Course Contents

Unit-02: Intel 8085

- Functional Block Diagram and Pin Configuration,
- Timing and control Unit, Registers, Data and Address Bus,
- Operation Code and Operands, Addressing Modes, Interrupts, Flags, Instructions and Data Flow inside 8085,
- Intel 8085 Instructions (8085 Instructions: Data Transfer: MOV, IN, OUT, STA, LDA, LXI, LDAX, STAX, XCHG ; Arithmetic and Logic:- ADD, SUB, INR, DCR, AND, OR, XOR, ; CMP, RLC, RRC, RAL, RAR ; Branching:- JMP, JNZ, JZ, JNC, JC)
- Basic Assembly Language Programming using 8085 Instruction Sets: Addition, Subtraction, Multiplication and Division, Simple Sequence Programs, Array Searching and Sorting using Branching and Looping, Conversion (BCD to ASCII)

8085 Functional Block Diagram and Pin Configuration:

On completion of this chapter:

- recognize the functions of various pins of the 8085 microprocessor.
- list the various internal units that make up 8085 architecture, and explain their functions decoding and executing an instruction.
- draw the block diagram of an 8085 base microcomputer.

The 8085 MPU:

The 8085 microprocessor can almost qualify as an MPU, but with the following two limitations.

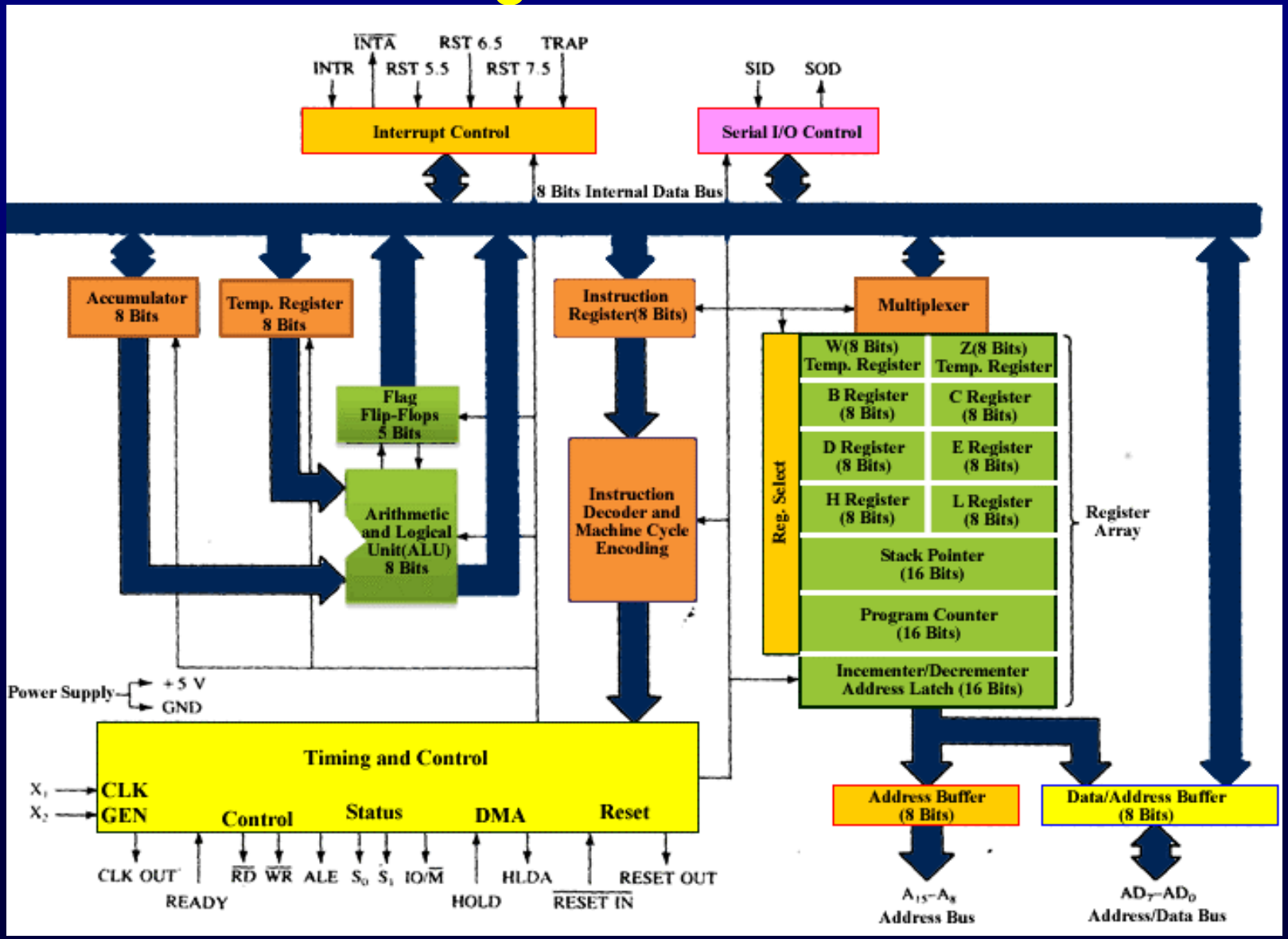
1. The low-order address bus of the 8085 microprocessor is multiplexed (time shared) with the data bus. The buses need to be demultiplexed.
2. Appropriate control signals need to be generated to interface memory and I/O with the 8085. (Intel has some specialized memory and I/O devices that do not require such control signals).

8085 Microprocessor and its operation

The 8085 Microprocessor:

- The 8085 is an 8-bit general purpose microprocessor capable of addressing 64K of memory.
- The device has 40 pins, requires a +5 V single power supply, and can operate with a 3-MHz single-phase clock.
- The 8085 is an enhanced version of predecessor, the 8080A; its instruction set is upward-compatible with that of the 8080A, meaning that the 8085 instruction set includes all the 8080A instructions plus some additional ones. Programs written for the 8080A will be executed by the 8085, but the 8085 and the 8080A are not pin compatible.

8085 MP Functional Block Diagram:



Functional Block Diagram and Pin Configuration

Functional 8085 block diagram:

- A 8085 microprocessor, is a second generation 8-bit microprocessor and is the base for studying and using all the microprocessor available in the market.
- 8085 is pronounced as "eighty-eighty-five" microprocessor. It is an 8-bit microprocessor designed by Intel in 1977 using NMOS technology. It has the following configuration –

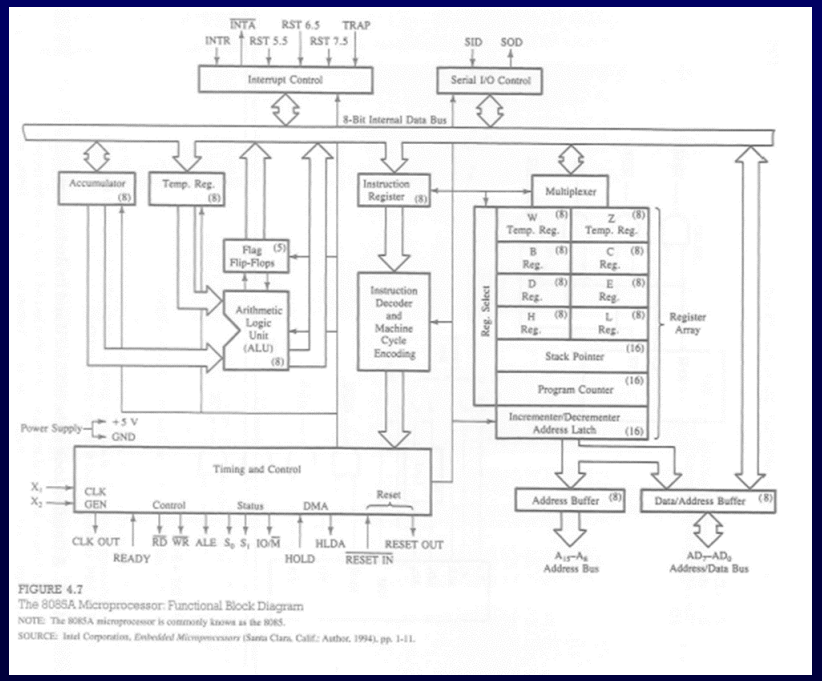
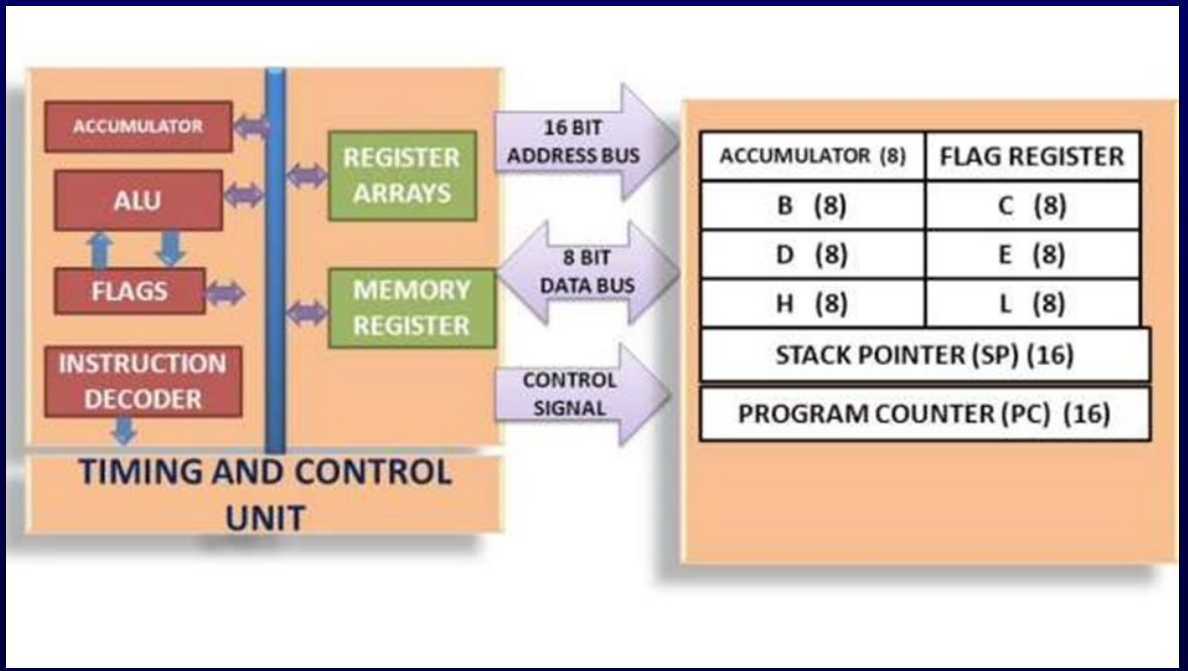


FIGURE 4.7
The 8085A Microprocessor: Functional Block Diagram
NOTE: The 8085A microprocessor is commonly known as the 8085.
SOURCE: Intel Corporation, *Embedded Microprocessors* (Santa Clara, Calif.: Author, 1994), pp. 1-11.



Functional Block Diagram and Pin Configuration:

Functional 8085 block diagram:

- 8-bit data bus
- 16-bit address bus, which can address upto 64KB
- A 16-bit program counter
- A 16-bit stack pointer
- Six 8-bit registers arranged in pairs: BC, DE, HL
- Requires +5V supply to operate at 3.2 MHZ single phase clock It is used in washing machines, microwave ovens, mobile phones, etc.

8085 Microprocessor and its operation

8085 consists of the following functional units:

Accumulator:

- It is an 8-bit register used to perform arithmetic, logical, I/O & LOAD/STORE operations. It is connected to internal data bus & ALU.

Arithmetic and logic unit(ALU):

As the name suggests, it performs arithmetic and logical operations like Addition, Subtraction, AND, OR, etc. on 8-bit data.

General purpose register:

There are 6 general purpose registers in 8085 processor, i.e. B, C, D, E, H & L. Each register can hold 8-bit data.

These registers can work in pair to hold 16-bit data and their pairing combination is like BC, DE & HL.

Functional Block Diagram and Pin Configuration:

Registers in 8085:

(a) General Purpose Registers:

The 8085 has six general-purpose registers to store 8-bit data; these are identified as- B, C, D, E, H, and L. These can be combined as register pairs – BC, DE, and HL, to perform some 16-bit operation. These registers are used to store or copy temporary data, by using instructions, during the execution of the program.

(b) Specific Purpose Registers

Accumulator:

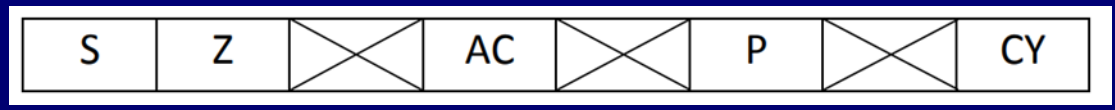
The accumulator is an 8-bit register (can store 8-bit data) that is the part of the arithmetic and logical unit (ALU). After performing arithmetical or logical operations, the result is stored in accumulator. Accumulator is also defined as register A.

Flag registers:

8085 Microprocessor and its operation

Flag registers:

The flag register contains 5-bit that are used as flags or indicator. Any time 8085 executes an arithmetic or logic instruction.



