus**: The address bus is a group of l6 lines (8085 microprocessor) identified as A0 to A15. Address bus is **unidirectional**: bits flow in one directional, from CPU to peripherals or memory. The MPU uses the address bus to identify a peripheral of memory location. Each peripheral or memory location is identified by an specific binary numbers is called address, an address bus is used to carry the address information from MPU to peripheral or memory locations. The 8085 microprocessor has 16 bit address bus so, by using these 16 address line MPU is capable to address 64K memory locations.

- **Data Bus:** The data bus is a group of 8 lines used for data flow. These lines are **bidirectional**: data flow in both direction i.e from MPU to peripheral or memory and vice-versa. The eight data lines enable the MPU to manipulate 8 bit data from 00h to FFh.

- **Control Bus:** The control bus is comprised of single lines that carry synchronization signals. Control bus are not group of lines like address or data bus, but individual lines that provide a pulse to indicate a MPU operation.

# 8085 Programming Model



## The ALU

- ALU performs Arithmetic operations like Addition, Subtraction, Increment, decrement and logical operations like logical OR, logical AND, logical XOR, complement, compare and shift operations.
- The ALU includes the accumulator which help ALU to perform arithmetic or logical operation
- The ALU also includes a temporary register used for holding data temporarily during the execution of the operation. This temporary register is not accessible by the programmer.

## Registers

- **General Purpose Registers**
  - B, C, D, E, H& L(8 bit registers)
  - Can be used as single 8 bit register
  - Can be used as 16 bit register pairs BC, DE, HL
  - BC, DC and HL can be used as a data pointer (holds memory address)
- **Special Purpose Register**
  - **Accumulator(8 bit register)**
    - Store 8 bit data
    - Store the result of an operation
    - Store 8 bit data during I/O transfer

- **Flag Register**
  - 8 bit register: shows the status of the microprocessor before/after an operation.
  - S (sign flag), Z (zero flag), AC (auxillary carry flag), P (parity flag) & CY (carry flag)

```
    10110011
+   01001101
  ------------
1   00000000
```

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

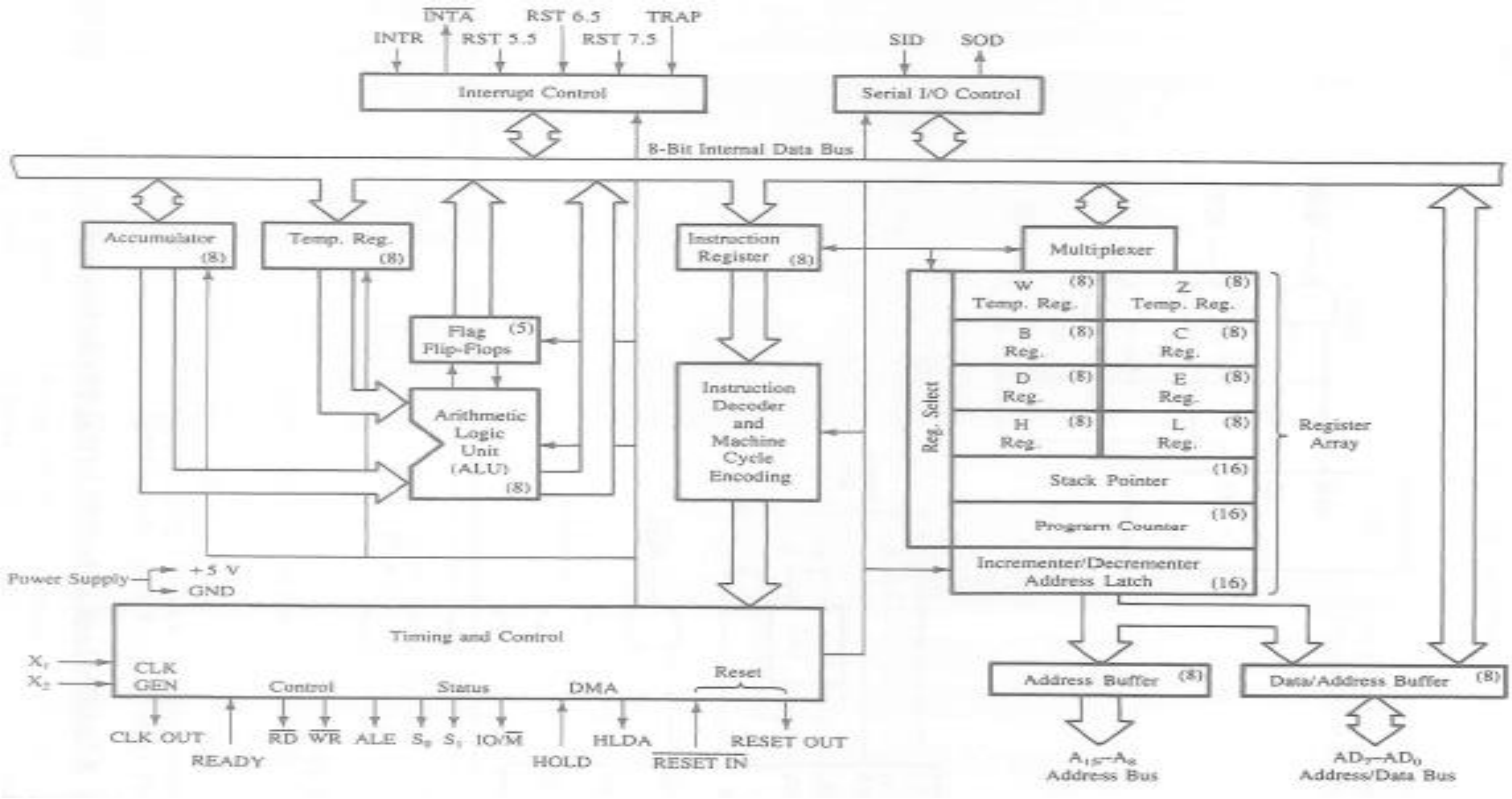
  - **Sign Flag:**
    - Used for indicating the sign of the data in the accumulator after some arithmetic or logical operation.
    - The sign flag is set if negative (1 –negative)
    - The sign flag is reset if positive (0 –positive)

```
          1011 0101
       +  0110 1100
          ------------
Carry 1   0010 0001
```

