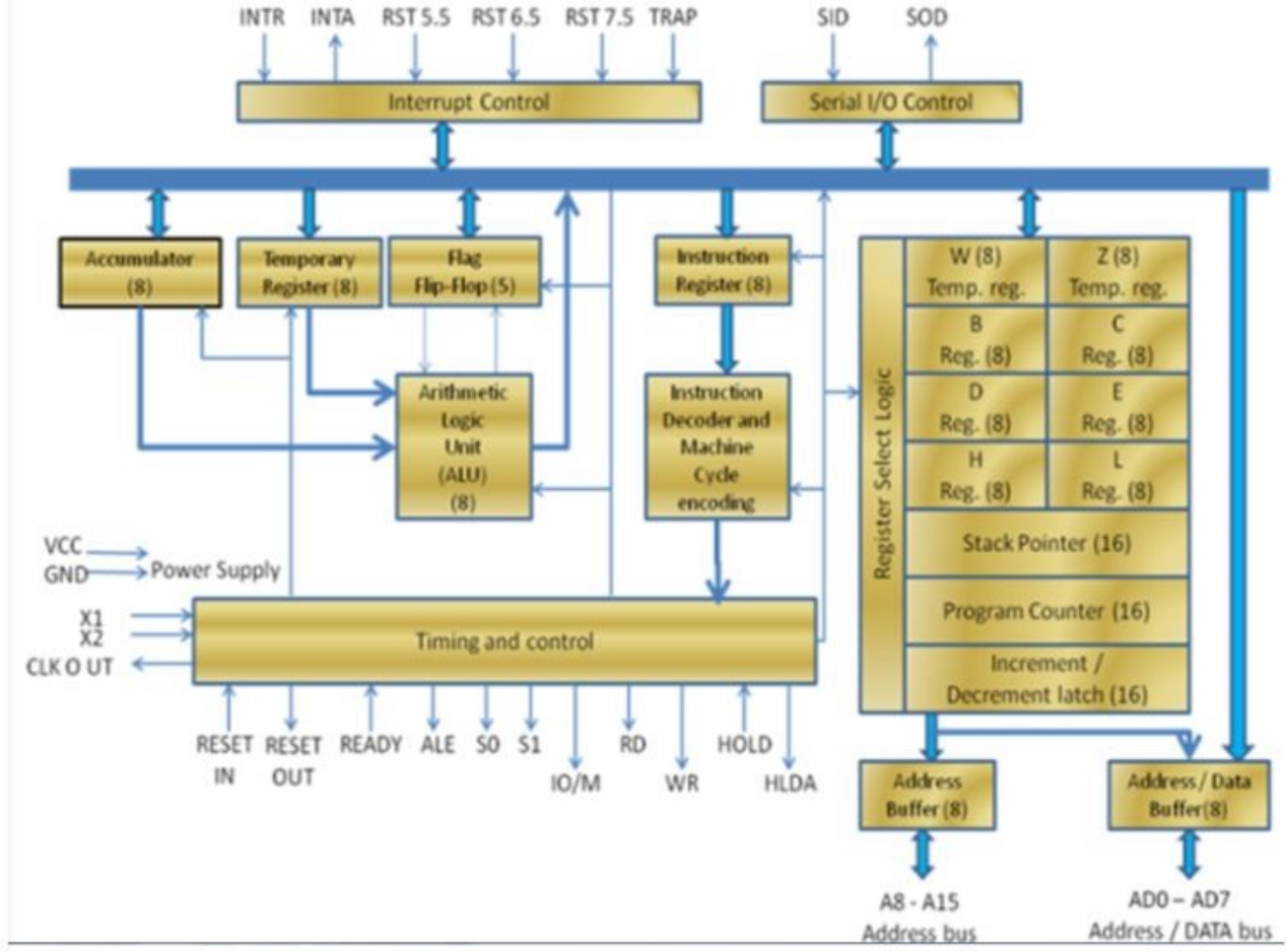
The architecture of 8085 microprocessor is shown below:



ALU

The **Arithmetic and Logic Unit**, ALU performs the arithmetic and logical operations:

- ❍Addition
- ❍Subtraction
- ❍Logical AND
- ❍Logical OR
- ❍Logical EXCLUSIVE OR
- ❍Complement (Logical NOT)
- ❍Increment (add 1)

- ❍Decrement (subtract 1)
- ❍Left shift, Rotate left, Rotate right
- ❍Clear, etc.

Timing and Control Unit

The timing and control unit is the section of the CPU.

- ❍It is used to generate timing and control signals which are necessary for the execution of instructions.
- ❍It is used to control data flow between CPU and peripherals (including memory).
- ❍It is used to provide status, control and timing signals which are required for the operation of memory and I/O devices.
- ❍It is used to control the entire operations of the microprocessor and peripherals connected to it.

Thus we can see that the control unit of the CPU acts as the brain of the computer system.

Registers

Registers are used for temporary storage and manipulation of data and instructions by the microprocessor. Data remain in the registers till they are sent to the I/O devices or memory. Intel 8085 microprocessor has the following registers:

- ❍One 8-bit accumulator (ACC) i.e. register A
- ❍Six general purpose registers of 8-bit, these are B,C, D, E, H and L
- ❍One 16-bit stack pointer, SP
- ❍One 16-bit Program Counter, PC
- ❍Instruction register
- ❍Temporary register

In addition to the above mentioned registers the 8085 microprocessor contains a set of five flip-flops which serve as flags (or status flags).