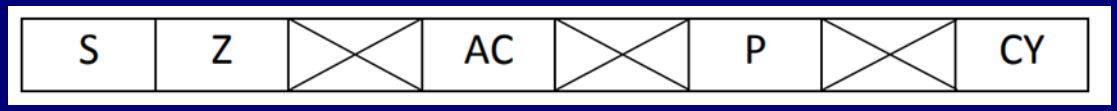
The flag register is a special purpose register and it is completely different from other registers in microprocessor. It consists of 8 bits and only 5 of them are useful. The other three are left vacant and are used in the future Intel versions. These 5 flags are set or reset (when value of flag is 1, then it is said to be set and when value is 0, then it is said to be reset) after an operation according to data condition of the result in the accumulator and other registers.

8085 Microprocessor and its operation

Flag registers: The 5 flag registers are:

S mean the sign bit and given:

- Logic 1 = (-ve).
- Logic 0 = (+ve).



Z mean zero flag and given:

- Logic 1 = zero result.
- Logic 0 = 1 result.

AC mean auxiliary carry and given:

- Logic 1 = there is a carry from bit 3 to bit 4.
- Logic 0 = no carry.

P mean parity flag and given:

- Logic 1 = the number of ones in accumulator is even.
- Logic 0 = the number of ones in accumulator is odd.

C mean carry flag and given:

- Logic 1 = there is a carry from last bit 7.
- Logic 0 = no carry from last 7 bit.

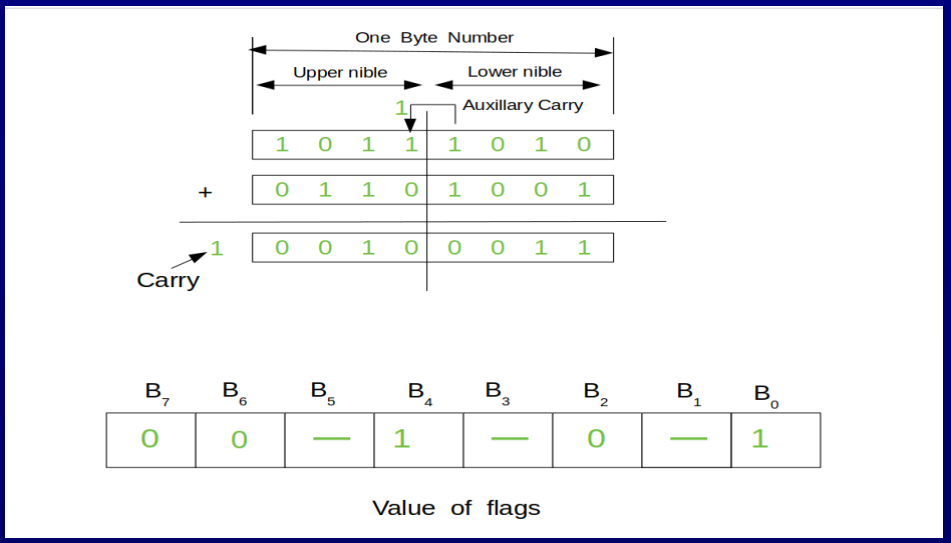
8085 Microprocessor and its operation

Uses of Flags:

- The ALU includes five flip-flops that are set or reset according to data conditions in the accumulator and other registers. The microprocessor uses them to perform the third operation; namely testing for data conditions.
- For example, after an addition of two numbers, if the sum in the accumulator is larger than eight bits, the flip-flop that is used to indicate a carry, carry the Carry flag (CY), is set to one. When an arithmetic operation results in zero, the flip-flop called the Zero flag (Z) is set to one.
- Flags have critical importance in the decision-making process of the microprocessor. The conditions (set or reset) of the flags are tested through software instructions.
- For example, the instruction JC (Jump On Carry) is implemented to change the sequence of a program when the CY flag is set. The implemented to flags cannot be emphasized enough; they will be discussed again in applications of conditional jump instructions.

Functional Block Diagram and Pin Configuration

Flag registers:



Example –

Here two binary numbers are added. The result produced is stored in the accumulator. Now lets check what each bit means. Refer to the below explanation simultaneously to connect them with the example.

- Sign Flag (7th bit): It is reset(0), which means number stored in the accumulator is positive.
- Zero Flag (6th bit): It is reset(0), thus result of the operations performed in the ALU is non-zero.
- Auxiliary Carry Flag (4th bit): We can see that b3 generates a carry which is taken by b4, thus auxiliary carry flag gets set (1).
- Parity Flag (2nd bit): It is reset(0), it means that parity is odd. The accumulator holds odd number of 1's.
- Carry Flag (0th bit): It is set(1), output results in more than 8 bit.

Functional Block Diagram and Pin Configuration

(c) Memory Registers

There are two 16-bit registers used to hold memory addresses. The size of these registers is 16 bits because the memory addresses are 16 bits. They are :-

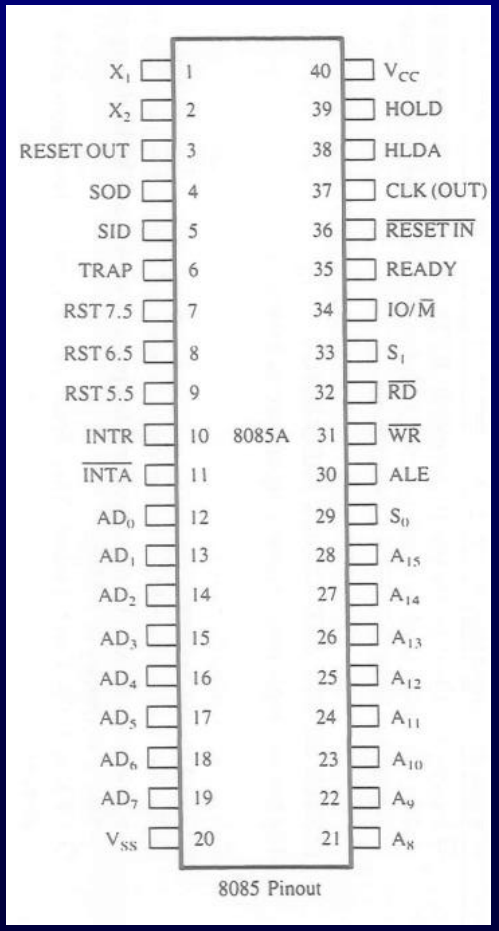
- **Program Counter:** This register is used to sequence the execution of the instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When a byte (machine code) is being fetched, the program counter is incremented by one to point to the next memory location.
- **Stack Pointer:** It is used as a memory pointer. It points to a memory location in read/write memory, called the stack. It is always incremented/decremented by 2 during push and pop operation.
- Temporary register: It is an 8-bit register, which holds the temporary data of arithmetic and logical operations.

Functional Block Diagram and Pin Configuration

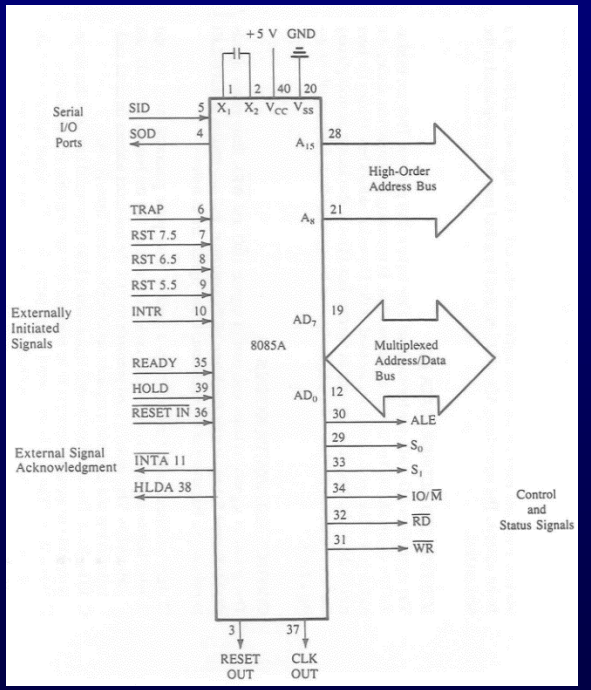
PIN Configuration of 8085 :

8085 is a 40 pin IC, DIP package. The signals from the pins can be grouped as follows

- Power supply and clock signals
- Address bus
- Data bus
- Control and status signals
- Interrupts and externally initiated signals
- Serial I/O ports

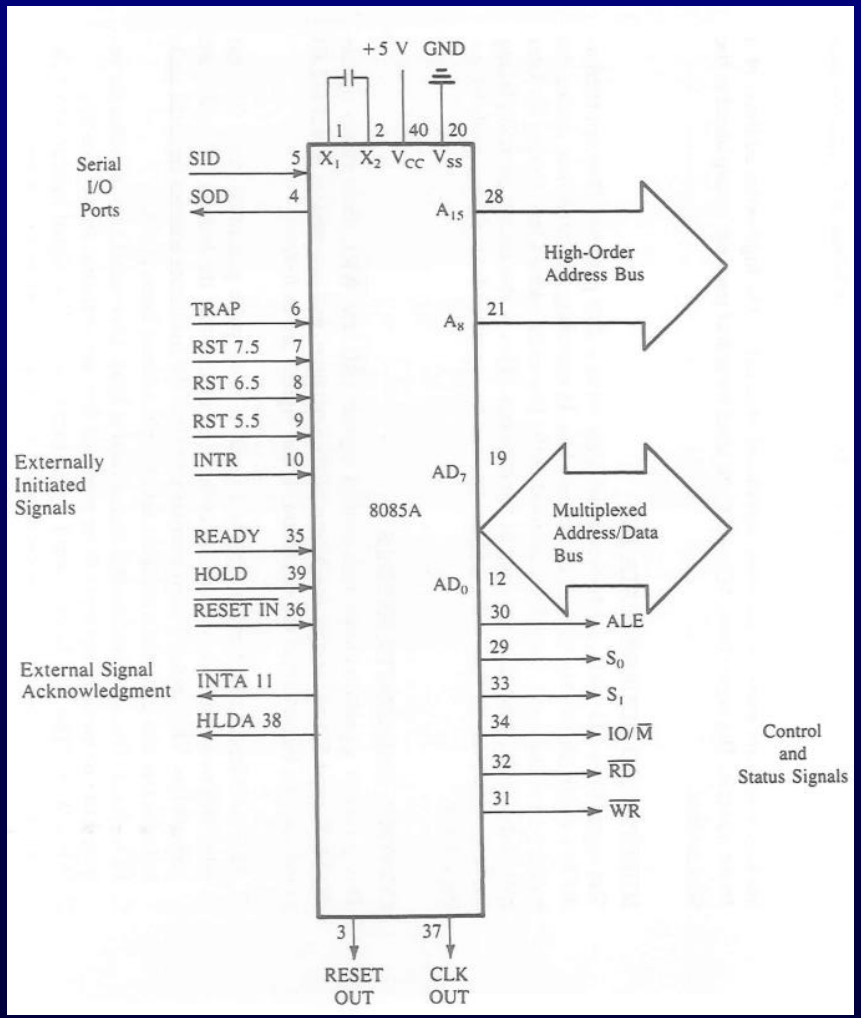


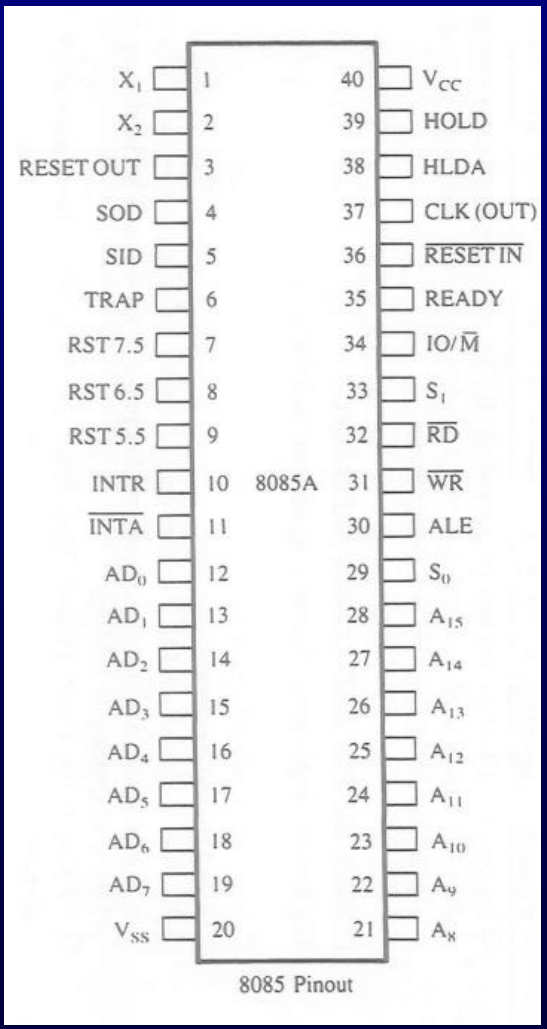
8085 Pinout



Functional Block Diagram and Pin Configuration:

