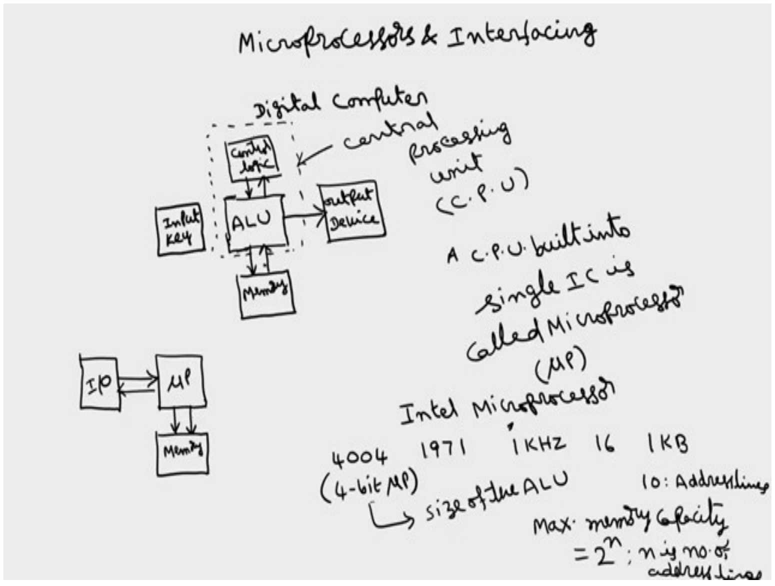
Microprocessors and Interfacing

Prof. Shaik Rafi Ahmed
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati

Lecture - 01
Microprocessor Operations

(Refer Slide Time 00:33)


Microprocessors & Interfacing
Digital Computer
control logic
ALU
Input Key
output Device
Memory
central processing unit (C.P.U)
A C.P.U. built into single IC is called Microprocessor (μP)
I/P
μP
Memory
Intel Microprocessor
4004 1971 1KHZ 16 1KB
(4-bit μP)
size of the ALU
10: Address lines
Max. memory capacity = 2^n ; n is no. of address lines

So, very good morning; so the title of my course is Microprocessors and Interfacing. So, first I will give you the introduction to the microprocessor; what is the microprocessor? If you consider a digital computer block diagram; so what are the different blocks in a digital computer?

So, you require some input device through which we will take the data input device such as keyboard and mouse and this input data will be given to a ALU; Arithmetic Logic Unit which performs various arithmetic and logical operations. Arithmetic operations such as addition, multiplication, subtraction and logical operations such as ANDing, ORing, inverting, compliment all these operations.

Then to store the results; we require some output device and you can store the programs in the memory; and you require some control logic to control all the operations. The combination of this ALU and control logic is called as a microprocessor which is called a Central Processing Unit CPU. So, a CPU can be made of the different circuits like adder, subtractor and AND operation, OR operation and all or otherwise you can built this CPU into single IC; that single IC is called a microprocessor. A CPU built into single IC integrated circuit is called a microprocessor.

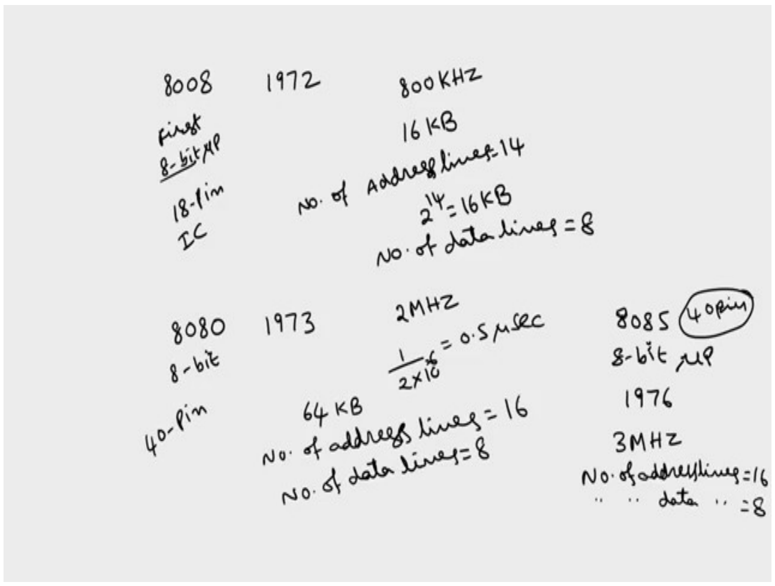
So, normally the microprocessor I will call as μ (mu) P because μ is the symbol for the micro and P is processor. So, what is the; I mean generation of these microprocessors? So, the first microprocessor was the Intel's microprocessors. So, in this course I am going to discuss only Intel's microprocessors. The first microprocessor that was manufactured by the Intel is 4004 in the year 1971 and this is 4-bit microprocessor.

So, what is the meaning of this 4-bit, 8-bit and 16-bit, 32-bit microprocessor? So, this basically decide the size of the ALU; this 4-bit implies the size of the ALU, that is if you want to perform addition of two 4-bit numbers; we can perform in a single clock cycle, so using this 4004. If you want to perform addition of two 8-bit numbers; it requires two clock cycles and so on. So, this is 4-bit microprocessor was first I mean fabricated in 1971. So, this is a 4-bit microprocessor and whose clock frequency is 1 kilo Hertz and the number of pins in this IC are 16 and this memory capacity of these 4004 was just 1 kilobyte means there are 10 address lines.

