13EC3201-MICROPROCESSORS AND INTERFACING

Unit-I
INTRODUCTION TO MICROPROCESSORS

INTRODUCTION:

The microprocessor is nothing but the CPU and it is an essential component of the computer. It is a silicon chip that comprises millions of transistors and other electronic components that process millions of instructions per second. A Microprocessor is a versatile chip, that is combined with memory and special purpose chips and preprogrammed by a software. It accepts digital data as input and processes it according to the instructions stored in the memory. The microprocessor has many functions like functions of data storage, interact with various other devices and other time related functions. But, the main function is to send and receive the data to make the function of the computer well.

Evolution of Microprocessor

The evolution of microprocessors was divided into five generations such as first, second, third, fourth and fifth generation and the characteristics of these generations are discussed below.

First Generation Microprocessors

The first generation microprocessors were introduced in the year 1971-1972. The instructions of these microprocessors were processed serially, they fetched the instruction, decoded and then executed it. When an instruction of the microprocessor was finished, then the microprocessor updates the instruction pointer & fetched the following instruction, performing this consecutive operation for each instruction in turn.

Second Generation Microprocessors

In the year 1970, small amount of transistors were available on the integrated circuit in the second generation microprocessors. Examples of the second generation microprocessors are 16-bit arithmetic 7 pipelined instruction processing, MC68000 Motorola microprocessor. These processors are introduced in the year 1979, and Intel 8080 processor is another example of the microprocessor. The second generation of the microprocessor is defined by overlapped fetch, decode and execute the steps. When the first generation is processed in the execution unit, then the second instruction is decoded and the third instruction is fetched.

The difference between the first generation microprocessor and second generation microprocessors was mainly the use of new semiconductor technologies to manufacture the chips. The result of this technology resulted in a fivefold increase in instruction, speed, execution and higher chip densities.

Third Generation Microprocessors

The third generation microprocessors were introduced in the year 1978, was denoted by Intel's 8086 and the Zilog Z8000. These were 16-bit processors with a performance like mini computers. These types of microprocessors were different from the previous generations of microprocessors in that all main workstation industrialists began evolving their own ISC based microprocessor architectures.

Fourth Generation Microprocessors

As many industries converted from commercial microprocessors to in house designs, the fourth generation microprocessors are entered with outstanding design with a million transistors. Leading edge microprocessors like Motorola’s 88100 and Intel’s 80960CA could issue & retire more than one instruction per clock cycle.

Fifth Generation Microprocessors

Fifth generation microprocessors employed decoupled super scalar processing, and their design soon exceeded 10 million transistors. In fifth generation, PCs are a low-margin, high volume business conquered by a single microprocessor.

Types of Microprocessor

Microprocessors are classified into five types, namely:
CISC-Complex Instruction Set Microprocessors,
RISC-Reduced Instruction Set Microprocessor,
ASIC- Application Specific Integrated Circuit,
Superscalar Processors,
DSP’s-Digital Signal Microprocessors.

Complex Instruction Set Microprocessors

The short term of Complex Instruction Set Microprocessors is CISM and they classify a microprocessor in which orders can be performed together along with other low level activities. These types of processors performs the different tasks like downloading, uploading, recalling data into the memory card and recalling data from the memory card. Apart from these tasks, it also does complex mathematical calculations in a single command.

Reduced Instruction Set Microprocessor

The short term of Reduced Instruction Set Microprocessor is RISC. These types of processors are made according to the function in which the microprocessor can carry out small things in specific command. In this way these processors completes more commands at a faster rate.

Superscalar Microprocessors

Superscalar processor facsimiles the hardware on the processor to perform various tasks at a time. These processors can be used for ALUs or multipliers. They have different operational units and these processors can carry out more than a one command by continuously transmitting several instructions to the extra operational units inside the processor.

The Application Specific Integrated Circuit

The short term of Application Specific Integrated Circuit processor is an ASIC. These processors are used for particular purposes that includes of automotive emissions control or personal digital assistant’s computer. This type of processor is made with proper specification, but apart from these it can also be made with off the shelf gears.

Digital Signal Multiprocessors

Digital signal processors are also called as DSP’s, these processors are used to encode and decode the videos or to convert the D/A (digital to analog) &A/D (analog to digital). They

need a microprocessor that is excellent in mathematical calculations. The chips of this processor are employed in RADAR, home theaters, SONAR, audio gears, TV set top boxes and Mobile phones.
There are many companies like Intel, Motorola, DEC (Digital Equipment Corporation), TI (Texas Instruments) associated with many microprocessors such as 8085 microprocessors, ASIC, CISM, RISC, DSPs and 8086 microprocessors like Intel

Advantages and Disadvantages of Microprocessors
The advantages of microprocessors are

- The processing speed is high
- Intelligence has been brought to systems
- Flexible.
- Compact size.
- Easy maintenance
- Complex mathematics

Some of the disadvantages of microprocessor are it might get overheated and the limitation of the microprocessor imposes on size of data.