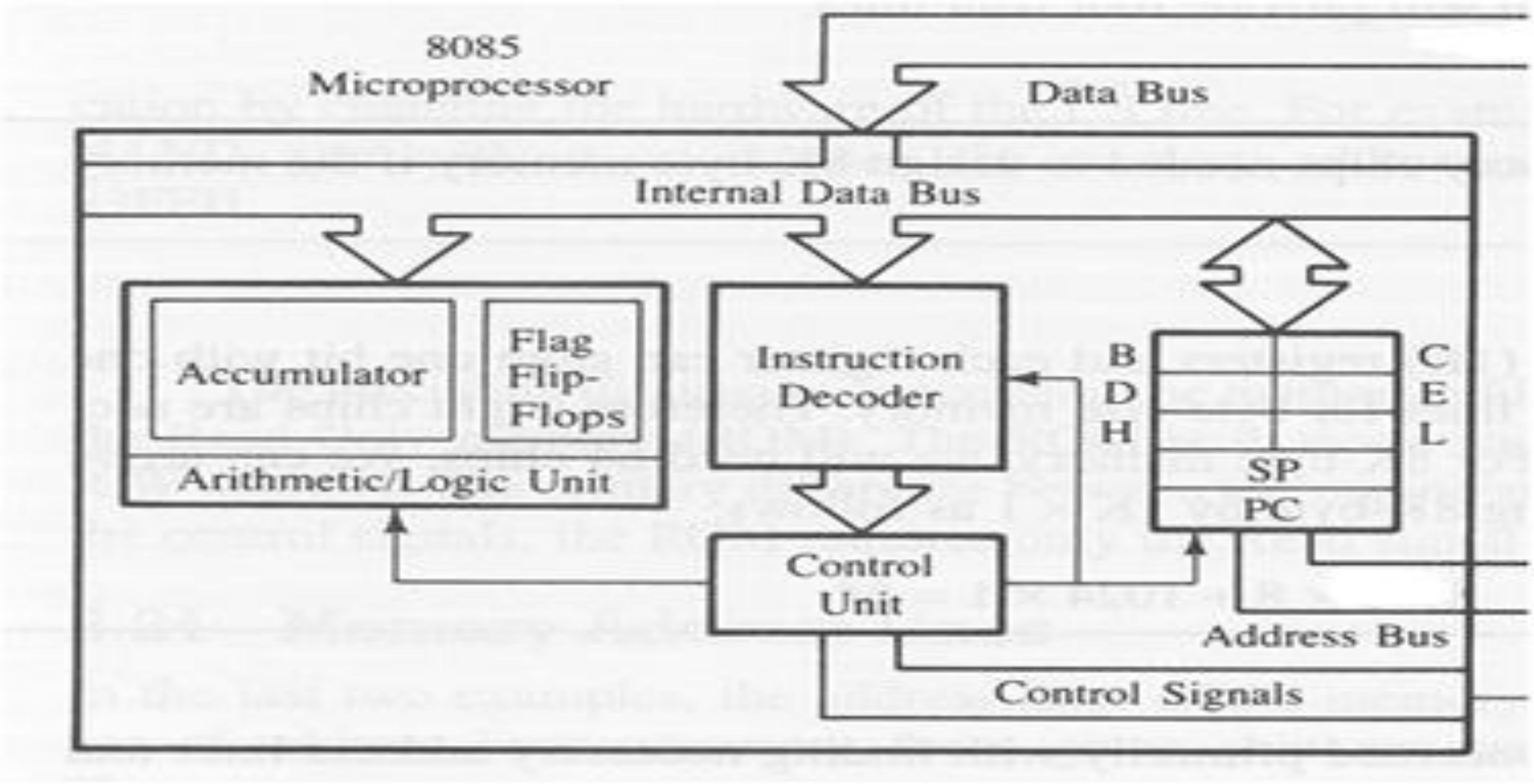
- **Zero Flag:**
  - Zero flag will be set to 1 if the result obtained after an arithmetic or logical operation is 0.
- **Carry Flag:**
  - Carry flag will be set to 1 if there is a carry or borrow from arithmetic operation, i.e. a carry generated from the D7 bit of the accumulator.
- **Auxillary Carry Flag:**
  - Auxillary Carry flag will be set to 1 if there is a carry out from D3 to D4 bit of the accumulator after some arithmetic operation.
- **Parity Flag:**
  - Parity flag will be set to 1 if the number of 1's in accumulator is even. I.e. set if parity is even, cleared if parity is odd

- **PC (PROGRAM COUNTER)**
  - The Program Counter (PC)This is a register that is used to control the sequencing of the execution of instructions.
  - This register always holds the address of the next instruction.
  - Since it holds an address, it must be 16 bits wide.
- **SP (STACK POINTER)**
  - The stack pointer is also a 16-bit register that is used to point into memory.
  - The memory this register points to is a special area called the stack.
  - The stack is an area of memory used to hold data that will be retrieved soon.
  - The stack is usually accessed in a Last In First Out (LIFO) fashion.
- **Non Programmable Registers**
  - **Instruction Register & Instruction Decoder:**
    - Instruction is stored in IR after fetched by processor
    - Instruction Decoder decodes instruction in IR (Instruction Register)