So, the relation between this number of address lines and the memory capacity is the maximum memory that can be connected to the microprocessor; here because the memory is outside the microprocessor. If you take the simplified circular diagram of the block diagram of this one; this is input device, you can call this IO in a single block. Then this is microprocessor, the combination of ALU and control logic; then we will be having memory. This is also bidirectional because its IO; from input, the data will be taken in this direction and from microprocessor the data will be outputted in this direction.

So, here this requires this 7-10 address lines and 1 kilobytes of the memory. The relation between the number of address lines and memory is the maximum memory capacity is given by is equal 2 raised to the power of n, where n is the number of address lines. And the number of data lines will be decided by this 4-bit microprocessor means the number of data lines will be 4-bits. So, this Intel 4004 was the first I mean Intel's microprocessor which is I mean generated which is fabricated in 1971.

(Refer Slide Time 06:31)



8008 1972 800KHZ

first 8-bit µP

18-pin IC

16 KB

No. of Address lines = 14

$2^{14} = 16KB$

No. of data lines = 8

8080 1973 2MHZ

8-bit

40-pin

$\frac{1}{2\times10^6} = 0.5\,\mu sec$

64 KB

No. of address lines = 16

No. of data lines = 8

8085 $\boxed{40\text{ pin}}$

8-bit µP

1976

3MHZ

No. of address lines = 16

" " data " = 8

So, the Intel's the first 8-bit microprocessor is 8008; so which is I mean developed in 1972. This is Intel's first microprocessor; first 8-bit microprocessor. As I have told 8-bit microprocessor means the size of ALU is 8-bit and this 8008 can operate at frequencies of 800 kilohertz. And the memory capacity of this one is 16 kilobytes; that means, we will be having address lines of 14 because 2 raised to the power of 14 is 16 kilobyte. And number of data lines of this 8-bit represents the number of data lines is 8 and this is the clock frequency the important parameters of any microprocessor.

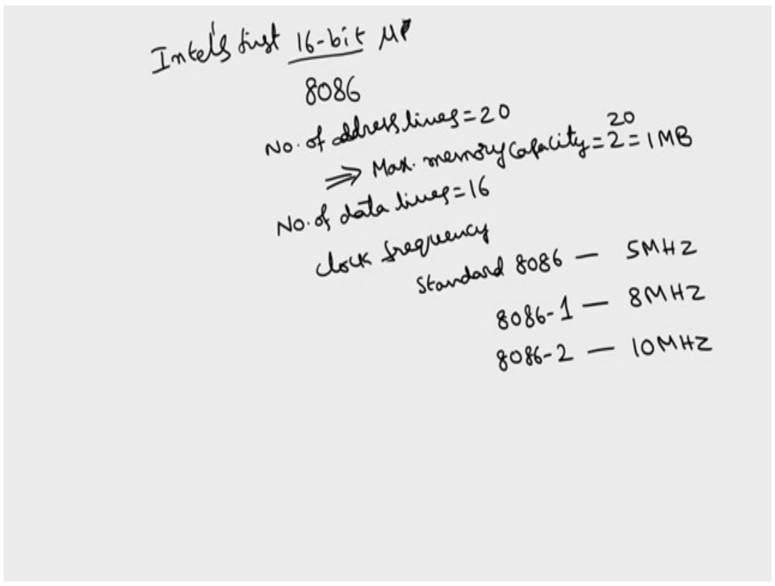
So, how many address lines, how many data lines, what is the clock frequency? So, here are the three important parameters and then this is actually of course, this is I mean 18 pin device this was 18 pin IC. So, later as the technology developments; so the next 8-bit microprocessor only which is 8080; so this is developed in 1973; this is also another 8-bit microprocessor only. So, the difference between 8008 and 8080 is the clock frequency here is 800 megahertz whereas here the clock frequency is 2 megahertz.

Means the speed of this 8080 is more when compared with 8008 because here one clock cycle will take 1 by 2 megahertz. So, this is equal to 0.5 microseconds whereas, here one clock cycle will take 1 by 800 kilohertz which is I mean more than this 0.5 microseconds. And another difference is here we will be having 16 kilobytes of the memory only whereas, here this can connect to 64 kilobytes of the memory that is the number of address lines are here 16. So, 2 raised to the power of 16 is 64 kilobytes.

The number of data lines here is because this is also 8-bit; same as that of 8008. So, the major advantage of this one is speed is more; as well the memory capacity is more. So, next same 8085 microprocessor which is one of the famous microprocessor I mean 8-bit microprocessor, this is also 8-bit; 8085 and also another 8-bit microprocessor developed in 1976.

So, the major difference between 8085 and 8088 is see here this clock frequency of 8088 is 2 megahertz; whereas, 8085 is 3 megahertz. So, one clock period is approximately 330 nanoseconds. And the memory capacity is same the remaining things out there only the speed of this one is more. So, all these 8080, 8085 all these are 40 pin devices. So, number of address and data lines are same; number of data lines will be 8. So, the only difference is here the speed of 8085 is more when compared with 8080.