PIN Configuration of 8085





8085 Pinout

8085 Microprocessor and its operation

8085 MP Functional Block Diagram:

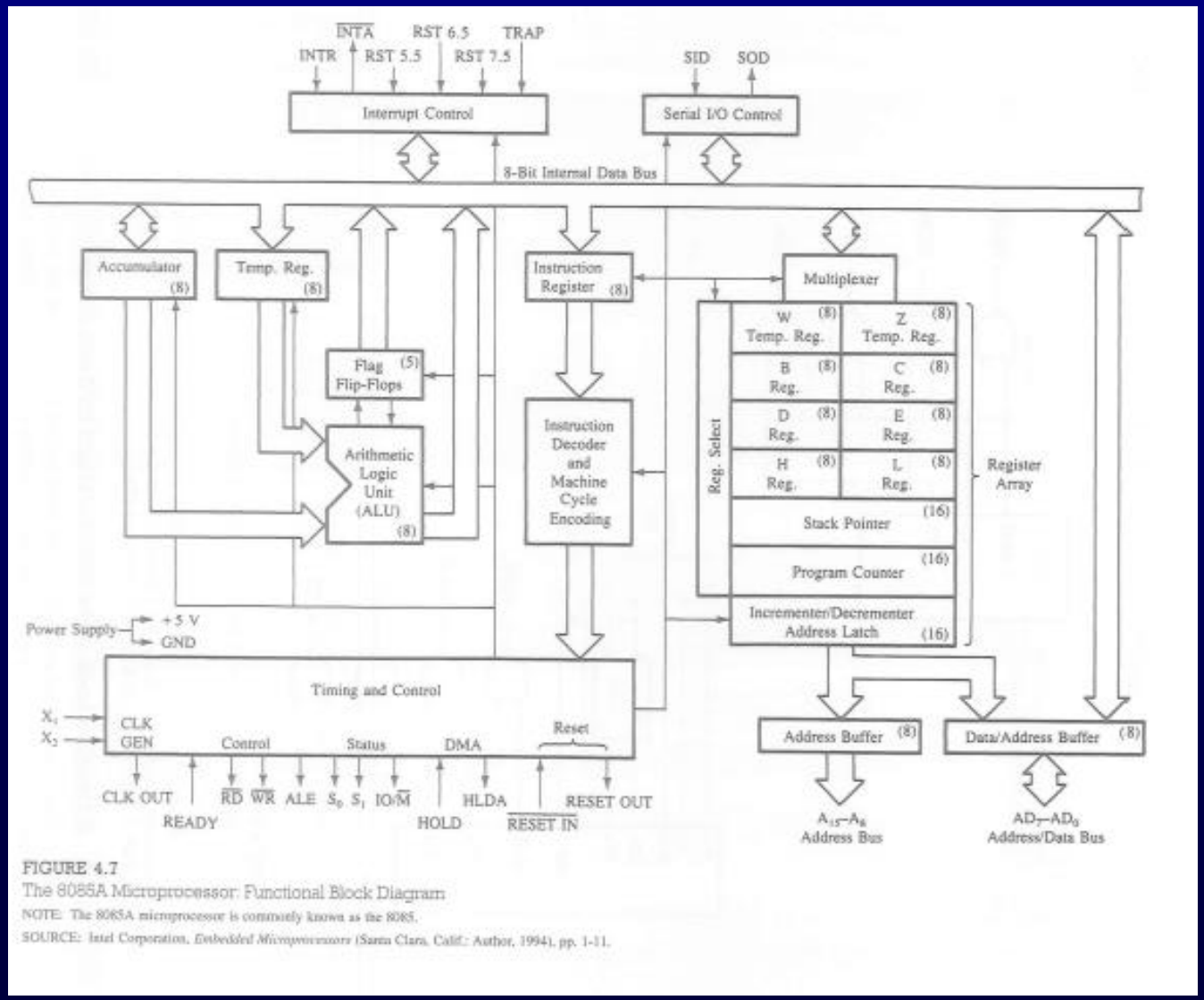


FIGURE 4.7
The 8085A Microprocessor: Functional Block Diagram

NOTE: The 8085A microprocessor is commonly known as the 8085.

SOURCE: Intel Corporation, *Embedded Microprocessors* (Santa Clara, Calif.: Author, 1994), pp. 1-11.

Functional Block Diagram and Pin Configuration:

1. **Power supply and Clock frequency signals:**
   - Vcc (Voltage Common Collector) + 5 volt power supply
   - Vss(Voltage Source Supply) Ground :+ve power supply
   - X1, X2 : Crystal or R/C(Resistor/Capacitor) network or LC(Inductor/Capacitor) network connections to set the frequency of internal clock generator.
   - The frequency is internally divided by two. Since the basic operating timing frequency is 3 MHz, a 6 MHz crystal is connected externally.
   - CLK (output)-Clock Output is used as the system clock for peripheral and devices interfaced with the microprocessor.

Functional Block Diagram and Pin Configuration:

2. Address Bus: A8 - A15

- It carries the most significant 8 bits of the memory address or the 8 bits of the I/O address.

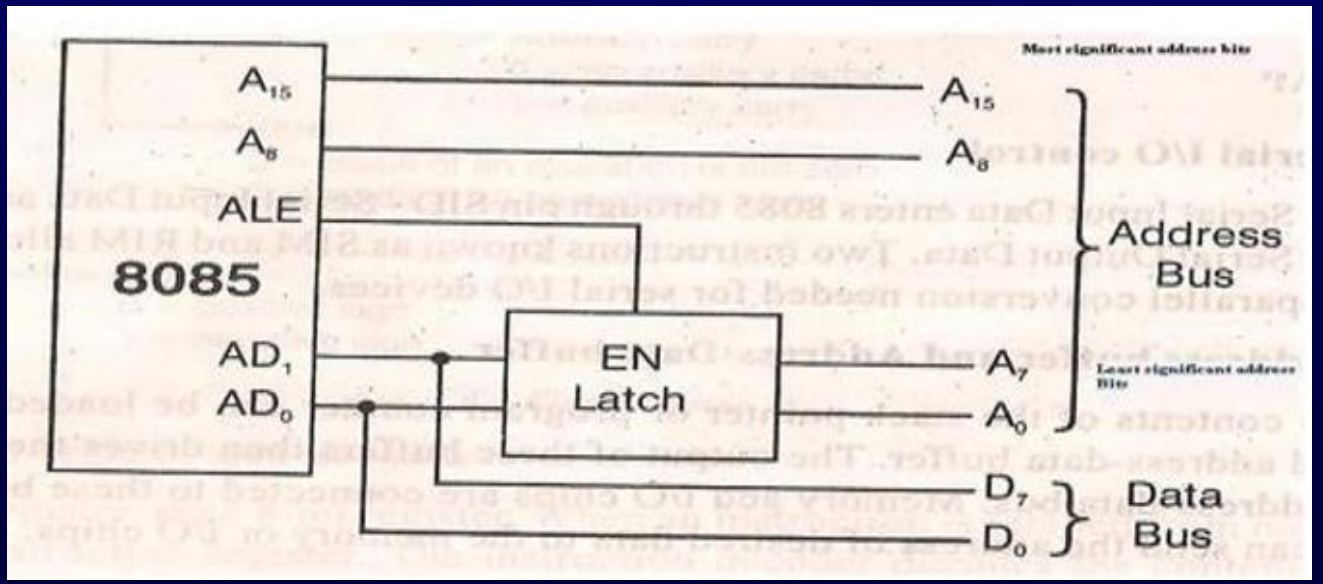
3. Multiplexed Address / Data Bus:

- AD0 - AD7
- These multiplexed set of lines used to carry the lower order 8 bit address as well as data bus.
- During the opcode fetch operation, in the first clock cycle, the lines deliver the lower order address A0 - A7.
- In the subsequent IO / memory, read / write clock cycle the lines are used as data bus. The CPU may read or write out data through these lines.

Functional Block Diagram and Pin Configuration

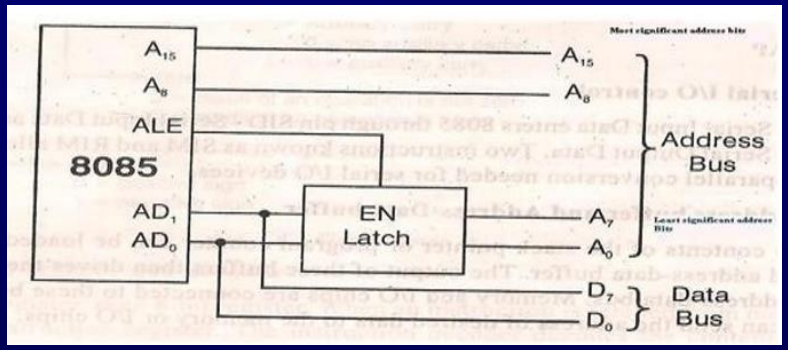
3. Multiplexed Address / Data Bus:

The signal lines AD7 to AD0 are bidirectional, they serve a dual purpose. They are used as the low-order address bus as well as the data bus. In executing an instruction, during the earlier part of the cycle, these lines are used as the low-order address bus. During the later part of the cycle, these lines are used as the data bus. (This is also known as multiplexing the bus). However, the low-order address bus can be separated from these signals by using a latch.



Functional Block Diagram and Pin Configuration:

Multiplexed Address / Data Bus:



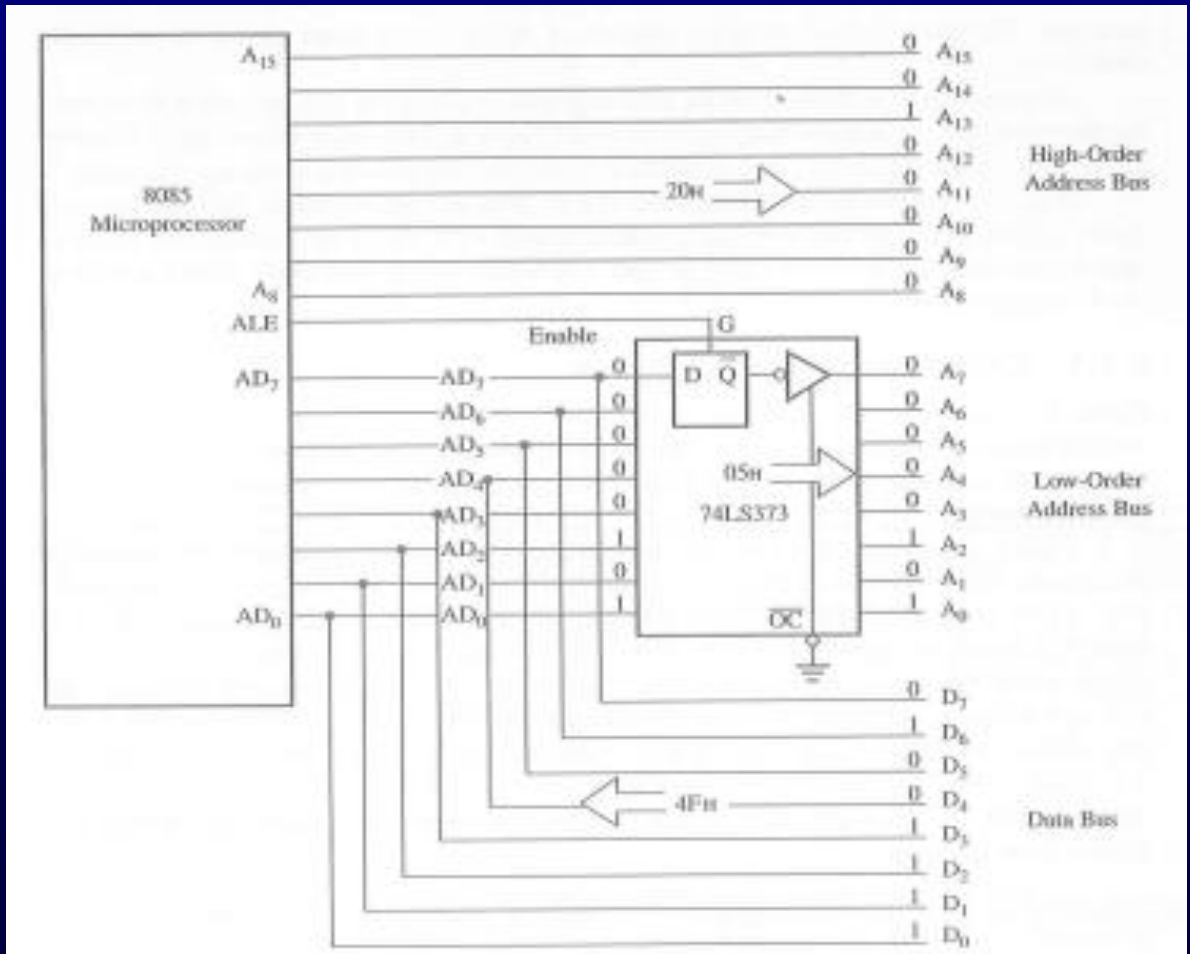


FIGURE 4.4
Schematic of Latching Low-Order Address Bus

Functional Block Diagram and Pin Configuration:

4. Control and Status signals:

These signals include two control signals ($\overline{RD}$ & $\overline{WR}$), three status signals (IO/ $\bar{M}$, S1 and So) to identify the nature of the operation and one special signal (ALE) to indicate the beginning of the operations.