A flag is a flip-flop which indicates some conditions which arises after the execution of an arithmetic or logical instruction.

1.**Accumulator (ACC):** The accumulator is an 8-bit register associated with the ALU. The register 'A' is an accumulator in the 8085. It is used to hold one of the operands of an arithmetic and logical operation.
The final result of an arithmetic or logical operation is also placed in the accumulator.

2.**General-Purpose Registers:** The 8085 microprocessor contains six 8-bit general purpose registers. They are: B, D, C, E, H and L register.
To hold data of 16-bit a combination of two 8-bit registers can be employed. The combination of two 8-bit registers is called **register pair**. The valid register pairs in the 8085 are: D-E, B-C and H-L. The H-L pair is used to act as a memory pointer.

3.**Program Counter (PC):** It is a 16-bit special purpose register. It is used to hold the address of memory of the next instruction to be executed. It keeps the track of the instruction in a program while they are being executed.
The microprocessor increments the content of the next program counter during the execution of an instruction so that at the end of the execution of an instruction it points to the next instruction?s address in the program.

4.**Stack Pointer (SP):** It is a 16-bit special function register used as memory pointer. A stack is nothing but a portion of RAM. In the stack, the contents of only those registers are saved, which are needed in the later part of the program.
The stack pointer (SP) controls the addressing of the stack. The Stack Pointer contains the address of the top element of data stored in the stack.

5.**Instruction Register:** The instruction register holds the opcode (operation code or instruction code) of the instruction which is being decoded and executed.

6.**Temporary Register:** It is an 8-bit register associated with the ALU. It holds data during an arithmetic/logical operation. It is used by the microprocessor. It is not accessible to programmer.

7.**Flags:** The Intel 8085 microprocessor contains five flip-flops to serve as a status flags. The flip-flops are reset or set according to the conditions which arise during an arithmetic or logical operation.

The five status flags of Intel 8085 are:

- Carry Flag (CS)
- Parity Flag (P)
- Auxiliary Carry Flag (AC)
- Zero Flag(Z)
- Sign Flag(S)

If a flip-flop for a particular flag is set, then it indicates 1. When it is reset, it indicates 0.

Data and Address Bus

- The Intel 8085 is an 8-bit microprocessor. Its **data bus** is 8-bit wide and therefore, 8 bits of data can be transmitted in parallel from or to the microprocessor.
- The Intel 8085 requires an **address bus** of 16-bit wide as the memory addresses are of 16-bits.
- The 8 most significant bits of the address are transmitted by the address bus, A-bus (pins A_8 ? A_{15}).
- The 8 least significant bits of the address are transmitted by data/address bus, AD-bus (pins AD_0 ? AD_7).

Pin Configuration

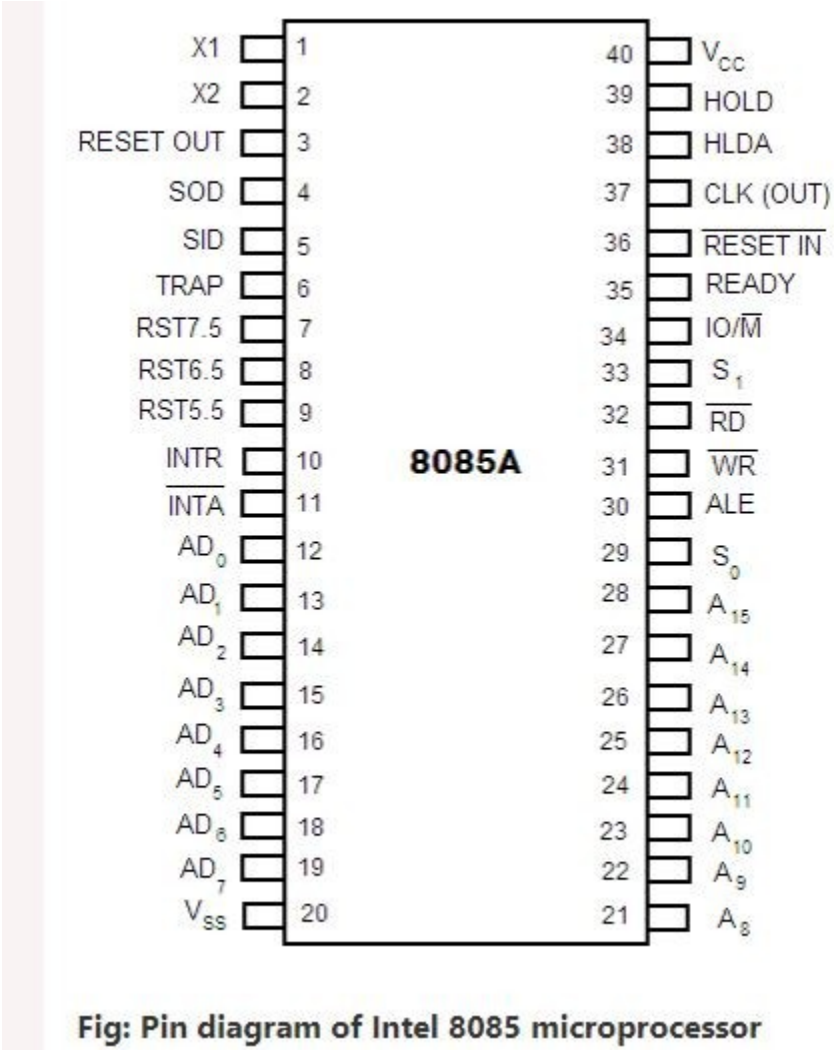


Fig: Pin diagram of Intel 8085 microprocessor

The descriptions of various pins are as follows:

Address Bus and Data Bus

- **A_8 ? A_{15} (Output):** These are **address bus** and are used for the most significant bits of the memory address or 8-bits of I/O address.