(Refer Slide Time 10:53)



Intel's first 16-bit μP

8086

No. of address lines = 20

$\Rightarrow$ Max. memory capacity = 2^{20} = 1MB

No. of data lines = 16

clock frequency

Standard 8086 — 5MHz

8086-1 — 8MHz

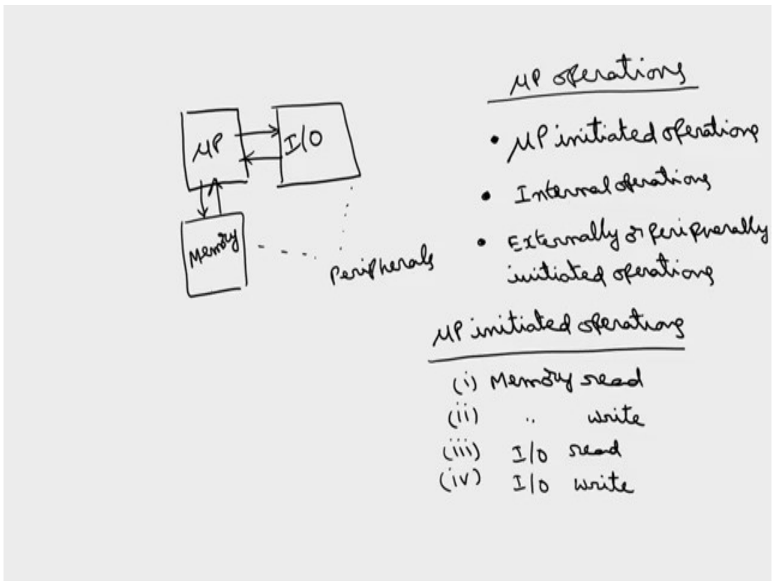
8086-2 — 10MHz

Then the Intel's first 16-bit microprocessor is 8086. So, in this course; so we are going to concentrate on 8086. So, this is Intel's first 16-bit microprocessor. So, this will be having the major differences between 8086 and 8085 is; one is the size of ALU here is 16-bit and the number of address lines in 8086 are 20; implies the maximum memory that can be connected to the 8086 is maximum memory capacity is 2 raised to the power of 20 which is 1 megabyte; this is one difference and the number of data lines are 16.

And the clock frequency is here the speed of 8086 compared with 8085; clock frequency we have three versions; the standard 8086 operates at 5 megahertz clock whereas, the version 8086-1 operates at 8 megahertz clock and the version 8086-2 operates at 10 megahertz clock. So, in this way the 8086 is more popular than 8085 which is 8-bit microprocessor.

So, in this course basically we are going to concentrate on 8086; the architecture, the addressing modes the; I mean instructions, the programming and interfacing to the peripheral devices; so, coming way for these microprocessor operations.

(Refer Slide Time 13:04)



So, the block diagram of this microcontroller; as I have told this is microprocessor IO and memory. This IO and memory are called peripherals. Without this memory and IO, the microprocessor cannot perform any function. Now, this is you have to I mean connect memory to the microprocessor. The technical work used to connect this memory to microprocessor is called interfacing.

So, how to interface the memory to the microprocessor? How to interface IO to the microprocessor? So, this also we are going to study in this course. So, coming for these microprocessor operations; so what are the operation that can be performed by the microprocessor? So, the microprocessor operations can be broadly classified into three categories.

So, one is microprocessor initiated operations there are some operations which will be initiated by the microprocessor itself; microprocessor initiated operations. I will discuss each I mean operations in detail; second one is the peripheral can interface can initiate the operation or there are some internal operations which does not need any communication with the peripheral devices.

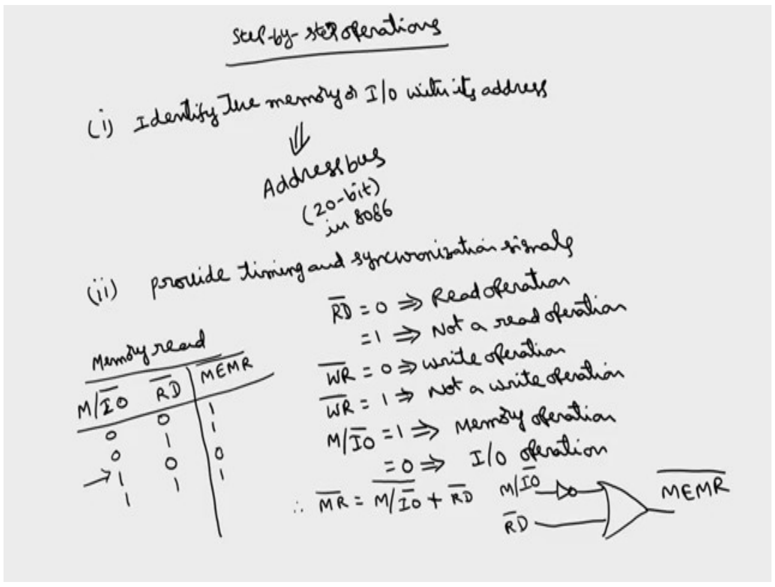
So, there are some internal operations which takes place inside the microprocessor and there are some operations initiated by the peripheral; externally or peripherally initiated operations. So, these are the basic microprocessor; you take any general microprocessor. So, these are the only three operations performed by the microprocessors. There are some microprocessor initiated operations there are some internal operations there are some operations initiated by the external devices or the peripherals which are memory and IO. ok.

So, coming for the first one, microprocessor initiated operations; here there are basically four microprocessor initiated operations. One is the microprocessor can read from the memory; memory read operation. Because this memory is read write memory; so you can read the data from the memory or you can write the data into the memory. So, the microprocessor wants to read the memory; memory read operation.

Second one is memory write operation; microprocessor can write some data into the memory. Similarly, IO read the microprocessor can read some data from the IO device. In fact, IO means it is only input device from input device it will read and IO write the microprocessor can write some data into the IO device and basically writing will be into the IO; I mean output device reading will be from input device. So, these are the three basic operations that will be initiated by the microprocessor.