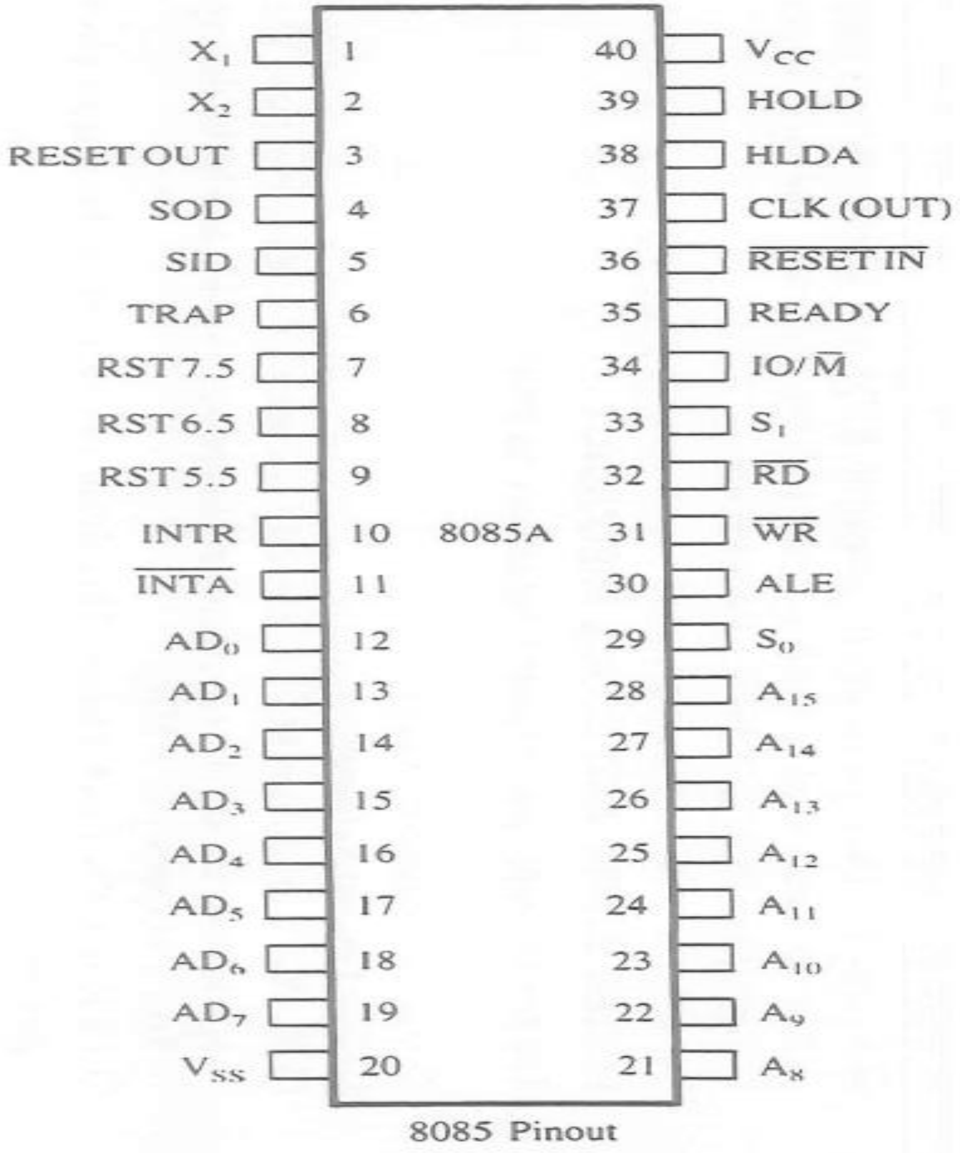
# Architecture of 8085 Microprocessor



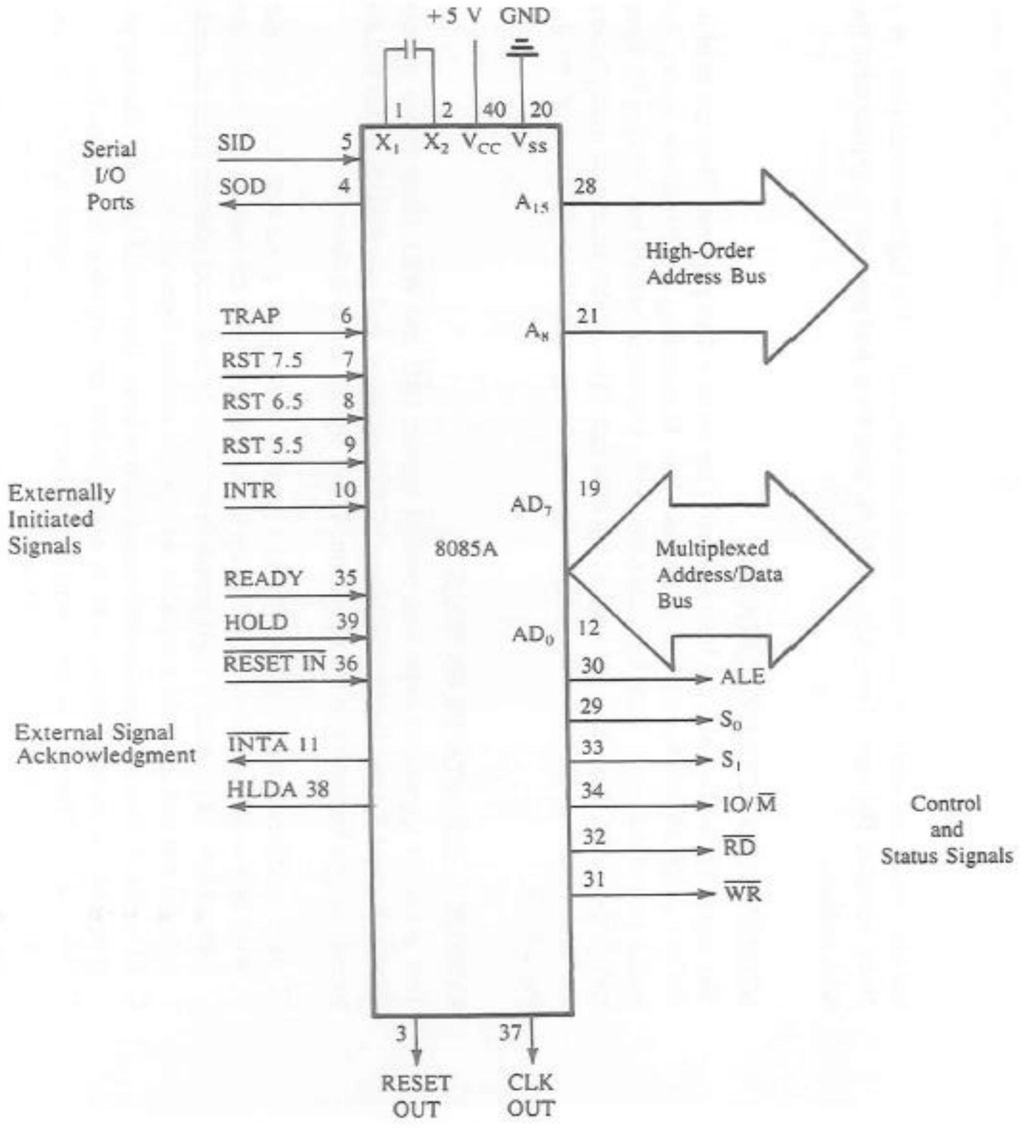
# Simplified Block Diagram



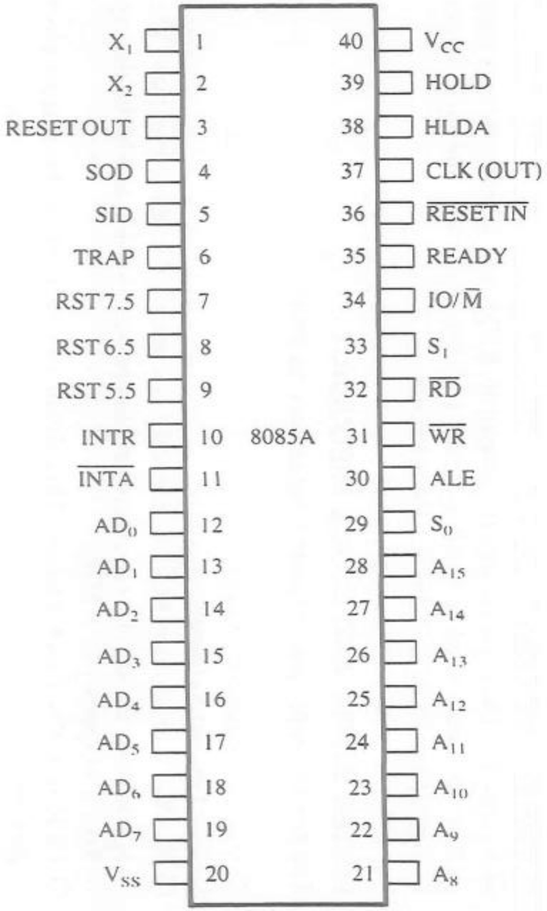
# Pin Description




8085 Pinout



# 8085 Microprocessor



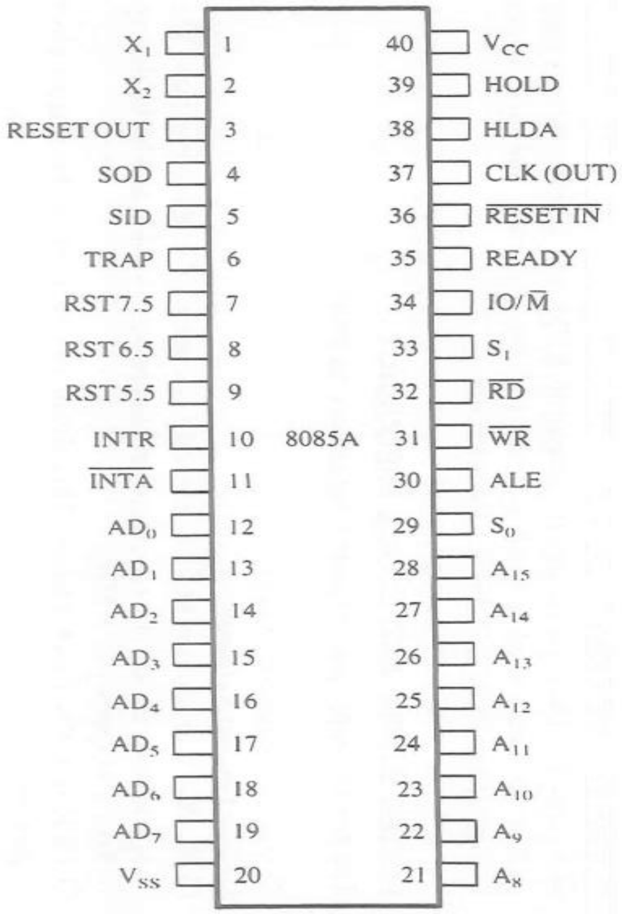
8085 Pinout

- The 8085A is an 8 bit general purpose microprocessor capable of addressing of 64KB of memory.
- The device has 40 pins.
- It requires a +5v single power supply.
- It can operate with 3 MHz single phase clock.
- 8085A-2 can operates at maximum frequency of 5 MHz.
- All the **signals can be classified into six groups (i)address bus (ii) data bus (iii) control and status signals (Iv) power supply and frequency signals (v) externally initiated signals and (vi) serial I/O ports.**
- **Address Bus:** The 8085 has 16 address lines, these lines are separated by two segments A15-A8 and AD7-AD0. The eight signal lines A15-A8 are unidirectional and used for most significant bits, called the higher order address of 16 bit address. The signal AD7-AD0 lines are used for dual purpose as stated bellow.
- **Multiplexed address/data bus:** The address lines AD7-AD0 are bidirectional: they serve a dual purpose. They are used as the lower order address bus and the data bus. In executing an instruction, during the earlier part of the cycle, these lines are used as the lower order address bus. During the later part of the cycle, these lines are used as the data bus (This is known as the multiplexing the bus). However the lower order address bus can be separated from these signals by using a latch.