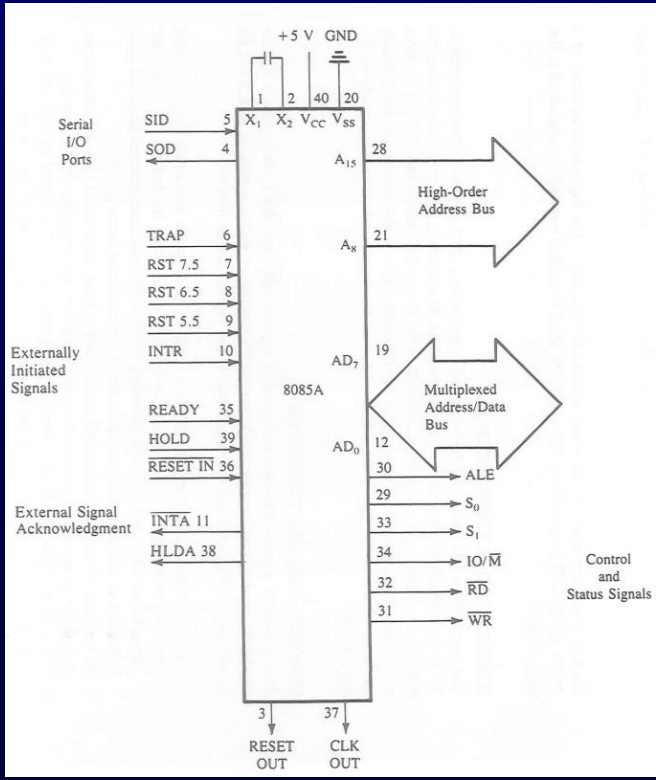


TABLE 4.1
8085 Machine Cycle Status and Control Signals

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

Functional Block Diagram and Pin Configuration

4. Control and Status signals:

ALE (output) - Address Latch Enable.

- This signal helps to capture the lower order address presented on the multiplexed address / data bus. When it is the pulse, 8085 begins an operation. It generates AD_0 - AD_7 as the separate set of address lines A_0 –A_7.

$\overline{RD}$ (active low) - Read memory or IO device.

- This indicates that the selected memory location or I/O device is to be read and that the data bus is ready for accepting data from the memory or I/O device.

$\overline{WR}$ (active low) - Write memory or IO device.

- This indicates that the data on the data bus is to be written into the selected memory location or I/O device.

IO/ $\overline{M}$ (output) - Select memory or an IO device.

- This status signal indicates that the read / write operation relates to whether the memory or I/O device.
- It goes high to indicate an I/O operation.
- It goes low for memory operations.

Functional Block Diagram and Pin Configuration

5. Status signals:

It is used to know the type of current operation of the microprocessor.

TABLE 4.1
8085 Machine Cycle Status and Control Signals

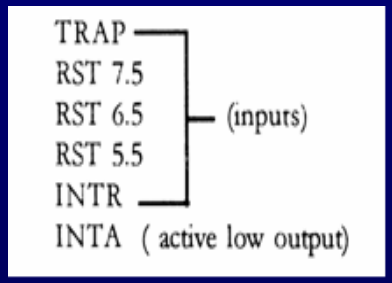
Machine Cycle	Status IO/$\overline{M}$	Status S_1	Status S_0	Control Signals
Opcode Fetch	0	1	1	$\overline{RD} = 0$
Memory Read	0	1	0	$\overline{RD} = 0$
Memory Write	0	0	1	$\overline{WR} = 0$
I/O Read	1	1	0	$\overline{RD} = 0$
I/O Write	1	0	1	$\overline{WR} = 0$
Interrupt Acknowledge	1	1	1	$\overline{INTA} = 0$
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

Functional Block Diagram and Pin Configuration:

6. Interrupts and Externally initiated operations:

They are the signals initiated by an external device to request the microprocessor to do a particular task or work. There are five hardware interrupts called,



On receipt of an interrupt, the microprocessor acknowledges the interrupt by the active low INTA (Interrupt Acknowledge) signal.

- **Hold (Input)**
  - This indicates peripheral controller requesting the bus.
- **HLDA (Output)**
  - This indicates the acknowledgement for the Hold request.

Functional Block Diagram and Pin Configuration

6. Interrupts and Externally initiated operations:

They are the signals initiated by an external device to request the microprocessor to do a particular task or work. There are five hardware interrupts.

8085 Interrupts and Externally Initiated Signals

☐ INTR (Input)	Interrupt Request: This is used as a general-purpose interrupt; it is similar to the INT signal of the 8080A.
☐ $\overline{\text{INTA}}$ (Output)	Interrupt Acknowledge: This is used to acknowledge an interrupt.
☐ RST 7.5 (Inputs) RST 6.5 RST 5.5	Restart Interrupts: These are vectored interrupts that transfer the program control to specific memory locations. They have higher priorities than the INTR interrupt. Among these three, the priority order is 7.5, 6.5, and 5.5.
☐ TRAP (Input)	This is a nonmaskable interrupt and has the highest priority.
☐ HOLD (Input)	This signal indicates that a peripheral such as a DMA (Direct Memory Access) controller is requesting the use of the address and data buses.
☐ HLDA (Output)	Hold Acknowledge: This signal acknowledges the HOLD request.
☐ READY (Input)	This signal is used to delay the microprocessor Read or Write cycles until a slow-responding peripheral is ready to send or accept data. When this signal goes low, the microprocessor waits for an integral number of clock cycles until it goes high.

Functional Block Diagram and Pin Configuration

6. Interrupts and Externally initiated operations:

- **READY (Input)**
  - It is used to delay the microprocessor read and write cycles until a slow responding peripheral is ready to send or accept data.
  - Memory and I/O devices will have slower response compared to microprocessors.
  - Before completing the present job such a slow peripheral may not be able to handle further data or control signal from CPU.
  - The processor sets the READY signal after completing the present job to access the data.
    - The microprocessor enters into WAIT state while the READY pin is disabled.
- **Reset In (input, active low)**

This signal is used to reset the microprocessor. The program counter inside the microprocessor is set to zero. The buses are tri-stated.

Functional Block Diagram and Pin Configuration

6. Interrupts and Externally initiated operations:

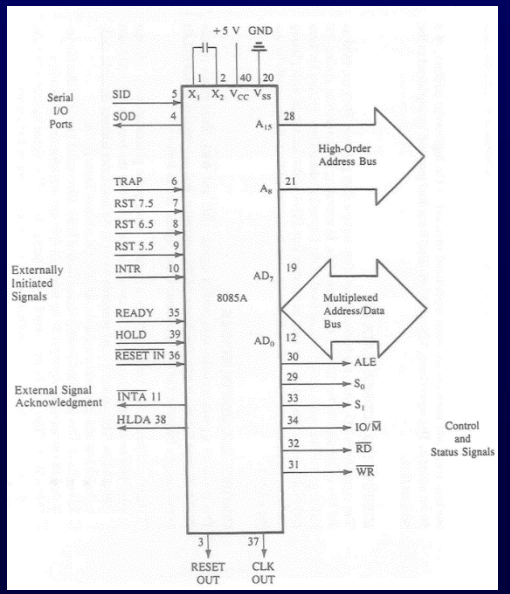
- **Reset In (input, active low)**

This signal is used to reset the microprocessor. The program counter inside the microprocessor is set to zero. The buses are tri-stated.

- **Reset Out (Output)**
  - It indicates CPU is being reset.
  - Used to reset all the connected devices when the microprocessor is reset.

7. Single Bit Serial I/O ports:

- SID (input) - Serial input data line
- SOD (output) - Serial output data line
- These signals are used for serial communication.



The 8085 CPU

Instruction, data format and storage :

An instruction is a command to the microprocessor to perform a given task on a specified data. Each instruction has two parts: one is task to be performed, called the operation code (opcode), and the second is the data to be operated on, called the operand. The operand (or data) can be specified in various ways. It may include 8-bit (or 16-bit) data, an internal register, a memory location, or 8-bit (or 16-bit) address. In some instructions, the operand is implicit. The 8085 microprocessor instruction set has 74 operation codes that result in 246 instructions.

The 8085 CPU

Instruction, data format and storage :

Instruction Word Size:

The 8085 instruction set is of three groups according to word size:

- One-word or one-byte instructions.
- Two-word or two-byte instructions.
- Three-word or three-byte instructions.

In the 8085 microprocessor, byte and words are synonymous because it is an 8-bit microprocessor. But, instructions are commonly referred to in terms of bytes rather than words.

The 8085 CPU

Instruction, data format and storage :

One-byte instructions:

A one-byte instruction includes a opcode and a operand in the same byte.

Task	Opcode	Operand	Binary Code	Hex code
Copy the content of accumulator in the register C.	MOV	C, A	0100 1111	4FH
Add the contents of register B to the contents of the accumulator.	ADD	B	1000 0000	80H
Invert each bit in the accumulator.	CMA	None	0010 1111	2FH

In the first instruction, operand and registers are specified. In the second instruction, the operand B is specific and the accumulator is not there. Similarly, in the third instruction, the accumulator is assuming to be the implicit operand.

The 8085 CPU

Instruction, data format and storage :

Two-byte instructions :

In a two-byte instruction, the first byte specifies the operation code and second byte specifies the operand.

Task	Opcode	Operand	Binary code	Hex code
Load an 8-bit data byte in the accumulator.	MVI	A, 35H	0011 1110 0011 0101	3EH First byte 35H Second byte

The 8085 CPU

Instruction, data format and storage :

Three-byte instructions :

In a three-byte instruction, the first byte specifies the opcode, and the following two bytes specify the 16-bit operand. The second byte is the low-order operand and the third byte is the high-order operand. If a 16-bit numeral is present in the instruction then that instruction will be of three-byte. For example, in LXI H,3500H and STA 2500H, etc.

Task	Opcode	Operand	Binary code	Hex code
Transfer the program sequence to the memory location 2085h	JMP	2550H	1100 0011 0101 0000 0010 0101	C3 First byte 50 Second byte 25 Third byte

The 8085 CPU

Instruction, data format and storage : Example

OPCODE	OPERAND	EXPLANATION	EXAMPLE
MOV	Rd, Rs	Rd = Rs	MOV A, B
MOV	Rd, M	Rd = Mc	MOV A, M
MOV	M, Rs	M = Rs	MOV M, A
MVI	Rd, 8-bit data	Rd = 8-bit data	MVI A, 50
MVI	M, 8-bit data	M = 8-bit data	MVI M, 50
LDA	16-bit address	A = contents at address	LDA 2050
STA	16-bit address	contents at address = A	STA 2050
LXI	r.p., 16-bit data	loads the specified register pair with data	LXI H, 3050
LDAX	r.p.	indirectly loads at the accumulator A	LDAX H
STAX	16-bit address	indirectly stores from the accumulator A	STAX 2050
XCHG	none	exchanges H with D, and L with E	XCHG
PUSH	r.p.	pushes r.p. to the stack	PUSH H
POP	r.p.	pops the stack to r.p.	POP H
OPCODE	OPERAND	EXPLANATION	EXAMPLE
IN	8-bit port address	inputs contents of the specified port to A	IN 15
OUT	8-bit port address	outputs contents of A to the specified port	OUT 15

The 8085 CPU

Instruction, data format and storage :

Opcode Format :

In the case of microprocessor, the instruction or operation are specified by using specific bit pattern unique for each instruction. These bit patterns contain all the information about operation, register used, memory to. The register and register pair are specified by using certain combination of bits. These combinations of bits are in table below.

Registers	Code
B	0 0 0
C	0 0 1
D	0 1 0
E	0 1 1
H	1 0 0
L	1 0 1
M (Memory)	1 1 0
A	1 1 1

Register Pairs	Code
BC	0 0
DE	0 1
HL	1 0
AF or SP	1 1

The 8085 CPU

Instruction, data format and storage :

Opcode Format :

In the case of microprocessor, the instruction or operation are specified by using specific bit pattern unique for each instruction. These bit patterns contain all the information about operation, register used, memory to. The register and register pair are specified by using certain combination of bits. These combinations of bits are in table below.