Now, the question is what are the I mean hardware elements or what are the circuitry that is required by the microprocessor to perform these operations. So, in order to perform these operations the microprocessor need to I mean follow three steps.

(Refer Slide Time 17:25)



Step-by-step operations

(i) Identify the memory or I/O with its address

$\Downarrow$

Address bus (20-bit) in 8086

(ii) provide timing and synchronization signals

$\overline{RD} = 0 \Rightarrow$ Read operation

$= 1 \Rightarrow$ Not a read operation

$\overline{WR} = 0 \Rightarrow$ write operation

$\overline{WR} = 1 \Rightarrow$ not a write operation

$M/\overline{IO} = 1 \Rightarrow$ Memory operation

$= 0 \Rightarrow$ I/O operation

Memory read

$M/\overline{IO}$	$\overline{RD}$	$\overline{MEMR}$
0	0	1
0	1	1
$\rightarrow$ 1	0	0
1	1	1

$\therefore \overline{MR} = \overline{M/\overline{IO}} + \overline{RD}$

So, in order to perform any of these operations the step by step procedure that has to be followed by the microprocessor to perform this memory initiated operations are. So, the first one is the microprocessor has to identify the memory or IO with its address; identify the memory or IO with its address. This is something like; so if you want to I mean deliver some item to a house in a street. So, initially you have to find out the house number; so this is also similar to that one. So, it has to identify the memory or IO with its address.

So, in order to achieve this operation microprocessor needs some address bus, through address bus this operation will be performed. So, we know that in case of 8086; the address bus capacity will be 20-bit address bus. So, the very first step in the microprocessor is to identify the memory or IO with its address. So, after identifying this memory or IO; then it has to generate a control signal, provide timing and synchronization signals; such as whether it has to perform the read operation or write operation.

So, we have two control signals read bar; so 8086 is having two control signals. There are some other control signals, but for this particular operation there is read bar signal; if read bar is 0, it implies read operation because this is active low signal. If read bar is 1, it is not a read operation. Similarly, write bar is 0 implies write operation; write bar is 1 means not a write operation.

Similarly, there is one signal; so we can perform read or write operation from either memory or IO. So, how to distinguish between the memory and IO? Further there is one control signal in 8086. So, I am discussing only 8086 signals only here, but this procedure is common to any microprocessor. So, there is M by IO bar signal; there is a signal called M by IO bar signal. If this is equal to 1; the operation refers to a memory; memory operation; if this is equal to 0 implies it is IO operation.

So, using these three control signals; one is read bar, write bar, M by IO bar; we can perform all the four operations that we have discussed in the last slide; that is memory read, memory write, IO read, IO write. Then how to perform memory read? If you want to perform memory read? So, what will be this one? This should be 1 and this should be 0; then only you can perform the memory read operation. So, for that we can generate the memory read signal; so memory read signal is not directly available from the microprocessor, but using these signals we can generate the memory read only for signals.

So, how to perform memory read operation? So, this depends upon read bar or M by IO bar; M by IO bar is one control signal and read bar; to generate memory read bar; memory read bar. So, I am generating memory read also active low signal; memory read bar is 0 means its memory read operation; if it is equal to 1, it is not a memory read operation.

So, there are four combinations here 0 0; so if M by IO bar is 0 read bar is 0. So, is this the memory read operation? If it is memory read operation, the memory read bar should be 0 otherwise memory read bar is 1. So, you see here this is M by IO bar is 0 means this is IO operation, not memory operation. So, we have to make this memory read bar as 1 so that it is not a memory read operation.

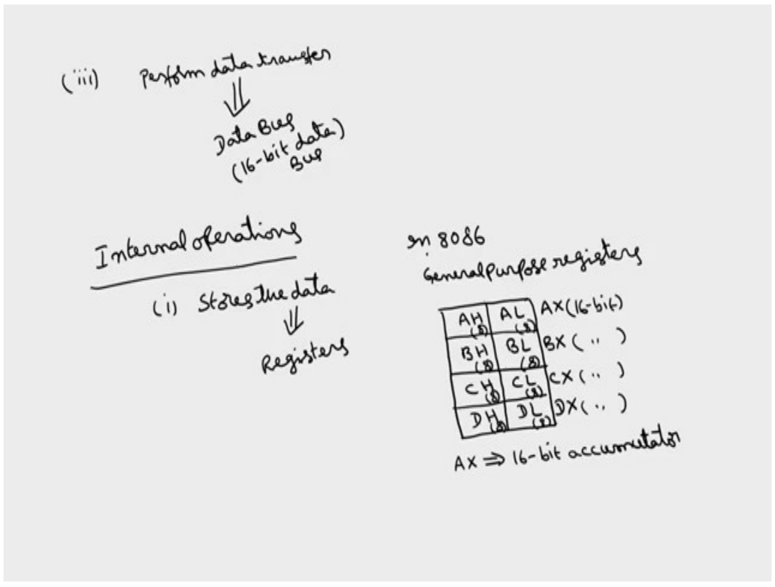
Similarly, if I take 0 1; still this IO bar is 0 means, this is IO operation. So, it is not a memory read operation; so memory read bar is 1. If this is 1 0; so, this is 1 means memory operation, read bar is 0 means read operation. So, you have to perform memory read operation; so memory read bar should be 0. Similarly, 1 1 which is memory only, but not read operation. So, memory read operation memory bar will be 1. So, using these three signals; so what is the logic circuitry required to generate memory read bar? So, the memory read bar logic expression will be; memory read bar is equal to, so this is 0; so we can have M by IO bar is 1. So, M by IO bar plus RD bar; RD bar is 0 means you have to take RD bar.