- Control and Status Signals:
- Three control signals are $\overline{RD}$, $\overline{WR}$& ALE.
- $\overline{\boldsymbol{RD}}$ – This signal indicates that the selected IO or memory device is to be read and is ready for accepting data available on the data bus.
- $\overline{\boldsymbol{WR}}$ – This signal indicates that the data on the data bus is to be written into a selected memory or IO location.
- **ALE** – It is a positive going pulse generated when a new operation is started by the microprocessor. When the **pulse goes high, it indicates address**. When **the pulse goes down it indicates data**.

- Three status signals are IO/M, S0 & S1.
- **IO/$\overline{M}$:** This signal is used to differentiate between IO and Memory operations, i.e. when it is high indicates IO operation and when it is low then it indicates memory operation.
- **S1 & S0**: Type equation here.These signals are used to identify the type of current operation.

| Machine Cycle | Status IO/$\overline{M}$ | Status $S_1$ | Status $S_0$ | Control Signals |
|---|---|---|---|---|
| Opcode Fetch | 0 | 1 | 1 | $\overline{RD} = 0$ |
| Memory Read | 0 | 1 | 0 | $\overline{RD} = 0$ |
| Memory Write | 0 | 0 | 1 | $\overline{WR} = 0$ |
| I/O Read | 1 | 1 | 0 | $\overline{RD} = 0$ |
| I/O Write | 1 | 0 | 1 | $\overline{WR} = 0$ |
| Interrupt Acknowledge | 1 | 1 | 1 | $\overline{INTA} = 0$ |
| Halt | Z | 0 | 0 | |
| Hold | Z | X | X | $\overline{RD}$, $\overline{WR}$ = Z and $\overline{INTA}$ = 1 |
| Reset | Z | X | X | |