In assembly language, this is expressed as

Opcode	Operand	Hex Code
ADD	B	80H

3. MOVE (Copy) the content of register Rs (source) to register Rd (destination) — 01 (2-bit Opcode for MOVE) — DDD (Reg. Rd) — SSS (Reg. Rs)

This instruction is completed by adding the codes of two registers. For example,

Move (copy) the content: 0 1
To register C : 0 0 1 (DDD)
From register A : 1 1 1 (SSS)
Binary Instruction : 0 1 | 0 0 1 | 1 1 1 → 4FH
Opcode Operand

In assembly language, this is expressed as

Opcode	Operand	Hex Code
MOV	C,A	4F

Registers	Code
B	0 0 0
C	0 0 1
D	0 1 0
E	0 1 1
H	1 0 0
L	1 0 1
M (Memory)	1 1 0
A	1 1 1

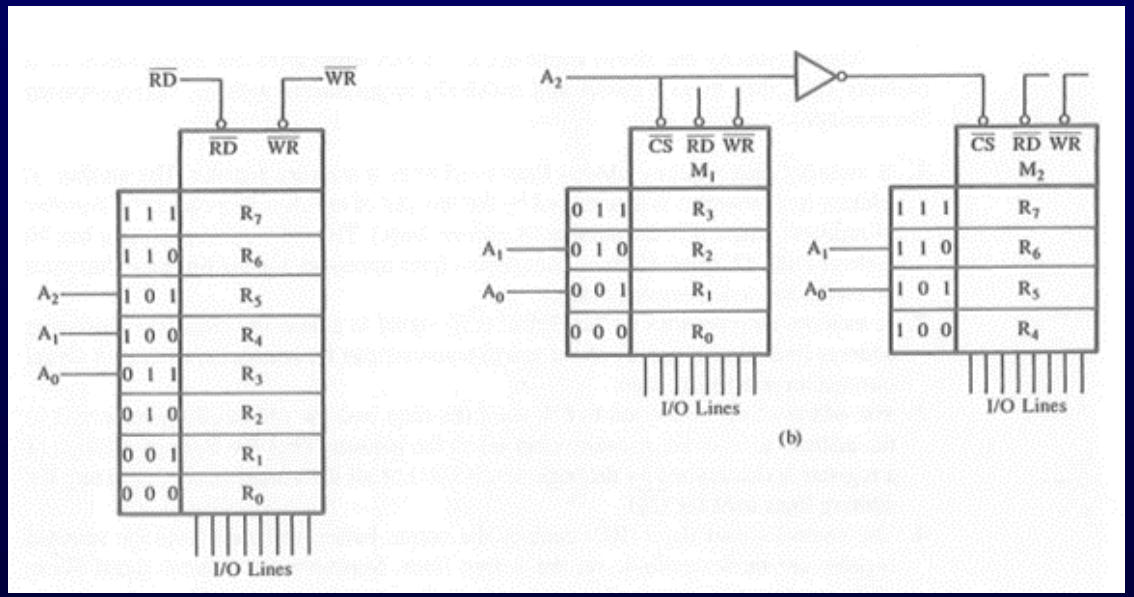
Register Pairs	Code
BC	0 0
DE	0 1
HL	1 0
AF or SP	1 1

The 8085 CPU

Instruction, data format and storage :

Data Storage:Memory

- Memory is a storage of binary bits. Memory chips used in most systems are nothing but 8-bit registers stacked one above the other.
- How to write, assemble and execute a simple program
- A program is a sequence of instruction written to tell a computer to perform a specific function.



The 8085 CPU

Overview of the 8085-instruction set:

The 8085-microprocessor instruction set has 74 operation codes that result in 246 instructions. Since the 8085 is an 8-bit device it can have up to 28 (256) instructions. However, the 8085 only uses 246 combinations that represent a total of 74 instructions. Most of the instructions have more than one format. These instructions can be grouped into five different groups:

1. Data Transfer Operations
2. Arithmetic Operations
3. Logic Operations
4. Branch Operations
5. Machine Control Operations

Addressing Modes

Overview of the 8085-instruction set:

The instructions from the 8085 instruction set introduced in this chapter are summarized below to provide an overview. After careful examination of these instructions, you will begin to see a pattern emerge from the mnemonics, the number of bytes required for the various instructions, and the tasks the 8085 can perform. Read the notations (Rs) as the contents of the source register, (Rd) as the contents of the destination register, (A) as the contents of the accumulator, and (R) as the contents of the register R.*

Instructions	Tasks	Addressing Mode
Data transfer (Copy) Instructions		
1. MOV Rd,Rs	Copy (Rs) into (Rd).	Register
2. MVI R,8-bit	Load register R with the 8-bit data.	Immediate
3. IN 8-bit port address	Read data from the input port.	Direct
4. OUT 8-bit port address	Write data in the output port.	Direct
Arithmetic Instructions		
1. ADD R	Add (R) to (A).	Register
2. ADI 8-bit	Add 8-bit data to (A).	Immediate
3. SUB R	Subtract (R) from (A).	Register
4. SUI 8-bit	Subtract 8-bit data from (A).	Immediate

*R, Rs, and Rd represent any one of the 8-bit registers—A, B, C, D, E, H, and L.

5. INR R	Increment (R).	Register
6. DCR R	Decrement (R).	Register

Overview of the 8085-instruction set :

Logic Instructions		
1. ANA R	Logically AND (R) with (A).	Register
2. ANI 8-bit	Logically AND 8-bit data with (A).	Immediate
3. ORA R	Logically OR (R) with (A).	Register
4. ORI 8-bit	Logically OR 8-bit data with (A).	Immediate
5. XRA R	Logically Exclusive-OR (R) with (A).	Register
6. XRI 8-bit	Logically Exclusive-OR 8-bit data with (A).	Immediate
7. CMA	Complement (A).	

Addressing Modes

Overview of the 8085-instruction set :

Branch Instructions

1. JMP 16-bit	Jump to 16-bit address unconditionally.	Immediate
2. JC 16-bit	Jump to 16-bit address if the CY flag is set.	Immediate
3. JNC 16-bit	Jump to 16-bit address if the CY flag is reset.	Immediate
4. JZ 16-bit	Jump to 16-bit address if the Zero flag is set.	Immediate
5. JNZ 16-bit	Jump to 16-bit address if the Zero flag is reset.	Immediate
6. JP 16-bit	Jump to 16-bit address if the Sign flag is reset.	Immediate
7. JM 16-bit	Jump to 16-bit address if the Sign flag is set.	Immediate
8. JPE 16-bit	Jump to 16-bit address if the Parity flag is set.	Immediate
9. JPO 16-bit	Jump to 16-bit address if the Parity flag is reset.	Immediate

Overview of the 8085-instruction set :

Machine Control Instructions

1. NOP	No operation.
2. HLT	Stop processing and wait.

The set of instructions listed here is used frequently in writing assembly language programs. The important points to be remembered about these instructions are as follows:

1. The data transfer (copy) instructions copy the contents of the source into the destination without affecting the source contents.
2. The results of the arithmetic and logic operations are usually placed in the accumulator.
3. The conditional Jump instructions are executed according to the flags set after an operation. Not all instructions set the flags; in particular, the data transfer instructions do not set the flags.

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

Every instruction is divided into two parts, one part is opcode and another part is operand. Opcode tells the type of operation and operand is the data on which operation is to be performed.

The way in which operand is specified in an instruction is called addressing mode. There are five types of addressing modes of 8085 microprocessor.

1. Register Addressing Mode
2. Immediate Addressing Mode
3. Direct Addressing Mode
4. Indirect Addressing Mode
5. Implied/Implicit Addressing Mode

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

1. Register Addressing Mode

In this mode the operands are registers of microprocessor. The operation is performed within different registers of the microprocessor. Examples of register addressing mode are given below.

```
ADD L
MOV C, D
SUB L
```

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

2. Immediate Addressing Mode

In this mode the 8 bit or 16 bit data is specified in the instruction itself as its one of the operand. Examples of immediate addressing mode are given below.

ADI 05H

MVI A, 33H

LXI H, 2050H ; 2050 is loaded into the HL register pair

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

3. Direct Addressing Mode

In this addressing mode one of the operand is data stored in the memory. The memory address of the data is directly given in the instruction itself. Examples of direct addressing are given below.

STA 3050H ; Store accumulator content in the memory address 3050H

OUT 02H ; Output Accumulator contents to an output port 8-bit address 02H

LDA 2345H ; Load Accumulator with the contents from memory address 2345H

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

4. Indirect Addressing Mode

In this mode also one of the operands is the data stored in the memory. The memory address of the data is specified by the register pair.

```
STAX B          //memory address is specified by BC register pair
LDAX D          //memory address is specified by DE register pair
MOV M, C        //memory address is specified by HL register pair
```

The 8085 CPU

Addressing Modes of 8085 Microprocessor:

5. Implied/Implicit Addressing Mode

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

In 8085 Instruction set, there is one mnemonic XCHG, which stands for eXCHanGe. This is an instruction to exchange contents of HL register pair with DE register pair. This instruction uses implied addressing mode. In the instruction, we don't mention as "XCHG HL, DE". It is implied that it will deal with HL and DEregister pairs. So we write only XCHG as mnemonic. That's why it is called an implied addressing mode.

```
XCHG    ;( exchange contents of HL register pair with DE register pair)
CMA     ;(finds and stores the 1's complement of the contains of accumulator A in A)
RRC     ;(rotate accumulator A right by one bit)
RLC     ;(rotate accumulator A left by one bit)
```

8085 instruction cycle, machine cycle, T states:

Operation of a microprocessor can be classified in to following four groups according to their nature.

- Opcode fetch
- Memory Read
- Memory Write
- I/O Read
- I/O Write
- Request acknowledgement

TABLE 4.1
8085 Machine Cycle Status and Control Signals

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

During Opcode fetch cycle, fetches the instructions from memory and delivers it to the instruction register of the microprocessor. For any instruction cycle, Opcode fetch is the first machine cycle. During three operations, microprocessor generates and receives different signals. These all operations are terms as machine cycle.

8085 instruction cycle, machine cycle, T states

- Clock Cycle (T state): It is defined as one subdivision of the operation performed in one clock period.
- Machine Cycle: It is defined as the time required to complete one operation of accessing memory, I/O, or acknowledging an external request. This cycle may consist of three to six T-states.
- Instruction Cycle: It is defined as the time required completing the execution of an instruction. The 8085-instruction cycle consists of one to six machine cycles or one to six operations.

8085 instruction cycle, machine cycle, T states

Opcode fetch Machine Cycle:

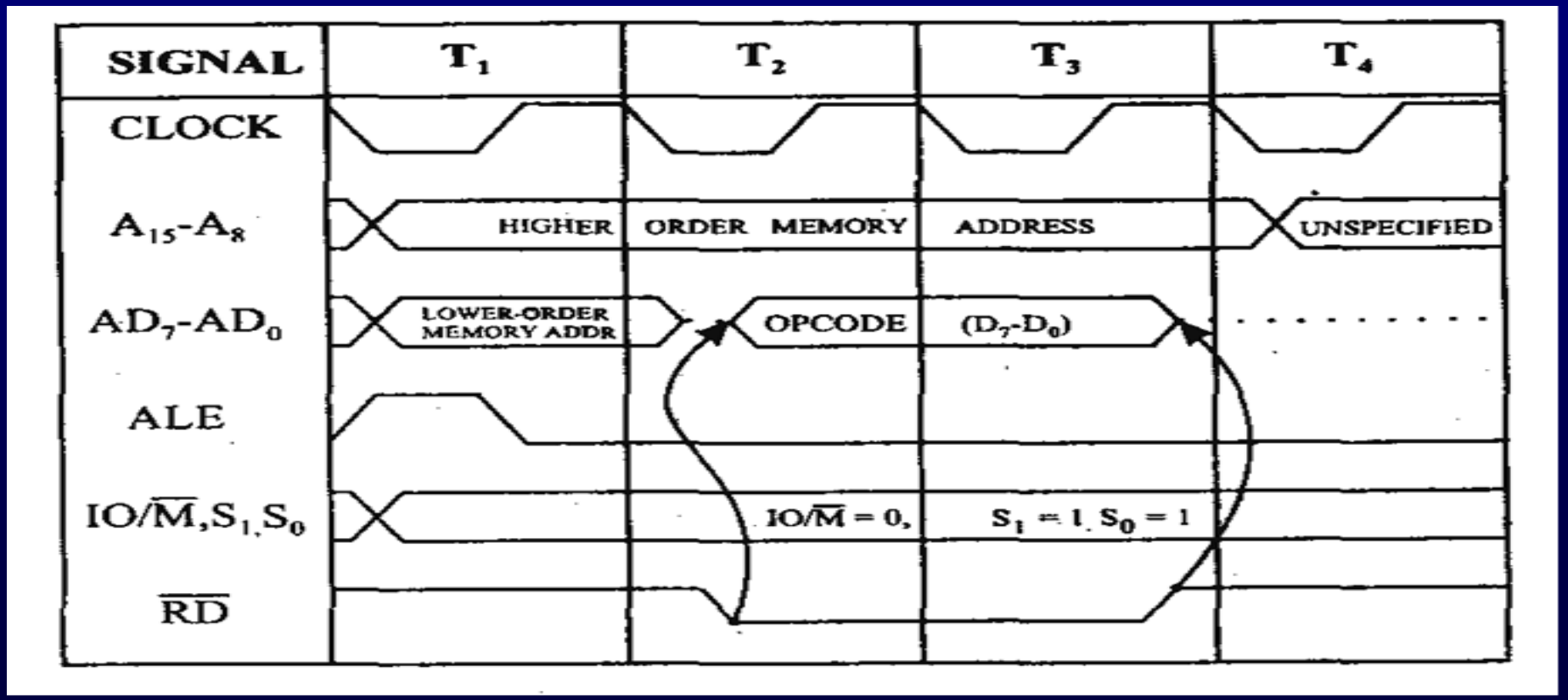
- The first operation in any instruction is Opcode fetch, The microprocessor needs to get(fetch) this machine code from the memory register where it is stored before the microprocessor can begin to execute the instruction.
- Let's consider the instruction MOV C, A stored at memory location 2005H. The Op-Code for the instruction is 4FH and Op-Code fetch cycle is of 4 clock cycles.