- **AD_0 ? AD_7 (Input/output):** These are time multiplexed **address/data bus** i.e. they serve dual purpose. They are used for the least significant 8 bits of the memory address or I/O address during the first cycle. Again they are used for data during 2nd and 3rd clock cycles.

Control and Status Signals

ALE (Output): ALE stands for **Address Latch Enable** signal. ALE goes high during first clock cycle of a machine cycle and enables the lower 8-bits of the address to be latched either into the memory or external latch.

- **IO/M (Output):** It is a **status signal** which distinguishes whether the address is for memory or I/O device.
- **S_0, S_1 (Output):** These are **status signals** sent by the microprocessors to distinguish the various types of operation given in table below:

Status codes for Intel 8085

S_1	S_0	Operations
0	0	HALT
0	1	WRITE
1	0	READ
1	1	FETCH

RD (Output): RD is a **signal to control READ operation**. When it goes low, the selected I/O device or memory is read.

- **WR (Output):** WR is a **signal to control WRITE operation**. When it goes low, the data bus' data is written into the selected memory or I/O location.
- **READY (Input):** It is used by the microprocessor to sense whether a peripheral is ready to transfer a data or not. If READY is high, the peripheral is ready. If it is low the micro processor waits till it goes high.

Interrupts and Externally Initiated Signals

- **HOLD (INPUT):** HOLD indicates that another device is requesting for the use of the address and data bus.

❍**HLDA (OUTPUT):** HLDA is a signal for **HOLD acknowledgement** which indicates that the HOLD request has been received. After the removal of this request the HLDA goes low.

❍**INTR (Input):** INTR is an **Interrupt Request Signal**. Among interrupts it has the lowest priority. The INTR is enabled or disabled by software.

❍**INTA (Output):** INTA is an **interrupt acknowledgement** sent by the microprocessor after INTR is received.

❍**RST 5.5, 6.5, 7.5 and TRAP (Inputs):** These **all are interrupts**. When any interrupt is recognized the next instruction is executed from a fixed location in the memory as given below:

Line	Location from which next instruction is picked up
TRAP	0024
RST 5.5	002C
RST 6.5	0034
RST 7.5	003C

RST 7.5, RST 6.5 and RST 5.5 are the restart interrupts which cause an internal restart to be automatically inserted.

The TRAP has the highest priority among interrupts. The order of priority of interrupts is as follows:

❍TRAP (Highest priority)

❍RST 7.5

❍RST 6.5

❍RST 5.5

❍INTR (Lowest priority).

Reset Signals

❍**RESET IN (Input):** It resets the program counter (PC) to 0. It also resets interrupt enable and HLDA flip-flops. The CPU is held in reset condition till RESET is not applied.

❍**RESET OUT (Output):** RESET OUT indicates that the CPU is being reset.

Clock Signals

❍**X_1, X_2 (Input):** X1 and X2 are terminals to be connected to an external crystal oscillator which drives an internal circuitry of the microprocessor. It is used to produce a suitable clock for the operation of microprocessor.

❍**CLK (Output):** CLK is a **clock output** for user, which can be used for other digital ICs. Its frequency is same at which processor operates.

Serial I/O Signals

❍**SID (Input):** SID is data line for **serial input**. The data on this line is loaded into the seventh bit of the accumulator when RIM instruction is executed.

❍**SOD (Output):** SOD is a data line for **serial output**. The seventh bit of the accumulator is output on SOD line when SIM instruction is executed.

Power Supply

V_{cc} : +5 Vlots supply

V_{ss} : ground reference

8085 Instructions

An **instruction** of computer is a command given to the computer to perform a specified operation on given data. Some instructions of Intel 8085 microprocessor are: MOV, MVI, LDA, STA, ADD, SUB, RAL, INR, MVI, etc.

Opcode and Operands

Each instruction contains two parts: Opcode (Operation code) and Operand.

The 1st part of an instruction which specifies the task to be performed by the computer is called Opcode.

The 2nd part of the instruction is the data to be operated on, and it is called Operand. The Operand (or data) given in the instruction may be in various forms such as 8-bit or 16-bit data, 8-bit or 16-bit address, internal registers or a register or memory location.

Instruction Word Size

A digital computer understands instruction written in binary codes (machine codes). The binary codes of all instructions are not of the same length.

According to the word size, the Intel 8085 instructions are classified into the following three types:

1.One byte instruction

2.Two byte instruction

3.Three byte instruction

1. One-byte instruction: Examples of one byte instructions are:

- **MOV A, B** - Move the content of the register B to register A.
- **ADD B** ? Add the content of register B to the content of the accumulator.

All the above two examples are only one byte long. All one-byte instructions contain information regarding operands in the opcode itself.

2. Two-byte instruction: In a two byte instruction the first byte of the instruction is its opcode and the second byte is either data or address.

Example:

MVI B, 05; 05 moved to register B.

06, 05; MVI B, 05 is in the code form.

The first byte 06 is the opcode for MVI B and second byte 05 is the data which is to be moved to register B.

