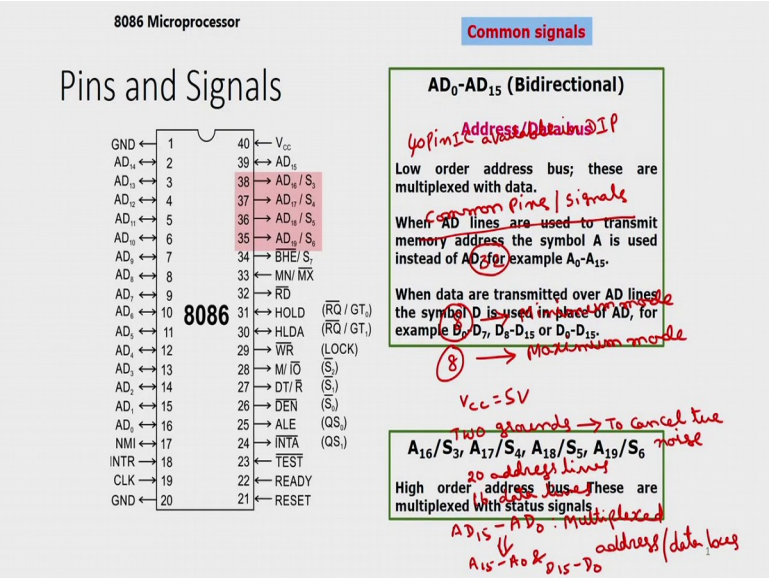
Microprocessors and Interfacing
Prof. Shaik Rafi Ahmed
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati

Lecture - 04
8086 Common and Minimum Mode Signals

So, last class we have discussed about the 8086 architecture. Now we will discuss about the different pins and the functions of the different pins. Of course, we have discussed most of the functions already while discussing about the architecture and the microprocessor operations.

(Refer Slide Time: 00:46)



So, coming for this 8086 is a 40 pin IC available in dual in line package, is a 40 pin IC available in dual in line package. Dual in line package, means; half of the pins will be on one side, half of the pins on the other side.

So, out of this 40 pins so in fact, this actually 48 because these 8 pins, they are exclusively for minimum mode or maximum mode, at a particular time we can use either this 8 pins or this 8 pins only. So, there are some common signals or common pins and signals. There are common signals, they are 32 total and 8 exclusively for minimum mode and 8 exclusively for maximum mode. As I have discussed yesterday; in minimum mode, it is a single user; single user this configuration whereas, maximum mode is multi user.

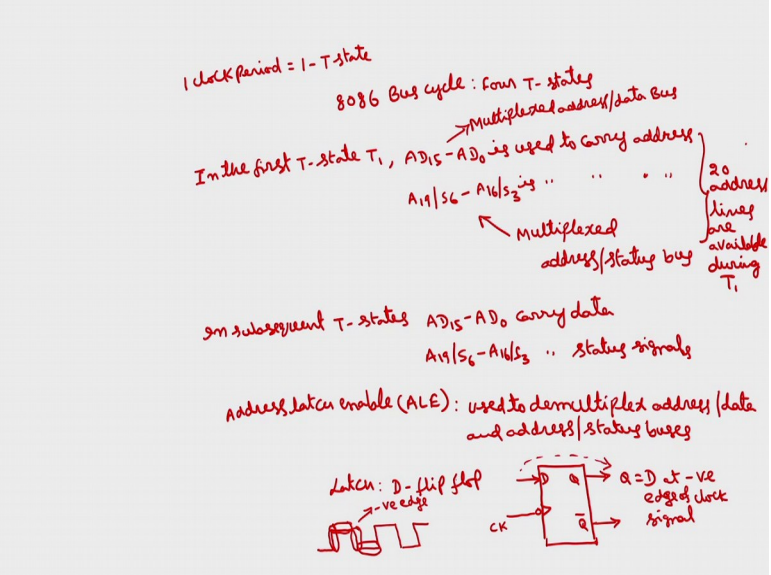
So, first I will discuss about the common pins. So, if you come for this common pins so, we require VCC and ground for any IC. So, we have one VCC which is equal to 5 volts and one specialty of this 8086 is this is having two grounds. So, the reason for having the two grounds is to cancel the noise. So, remaining are, there are we know that microprocessor 8086 having 20 address lines and 16 data lines.

The lower order 16 address lines are multiplexed with 16 data lines and that will be available as AD15 to AD0. This is multiplexed address data bus; this is called as multiplexed address data bus. The reason for multiplexing this signals is we can reduce the number of pins. As the number of pins reduces, the area reduces, power reduces ok.

So, even if you multiplex this address and data bus the function of the microprocessor will not be affected that is the another condition. So, here this will be actually if I split into two parts, this is A15 to A0, which is the lower address 16 bits of address and D15 to D0; the entire data bus 16 bit data bus.