MOV C, A	2005H	4F

8085 instruction cycle, machine cycle, T states

Timing Diagram for Opcode Fetch Machine Cycle :

MOV C, A 2005H 4F

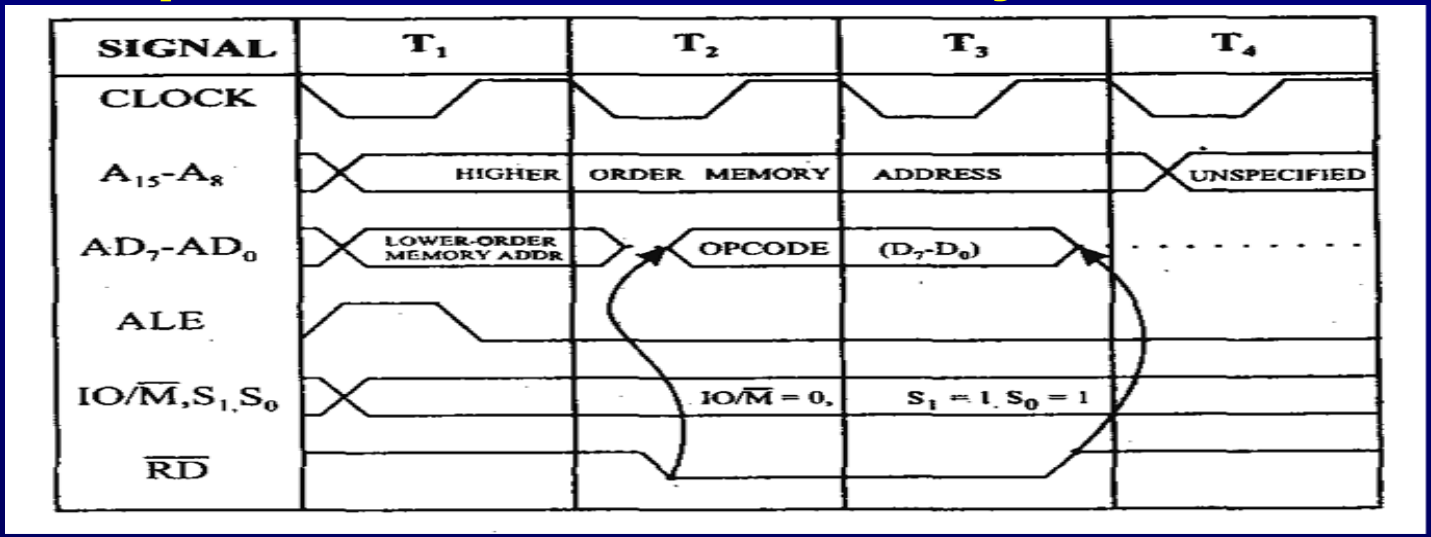


8085 instruction cycle, machine cycle, T states

Timing Diagram for Opcode Fetch Machine Cycle :

MOV C, A

2005H 4F



- **Step1:** Microprocessor places the 16 bit memory address from Program Counter on the address bus. At T_1, high order address (20) is placed at A_8-A_{15} and lower order address (05) is placed at AD_0-AD_7 ALE signal goes high. IO/M goes low and both S_0 and S_I goes high for Op-Code fetch.
- **Step 2:** The control unit sends the control signal RD to enable the memory chip and active during T_2 and T_3.
- **Step 3:** The byte from the memory location is placed on the data bus that is 4f into D_0-D_7 and RD goes high impedance.
- **Step4:** The instruction 4FH is decoded and content of accumulator will be copied into register C during clock cycle T_4.

8085 instruction cycle, machine cycle, T states

Memory Read Cycle:

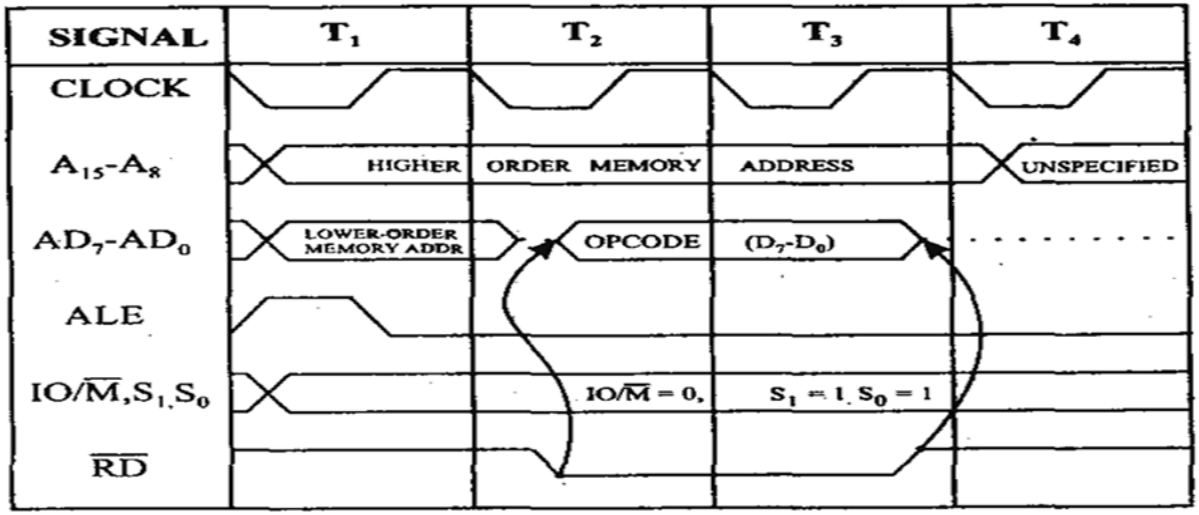
Let's consider the instruction MVI A, 32 H stored at memory location 2000H.

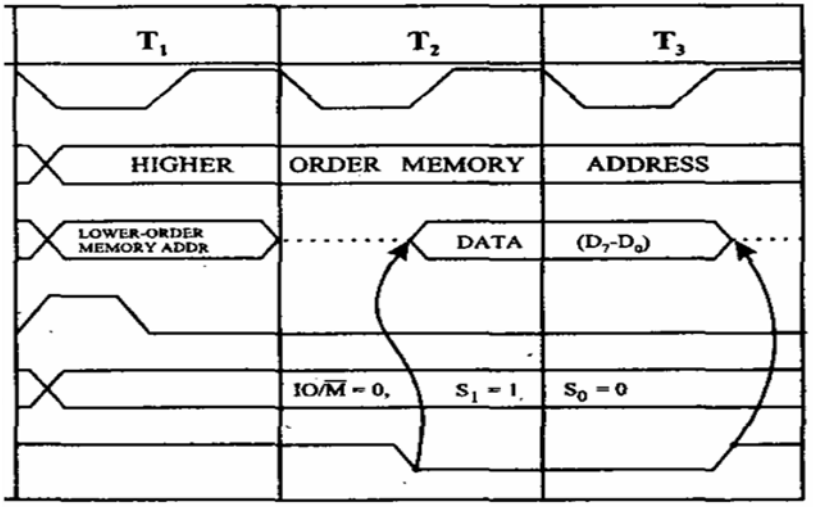
MVI A, 32H

2000 3EH

2001 32H

Here two machine cycles are presented, first is opcode fetch which consists of 4 clock cycles and second is memory read consist of 4 clock cycle.





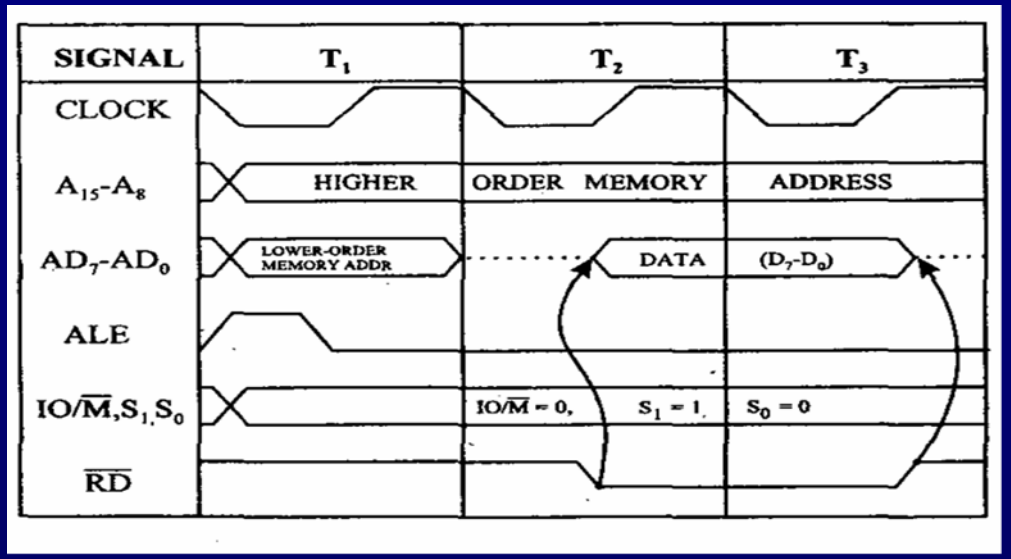
8085 instruction cycle, machine cycle, T states

Memory Read Cycle:

MVI A, 32H

2000 3EH

2001 32H



- Step 1: First machine cycle (Op-Code fetch) is identical for timing diagram of Op-Code fetch cycle.
- Step 2: After completion of Op-Code fetch cycle, 8085 places the address 2001 on the address bus and increments PC to 2002H. ALE is asserted high, IO/M' =0, S1=1, S0=0 for memory read cycle. When RD' =0, memory places the data byte 32H on the dada bus.

8085 instruction cycle, machine cycle, T states

Memory Read Cycle:

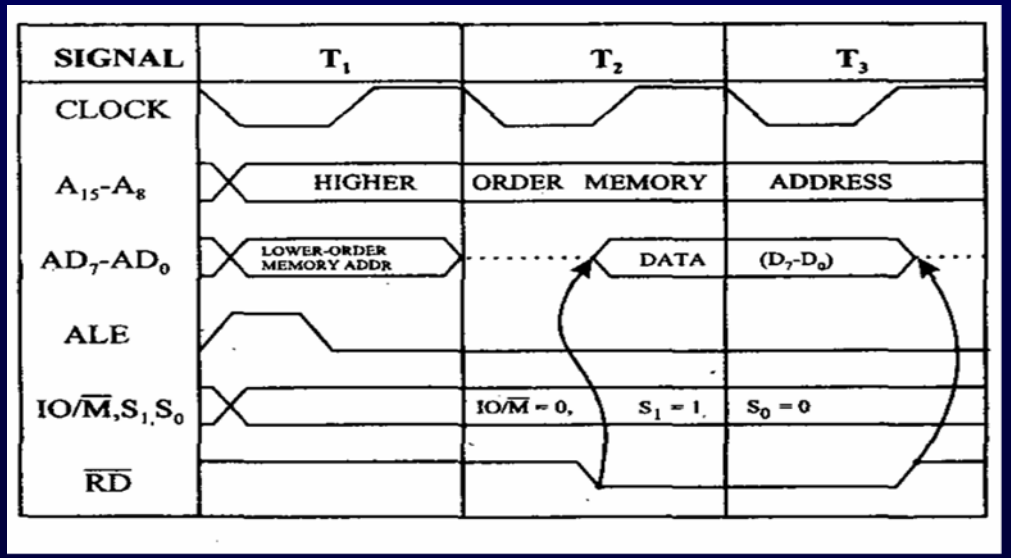
Let's consider the instruction MVI A, 32 H stored at memory location 2000H.

MVI A, 32H

2000 3EH

2001 32H

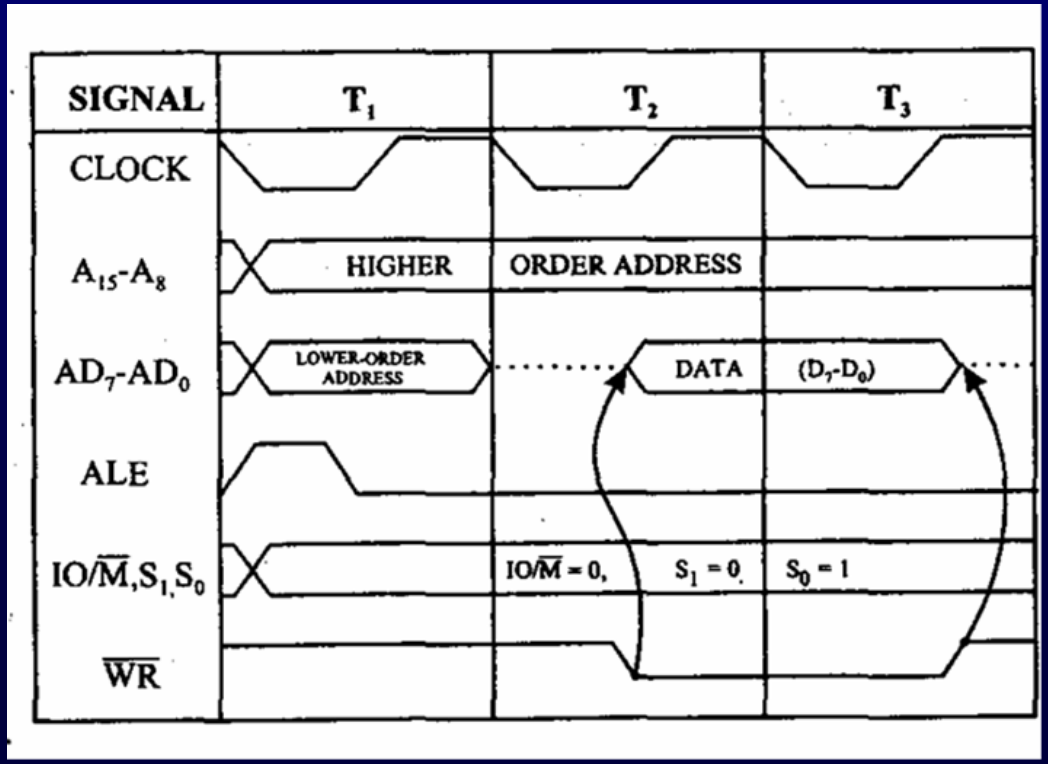
Here two machine cycles are presented, first is opcode fetch which consists of 4 clock cycles and second is memory read consist of 4 clock cycle.