3. Three-byte instruction: The first byte of the instruction is its opcode and the second and third bytes are either 16-bit data or 16-bit address.

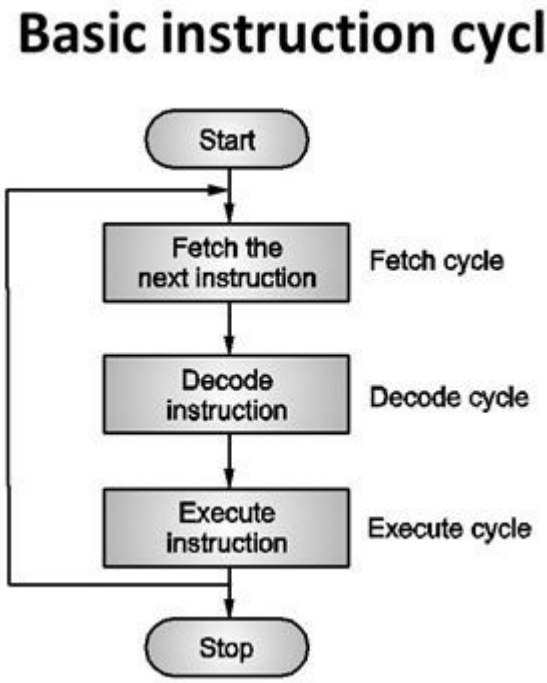
Example:

LXI H, 2400H; Load H-L Pair with 2400H

21, 00, 24; LXI H, 2400H in the code form

The first byte 21 is the opcode for the instruction LXI H. The second 00 is 8 LSBs of the data (2400H), which is loaded into register L. The third byte 24 is 8 MSBs of the data (2400H), which is loaded into register H.

Instruction Cycle

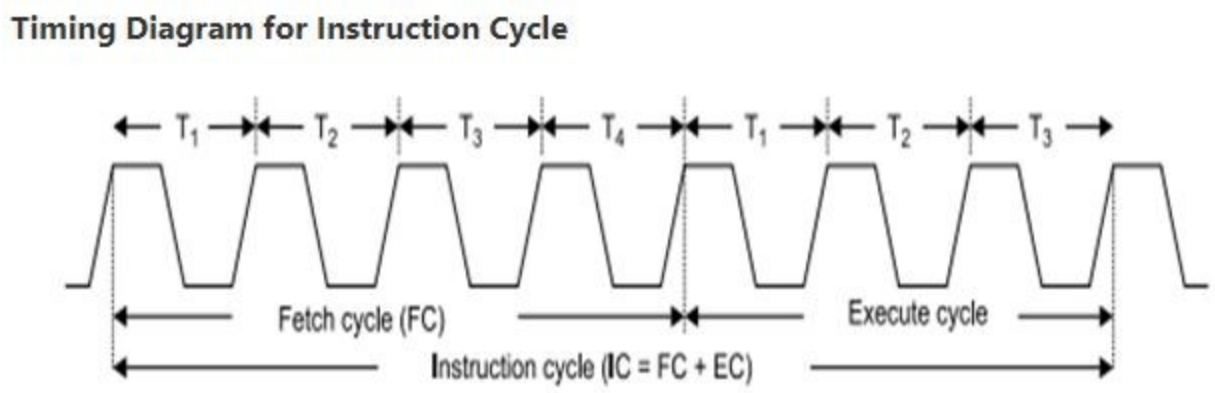


The time required to fetch an instruction and necessary data from memory and to execute it, is called an instruction cycle. Or the total time required to execute an instruction is given by:

```
IC = FC + EC
```

Where,

```
IC = Instruction Cycle
FC = Fetch Cycle
```




```
EC = Execute Cycle
```

Fetch the instruction (Fetch Cycle)

In the beginning of the fetch cycle, the content of the program counter (PC), which is the address of the memory location where opcode is available, is sent to the memory. The memory puts the opcode on the data bus so as to transfer it to the CPU.

The whole operation of fetching an opcode takes three clock cycles. A slow memory may take more time.

- **Decode the instruction (Decode Cycle)**

The opcode fetched from the memory goes to the data register, DR and then to instruction register, IR. From the IR it goes to the decoder circuitry which decodes the instruction. Decoder circuitry is within the microprocessor.

- **Execute the Instruction (Execute Cycle)**

After the instruction is decoded, execution begins.

If the operand is reside the general purpose registers, execution is immediately performed. The time taken in decoding and execution of an instruction is one clock cycle.

In some situations, an execute cycle may involve one or more read or write cycles or both.

Read Cycle: If an instruction contains data or operand address which are in the memory, the CPU has to perform some read operations to get the desired data. In case of a read cycle the instruction received from the memory are data or operand address instead of an opcode.

Write Cycle: In write cycle data are sent from the CPU to the memory or an output device.

