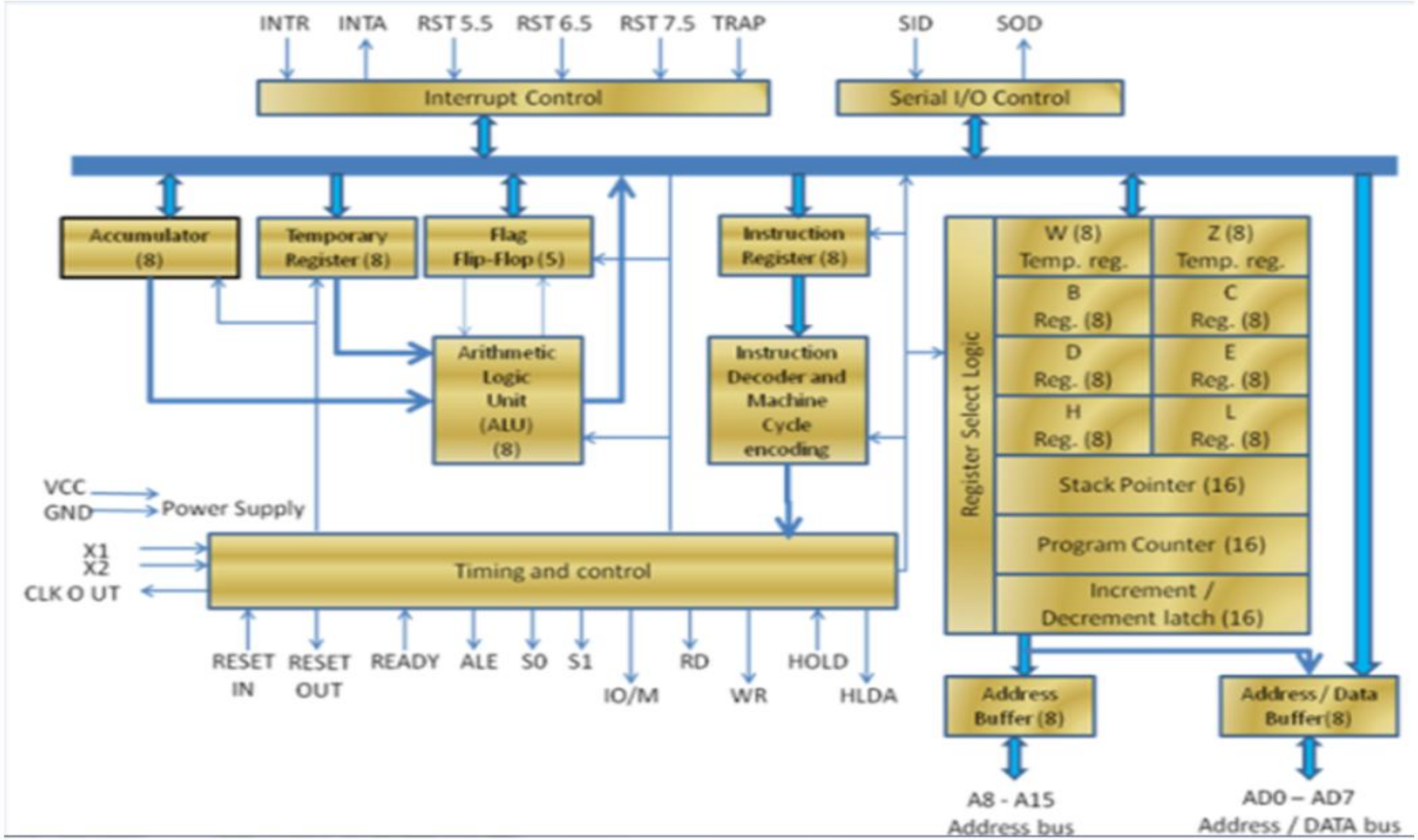
Microprocessor Architecture:

What is the 8085 Microprocessor?

- The 8085 is an 8-bit microprocessor, and it was launched by the Intel team in the year of 1976 with the help of NMOS technology.
- This processor is the updated version of the microprocessor. The configurations of 8085 microprocessor mainly include data bus-8-bit, address bus-16 bit, program counter-16-bit, stack pointer-16 bit, registers 8-bit, +5V voltage supply, and operates at 3.2 MHz single segment CLK.
- The applications of 8085 microprocessor are involved in microwave ovens, washing machines, gadgets, etc.

8085 Architecture



8085 is an 8-bit, general purpose microprocessor. It consists of following functional units:-

1. **Arithmetic and Logic Unit (ALU) :**
   It is used to perform mathematical operations like: addition, multiplication, subtraction, division, decrement, increment, etc.