The applications of the microprocessors mainly involve in controllers in home appliances, wireless communication equipments, office publication and automation, consumer electronic goods, calculators, accounting system, video games, industrial controllers and data acquisition systems.

8085 Architecture:

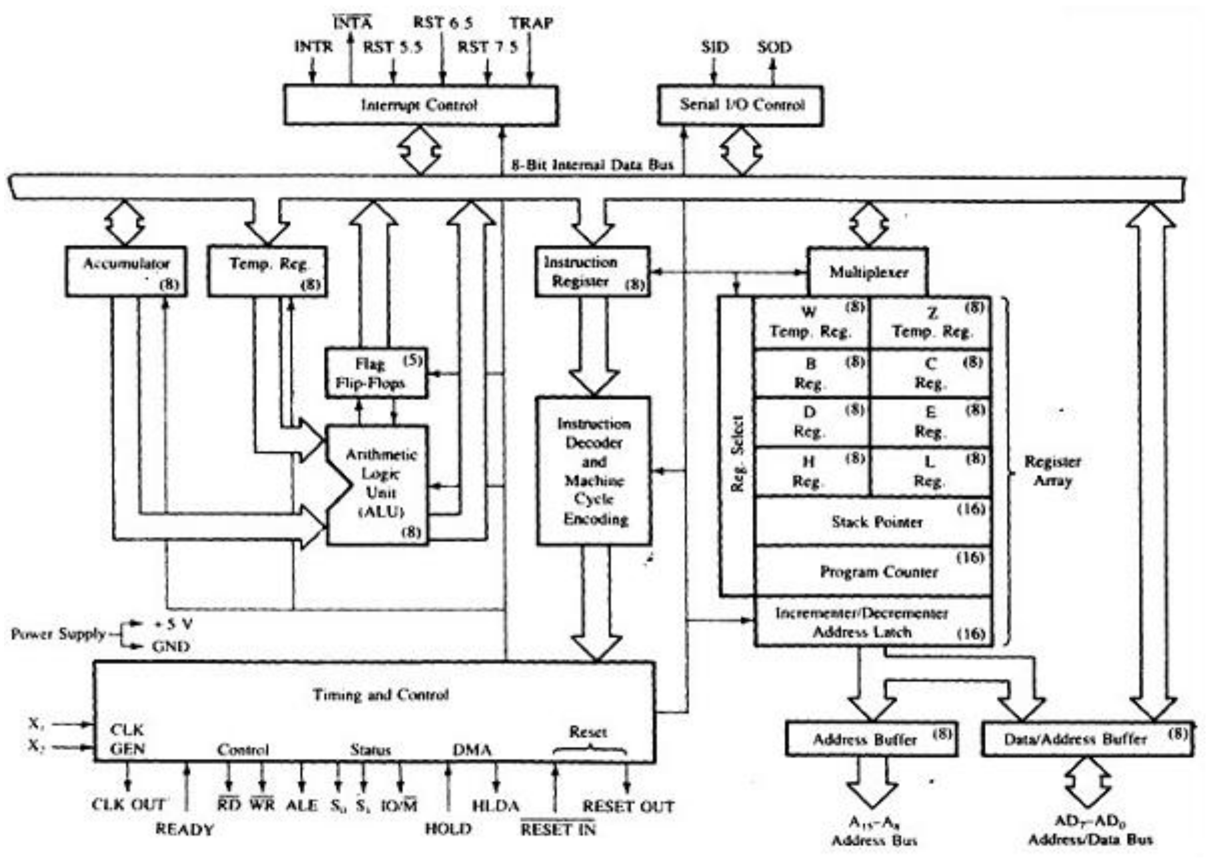

Figure:1- 8085 Architecture

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 −

- 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 – Functional Units

8085 consists of the following functional units as shown in figure:1.

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

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 B-C, D-E & H-L.

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.

Stack pointer

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

Temporary register

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

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.

These are the set of 5 flip-flops −

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

Its bit position is shown in the following table :

D7	D6	D5	D4	D3	D2	D1	D0
S	Z		AC		P		CY

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.

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 as shown in figure:1.

- Control Signals: READY, RD’, WR’, ALE
- Status Signals: S0, S1, IO/M’
- DMA Signals: HOLD, HLDA
- RESET Signals: RESET IN, RESET OUT

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.

Serial Input/output control

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

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.