- **Machine Cycle and State**

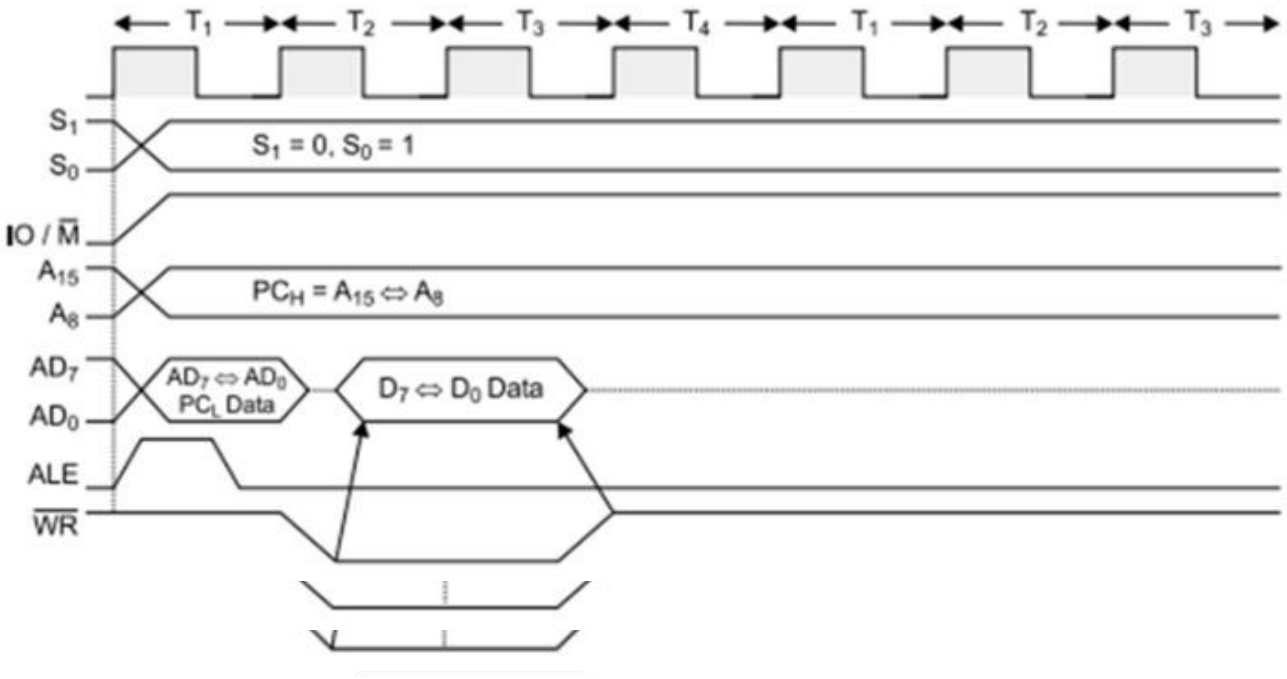
The necessary steps carried out to perform the operation of accessing either memory or input output device, constitute a **machine cycle**. In other words, necessary steps carried out to perform a fetch, a read or a write operation constitutes a machine cycle.

One sub-division of an operation performed in one clock cycle is called a state or **T-state**. In short, one clock cycle of the system clock is referred to as a state.

Timing Diagram

The necessary steps carried out in a machine cycle can be represented graphically. Such a graphical representation is called **timing diagram**. The timing diagram for opcode fetch, memory read, memory write, I/O read and I/O write will be discussed below:

- **Timing Diagram for I/O Write**



In the above diagrams, the basic used parameters are:

ALE: ALE indicates the availability of a valid address on the multiplexed address/data lines. When it is high or 1, then it acts as an address bus and when low or 0, then it acts as a data bus.

RD (low active): If it is high or 1, then no data is read by the microprocessor. If signal is low or 0, then data is read by the microprocessor.

WR (low active): If it is high or 1, then no data is written by the microprocessor. If signal is low or 0, then data is written by the microprocessor.

IO/M (low active): A high or 1 on this signal indicates I/O operation while a low or 0 indicates memory operation.

S_0, S_1: S0 and S1 Indicate the type of machine cycle in progress.

The below table, shows the status of different control signal for different operation:

Machine cycle	Status			Controls		
	IO / $\overline{M}$	S_1	S_0	$\overline{RD}$	$\overline{WR}$	$\overline{INTA}$
Opcode Fetch (OF)	0	1	1	0	1	1
Memory Read	0	1	0	0	1	1
Memory Write	0	0	1	1	0	1
I/O Read (I/OR)	1	1	0	0	1	1
I/O Write (I/OW)	1	0	1	1	0	1

Instruction Set of 8085

Instruction and Data Formats

The various techniques to specify data for instructions are:

1.8-bit or 16-bit data may be directly given in the instruction itself.

2.The address of the memory location, I/O port or I/O device, where data resides, may be given in the instruction itself.

3.In some instructions, only one register is specified. The content of the specified register is one of the operands.

4.Some instructions specify two registers. The contents of the registers are the required data.

5.In some instructions, data is implied. The most instructions of this type operate on the content of the accumulator.

Due to different ways of specifying data for instructions, the machine codes of all instructions are not of the same length. It may 1-byte, 2-byte or 3-byte instruction.

Addressing Modes

Each instruction requires some data on which it has to operate. There are different techniques to specify data for instructions. These techniques are called **addressing modes**. Intel 8085 uses the following addressing modes:

- **Direct Addressing**

In this addressing mode, the address of the operand (data) is given in the instruction itself.

Example

STA 2400H: It stores the content of the accumulator in the memory location 2400H.

32, 00, 24: The above instruction in the code form.

In this instruction, 2400H is the memory address where data is to be stored. It is given in the instruction itself. The 2nd and 3rd bytes of the instruction specify the address of the memory location. Here, it is understood that the source of the data is accumulator.

- **Register Addressing**

In register addressing mode, the operand is in one of the general purpose registers. The opcode specifies the address of the register(s) in addition to the operation to be performed.

Example:

MOV A, B: Move the content of B register to register A.