So, using these signals you can generate memory read bar if memory read bar is a 0; memory read bar is 0, so then this will be 1. So, memory read bar is 1 plus anything is 1 ok. So, similarly for these two combinations memory read bar is equal to 1 and for these two operations M by IO bar is 1; so 1 bar becomes 0. If read bar is also 0; then only this MR bar is 0; otherwise MR bar is 1.

So, if I substitute these all four combinations for only this particular combination; we can find MR bar as 0, otherwise MR bar is 1. So, this is logic circuitry that is required is; you take M by IO bar, invert; then read bar, you directly give to the OR gate. This is M by IO bar, then read bar these two are available from the microprocessor and you can generate memory read bar; I recall MEMR bar; here also you can write MEM.

Similarly, you can generate memory write bar; the difference is here in place of read bar, you will be having write bar; so similar type of the circuit will be there. Similarly, you can generate IO read bar; IO write bar. So, this is how; so we can perform all the four operations. First, we are identifying the memory (Refer Slide Time 25:46) with the address, then you are providing timing and control signals.

(Refer Slide Time 25:46)



Then the third operation will be-perform the data transfer. So, this will be performed by a data bus. So, in case of 8086 we have 16-bit data bus. So, in order to perform the very first operation which is microprocessor initiated operations; such as memory read, memory write, IO read, IO write. So, we need address bus, we need some control signals, we need some data bus. So, finally, after discussing about all these operations I will draw the overall architecture of 8086.

So, in the second operation is internal operations; as I have told there are three basic microprocessor operations; one is microprocessor will initiate the operations, second one is there are some operations which takes place inside the microprocessor itself. So, what are the operation that takes place inside the microprocessor? What are the internal operation? One is stores the data; so in order to store the data, what is required? What are the hardware elements required to store the data inside the microprocessor? So, for this we require registers.

So, registers are basically used to store the data. So, register is nothing, but a combination of the flip flops; each flip flop is capable of storing 1 bit of the information. So, here if you want to store 16-bit of the information; we require 16 flip flops connected together D flip flops, then if you want 8-bit; so 8 flip flops you have to connect. So, you might have studied these registers in your digital logic course. So, we require some registers; so in 8086 what are the different types of the registers available?

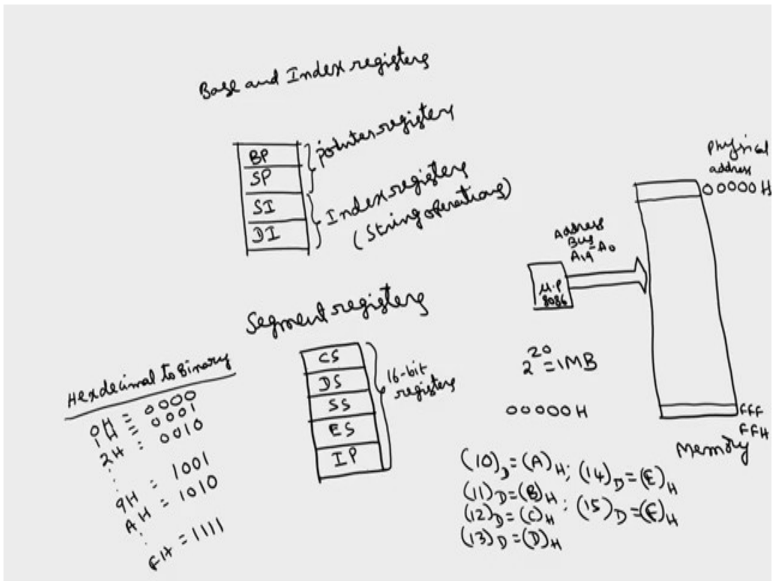
So, there are two types of registers available in 8086. So, one are called general purpose registers; they are basically we have AH, AL; BH, BL; CH, CL; DH DL. This is register array; this AH, AL combination is called AX and this combination is called BX; this is called CX, this is called DX. We can store either 8-bit of the data or 16-bit of the data. So, even though 8086 is a 16-bit microprocessor; it can perform 8-bit operations also; so in case of data is 8-bit, we can use either AH or AL. This AX is 16-bit; so all these are 16-bit X will be all 16-bits. In this AH is the higher order 8-bits; so each one here is 8-bit; this 8 represent this is 8-bit.

The higher order 8-bits of the AX will be stored in AH, the lower order 8-bits will be in AL. Similarly, higher order 8-bits of this BX is in BH, lower order will be in BL. The higher order 8-bits of CX is in CH and lower order will be in CL; similarly, here and here. So, this 8 represents the size of this particular register. So, for 8-bit operations; we will use AH, AL, BH, BL, CH, CL, DH, DL whereas, for 16-bit operations we will use AX, BX, CX, DX. Here this AX will act as accumulator; 16-bit accumulator.

Later, we are going to show that in case arithmetic logic unit. So, if you want to perform any arithmetic logical operation; we require two operations, two operands; two data. So, one of the operand has to be taken into the accumulator. So, I will discuss this while discussing about the arithmetic logic unit. So, this AX will act as a accumulator.

So, the remaining BH, BL, DH, DL or BX, CX, DX; they can be used as general purpose registers to store any data. And there is some special purpose also CX will be used as a counter in some applications and DX will be used in some division operations. So, the specialized applications of this BX, CH, DX will be clear while discussing about the instruction set of 8086. So, these are called general purpose registers; there are some other registers called base and index registers.