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.

pin diagram of 8085 Microprocessor:

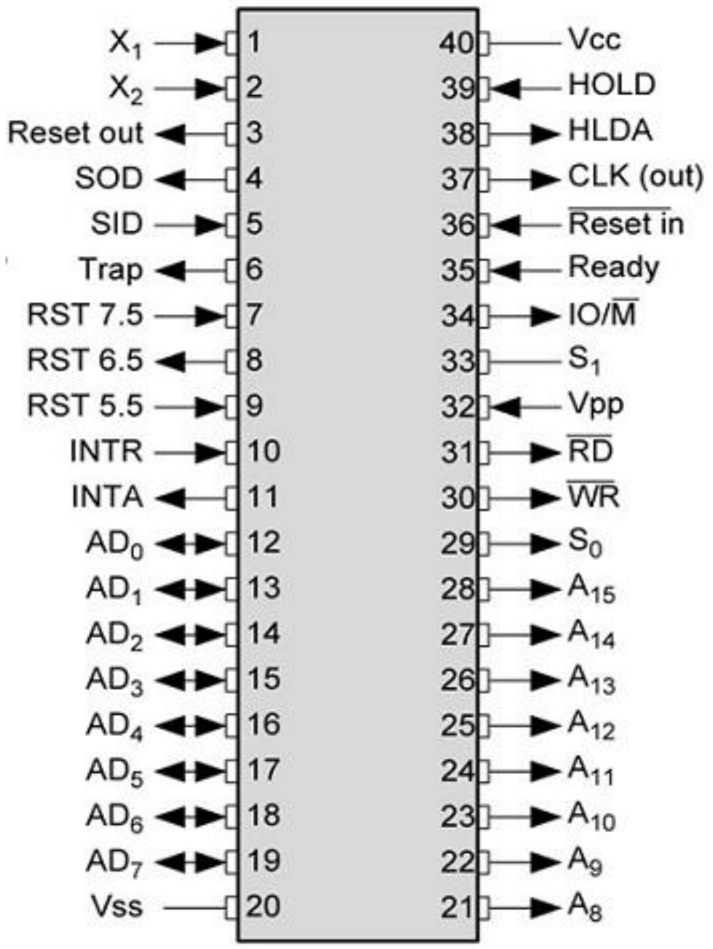

Figure:2 pin diagram of 8085

The pins of a 8085 microprocessor can be classified into seven groups :

Address bus

A15-A8, it carries the most significant 8-bits of memory/IO address.

Data bus

AD7-AD0, it carries the least significant 8-bit address and data bus.

Control and status signals

These signals are used to identify the nature of operation. There are 3 control signal and 3 status signals.

Three control signals are RD, WR & ALE.

- **RD** − This signal indicates that the selected IO or memory device is to be read and is ready for accepting data available on the data bus.
- **WR** − This signal indicates that the data on the data bus is to be written into a selected memory or IO location.

- **ALE** − It is a positive going pulse generated when a new operation is started by the microprocessor. When the pulse goes high, it indicates address. When the pulse goes down it indicates data.

Three status signals are IO/M, S0 & S1.

IO/M

This signal is used to differentiate between IO and Memory operations, i.e. when it is high indicates IO operation and when it is low then it indicates memory operation.

S1 & S0

These signals are used to identify the type of current operation.

S0	S1	STATS
0	0	HALT
0	1	READ
1	0	WRITE
1	1	IDLE

Power supply

There are 2 power supply signals − V_{CC} & V_{SS}. V_{CC} indicates +5v power supply and V_{SS} indicates ground signal.

Clock signals

There are 3 clock signals, i.e. X1, X2, CLK OUT.

- **X1, X2** − A crystal (RC, LC N/W) is connected at these two pins and is used to set frequency of the internal clock generator. This frequency is internally divided by 2.
- **CLK OUT** − This signal is used as the system clock for devices connected with the microprocessor.

Interrupts & externally initiated signals

Interrupts are the signals generated by external devices to request the microprocessor to perform a task. There are 5 interrupt signals, i.e. TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR. We will discuss interrupts in detail in interrupts section.

- **INTA** − It is an interrupt acknowledgment signal.
- **RESET IN** − This signal is used to reset the microprocessor by setting the program counter to zero.
- **RESET OUT** − This signal is used to reset all the connected devices when the microprocessor is reset.
- **READY** − This signal indicates that the device is ready to send or receive data. If READY is low, then the CPU has to wait for READY to go high.
- **HOLD** − This signal indicates that another master is requesting the use of the address and data buses.
- **HLDA (HOLD Acknowledge)** − It indicates that the CPU has received the HOLD request and it will relinquish the bus in the next clock cycle. HLDA is set to low after the HOLD signal is removed.

Serial I/O signals

There are 2 serial signals, i.e. SID and SOD and these signals are used for serial communication.

- **SOD** (Serial output data line) − The output SOD is set/reset as specified by the SIM instruction.
- **SID** (Serial input data line) − The data on this line is loaded into accumulator whenever a RIM instruction is executed.

Clock cycle

The speed of a computer processor, or CPU, is determined by the **clock cycle**, which is the amount of time between two pulses of an oscillator. Generally speaking, the higher number of pulses per second, the faster the computer processor will be able to process information. The clock speed is measured in Hz, typically either megahertz (MHz) or gigahertz (GHz). For example, a 4GHz processor performs 4,000,000,000 clock cycles per second. Computer processors can execute one or more instructions per clock cycle, depending on the type of processor. Early computer processors and slower processors can only execute one instruction per clock cycle, but faster, more advanced processors can execute multiple instructions per clock cycle, processing data more efficiently.