2. **Flag Register :**
   It is an 8-bit register that stores either 0 or 1 depending upon which value is stored in the accumulator.

3. **Accumulator :**
   Accumulator is used to perform I/O, arithmetic and logical operations. It is connected to ALU and internal data bus.

4. **General Purpose Registers :**
   There are 6 general purpose registers. These registers can hold 8 bit values. These 8-bit registers are B,C,D,E,H,L. These registers work as 16-bit registers when they work in pair like: B-C, D-E, H-L.

8085 is an 8-bit, general purpose microprocessor. It consists of following functional units:-

5. Program counter: It is a 16-bit register used to store the memory address location of the next instruction to be executed. Microprocessor increments the program whenever an instruction is being executed, so that the program counter points to the memory address of the next instruction that is going to be executed.

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

7. Temporary register: It is an 8-bit register, which holds the temporary data of arithmetic and logical operations.

8. Flag register : It is an 8-bit register having five 1-bit flip-flops, which holds either 0 or 1 depending upon the result stored in the accumulator.

8085 is an 8-bit, general purpose microprocessor. It consists of following functional units:-

9. Instruction register and decoder:

- It is an 8-bit register. When an instruction is fetched from memory then it is stored in the Instruction register. Instruction decoder decodes the information present in the Instruction register.

10.Timing and control unit:

- It provides timing and control signal to the microprocessor to perform operations. Following are the timing and control signals, which control external and internal circuits −
- Control Signals: READY, RD’, WR’, ALE
- Status Signals: S0, S1, IO/M’
- DMA Signals: HOLD, HLDA
- RESET Signals: RESET IN, RESET OUT

8085 is an 8-bit, general purpose microprocessor. It consists of following functional units:-

11.Interrupt control

- As the name suggests it controls the interrupts during a process. 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 in 8085 microprocessor: INTR, RST 7.5, RST 6.5, RST 5.5, TRAP.

12. Serial Input/output control

- It controls the serial data communication by using these two instructions: SID (Serial input data) and SOD (Serial output data).

8085 is an 8-bit, general purpose microprocessor. It consists of following functional units:-

14. Address buffer and address-data buffer:

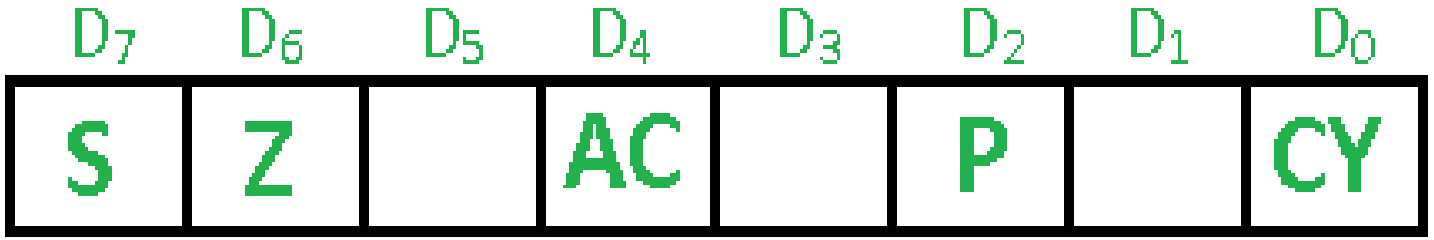
- The content stored in the stack pointer and program counter is loaded into the address buffer and address-data buffer to communicate with the CPU. The memory and I/O chips are connected to these buses; the CPU can exchange the desired data with the memory and I/O chips.

15. Address bus and data bus:

- Data bus carries the data to be stored. It is bidirectional, whereas address bus carries the location to where it should be stored and it is unidirectional. It is used to transfer the data & Address I/O devices.

8085 Flag Register

- The **Flag register** is a Special Purpose Register. Depending upon the value of result after any arithmetic and logical operation the flag bits become set (1) or reset (0). In 8085 microprocessor, flag register consists of 8 bits and only 5 of them are useful.
- The 5 flags are:



8085 Flag Register

Sign Flag (S) – After any operation if the MSB (B(7)) of the result is 1, it in indicates the number is negative and the sign flag becomes set, i.e. 1. If the MSB is 0, it indicates the number is positive and the sign flag becomes reset i.e. 0.

- **1- MSB is 1 (negative)**
- **0- MSB is 0 (positive)**

- **Example:**

    MVI A 30 (load 30H in register A)

    MVI B 40 (load 40H in register B)

    SUB B (A = A – B)

These set of instructions will set the sign flag to 1 as 30 – 40 is a negative number.