8085 instruction cycle, machine cycle, T states

Memory Write Cycle:

- The memory write machine cycle is executed by the processor to write a data byte in a memory location.
- The processor takes, 3T states to execute this machine cycle.



8085 instruction cycle, machine cycle, T states

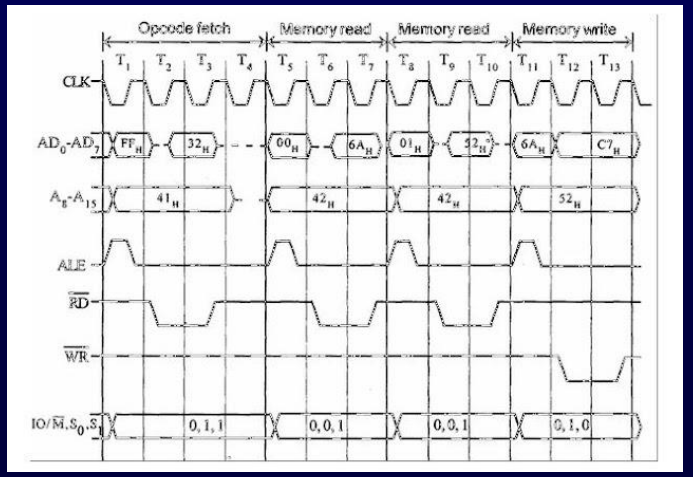
Memory Write Cycle: Timing Diagram for STA 526AH

STA means Store Accumulator -The content of the accumulator is stored in the specified address (526A).

- The op-code of the STA instruction is said to be 32H. It is fetched from the memory 41FFH (see fig). – Opcode Fetch machine cycle
- Then the lower order memory address is read (6A). – Memory Read Machine Cycle
- Read the higher order memory address (52).- Memory Read Machine Cycle
- The combination of both the addresses is considered and the content from accumulator is written in 526A. – Memory Write Machine Cycle

Assume the memory address for the instruction and let the content of accumulator is C7H. So, C7H from accumulator is now stored in 526A

Address	Memories	Hex-code
41FFH	STA	32H
4200H	6AH	6AH
4201H	52H	52H
526AH	C7H	C7H

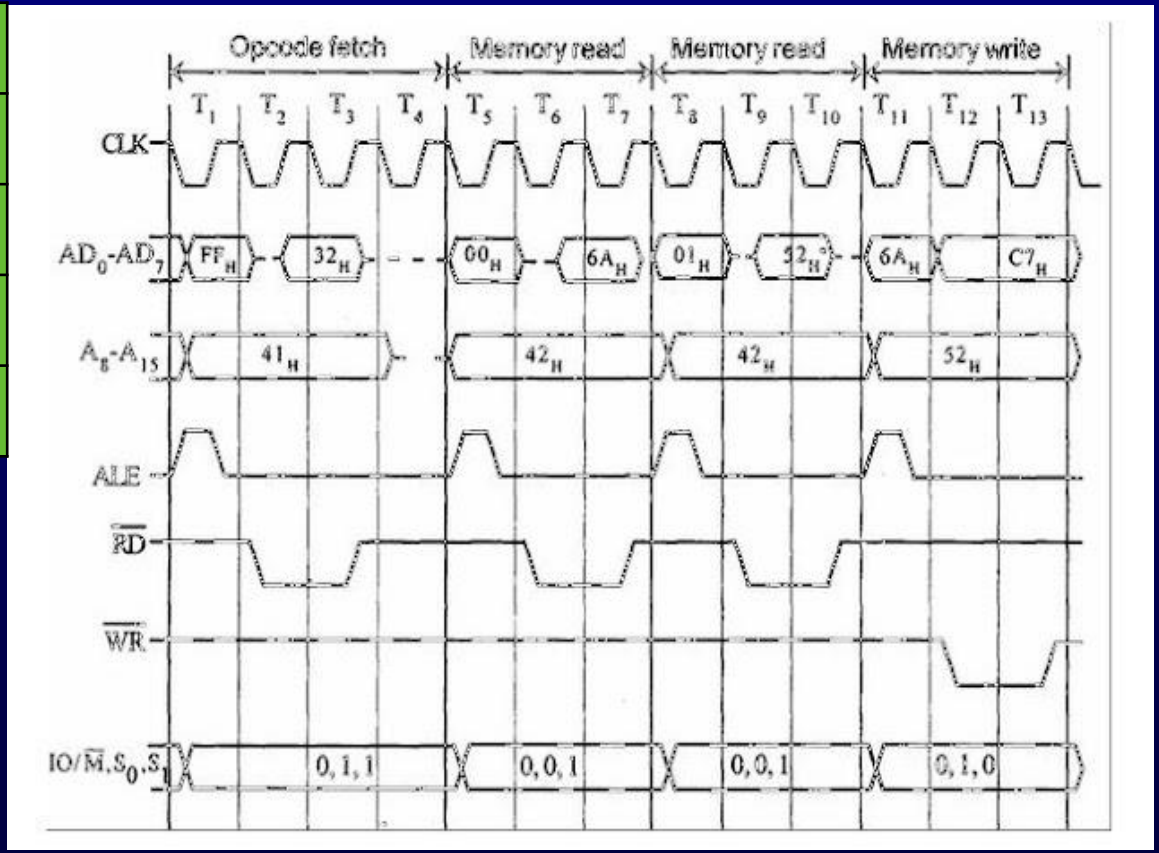


8085 instruction cycle, machine cycle, T states

Memory Write Cycle: Timing Diagram for STA 526AH

STA means Store Accumulator -The content of the accumulator is stored in the specified address (526A).

Address	Memories	Hex-code
41FFH	STA	32H
4200H	6AH	6AH
4201H	52H	52H
526AH	C7H	C7H



Flags:

Flag registers:

1. Sign Flag: It occupies the seventh bit of the flag register, which is also known as the most significant bit. It helps the programmer to know whether the number stored in the accumulator is positive or negative. If the sign flag is set, it means that number stored in the accumulator is negative, and if reset, then the number is positive.

2. Zero Flag: It occupies the sixth bit of the flag register. It is set, when the operation performed in the ALU results in zero(all 8 bits are zero), otherwise it is reset. It helps in determining if two numbers are equal or not.

3. Auxiliary Carry Flag: It occupies the fourth bit of the flag register. In an arithmetic operation, when a carry flag is generated by the third bit and passed on to the fourth bit, then Auxiliary Carry flag is set. If not flag is reset. This flag is used internally for BCD(Binary-Coded decimal Number) operations.

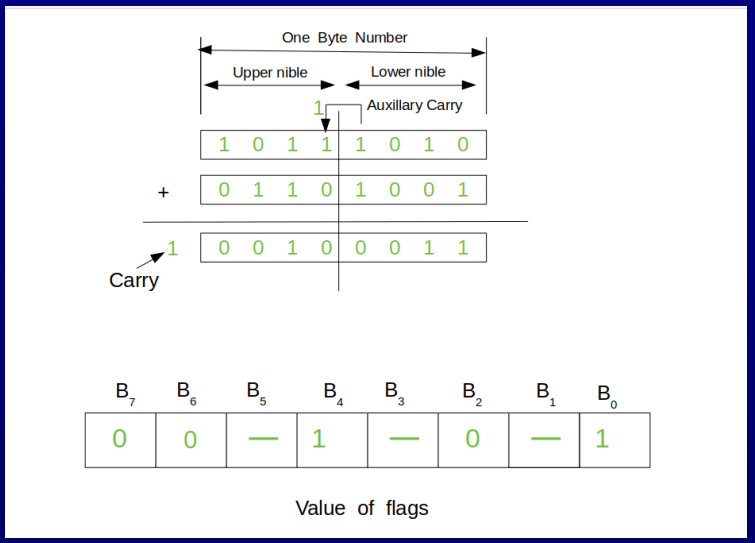
Note – This is the only flag register in 8085 which is not accessible by user.

4. Parity Flag: It occupies the second bit of the flag register. This flag tests for number of 1's in the accumulator. If the accumulator holds even number of 1's, then this flag is set and it is said to even parity. On the other hand if the number of 1's is odd, then it is reset and it is said to be odd parity.

5. Carry Flag: It occupies the zeroth bit of the flag register. If the arithmetic operation results in a carry(if result is more than 8 bit), then Carry Flag is set; otherwise it is reset.

Flags:

Flag registers:



Example –

Here two binary numbers are added. The result produced is stored in the accumulator. Now lets check what each bit means. Refer to the below explanation simultaneously to connect them with the example.

- Sign Flag (7th bit): It is reset(0), which means number stored in the accumulator is positive.
- Zero Flag (6th bit): It is reset(0), thus result of the operations performed in the ALU is non-zero.
- Auxiliary Carry Flag (4th bit): We can see that b3 generates a carry which is taken by b4, thus auxiliary carry flag gets set (1).
- Parity Flag (2nd bit): It is reset(0), it means that parity is odd. The accumulator holds odd number of 1's.
- Carry Flag (0th bit): It is set(1), output results in more than 8 bit.

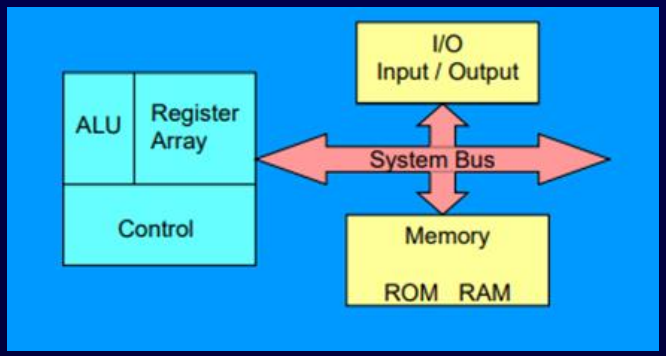
3. Microprocessor architecture and operation

Microprocessor architecture and operation:

2. Internal Data Operations and the 8085/8080A Registers

The internal architecture of the 8085/8080A microprocessor determines how and what operations can be performed with the data. These operations are

- Store 8-bits data.
- Perform arithmetic and logical operations.
- Test for conditions.
- Sequence the execution of instructions.
- Store data temporarily during execution in the defined R/W memory locations called the stack.



3. Microprocessor architecture and operation

Microprocessor architecture and operation:

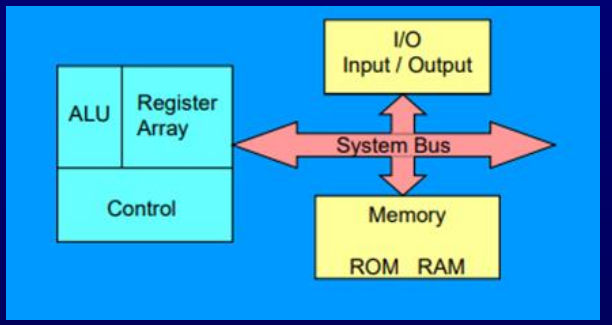
2. Internal Data Operations and the 8085/8080A Registers

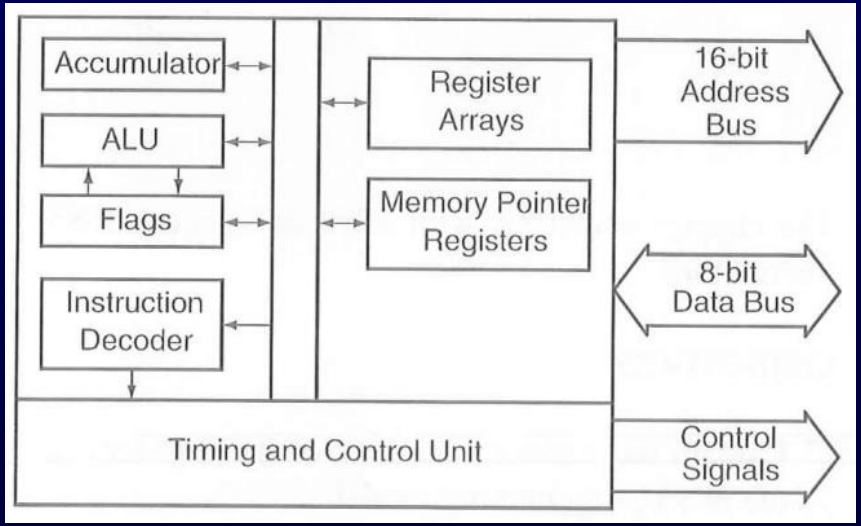
To perform these operations, the microprocessor requires:

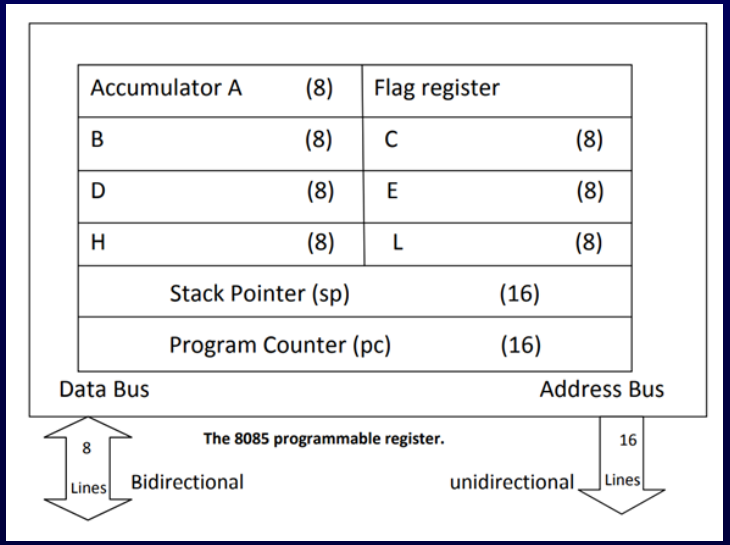
a) Registers.

b) An arithmetic logic unit (ALU) & control logic.

c) Internal buses (paths for information flow).