Machine cycle

The steps performed by the computer processor for each machine language instruction received. The **machine cycle** is a 4 process cycle as shown in figure:3,that includes reading and interpreting the machine language, executing the code and then storing that code.

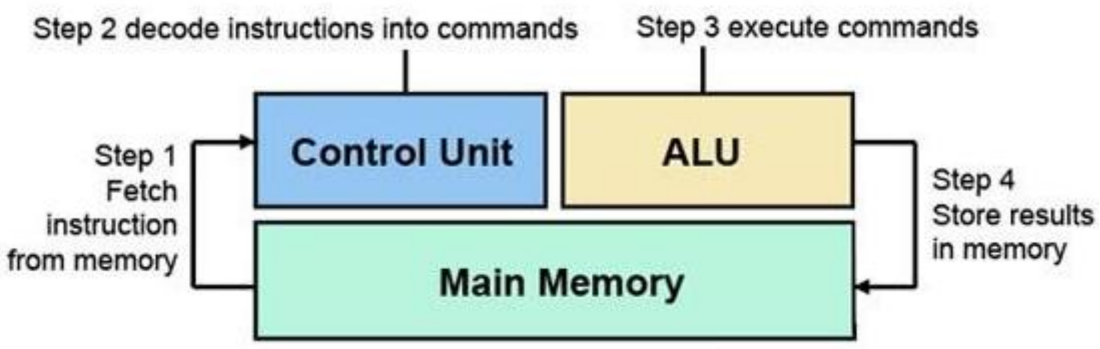
Step 2 decode instructions into commands
Step 3 execute commands
Step 1 Fetch instruction from memory
Control Unit
ALU
Step 4 Store results in memory
Main Memory

Figure:3 machine cycle

Four steps of Machine cycle

1. **Fetch** - Retrieve an instruction from the memory.
2. **Decode** - Translate the retrieved instruction into a series of computer commands.
3. **Execute** - Execute the computer commands.
4. **Store** - Send and write the results back in memory.

Instruction cycle

The sequence of operations that the cpu has to carry out while execution is called instruction cycle.
1:- Read an Instruction
2:- Decode the instruction
3:- Find the address of operand
4:- retrieve an operand
5:- perform desired operation
6:- find the address of destination
7:- store the result into the destination

Timing diagram of 8085:

It is one of the best way to understand to process of micro-processor/controller. With the help of timing diagram we can understand the working of any system, step by step working of each instruction and its execution, etc.

It is the graphical representation of process in steps with respect to time. The timing diagram represents the clock cycle and duration, delay, content of address bus and data bus, type of operation ie. Read/write/status signals.

Important terms related to timing diagrams:

1. **Instruction cycle**: this term is defined as the number of steps required by the cpu to complete the entire process ie. Fetching and execution of one instruction. The fetch and execute cycles are carried out in synchronization with the clock.
2. **Machine cycle:** It is the time required by the microprocessor to complete the operation of accessing the memory devices or I/O devices. In machine cycle various operations like opcode fetch, memory read, memory write, I/O read, I/O write are performed.
3**. T-state:** Each clock cycle is called as T-states.

Rules to identify number of machine cycles in an instruction:

1. If an addressing mode is direct, immediate or implicit then No. of machine cycles = No. of bytes.

2. If the addressing mode is indirect then No. of machine cycles = No. of bytes + 1. Add +1 to the No. of machine cycles if it is memory read/write operation.

3. If the operand is 8-bit or 16-bit address then, No. of machine cycles = No. of bytes +1.

4. These rules are applicable to 80% of the instructions of 8085.

Timing Diagram:

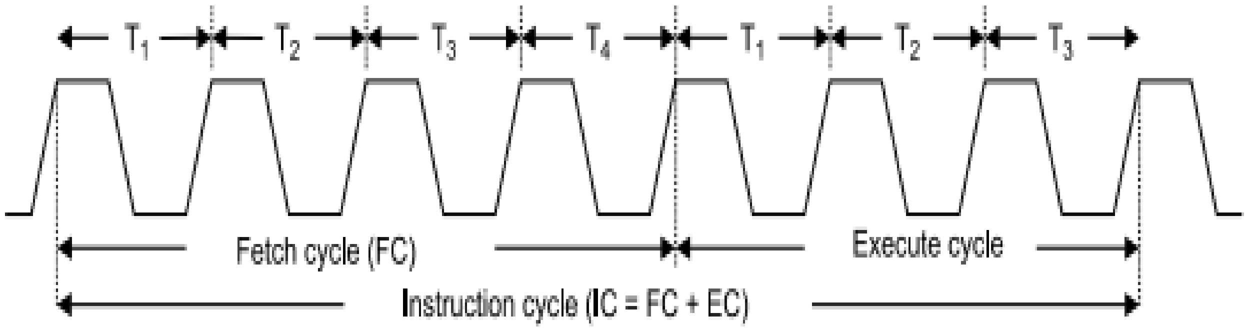

Figure:4 timing diagram of fetch,machine,instruction cycle

Opcode fetch:

The microprocessor requires instructions to perform any particular action. In order to perform these actions microprocessor utilizes Opcode which is a part of an instruction which provides detail(ie. Which operation µp needs to perform) to microprocessor.