NOTE: Z = Tri-state (high impedance)
X = Unspecified



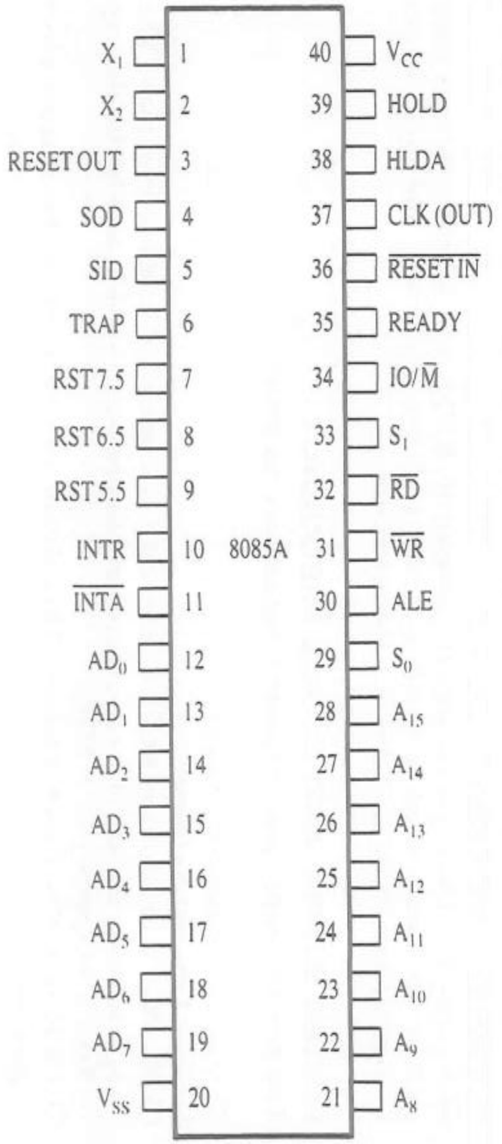
8085 Pinout

- **Power Supply and Frequency Signals:**
- **Power Supply:** There are 2 power supply signals − VCC & VSS. VCC indicates +5v power supply and VSS indicates ground signal.
- **Clock signals:** There are 3 clock signals, i.e. X1, X2, CLK OUT.
- **X1, X2** − A crystal (RC, LC N/W) is connected at these two pins and is used to set frequency of the internal clock generator. This frequency is internally divided by 2.(clock - Input , feedback)
- **CLK OUT** − This signal is used as the system clock for devices connected with the microprocessor.
- **Serial I/O signals:**There are 2 serial signals, i.e. SID and SOD and these signals are used for serial communication.
- **SOD** (Serial output data line) − The output SOD is set/reset as specified by the SIM instruction.

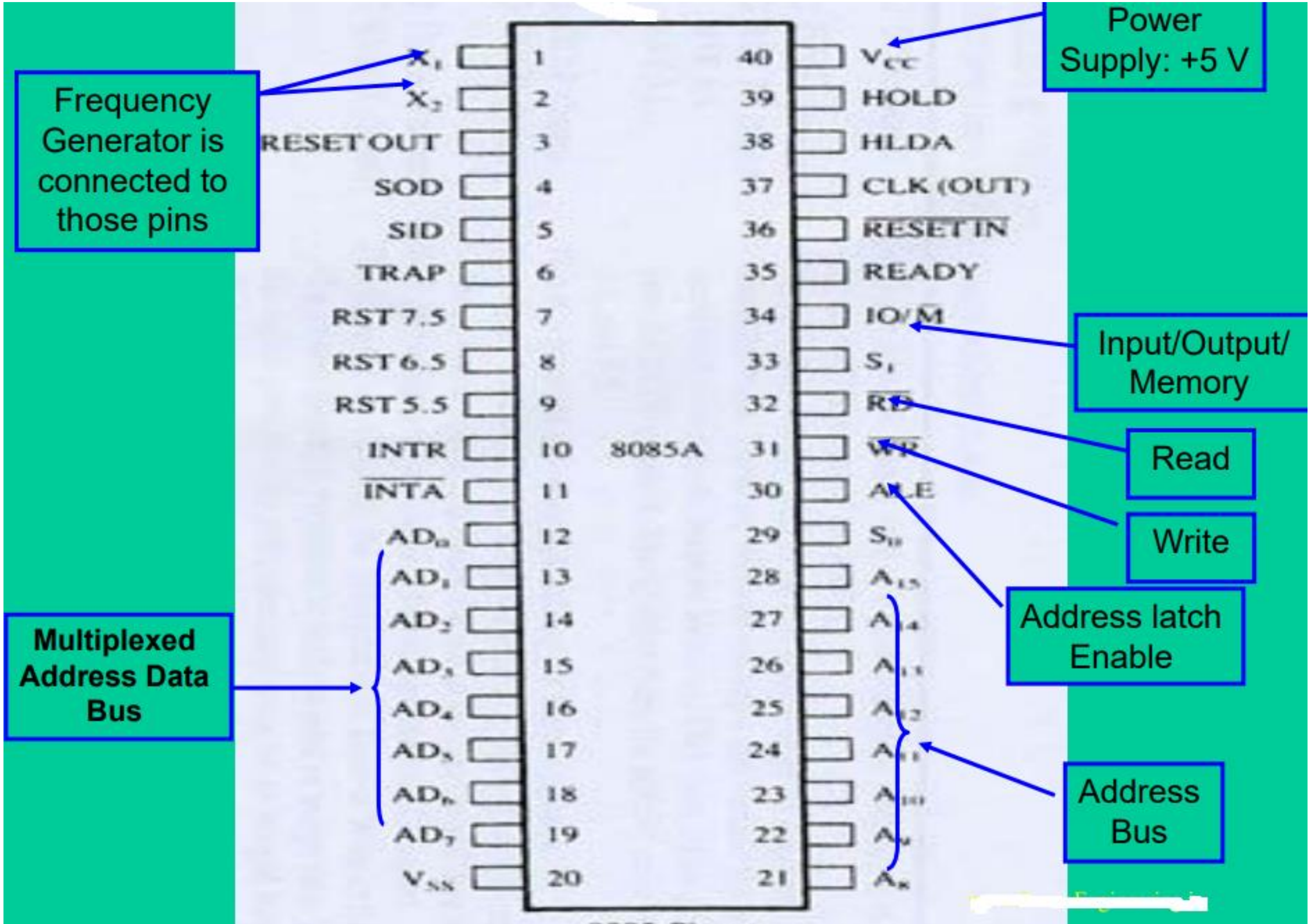
- **SID** (Serial input data line) − The data on this line is loaded into accumulator whenever a RIM instruction is executed.
- **Interrupts & externally initiated signals:** When a microprocessor is executing a main program and whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request. After the request is completed, the control goes back to the main program. There are 5 interrupt signals, i.e. TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.
- **INTA** − It is an interrupt acknowledgment signal.
- **RESET IN** − This signal is used to reset the microprocessor by setting the program counter to zero.
- **RESET OUT** − This signal is used to reset all the connected devices when the microprocessor is reset.
- **READY** − This signal indicates that the device is ready to send or receive data. If READY is low, then the CPU has to wait for READY to go high.