8085 Flag Register

Zero Flag (Z) – After any arithmetical or logical operation if the result is 0 (00)H, the zero flag becomes set i.e. 1, otherwise it becomes reset i.e. 0.

00H zero flag is 1.

from 01H to FFH zero flag is 0

1- zero result

0- non-zero result

Example:

MVI A 10 (load 10H in register A)

SUB A (A = A – A)

These set of instructions will set the zero flag to 1 as 10H – 10H is 00H

8085 Flag Register

Auxiliary Cary Flag (AC) – This flag is used in BCD number system(0-9). If after any arithmetic or logical operation D(3) generates any carry and passes on to B(4) this flag becomes set i.e. 1, otherwise it becomes reset i.e. 0. This is the only flag register which is not accessible by the programmer

- **1-carry out from bit 3 on addition or borrow into bit 3 on subtraction 0-otherwise**

Example:

MOV A 2B (load 2BH in register A)

MOV B 39 (load 39H in register B)

ADD B (A = A + B)

These set of instructions will set the auxiliary carry flag to 1, as on adding 2B and 39, addition of lower order nibbles B and 9 will generate a carry.

8085 Flag Register

Parity Flag (P) – If after any arithmetic or logical operation the result has even parity, an even number of 1 bits, the parity register becomes set i.e. 1, otherwise it becomes reset i.e. 0.

1-accumulator has even number of 1 bits

0-accumulator has odd parity

Example:

MVI A 05 (load 05H in register A)

This instruction will set the parity flag to 1 as the BCD code of 05H is 00000101, which contains even number of ones i.e. 2.

8085 Flag Register

Carry Flag (CY) – Carry is generated when performing n bit operations and the result is more than n bits, then this flag becomes set i.e. 1, otherwise it becomes reset i.e. 0.

During subtraction (A-B), if A>B it becomes reset and if (A<B) it becomes set. Carry flag is also called borrow flag.

1-carry out from MSB bit on addition or borrow into MSB bit on subtraction

0-no carry out or borrow into MSB bit

Example:

MVI A 30 (load 30H in register A)

MVI B 40 (load 40H in register B)

SUB B (A = A – B)

These set of instructions will set the carry flag to 1 as 30 – 40 generates a carry/borrow.

Instruction Execution and Instruction Cycles,

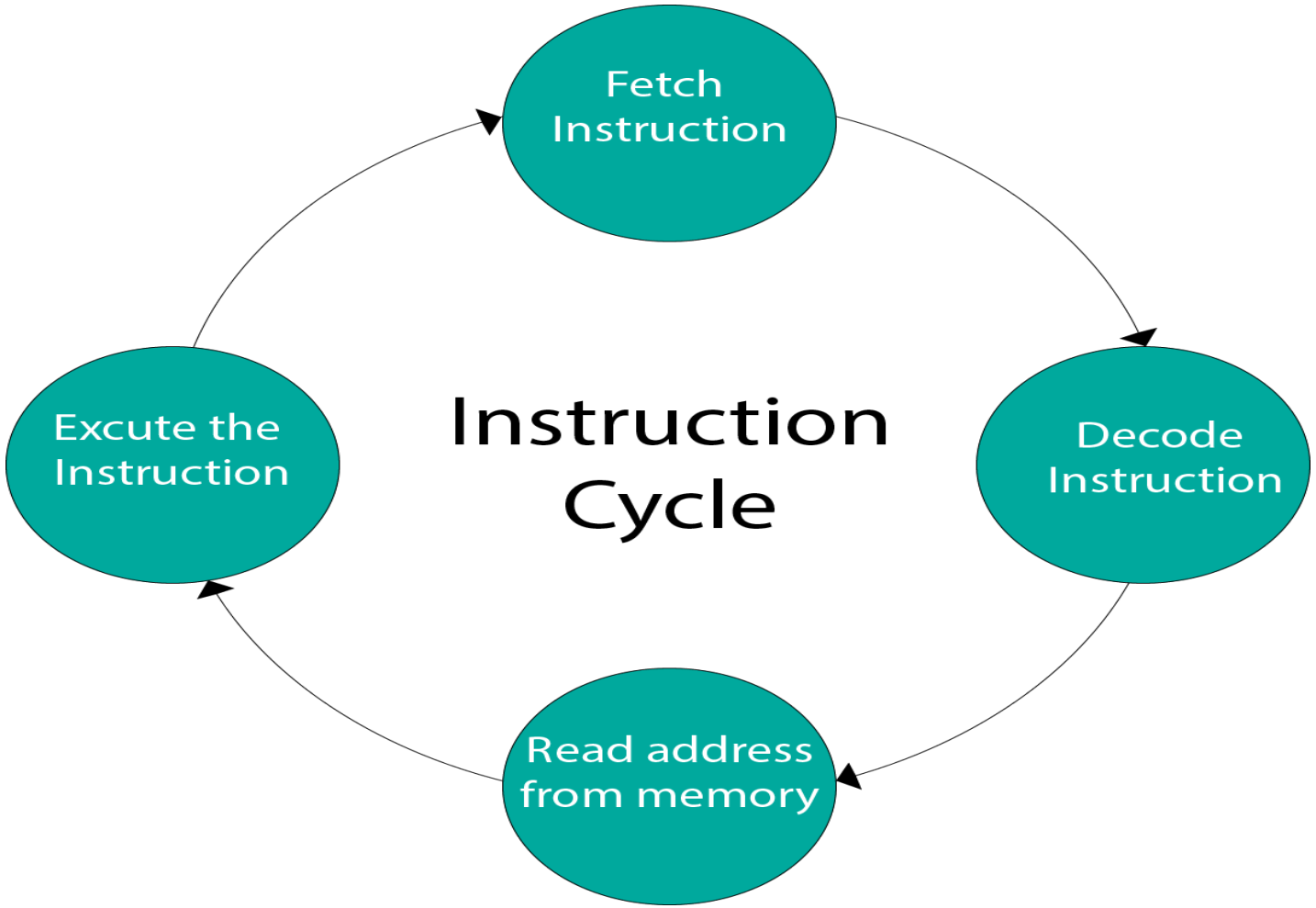
1.Fetch instruction from memory.