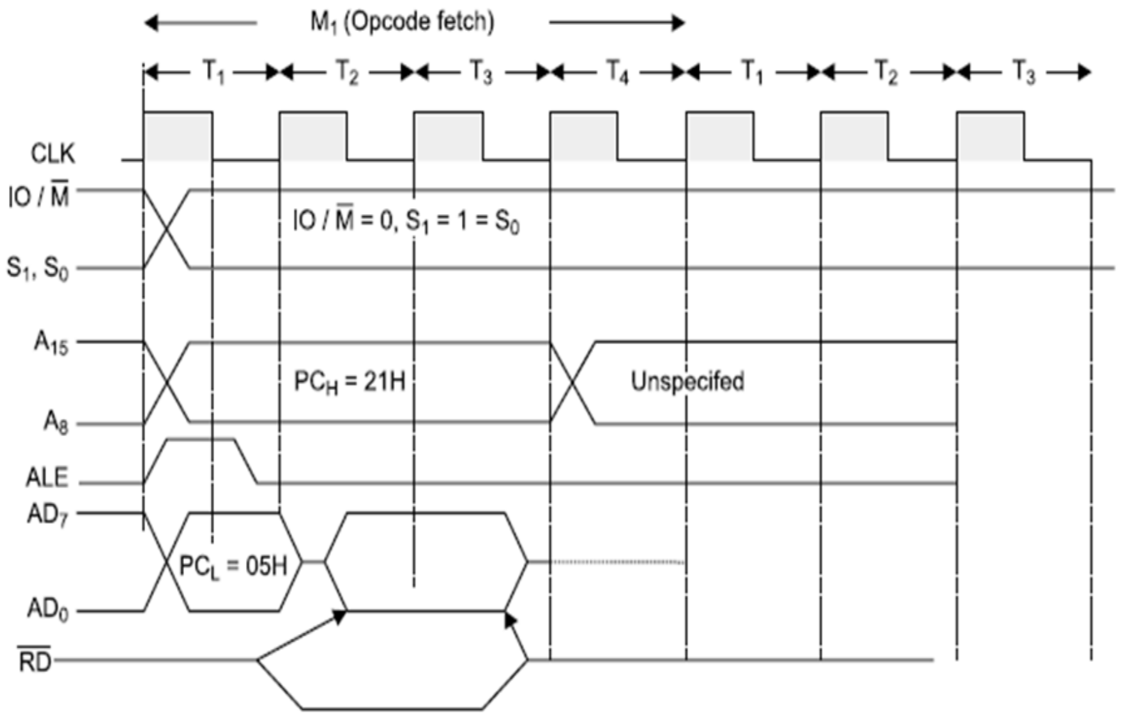

Figure 5: Opcode fetch timing diagram

Operation:

- During T1 state, microprocessor uses IO/M(bar), S0, S1 signals are used to instruct microprocessor to fetch opcode.
- Thus when IO/M(bar)=0, S0=S1= 1, it indicates opcode fetch operation.
- During this operation 8085 transmits 16-bit address and also uses ALE signal for address latching.

- At T2 state microprocessor uses read signal and make data ready from that memory location to read opcode from memory and at the same time program counter increments by 1 and points next instruction to be fetched.
- In this state microprocessor also checks READY input signal, if this pin is at low logic level ie. '0' then microprocessor adds wait state immediately between T2 and T3.
- At T3, microprocessor reads opcode and store it into instruction register to decode it further.
- During T4 microprocessor performs internal operation like decoding opcode and providing necessary actions.
- The opcode is decoded to know whether T5 or T6 states are required, if they are not required then µp performs next operation.