- **HOLD** − This signal indicates that another master is requesting the use of the address and data buses.

- **HLDA (HOLD Acknowledge)** − It indicates that the CPU has received the HOLD request and it will relinquish the bus in the next clock cycle. HLDA is set to low after the HOLD signal is removed.



8085 Pinout


Power Supply: +5 V
Frequency Generator is connected to those pins
Input/Output/ Memory
Read
Write
Address latch Enable
Multiplexed Address Data Bus
Address Bus
X1 1
X2 2
RESET OUT 3
SOD 4
SID 5
TRAP 6
RST 7.5 7
RST 6.5 8
RST 5.5 9
INTR 10
INTA 11
AD0 12
AD1 13
AD2 14
AD3 15
AD4 16
AD5 17
AD6 18
AD7 19
VSS 20
8085A
40 VCC
39 HOLD
38 HLDA
37 CLK (OUT)
36 RESET IN
35 READY
34 IO/M
33 S1
32 RD
31 WR
30 ALE
29 S0
28 A15
27 A14
26 A13
25 A12
24 A11
23 A10
22 A9
21 A8
8085 Pinout

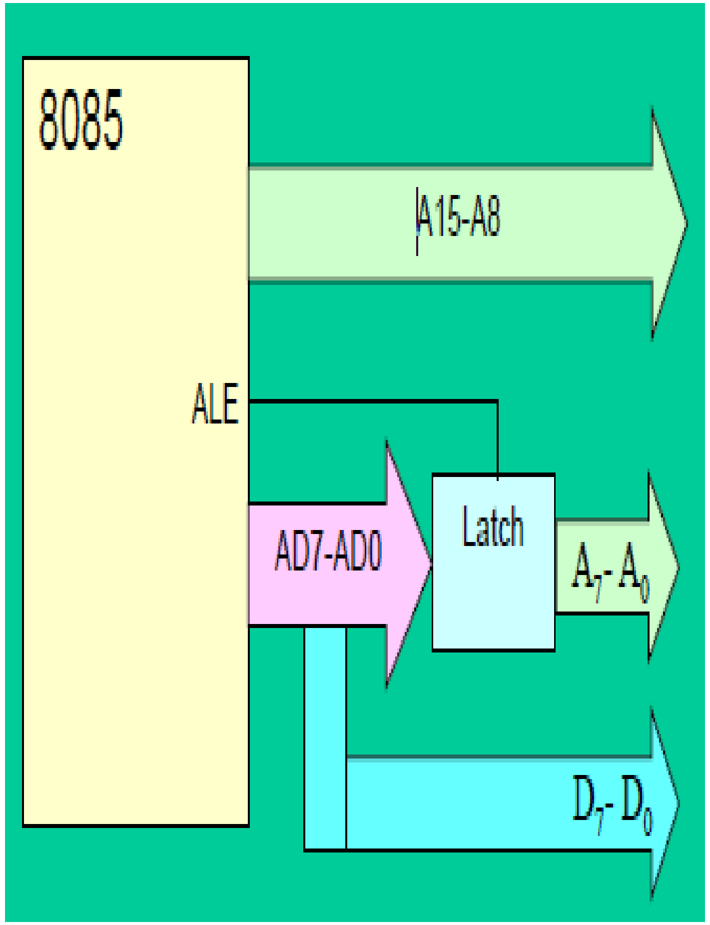
## Address and Data Busses

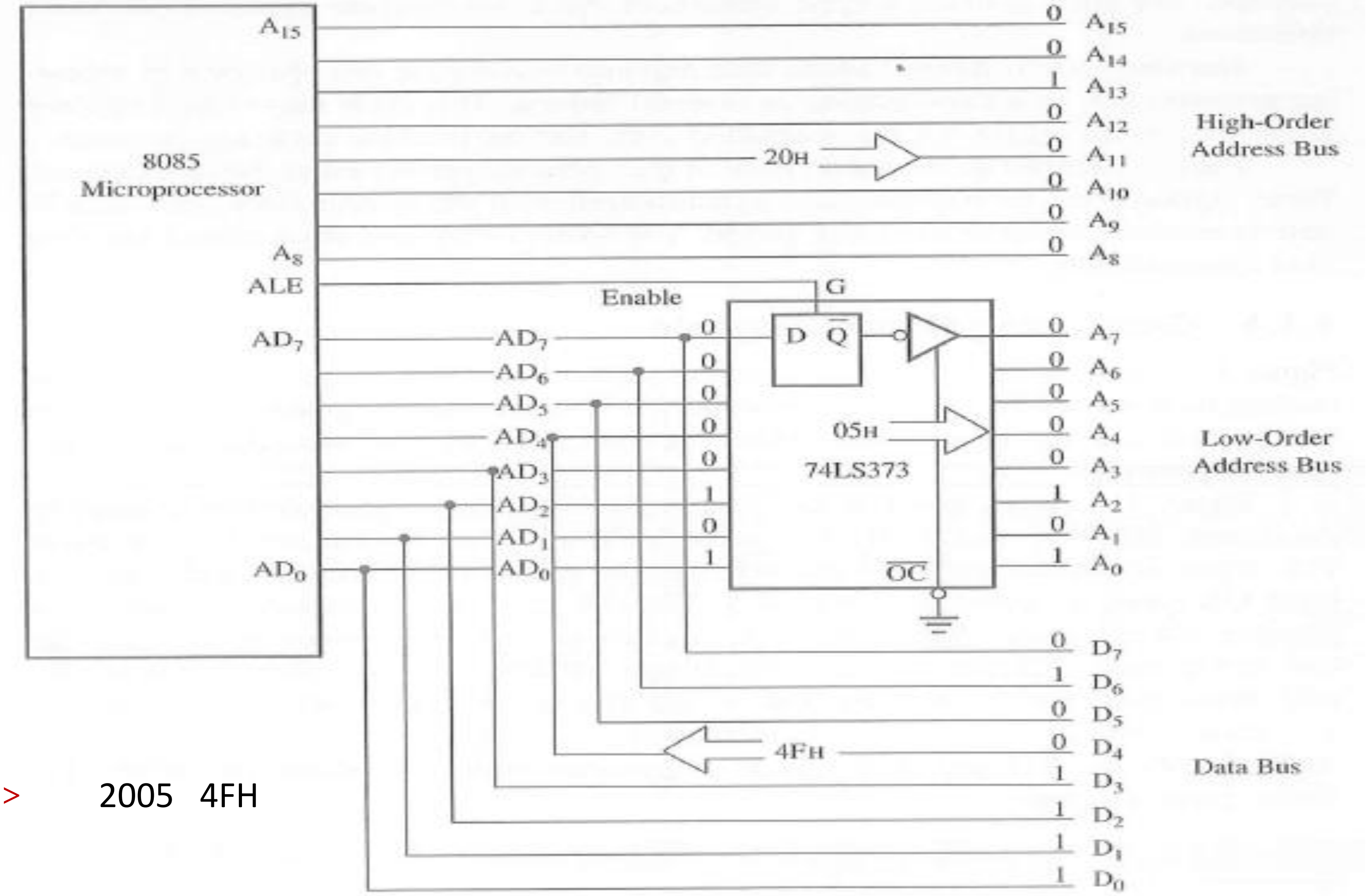
- The address bus has 8 signal lines A8 –A15which are unidirectional.
    - The other 8 address bits are multiplexed(time shared) with the 8 data bits. So, the bits AD0 –AD7are bi-directional and serve as A0 –A7and D0 –D7at the same time. During the execution of the instruction, these lines carry the address bits during the early part, then during the late parts of the execution, they carry the 8 data bits.
- In order to separate the address from the data, we can use a latch to save the value before the function of the bits changes.