2.Decode the instruction.

3.Read the effective address from memory.

4.Execute the instruction.

Instruction Execution and Instruction Cycles,



Machine Cycles & Timing Diagram,

Timing Diagram : Timing Diagram is a graphical representation. It represents the execution time taken by each instruction in a graphical format. The execution time is represented in T-states.

Instruction Cycle: The time required to execute an instruction is called instruction cycle.

Machine Cycle: The time required to access the memory or input/output devices is called machine cycle.

T-State:

- The machine cycle and instruction cycle takes multiple clock periods.
- A portion of an operation carried out in one system clock period is called as T-state.

Machine cycles of 8085

- The 8085 microprocessor has 5 basic machine cycles. They are

1. Opcode fetch cycle (4T)
2. Memory read cycle (3 T)
3. Memory write cycle (3 T)
4. I/O read cycle (3 T)
5. I/O write cycle (3 T)

Machine cycles of 8085

Time period, T = 1/f ; where f = Internal clock frequency

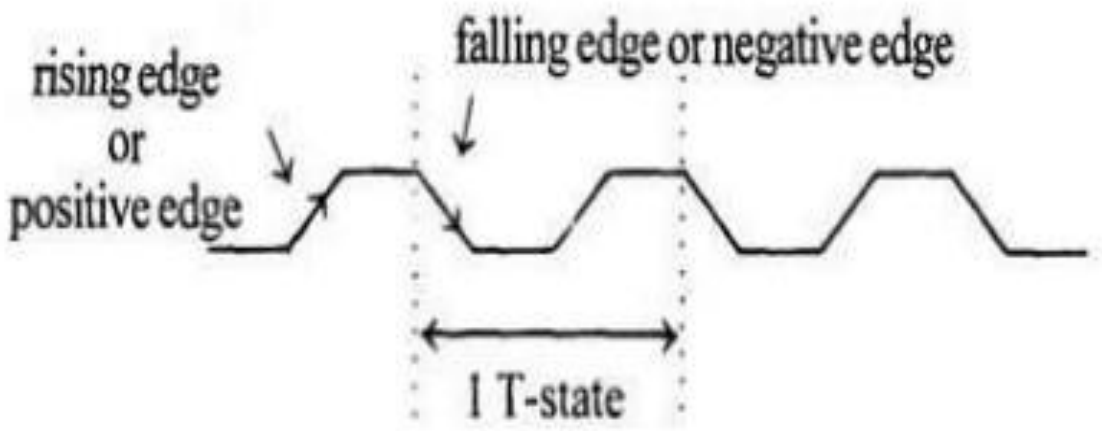


Fig 1.7 Clock Signal

Machine cycles of 8085

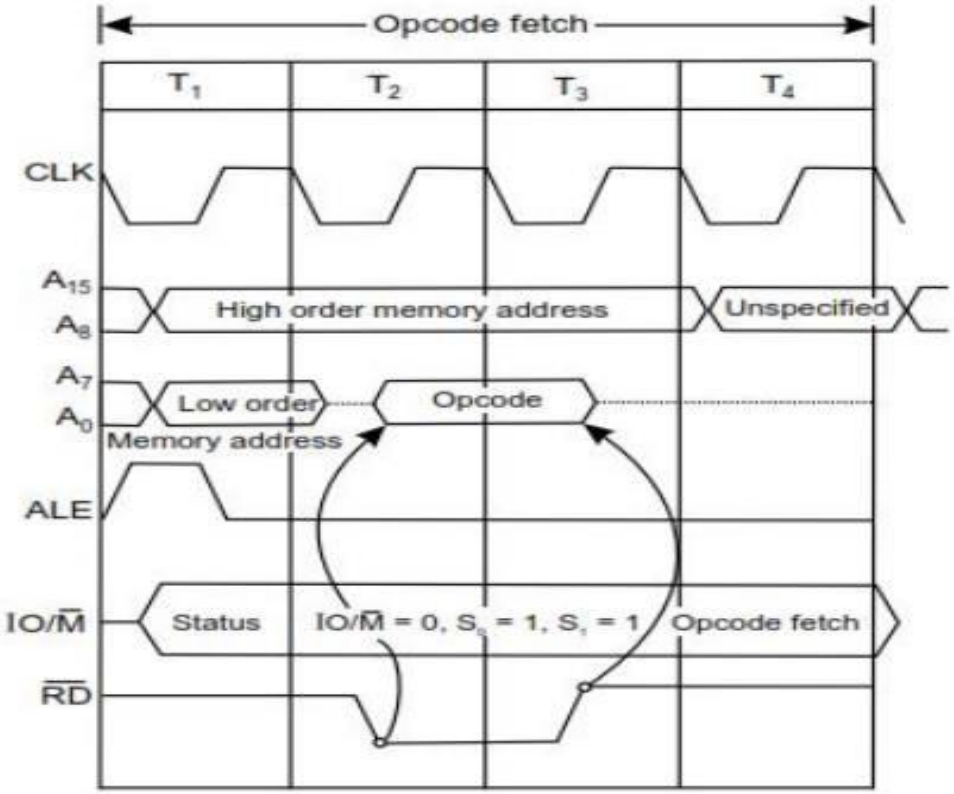


Fig 1.8 Opcode fetch machine cycle

- Each instruction of the processor has one byte opcode.
- The opcodes are stored in memory. So, the processor executes the opcode fetch machine cycle to fetch the opcode from memory.
- Hence, every instruction starts with opcode fetch machine cycle.
- The time taken by the processor to execute the opcode fetch cycle is 4T.
- In this time, the first, 3 T-states are used for fetching the opcode from memory and the remaining T-states are used for internal operations by the processor.

Machine cycles of 8085

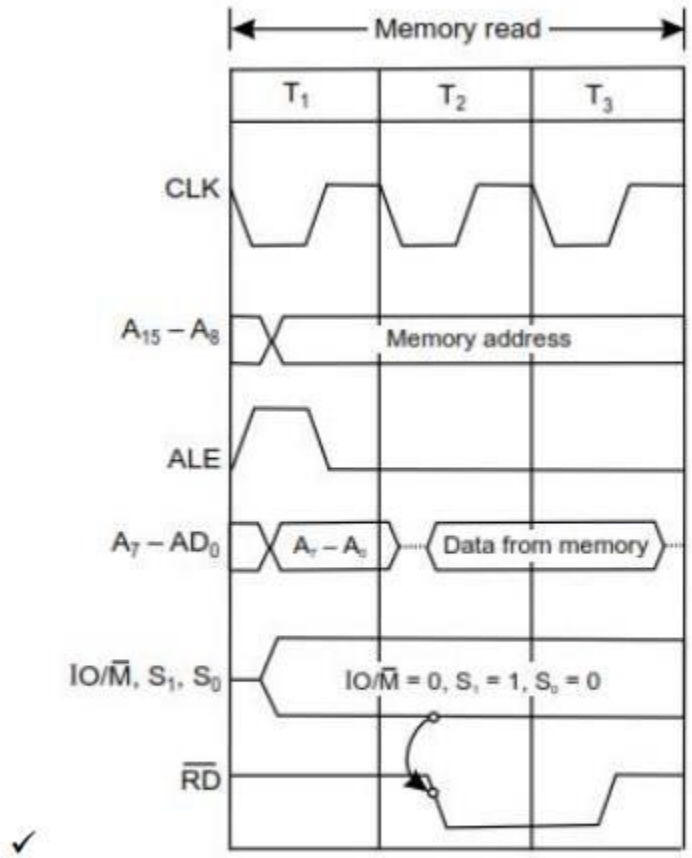


Fig 1.9 Memory Read Machine Cycle

- The memory read machine cycle is executed by the processor to read a data byte from memory.
- The processor takes 3T states to execute this cycle.
- The instructions which have more than one byte word size will use the machine cycle after the opcode fetch machine cycle.