Read and write timing diagram for memory and I/O Operation
Memory Read:

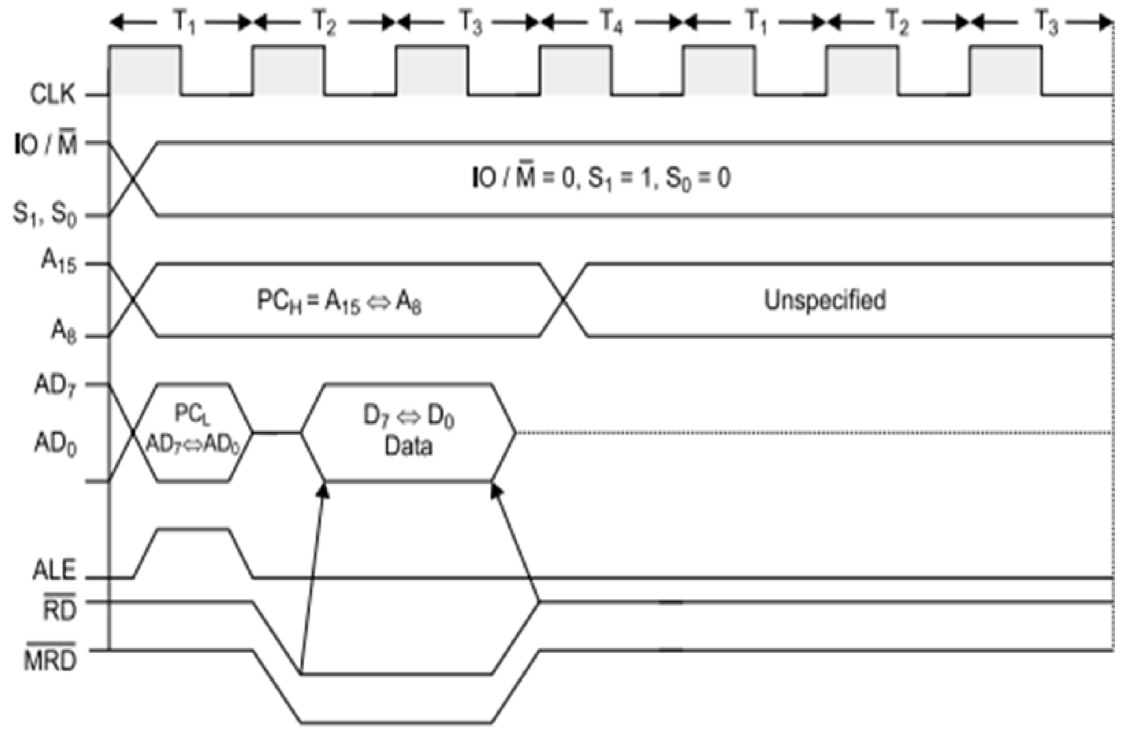

Figure 6: Memory read timing diagram

Operation:

- It is used to fetch one byte from the memory.
- It requires 3 T-States.
- It can be used to fetch operand or data from the memory.
- During T1, A8-A15 contains higher byte of address. At the same time ALE is high. Therefore Lower byte of address A0-A7 is selected from AD0-AD7.
- Since it is memory ready operation, IO/M(bar) goes low.
- During T2 ALE goes low, RD(bar) goes low. Address is removed from AD0-AD7 and data D0-D7 appears on AD0-AD7.
- During T3, Data remains on AD0-AD7 till RD(bar) is at low signal.

Memory Write:

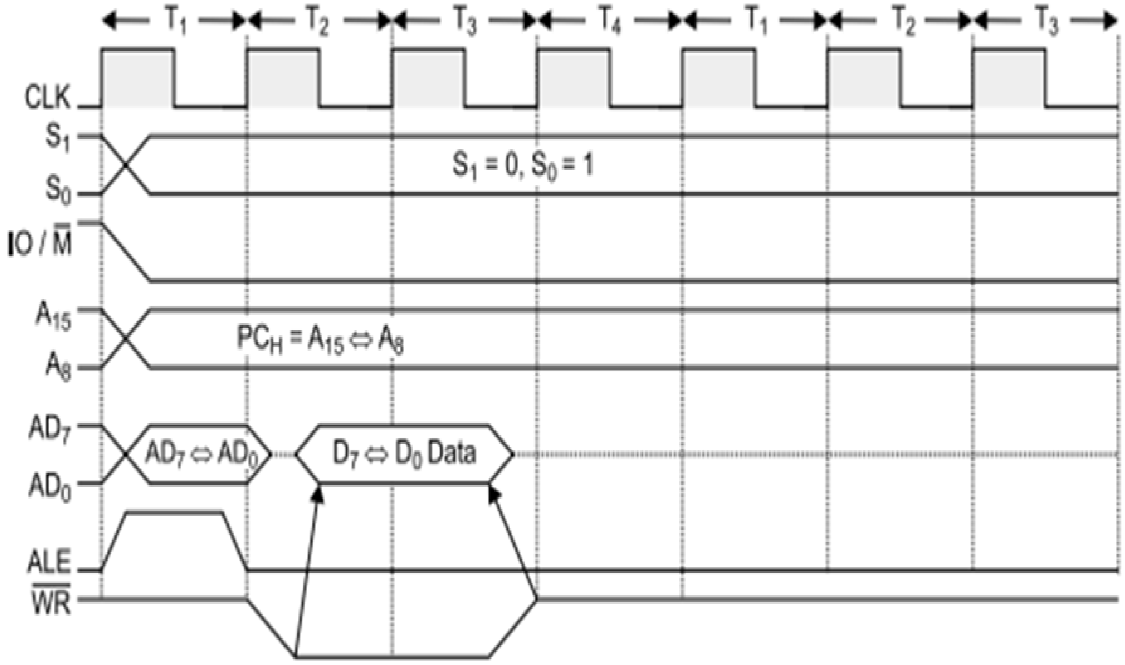

Figure 7: Memory write timing diagram

Operation:

- It is used to send one byte into memory.
- It requires 3 T-States.
- During T1, ALE is high and contains lower address A0-A7 from AD0-AD7.
- A8-A15 contains higher byte of address.
- As it is memory operation, IO/M(bar) goes low.
- During T2, ALE goes low, WR(bar) goes low and Address is removed from AD0-AD7 and then data appears on AD0-AD7.
- Data remains on AD0-AD7 till WR(bar) is low.

IO Read:

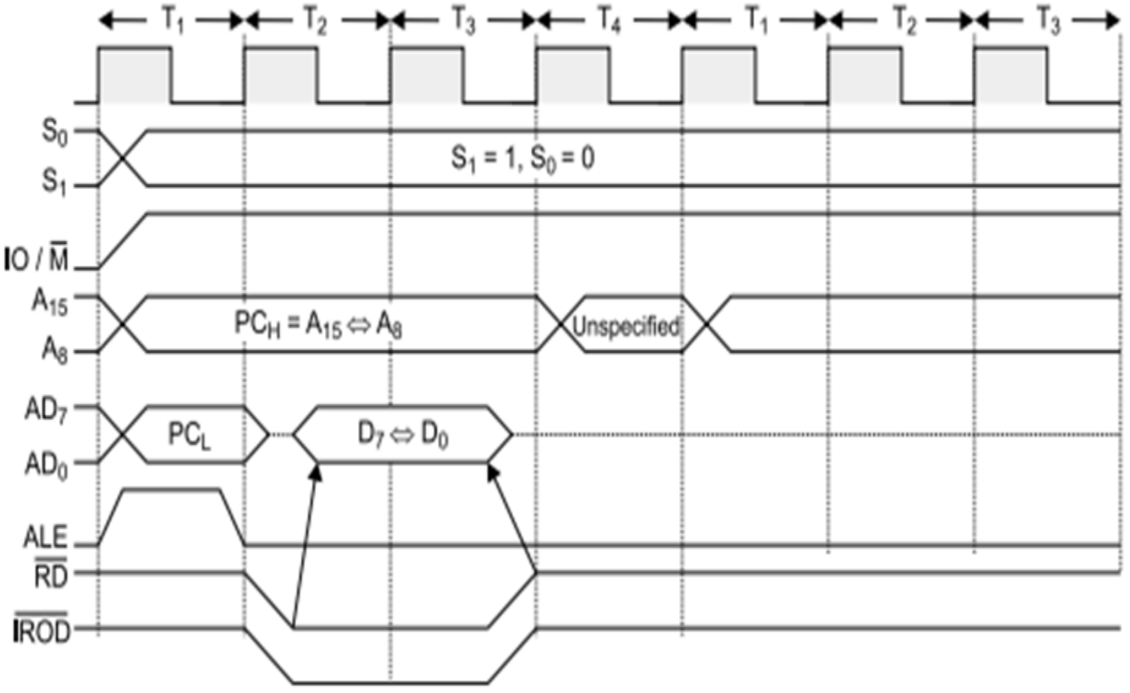

Figure 8: I/O read timing diagram

Operation:

It is used to fetch one byte from an IO port.

- It requires 3 T-States.
- During T1, The Lower Byte of IO address is duplicated into higher order address bus A8-A15.
- ALE is high and AD0-AD7 contains address of IO device.
- IO/M (bar) goes high as it is an IO operation.
- During T2, ALE goes low, RD (bar) goes low and data appears on AD0-AD7 as input from IO device.
- During T3 Data remains on AD0-AD7 till RD(bar) is low.

IO Write:

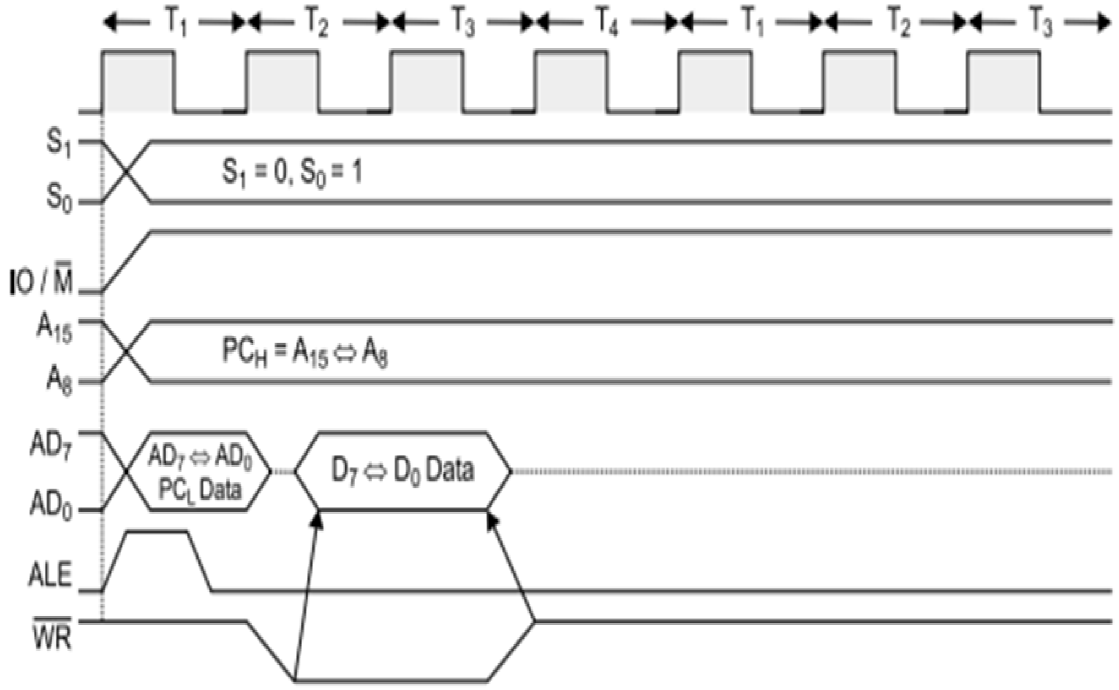

Figure 9:I/O write timing diagram

Operation:

- It is used to writ one byte into IO device.
- It requires 3 T-States.
- During T1, the lower byte of address is duplicated into higher order address bus A8-A15.
- ALE is high and A0-A7 address is selected from AD0-AD7.
- As it is an IO operation IO/M (bar) goes low.
- During T2, ALE goes low, WR (bar) goes low and data appears on AD0-AD7 to write data into IO device.
- During T3, Data remains on AD0-AD7 till WR(bar) is low.

The STACK

The stack is one of the most important things you must know when programming. Think of the stack as a deck of cards. When you put a card on the deck, it will be the top card. Then you put another card, then another. When you remove the cards, you remove them backwards, the last card first and so on. The stack works the same way, you put (push) words (addresses or register pairs) on the stack and then remove (pop) them backwards. That's called LIFO, Last In First Out.

The 8085 uses a 16 bit register to know where the stack top is located, and that register is called the SP (Stack Pointer). There are instructions that allow you to modify it’s contents.

PUSH & POP:

As you may have guessed, push and pop “pushes” bytes on the stack and then takes them off. When you push something, the stack counter will decrease with 2 (the stack "grows" down, from higher addresses to lower) and then the register pair is loaded onto the stack. When you pop, the register pair is first lifted of the stack, and then SP increases by 2. N.B: Push and Pop only operate on words (2 bytes ie: 16 bits).

You can push (and pop) all register pairs: BC, DE, HL and PSW (Register A and Flags). When you pop PSW, remember that all flags may be changed. You can't push an immediate value. If you want, you'll have to load a register pair with the value and then push it. Perhaps it's worth noting that when you push something, the contents of the registers will still be the same; they won't be erased or something. Also, if you push DE, you can pop it back as HL (you don't have to pop it back to the same register where you got it from).

The stack is also updated when you CALL and RETurn from subroutines. The PC (program counter which points at the next instruction to be executed) is pushed to the stack and the calling address is loaded into PC. When returning, the PC is loaded with the word popped from the top of the stack (TOS).So, when is this useful? It's almost always used when you call subroutines. For example, you have an often used value stored in HL. You have to call a subroutine that you know will destroy HL (with destroy I mean that HL will be changed to another value, which you perhaps don't know). Instead of first saving HL in a memory location and then loading it back after the subroutine, you can push HL before calling and directly after

the calling pop it back. Of course, it's often better to use the pushes and pops inside the subroutine. All registers you know will be changed are often pushed in the beginning of a subroutine and then popped at the end, in reverse order! Don't forget - last in first out. If you want to only push one 8 bit register, you still have to push it's "friend". Therefore, be aware that if you want to store away D with pushing and popping, remember that E will also be changed back to what it was before. In those cases, if you don't want that to happen, you should try first to change register (try to store the information in E in another register if you can) or else you have to store it in a temporary variable.
Before executing a program, you should keep track of your pushes and pops, since they are responsible for 99% of all computer crashes! For example, if you push HL and then forget to pop it back, the next RET instruction will cause a jump to HL, which can be anywhere in the ROM/RAM and the ccomputer will crash. Note however, it’s also a way to jump to the location stored in HL, but then you should really use the JMP instruction, to do the same thing. Push and pop doesn't change any flags, so you can use them between a compare and jump instructions, depending on a condition, which is often very useful.