The 8085 programmable register.

3. Microprocessor architecture and operation

Microprocessor architecture and operation:

2. Internal Data Operations and the 8085/8080A Registers

- **A) Registers:** The 8085 has 6 general purpose registers to perform its operation (store 8-bit data during program execution) these are: (B, C, D, E, H & L) or in pair (BC, DE & HL) to perform 16-bit operations it could be viewed as memory locations.
- **B) Accumulator(A):** 8-bit register that is part of (ALU), this register used to store 8-bit data to perform arithmetic & logic operation, the result of operation is stored in the accumulator.
- **C) Program counters (PC):** This 16-bit register used in sequencing the execution of instructions, this register is memory pointer. The Mp uses this register to sequence the execution of the instruction. The function of the program counter is to point to the memory address for which the next byte is to be fetched.

The 8085 CPU

Memory and Memory Operations:

The instruction code 0100 1111 (4FH) is stored in memory location 2005H. Illustrate the data flow and list the sequence of events when the instruction code is fetched by the MPU.

To fetch the instruction located in memory location 2005H, the following steps are performed:

1. The program counter places the 16-bit address 2005H of the memory location on the address bus (Figure 3.12).
2. The control unit sends the Memory Read control signal ($\overline{\text{MEMR}}$, active low) to enable the output buffer of the memory chip.
3. The instruction (4FH) stored in the memory location is placed on the data bus and transferred (copied) to the instruction decoder of the microprocessor.
4. The instruction is decoded and executed according to the binary pattern of the instruction.

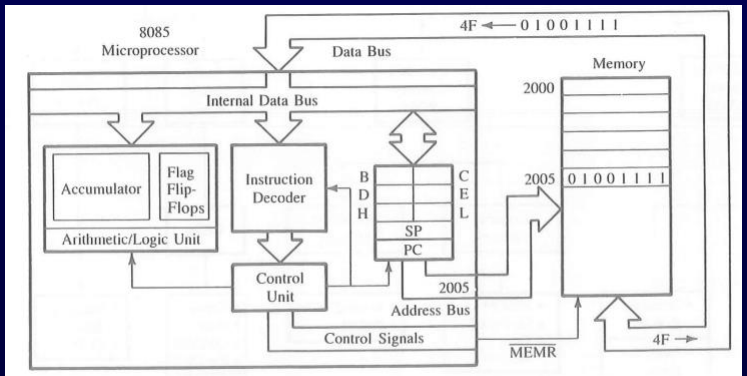


FIGURE 3.12
Instruction Fetch Operation

The 8085 CPU

Memory and Memory Operations:

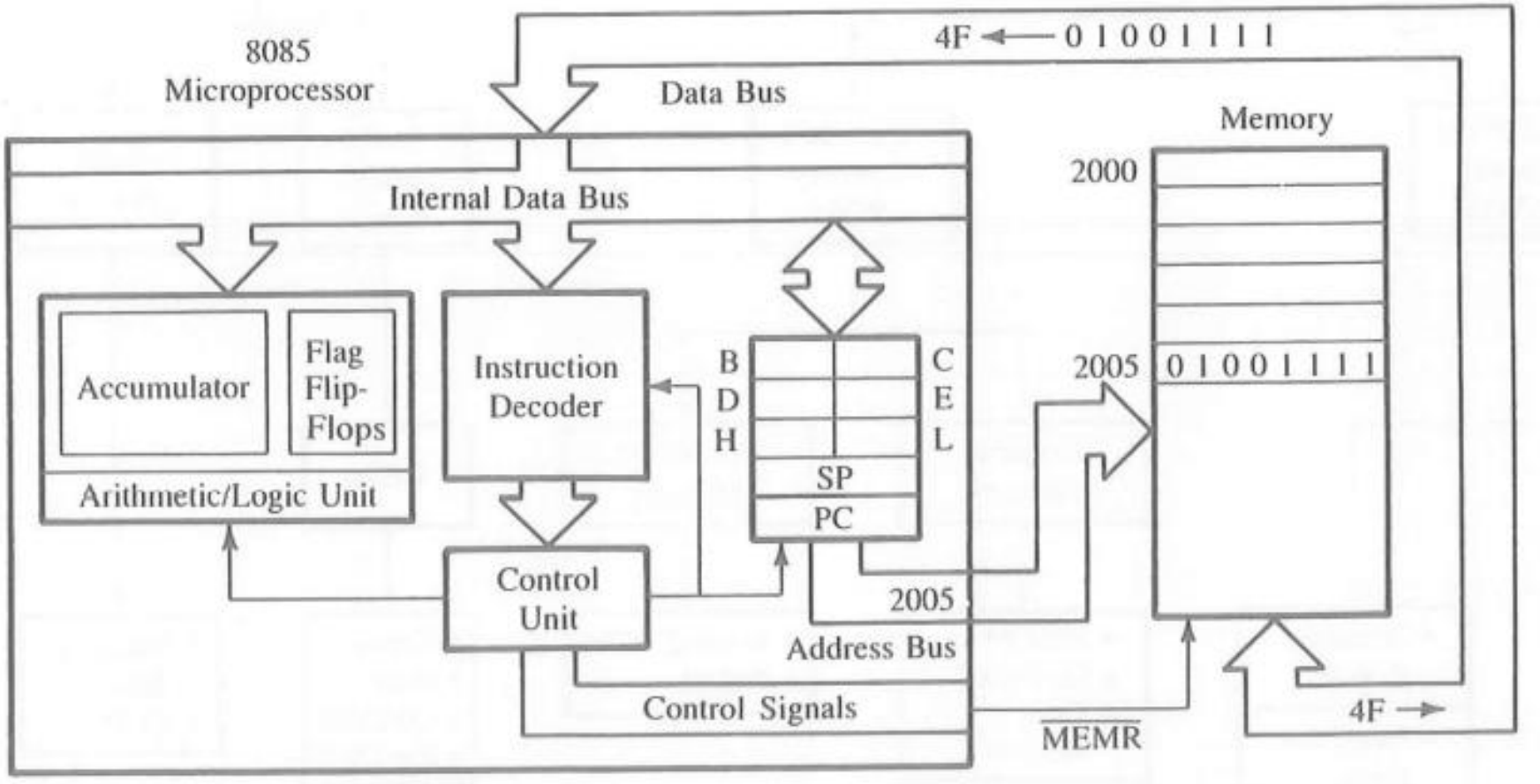


FIGURE 3.12
Instruction Fetch Operation

Basic Assembly Language Programming Using 8085 Instruction Sets:

- Data Transfer: MOV, IN, OUT, STA, LDA, LXI, LDAX, STAX, XCHG
- Arithmetic and Logic:- ADD, SUB, INR, DCR, AND, OR, XOR, ; CMP, RLC, RRC, RAL, RAR ;
- Branching:- JMP, JNZ, JZ, JNC, JC

Basic Assembly Language Programming Using 8085 Instruction Sets:

1. Data transfer instructions:

OPCODE	EXPLANATION	EXAMPLE
MOV	Rd = Rs	MOV A, B
MOV	Rd = Mc	MOV A, M
MOV	M = Rs	MOV M, A
MVI	Rd = 8-bit data	MVI A, 50
MVI	M = 8-bit data	MVI M, 50
LDA	A = contents at address	LDA 2050
STA	contents at address = A	STA 2050
LXI	loads the specified register pair with data	LXI H, 3050
LDAX	indirectly loads at the accumulator A	LDAX H
STAX	indirectly stores from the accumulator A	STAX 2050
XCHG	exchanges H with D, and L with E	XCHG
PUSH	pushes r.p. to the stack	PUSH H
POP	pops the stack to r.p.	POP H
IN	inputs contents of the specified port to A	IN 15
OUT	outputs contents of A to the specified port	OUT 15

Basic Assembly Language Programming Using 8085 Instruction Sets:

2. Arithmetic Operations:

OPCODE	OPERAND	EXPLANATION	EXAMPLE
ADD	R	A = A + R	ADD B
ADD	M	A = A + Mc	ADD M
ADI	8-bit data	A = A + 8-bit data	ADD 50
ADC	R	A = A + R + prev. carry	ADC B
ADC	M	A = A + Mc + prev. carry	ADC M
ACI	8-bit data	A = A + 8-bit data + prev. carry	ACI 50
SUB	R	A = A – R	SUB B
SUB	M	A = A – Mc	SUB M
SUI	8-bit data	A = A – 8-bit data	SUI 50
SBB	R	A = A – R – prev. carry	SBB B
SBB	M	A = A – Mc -prev. carry	SBB M
SBI	8-bit data	A = A – 8-bit data – prev. carry	SBI 50
INR	R	R = R + 1	INR B
INR	M	M = Mc + 1	INR M
INX	r.p.	r.p. = r.p. + 1	INX H
DCR	R	R = R – 1	DCR B
DCR	M	M = Mc – 1	DCR M
DCX	r.p.	r.p. = r.p. – 1	DCX H
DAD	r.p.	HL = HL + r.p.	DAD H

Basic Assembly Language Programming Using 8085 Instruction Sets:

3. Logical instructions:

OPCODE	OPERAND	DESTINATION	EXAMPLE
ANA	R	A = A AND R	ANA B
CMC	none	Compliments the carry flag	CMC
STC	none	Sets the carry flag	STC
ANA	M	A = A AND Mc	ANA 2050
ANI	8-bit data	A = A AND 8-bit data	ANI 50
ORA	R	A = A OR R	ORA B
ORA	M	A = A OR Mc	ORA 2050
ORI	8-bit data	A = A OR 8-bit data	ORI 50
XRA	R	A = A XOR R	XRA B
XRA	M	A = A XOR Mc	XRA 2050
XRI	8-bit data	A = A XOR 8-bit data	XRI 50
CMA	none	A = 1's compliment of A	CMA
CMP	R	Compares R with A and triggers the flag register	CMP B
CMP	M	Compares Mc with A and triggers the flag register	CMP 2050
CPI	8-bit data	Compares 8-bit data with A and triggers the flag register	CPI 50
RRC	none	Rotate accumulator right without carry	RRC
RLC	none	Rotate accumulator left without carry	RLC
RAR	none	Rotate accumulator right with carry	RAR
RAL	none	Rotate accumulator left with carry	RAR

Basic Assembly Language Programming Using 8085 Instruction Sets:

4. Branching instructions:

•Jump Instructions – The jump instruction transfers the program sequence to the memory address given in the operand based on the specified flag. Jump instructions are 2 types: Unconditional Jump Instructions and Conditional Jump Instructions.

•**Unconditional Jump Instructions:** Transfers the program sequence to the described memory address.

•**Conditional Jump Instructions:** Transfers the program sequence to the described memory address only if the condition in satisfied.

OPCODE	OPERAND	EXPLANATION	EXAMPLE
JMP	address	Jumps to the address	JMP 2050
JC	address	Jumps to the address if carry flag is 1	JC 2050
JNC	address	Jumps to the address if carry flag is 0	JNC 2050
JZ	address	Jumps to the address if zero flag is 1	JZ 2050
JNZ	address	Jumps to the address if zero flag is 0	JNZ 2050
JPE	address	Jumps to the address if parity flag is 1	JPE 2050
JPO	address	Jumps to the address if parity flag is 0	JPO 2050
JM	address	Jumps to the address if sign flag is 1	JM 2050
JP	address	Jumps to the address if sign flag 0	JP 2050

Basic Assembly Language Programming Using 8085 Instruction Sets:

5. The Machine Control :

HLT - Halt

NOP - No Operation