Machine cycles of 8085

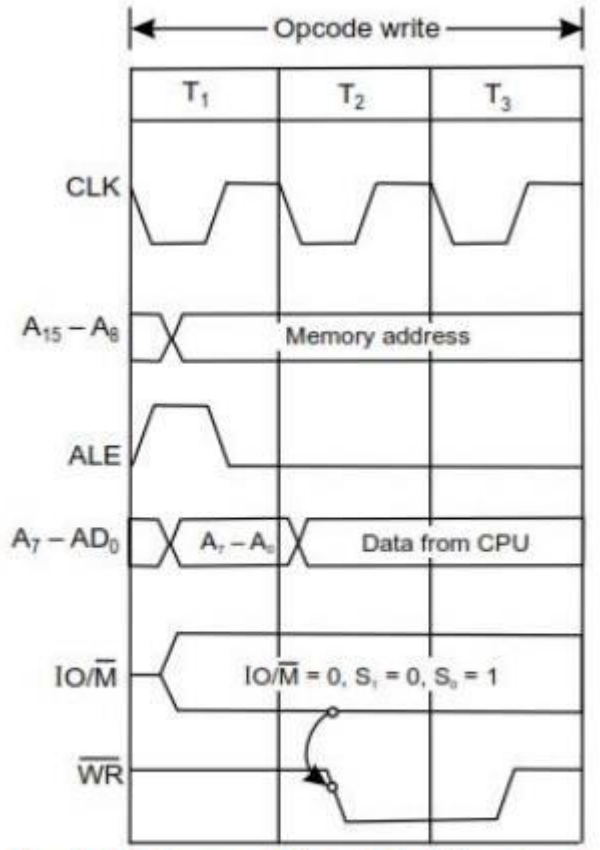


Fig 1.10 Memory Write Machine Cycle

- 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.