(Refer Slide Time 31:06)



So, they are called four base and index registers, two base registers and two index registers, they are all 16-bit registers. So, one is called base pointer BP; start pointer SP, source index SI, destination index DI. So, the base pointer I mean a pointer register; these two are called pointer registers; they will be used in stack applications and also some address calculation applications. And, these two are index registers, they are used in some string operations; I will discuss what is meant by string operations later; they are basically used in string operations, they are used for address calculation.

So, in case of any physical address calculation or stack related. So, stack pointer is again; this is same as 8085 stack pointer. So, stack pointer always address to the stack top; so I will discuss that details in the; I mean while discussing about this architecture, as well as the addressing modes and the instructions. And, there are some other registers called segment registers; these are very important registers in 8086.

We have four segment registers and along with this segment register which is called code segment, data segment, stack segment, extra segment and along with this; there will be one instruction pointer. So, this instruction pointer is similar to the program counter in case of 8085. So, the total memory as I have told the microprocessor 8086 is having 20 address lines; so 20-bit address bus is there A19 to A0; this is the symbol for bus; the maximum memory that can be connected to this 8086 is 2 raised to the power of 20.

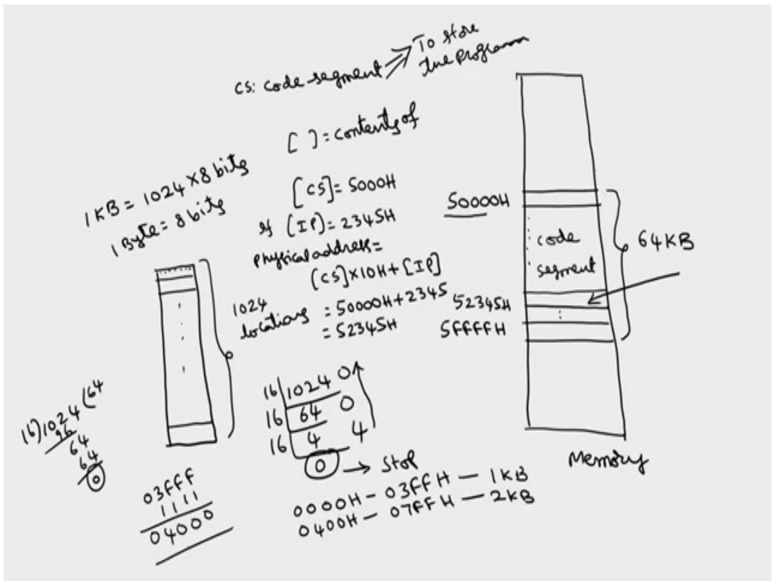
So, the maximum memory capacity is 2 raised to the power of 20; which is equal to 1 megabyte. And the physical address of each memory location; 1 megabyte means? If you start with 0000H; normally the memory will be addressed by using hexadecimal numbers. So, you know this hexadecimal number system; so 0 to 9 is same as decimal and 10 will be represented by 10 decimal is equivalent to A in hexadecimal. This D stands for decimal, this hexadecimal and 11 is B, 12 is C, 13 is D, 14 is E and 15 is F.

So, in binary form 0 to 9 is same as this is 4-bits B, C, D; so if I take the B C D numbers 0 is nothing, but 0000H. How to convert the hexadecimal number into binary number? Hexadecimal to binary conversion; if it is 0H; the binary we will have to write 4-bit binary equivalent this is binary. 1H will be 0001; 2H will be 0010 so on up to; 9H will be 1 0 0 1; A in hexadecimal will be 1010; 10, similarly F means 1111.

So here if I take the starting address; 20 address lines are there. So, 20 address lines 20 zeros, the starting will be having all address lines will be zeros and ending will be having all ones, 20 ones. Here the address of this one is 20 zeros; the address of this one is 20 ones. So, in hexadecimal number system, this will become 5 zeros H because each 0 is equal into 4-bits 5 into 4; 20 bits. So, the starting address of this memory is 00000H and the ending address will be FFFFFH; 5 F's; each F is equivalent to 4-bits; so total 5 Fs. So, each I mean F is 4-bits, total 20 bits all 20 ones, So, this is starting and ending address; this is called physical address.

Physical address of this 8086 microprocessor is 20 bit because here the segment registers are all 16-bit registers; all segment register as well as instruction pointers are 16-bit registers. Even though this I mean address is; physical address is 20 bit; so in the processing we are going to use only 16-bit addresses only. So, how to generate these physical address from the effective address?

(Refer Slide Time 38:02)



So, the calculation for these addresses is in case of code segment; this is code segment. So, the entire memory can be divided into segments; the entire 1 megabyte of the memory can be divided into segments of the maximum capacity 64 kilobytes. So, if I take a code segment with 64 kilobytes of the memory; out of this 1 megabyte, suppose if I use from here to here; this is one location, this is another location, this total capacity is 64 kilobytes. So, the starting addresses is some 50000 H. So, in case of 64 kilobytes; the ending address will be 5FFFF H.

So, how to find out this? This is code segment of the memory. So, in general 1 kilo means in a digital terminology; 1 kilobyte means 1024 by 8-bits; 1 byte is 8-bits. So, 1 kilo means it is not exactly equal to 1000, but it is 1024. So, the meaning of this one is 1 kilobyte means; so we will be having 1024 locations; this is 1, 2 so on up to; so this will be having 1024 locations and each location is capable of storing 8-bits of information. So, this can store the 8-bit, this can store 8-bit; so this is the meaning of this 1 kilobyte.