78: The instruction in the code form.

In the above example, MOV A, B is 78H. Besides the operation to be performed the opcode also specifies source and destination registers.

The opcode 78H can be written in binary form as 01111000. The first two bits, i.e. 0 1 are for MOV operation, the next three bits 1 1 1 are the binary code for register A, and the last three bits 000 are the binary code for register B.

- **Register Indirect Addressing**

In Register Indirect mode of addressing, the address of the operand is specified by a register pair.

Example

- **LXI H, 2500 H** - Load H-L pair with 2500H.
- **MOV A, M** - Move the content of the memory location, whose address is in H-L pair (i.e. 2500 H) to the accumulator.
- **HLT** - Halt.

In the above program the instruction MOV A, M is an example of register indirect addressing. For this instruction, the operand is in the memory. The address of the memory is not directly

given in the instruction. The address of the memory resides in H-L pair and this has already been specified by an earlier instruction in the program, i.e. LXI H, 2500 H.

- **Immediate Addressing**

In this addressing mode, the operand is specified within the instruction itself.

Example

LXI H, 2500 is an example of immediate addressing. 2500 is 16-bit data which is given in the instruction itself. It is to be loaded into H-L pair.

Implicit Addressing

There are certain instructions which operate on the content of the accumulator. Such instructions do not require the address of the operand.

Example

CMA, RAL, RAR, etc.

Status Flags

There is a set of five flip-flops which indicate status (condition) arising after the execution of arithmetic and logic instructions. These are:

- Carry Flag (CS)
- Parity Flag (P)
- Auxiliary Carry Flags (AC)
- Zero Flags (Z)
- Sign Flags (S)

Symbols and Abbreviations

The symbol and abbreviations which have been used while explaining Intel 8085 instructions are as follows:

H	Appearing at the end of the group of digits specifies hexadecimal, e.g. 2500H
Rp	One of the register pairs.
Rh	The high order register of a register pair
Rl	The low order register of a register pair
PC	16 bit program counter, PCH is high order 8 bits and PCL low order 8 bits of register PC.
CS	Carry Status
[]	The contents of the register identified within bracket
[[]]	The content of the memory location whose address is in the register pair identified within brackets
^	AND operation
v	OR operation

⊕ or ∀	Exclusive OR
←	Move data in the direction of arrow
⇔	Exchange contents

Intel 8085 Instructions

An **instruction** of a computer is a command given to the computer to perform a specified operation on given data. In microprocessor, the **instruction** set is the collection of the instructions that the microprocessor is designed to execute.

The programmer writes a program in assembly language using these instructions. These instructions have been classified into the following groups:

Data Transfer Group

Instructions which are used to transfer the data from a register to another register from memory to register or register to memory come under this group.

Instruction Set	Explanation	States	Flags	Addre-ssing	Machine Cycles	Example
MOV r_1, r_2 [r1] ← [r2]	Move the content of the one register to another	4	none	Register	1	MOV A, B
MOV r, M [r]←[[H-L]]	Move the content of memory to register	7	none	Register Indirect	2	MOV B, M
MOV M, r [[H-L]]←[r]	Move the content of register to memory	7	none	Register Indirect	2	MOV M, C
MVI r, data [r] ←data	Move immediate data to register	7	None	Immediate Register	3	MVI M, 08
LXI rp, data 16 [rp] ←data 16 bits, [rh] ←8 MSBs, [rl]	Load Register pair immediate	10	None	Immediate	3	LXI H, 2500H

LDA addr [A] ←[addr]	Load Accumulator direct	13	None	Direct	4	LDA 2400 H
STA Addr [addr] ←[A]	Store accumulator direct	13	None	Direct	4	STA 2000H
LHLD addr [L] ←[addr], [H] ← [addr + 1]	Load H-L pair direct	16	None	Direct	5	LHLD 2500H
SHLD addr [addr] ←[L], [addr +1] ← [H]	Store H-L pair direct	16	None	Direct	5	SHLD 2500 H
LDAX rp [A] ←[[rp]]	Load accumulator indirect	7	None	Register Indirect	2	LDAX B
STAX rp [[rp]] ←[A]	Store accumulator indirect	7	None	Register Indirect	2	STAX D
XCHG [H-L] ↔[D-E]	Change the contents of H-L	4	None	Register	1	

Arithmetic Group

The instructions of this group perform arithmetic operations such as addition, subtraction, increment or decrement of the content of a register or a memory.

SBI data [A] ←[A]-data-[CS]	Subtract immediate data from accumulator with borrow	7	All	Immediate	2	XCHG
INR r [r] ←[r]+1	Increment register content	4	All except carry flag	Register	1	INR K
INR M [[H-L]] ←[[H-L]]+1	Increment memory content	10	All except carry flag	Register indirect	3	INR K
DCR r [r] ←[r] -1	Decrement register content	4	All except carry flag	Register	1	DCR K
DCR M [[H-L]] ← [[H-L]]-1	Decrement memory content	10	All except carry flag	Register indirect	3	DCR K
INX rp [rp] ←[rp]+1	Increment memory content	6	None	Register	1	INX K
DCX rp [rp] ←[rp]-1	Decrement register pair	6	None	Register	1	DCX K
DAA	Decimal adjust accumulator	4			1	DAA

Logical Group

The instructions in this group perform logical operation such as AND, OR, compare, rotate, etc.