Machine cycles of 8085

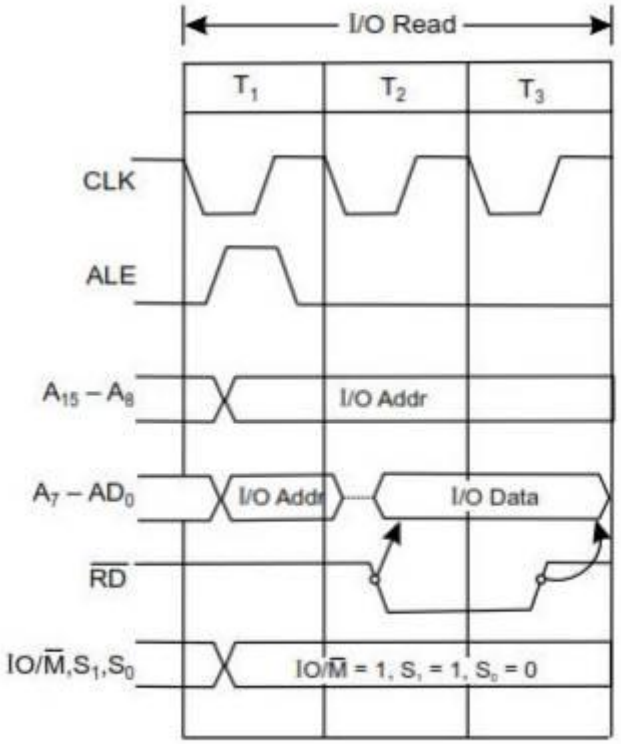


Fig 1.11 I/O Read Cycle

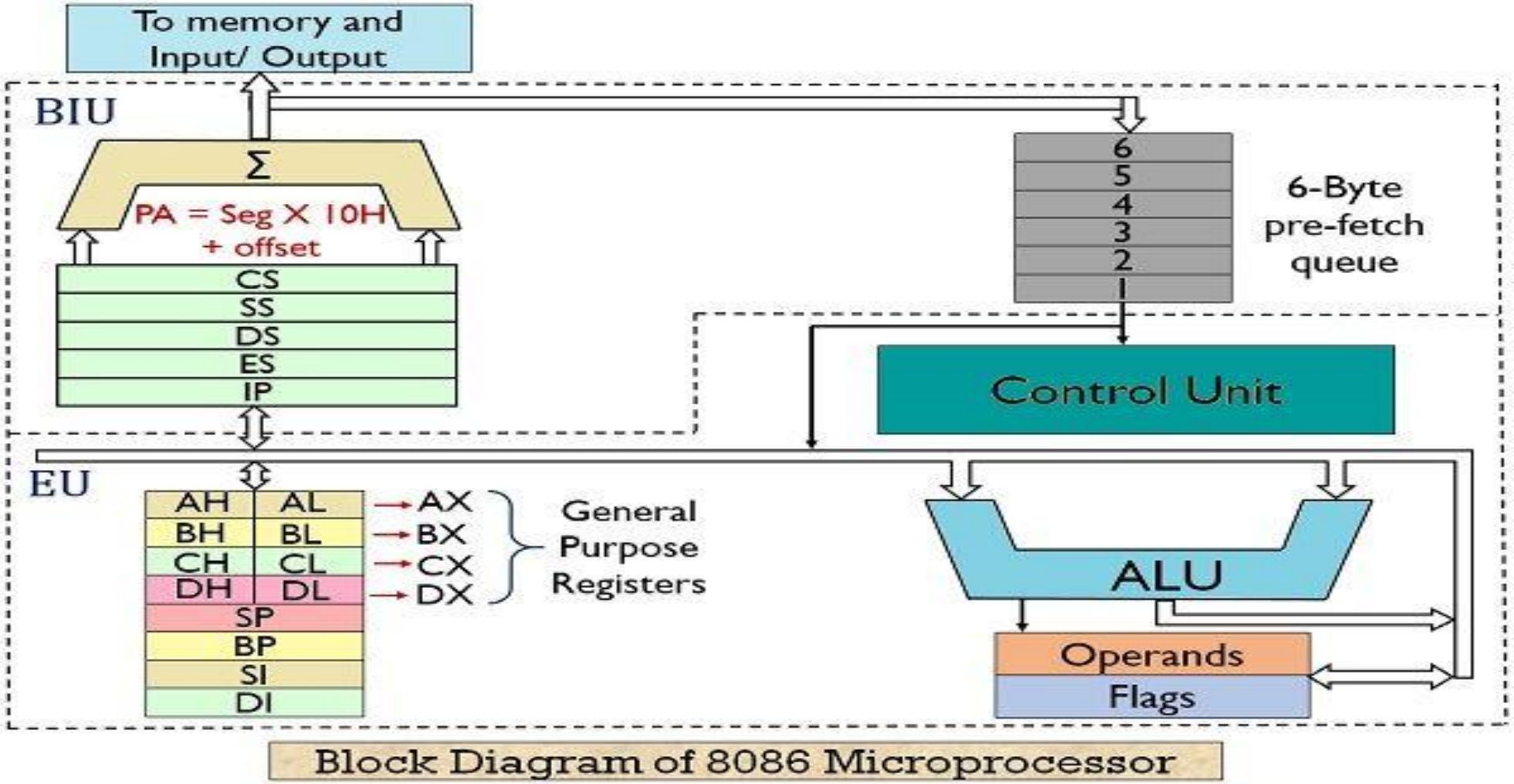
- The I/O Read cycle is executed by the processor to read a data byte from I/O port or from the peripheral, which is I/O, mapped in the system.

- The processor takes 3T states to execute this machine cycle.
- The IN instruction uses this machine cycle during the execution.

16 bit Microprocessor

- A 16 bit microprocessor is having 16bit register set. It have 16 address and data lines to transfer address and data both. Hence it is 16 address lines. The maximum addresses are 2^16 means 65536.

8086 Architecture



8086 Architecture

- 8086 Microprocessor is an enhanced version of 8085Microprocessor that was designed by Intel in 1976. It is a 16-bit Microprocessor having 20 address lines and16 data lines that provides up to 1MB storage. It consists of powerful instruction set, which provides operations like multiplication and division easily.
- It supports two modes of operation, i.e. Maximum mode and Minimum mode. Maximum mode is suitable for system having multiple processors and Minimum mode is suitable for system having a single processor.

Comparison between 8085 & 8086 Microprocessor

- **Size** − 8085 is 8-bit microprocessor, whereas 8086 is 16-bit microprocessor.
- **Address Bus** − 8085 has 16-bit address bus while 8086 has 20-bit address bus.
- **Memory** − 8085 can access up to 64Kb, whereas 8086 can access up to 1 Mb of memory.
- **Instruction** − 8085 doesn’t have an instruction queue, whereas 8086 has an instruction queue.
- **Pipelining** − 8085 doesn’t support a pipelined architecture while 8086 supports a pipelined architecture.
- **I/O** − 8085 can address 2^8 = 256 I/O's, whereas 8086 can access 2^16 = 65,536 I/O's.
- **Cost** − The cost of 8085 is low whereas that of 8086 is high.