So, if I convert these 1024 into hexadecimal; 1024 is decimal, if I convert this 1024 into hexadecimal. So, in order to convert a decimal number into hexadecimal you successively divide with 16 till the quotient is 0; 16, so this will be 16; 15 times. So, this will be 16, if we want to divide by 1024; 1024 by 16; 16; 6s are 96. Remainder is 6, 4; 16 4’s are; so 16; 64 times this is 0 and 16; 4s 0; 16 0’s; 4; so you have to stop here.

So, you see the way to convert this decimal to hexadecimal number. So, this is 16; 64’s are 1024; so reminder is 0, 16 4’s are 64, reminder is 0; 16 0s, reminder is 4. So, totally this will be 4 0 0. So, if you want to start with 0 0 0 0 H. So, because this is 4 0 0 0; 1 less than this one is called 3FF; for these 3 FFFH, if I add 1; F is 15, maximum number.

So, in case of decimal 9 is the maximum number; if I add 1, it becomes 1 0. So, this F plus 1 becomes 1 0; this means 1 0, this is 1 0 4. If you start with 0; so you have to take the value which is 1 less than this 400H; so this 0000H to 03FFH is called as 1 kilobyte. If you want 2 kilobytes; the next address is 0400H and if you add 3FFH, this is 07FFH, this will be 2 kilobytes.

So, starting from 0000H to 07FFH will 2 kilo, up to here this is 1 kilo. So, for this again if we add another 7FFH for the next address, you will get 4 kilobytes. So, like that if I take 64 kilobytes that we have to add FFFF H is equivalent to 64 kilobytes. If starting address is this, to get the ending address of 64 kilobytes; you have to add FFFF H. So, this is physical address is 20 bit.

Now, code segment stores the first 16-bits of the starting of the code segment; if the code segment starting addresses is this, the contents of code segment register. This is the symbol for the contents of; this is contents of, in microprocessor terminology this bracket represents contents of. So, contents of code segment register will be 5000 H; that is the starting 16-bit address of the starting of the code segment; then to access any bit; so we will take the instruction pointer; suppose if instruction pointer is having address of some 2345 H.

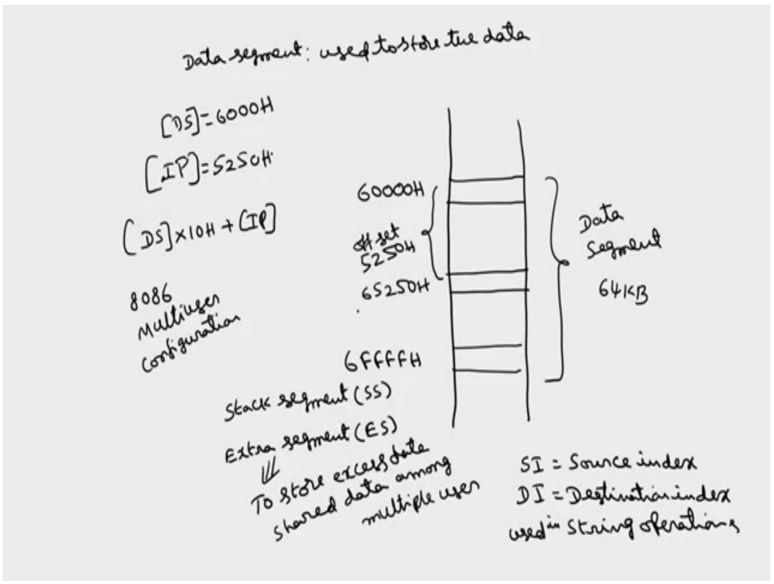
The physical address from where we are going to either read or write the data; this is the memory can be obtained by using a physical address, this is called effective address. Physical address is given by CS code segments; data into 10H plus contents of instruction pointer. So, this will become 5000H into 10H means; 50000H plus instruction pointer is 2345. So, this will be 52345H. So, somewhere here will be having say 5; 52345 H.

To access the data from here whether you have to read or write the information. So, you have to take 5000; the starting address, this is the starting 16-bits of this code segment; you have take into code segment register. Then the offset from here; from here to here what is the offset? What is the difference between this address and this are 2345H, that you have to take in instruction pointer.

So, this is the way to store the physical I mean to compute the physical address knowing the code segment register and instruction point registers. So, basically code segment register stores the programs; the purpose of the code segment register is to store the programs. In case of 8085, only 64 kilobytes of the memory is there; within that 64 kilobytes of the memory we will store the program, as well as we will store the data. whereas, in case of 8086; we can have the separate segments.

To store the memory; to store the I mean program, we have code segment; to store the data we have data segment as well as extra segment. So, you see about the first register which is code segment register which holds the starting 16-bit address of the; the particular code segment. So, the code segment size can be; it can go up to the 64 kilobytes of the memory.

(Refer Slide Time 46:04)



And, the second segment register is data segment, as the name implies this data segment used to store the data; again the same thing. So, if I take this data segment out of this memory; if I define from here to here as data segment, we can go up to the maximum of 64 kilobytes. If the starting address is something, lets say 60000H ending address of 64 kilobytes will be 6FFFFH. To read the data or to write the data into this data segment; this is basically used to store the data, to read or write the data; we can call instead of store we can write we can store the data after that we can either read or write the data.