XRA r [A] ← [A]∀[r]	XOR register with accumulator	4	All	Register	1
XRA M [A] ← [A] ∀ [[H-L]]	XOR memory with accumulator	7	All	Register indirect	2
XRI data [A] ←[A] ∀ [data]	XOR immediate data with accumulator	7	All	Immediate	2
CMA [A] ←[A]	Complement the accumulator	4	None	Implicit	1
CMC [CS] ←[CS]	Complement the carry status	4	CS		1
STC [CS] ← 1	Set carry status	4	CS		1
RAR $[A^n]$ ←$[A^{n+1}]$, [CS] ←$[A^0]$, $[A^7]$ ←[CS]	Rotate accumulator right through carry		CS	Implicit	1

Branch Control Group

This group contains the instructions for conditional and unconditional jump, subroutine call and return, and restart.

Unconditional Jump

Instruction Set	Explanation	States	Flags	Addressing	Machine Cycles
JMP addr(label) [PC] ← Label	Unconditional jump: jump to the instruction specified by the address	10	None	Immediate	3

Unconditional CALL

Instruction Set	Explanation	States	Flags	Addressing	Machine Cycles
CALL addr (label) [SP]-1] ← [PCH] ,[[SP-2] ← [PCL], [SP] ← [SP]-2, [PC] ← addr(label)	Unconditional CALL: Call the subroutine identified by the address	18	None	Immediate /register	5

Conditional CALL

Instruction Set	Explanation	States	Machine Cycles
CALL addr (label) [SP]-1] ← [PCH] , [[SP-2] ← [PCL], [PC] ← addr (label), [SP] ← [SP]-2	Unconditional CALL: Call the subroutine identified by the address if the specified condition is fulfilled	18, if true and 9, if not true	5, if true and 2, if not true
JPO addr (label)	Jump if odd	The parity	7/10	None	Immediate	2/3

Unconditional CALL

Instruction Set	Explanation	States	Flags	Addressing	Machine Cycles
CALL addr (label) [SP]-1] ← [PCH] ,[[SP-2] ← [PCL], [SP] ← [SP]-2, [PC] ← addr(label)	Unconditional CALL: Call the subroutine identified by the address	18	None	Immediate /register	5

Conditional CALL

Instruction Set	Explanation	States	Machine Cycles
CALL addr (label) [SP]-1] ← [PCH] , [[SP-2] ← [PCL], [PC] ← addr (label), [SP] ← [SP]-2	Unconditional CALL: Call the subroutine identified by the address if the specified condition is fulfilled	18, if true and 9, if not true	5, if true and 2, if not true

CPE addr(label)	Call subroutine if even parity	Parity Status P=1	9/18	None	Immediate /register	2/5
CPO addr(label)	Call subroutine if odd parity	Parity Status P= 0	9/18	None	Immediate /register	2/5

Unconditional Return

Instruction Set	Explanation	States	Flags	Addressing	Machine Cycles
RET [PCL] ← [[SP]], [PCH] ← [[SP] + 1], [SP] ← [SP] + 2	Unconditional RET: Return from subroutine	10	None	Indirect	3

(label)	result is plus	status S=0			/register	
CM addr (label)	Call Subroutine if the result is minus	Sign status S= 1	9/18	None	Immediate /register	2/5

CPE addr(label)	Call subroutine if even parity	Parity Status P=1	9/18	None	Immediate /register	2/5
CPO addr(label)	Call subroutine if odd parity	Parity Status P= 0	9/18	None	Immediate /register	2/5

Unconditional Return

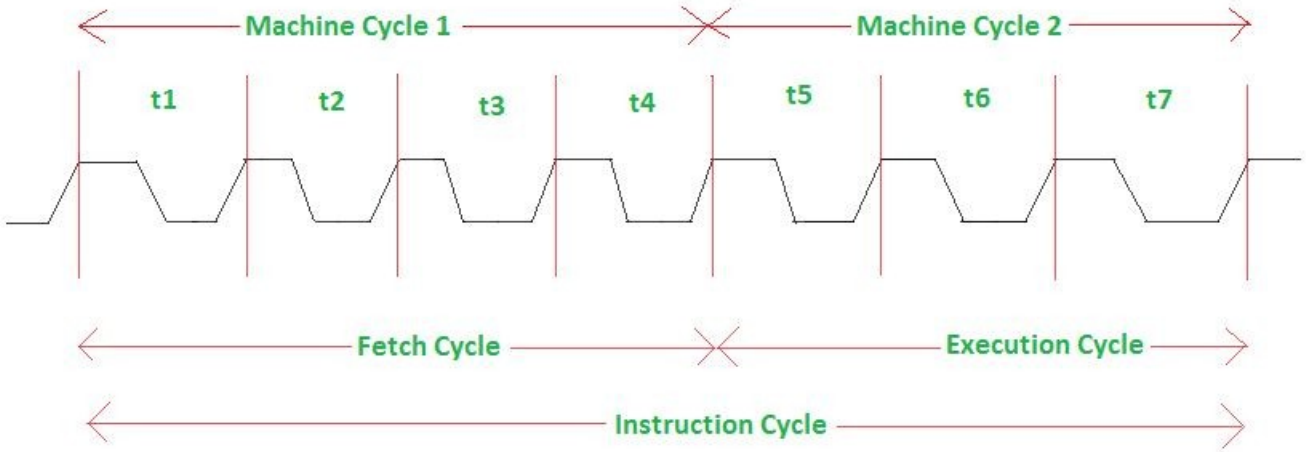
Instruction Set	Explanation	States	Flags	Addressing	Machine Cycles
RET [PCL] ← [[SP]], [PCH] ← [[SP] + 1], [SP] ← [SP] + 2	Unconditional RET: Return from subroutine	10	None	Indirect	3