8086 Architecture

EU (Execution Unit)

- Execution unit gives instructions to BIU stating from where to fetch the data and then decode and execute those instructions. Its function is to control operations on data using the instruction decoder & ALU. EU has no direct connection with system buses as shown in the above figure, it performs operations over data through BIU.

ALU

ALU

- It handles all arithmetic and logical operations, like +, −, ×, /, OR, AND, NOT operations.

Flag Register(Conditional Flags)

It is a 16-bit register that behaves like a flip-flop, i.e. it changes its status according to the result stored in the accumulator. It has 9 flags and they are divided into 2 groups − Conditional Flags and Control Flags.

Conditional Flags

It represents the result of the last arithmetic or logical instruction executed. Following is the list of conditional flags −

- **Carry flag** − This flag indicates an overflow condition for arithmetic operations.
- **Auxiliary flag** − When an operation is performed at ALU, it results in a carry/barrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7), then this flag is set, i.e. carry given by D3 bit to D4 is AF flag. The processor uses this flag to perform binary to BCD conversion.
- **Parity flag** − This flag is used to indicate the parity of the result, i.e. when the lower order 8-bits of the result contains even number of 1’s, then the Parity Flag is set. For odd number of 1’s, the Parity Flag is reset.
- **Zero flag** − This flag is set to 1 when the result of arithmetic or logical operation is zero else it is set to 0.
- **Sign flag** − This flag holds the sign of the result, i.e. when the result of the operation is negative, then the sign flag is set to 1 else set to 0.
- **Overflow flag** − This flag represents the result when the system capacity is exceeded.

Flag Register(Control Flags)

Control Flags

Control flags controls the operations of the execution unit. Following is the list of control flags −

- **Trap flag** − It is used for single step control and allows the user to execute one instruction at a time for debugging. If it is set, then the program can be run in a single step mode.
- **Interrupt flag** − It is an interrupt enable/disable flag, i.e. used to allow/prohibit the interruption of a program. It is set to 1 for interrupt enabled condition and set to 0 for interrupt disabled condition.
- **Direction flag** − It is used in string operation. As the name suggests when it is set then string bytes are accessed from the higher memory address to the lower memory address and vice-a-versa.

General purpose register

There are 8 general purpose registers, i.e., AH, AL, BH, BL, CH, CL, DH, and DL. These registers can be used individually to store 8-bit data and can be used in pairs to store 16bit data. The valid register pairs are AH and AL, BH and BL, CH and CL, and DH and DL. It is referred to the AX, BX, CX, and DX respectively.

- **AX register** − It is also known as accumulator register. It is used to store operands for arithmetic operations.
- **BX register** − It is used as a base register. It is used to store the starting base address of the memory area within the data segment.
- **CX register** − It is referred to as counter. It is used in loop instruction to store the loop counter.
- **DX register** − This register is used to hold I/O port address for I/O instruction.

Stack pointer register

- It is a 16-bit register, which holds the address from the start of the segment to the memory location, where a word was most recently stored on the stack.

BIU (Bus Interface Unit)

BIU takes care of all data and addresses transfers on the buses for the EU like sending addresses, fetching instructions from the memory, reading data from the ports and the memory as well as writing data to the ports and the memory. EU has no direction connection with System Buses so this is possible with the BIU. EU and BIU are connected with the Internal Bus.

It has the following functional parts −

- **Instruction queue** − BIU contains the instruction queue. BIU gets up to 6 bytes of next instructions and stores them in the instruction queue. When EU executes instructions and is ready for its next instruction, then it simply reads the instruction from this instruction queue resulting in increased execution speed.
- Fetching the next instruction while the current instruction executes is called **pipelining**.
- **Segment register** − BIU has 4 segment buses, i.e. CS, DS, SS& ES. It holds the addresses of instructions and data in memory, which are used by the processor to access memory locations. It also contains 1 pointer register IP, which holds the address of the next instruction to executed by the EU.

- **CS** − It stands for Code Segment. It is used for addressing a memory location in the code segment of the memory, where the executable program is stored.
- **DS** − It stands for Data Segment. It consists of data used by the program andis accessed in the data segment by an offset address or the content of other register that holds the offset address.
- **SS** − It stands for Stack Segment. It handles memory to store data and addresses during execution.
- **ES** − It stands for Extra Segment. ES is additional data segment, which is used by the string to hold the extra destination data.

- **Instruction pointer** − It is a 16-bit register used to hold the address of the next instruction to be executed.