So, to perform either read or write operation; again you have take the data segment register holds the upper 16-bits of the starting out of that particular segment; starting of that particular segment; so 6000H. If we want access the data from some location whose offset; is say some 5250. So, this address will be now 65250; the difference is called offset, that offset will be hold by instruction pointer.

So, now if we want to compute the physical address; the same information data segment into 10H plus contents of instruction pointer. So, there are some advantages of this memory segmentation; one is we can access the four segments which is total of maximum of 64 kilobytes here; so, 4 into 64 kilobytes; so 256 kilobits of the data can be accessed simultaneously; that is one advantage of this segmentation.

So, another way is another advantage of this one is; we can use this; this same memory can be accessed by multiple users. So, this microprocessor 8086 can operate in maximum mode configuration which is multi processor configuration; multi user configuration. 8085 is only single user; only one user can program that particular 8085; whereas, 8086 is we can have multi users; more than one user can access the same 8086. So, this memory segmentation enables the 8085 to access by multiple users; that is another advantage of the memory segmentation.

So, one more advantage is using the 16-bit address of the segments; we can access the 20 bit physical address. So, these are some special features of this 8086 compared with 8085, where segmentation is not possible. So, next segment register is this is data segment register and we have stack segment register. So, stack is basically; so the same definition as that in 8085. So, stack is a temporary memory; so during this either sub routines or the interrupt operations. So, some data need to be stored in temporarily in some locations that will be stored in stack. So, I will explain this stack; I mean register and stack operations in detail while discussing about the architecture.

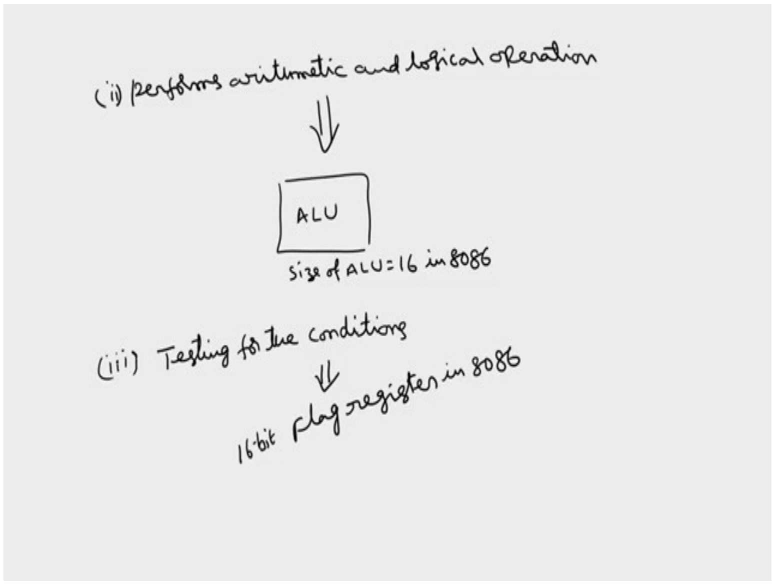
So, stack segment same; this holds the 16-bit starting with 16-bit of this; this I mean stack segment register in hexa similar manner. There is one more called extra segment; extra segment basically stores the excessive data. There is a data segment which can store 64 kilobytes of the data. If the data is in excess of 64 kilobytes; that excess data can be stored in extra segment. Or as I have told in multi user configuration, if two users want to I mean share a common data that common data also can be stored in extra segment.

So, extra segment can be used for storing the excess data and also you can store the shared data; shared data among the multiple users. And, also in extra segment will be used in this string operations; a string is if we want to transfer a huge amount block of data from one location to another location that is called string ok. So, in that case also we can use this extra segment.

So, in extra segment we will use this offset will be stored by source index and destination index; SI and DI registers as I have discussed in the earlier slide. So, there are two register called source index and destination index. So, I will discuss this in detail while discussing about the instruction set; this is source index and destination index. So, these are used in string operations; where a block of data has to be moved from one location to another location or you have to I mean process a block of data; the type of operation called string operations. So, in these string operations SI and DI will be used along with this extra segment register.

So, you have the different I mean registers that are available in 8086. So, all these registers are required; so we are discussing about the second microprocessor operation which is internal data operations.

(Refer Slide Time 52:37)



(ii) performs arithmetic and logical operation

ALU

size of ALU = 16 in 8086

(iii) Testing for the conditions

16-bit flag register in 8086

So, internal data operation; the first operation stores the data, second internal data operation is: performs arithmetic and logical operations; this is the internal operation. As I have told in the first very first slide; so the ALU is a part of a microprocessor; ALU and control unit is called microprocessor.

So, in ALU we can perform various arithmetic operation such as addition, subtraction, increment, decrement. We will discuss in detail those operations in the instruction set and logical operation such as rotating the data, shifting the data. So, there are some operations like compliment operation; so all these operations will be performed. So, to perform this operation, the microprocessor needs hardware device called ALU.

So, in case of 8086; the size of ALU is 16-bit that is why it is called 16-bit microprocessor; means in order to perform 16-bit addition or 16-bit; any 16-bit operation, this will take only 1 clock cycle. This is the second operation which is performs arithmetic and logical operations.

Third one is testing for the conditions; if you want to check between the two numbers which one is larger number. So, in such applications we have to test some conditions. So, in order to test conditions, the microprocessor requires a register called which is flag register. So, in 8086; the flag register length will be 16-bit, there is a 16-bit flag register in 8086. So, even though the length of this flag register is 16-bit; there are only 9 different flags.

So, I will discuss that flags and what is the condition for each I mean flag to be set or reset that in the next class.

Thank you.