Instruction cycle in 8085 microprocessor

Time required to execute and fetch an entire instruction is called instruction cycle. It consists:

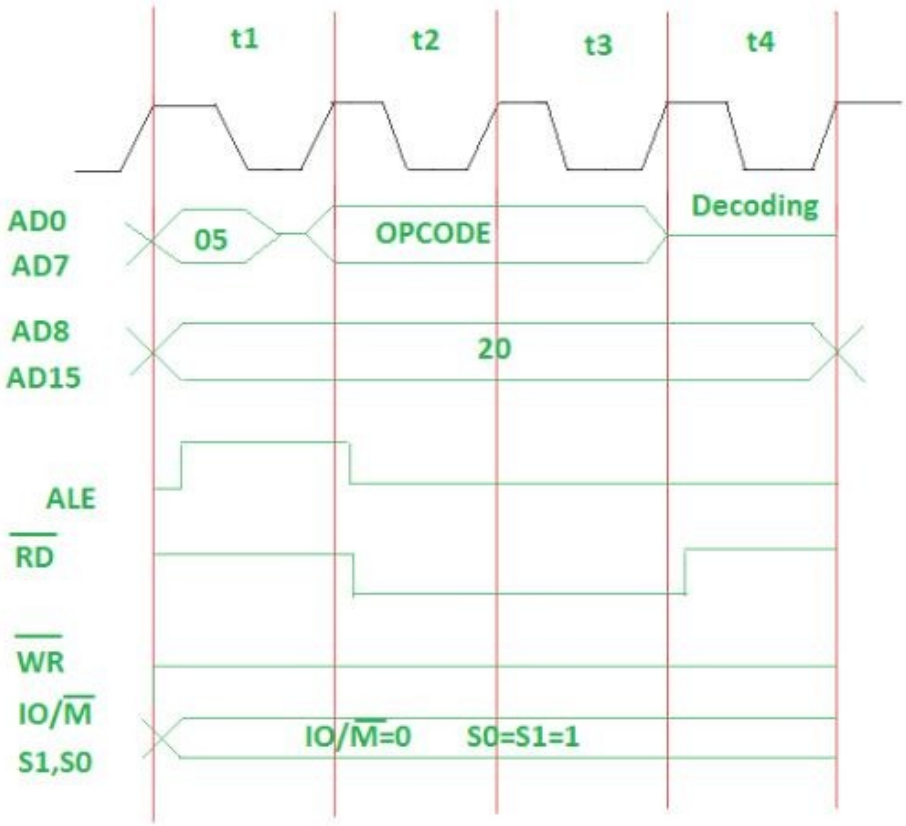
- **Fetch cycle –** The next instruction is fetched by the address stored in program counter (PC) and then stored in the instruction register.
- **Decode instruction –** Decoder interprets the encoded instruction from instruction register.
- **Reading effective address –** The address given in instruction is read from main memory and required data is fetched. The effective address depends on direct addressing mode or indirect addressing mode.
- **Execution cycle –** consists memory read (MR), memory write (MW), input output read (IOR) and input output write (IOW)

The time required by the microprocessor to complete an operation of accessing memory or input/output devices is called machine cycle. One time period of frequency of microprocessor is called t-state. A t-state is measured from the falling edge of one clock pulse to the falling edge of the next clock pulse.
Fetch cycle takes four t-states and execution cycle takes three t-states.



Instruction cycle in 8085 microprocessor

Timing diagram for fetch cycle or opcode fetch:



Timing diagram for opcode fetch

Above diagram represents:

- **05 –** lower bit of address where opcode is stored. Multiplexed address and data bus AD0-AD7 are used.
- **20 –** higher bit of address where opcode is stored. Multiplexed address and data bus AD8-AD15 are used.
- **ALE –** Provides signal for multiplexed address and data bus. If signal is high or 1, multiplexed address and data bus will be used as address bus. To fetch lower bit of address, signal is 1 so that multiplexed bus can act as address bus. If signal is low or 0, multiplexed bus will be used as data bus. When lower bit of address is fetched then it will act as data bus as the signal is low.
- **RD (low active) –** If signal is high or 1, no data is read by microprocessor. If signal is low or 0, data is read by microprocessor.

•**WR (low active) –** If signal is high or 1, no data is written by microprocessor. If signal is low or 0, data is written by microprocessor.

•**IO/M (low active) and S1, S0 –** If signal is high or 1, operation is performing on input output. If signal is low or 0, operation is performing on memory.

Machine Cycle	Status			Control Signals		
	$IO/\overline{M}$	S1	S0	$\overline{RD}$	$\overline{WR}$	$\overline{INTA}$
Opcode Fetch	0	1	1	0	1	1
Memory Read	0	1	0	0	1	1
Memory Write	0	0	1	1	0	1
I/0 Read	1	1	0	0	1	1
I/O Write	1	0	1	1	0	1
Interrupt Acknowledge	1	1	1	1	1	0
HALT	Z	0	0	Z	Z	1
HOLD	Z	X	X	Z	Z	1
RESET	Z	X	X	Z	Z	1

Where Z is tri state (pin neither connected to supply nor ground. High impedance) and X represents do not care.

8085 machine cycle status and control signals