## Demultiplexing AD7-AD0

- AD7–AD0 lines are serving as a dual purpose and that they need to be demultiplexed to get all the information.
- The high order bits of the address remain on the bus for three clock periods. However, the low order bits remain for only one clock period and they would be lost if they are not saved externally. Also, notice that the low order bits of the address disappear when they are needed most.
- To make sure we have the entire address for the full three clock cycles, we will use an external latch to save the value of AD7–AD0 when it is carrying the address bits. We use the ALE signal to enable this latch.
- Given that ALE operates as a pulse during T1, we will be able to latch the address. Then when ALE goes low, the address is saved and the AD7–AD0 lines can be used for their purpose as the bi-directional data lines.
- The high order address is placed on the address bus and hold for 3 clk periods,
- The low order address is lost after the first clk period, this address needs to be hold however we need to use latch
- The address AD7 –AD0 is connected as inputs to the latch 74LS373.
- The ALE signal is connected to the enable (G) pin of the latch and the OC –Output control –of the latch is grounded

# Demultiplexing AD7-AD0



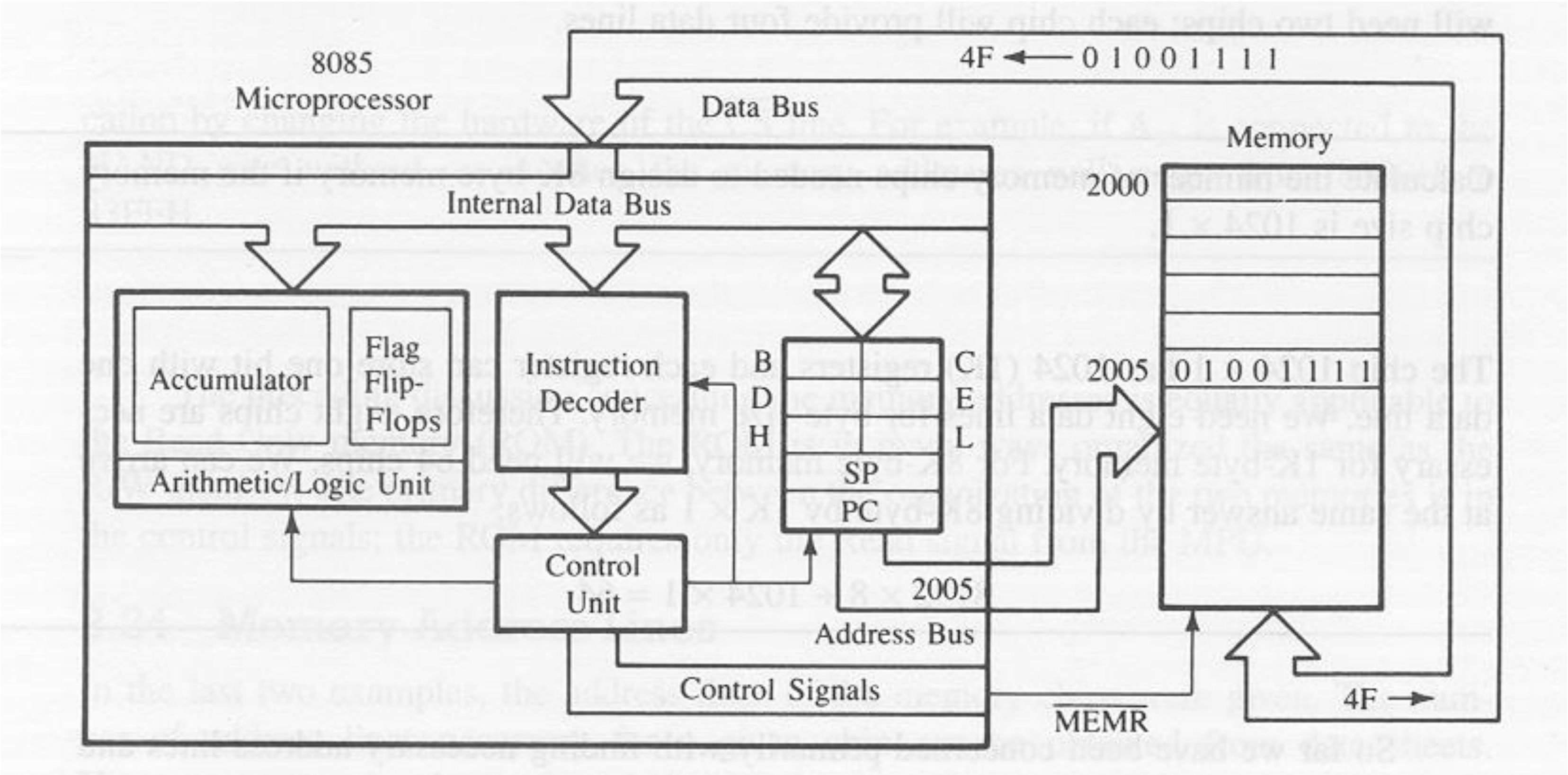




Example:

Instruction **: MOV C,A** **<A>→<C>** 2005 4FH

PC **: 2005**

# Instruction Fetch Operation




Example:

Instruction **: MOV C,A** **<A> → <C>** 2005 4FH

PC **: 2005**

# Instruction Fetch Operation

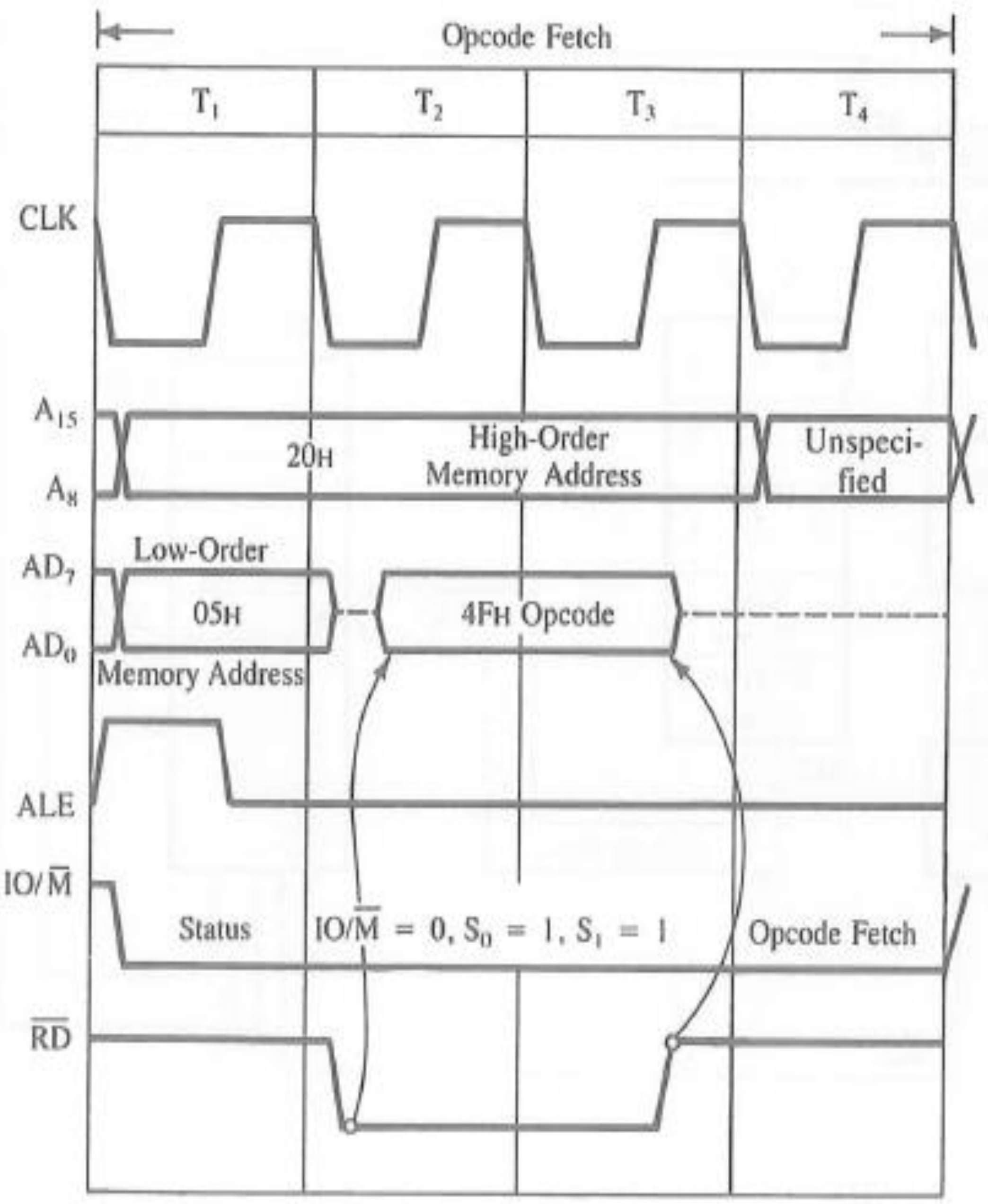
- All instructions (program steps) are stored in memory.
- To run a program, the individual instructions must be read from the memory in sequence, and executed.
- Program counter puts the 16-bit memory address of the instruction on the address bus
- Control unit sends the Memory Read Enable signal to access the memory
- The 8-bit instruction stored in memory is placed on the data bus and transferred to the instruction decoder
- Instruction is decoded and executed

Example:
Instruction : **MOV C,A** <A>→<C>
PC : **2005**

2005 4FH

| Machine Cycle | Status: $IO/\overline{M}$ | Status: $S_1$ | Status: $S_0$ | Control Signals |
|---|---|---|---|---|
| Opcode Fetch | 0 | 1 | 1 | $\overline{RD} = 0$ |
| Memory Read | 0 | 1 | 0 | $\overline{RD} = 0$ |
| Memory Write | 0 | 0 | 1 | $\overline{WR} = 0$ |
| I/O Read | 1 | 1 | 0 | $\overline{RD} = 0$ |
| I/O Write | 1 | 0 | 1 | $\overline{WR} = 0$ |
| Interrupt Acknowledge | 1 | 1 | 1 | $\overline{INTA} = 0$ |
| Halt | Z | 0 | 0 | $\overline{RD}$, $\overline{WR}$ = Z and $\overline{INTA}$ = 1 |
| Hold | Z | X | X | |
| Reset | Z | X | X | |

NOTE: Z = Tri-state (high impedance)
X = Unspecified



The process occurring in 4 t states of opcode fetch are:

**T1 :**

8085 transmits 16-bit address and also uses ALE signal for address latching.

**T2 :**

At T2 state microprocessor uses read signal and make data ready from that memory location to read opcode from memory and at the same time program counter increments by 1 and points next instruction to be fetched

**T3 :**

**At T3, the microprocessor reads the opcode and store it into instruction register to decode it further.**

**T4 :**

In this T state, the microprocessor decodes the opcode.

# Classification of instruction

## According to size

- 1 byte instruction (8 bits)
- 2 bytes instruction (16 bits)
- 3 bytes instruction (24 bits)

## According to function

- Data Transfer Instruction
- Arithmetic Instruction
- Logical Instruction
- Branching Instruction
- Machine Control Instruction