So, when does this multiplexed bus will be used for carrying the address, when does this multiplexed address bus will be used to carry the data? So, normally in the first T cycle; if we take this the entire micro processor operation so, here we have lower address bus. So, whatever I have discussed is already written here. So, when does this particular I mean multiplexed address data bus will be used for carrying the address is during T1.

(Refer Slide Time: 04:49)



So, T is actually T state, 1 clock period of any microprocessor is called as 1 T state. So, if I take the microprocessor cycle, it takes 4 T states 8086 bus cycle which is called as bus cycle. This will take 4 T states; I will show what are the T states.

So, if I want to read or write the data so, we required the address and the data buses that is we have discussed in the microprocessor operations. The first operation is microprocessor initiated operations which are memory read, memory write, IO read, IO write. So, for all this operations what is required is the microprocessor requires the address bus, data bus and some sort of control signals.

So, in that also there is a three step procedure to perform any of these four operations which we have discussed in the earlier classes. So, the first step is we have to identify the device with this address; device can be memory or IO. So, you have to identify the device with its address.

So, once if you identify the device with its address, then you generate the control signal whether it is read operation, write operation you generate the control signal; then third operation is third step is to transfer the data. So, among these three steps, in the first step identifying the device with its address, there only we require the address bus.

So, after that generating the control signals transfer of the data so, we do not need the address bus. That is why in the first T state so, this in the first T state, that is T1, AD15 to AD0 is used to carry address information. Because the first step in any microprocessor initiator operation is either memory read, memory write, IO read, IO write. So, the very first step is you have to identify the device with its address. So, we need address bus.

So, in the first T state this will be used to carry the address. And similarly we have the other total 20 bit physical address is to be required. So, these are the 16 bits so, remaining 4 bits are multiplexed with status signals. So, those are called as A19/S6 to A16/S3 is used to carry the address. So, these are state this is called actually the multiplexed, this is multiplexed address data bus whereas, this is multiplexed address status bus.

So, during the T1 say AD15 to AD0 will carry address that is A15 to A0 and A19/ S6 to A16/ S3 carries A19 to S6 is total we will get 20 address lines during the T1, total 20

address lines will be available during the T1. So, in T1 we will send the address; we will identify the device with its address.

So, in second step is you have to generate the control signals, then the next step is you have to transfer the data. So, in subsequent T states AD15 to AD0 carry, data and A19/S6 to A16/S3 carry status signals, because here we require the status signals and data bus for transferring of the data only in the first T states, we require the address bus.

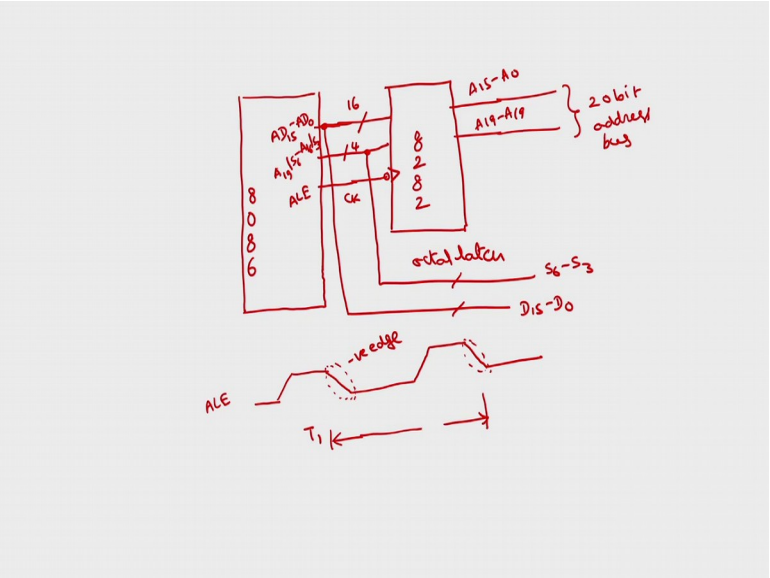
So, how this will be latched? So, for that we are going to use another signal called address latch enable, called ALE. So, ALE is required to demultiplex address data bus as well as address status bus. These are multiplexed, to demultiplex this we will use ALE used to demultiplex address data bus and address status bus. How exactly this will be demultiplexed? For that we require the latches. So, we know what is meant by latch.

Latch is some sort of D-flip flop. D-flip flop will act as a latch clocked D flip flop. So, what is the operation of the D-flip flop? So, if this is negative edge trigger or positive edge trigger; if I assumed that this is negative edge, this bubble represent negative edge trigger. This is clock, this is D, this is Q and Q bar.

So, if it is negative edge triggered flip-flop during the clock, as I have discussed in the last class also there are total four portions of the clock; this is ideal clock. So, there is one positive edge, positive level; negative edge, negative level. So, this particular flip flop will be enabled during negative edge. So, if I apply the clock signal at the negative edge, what happens? This flip flop will be enabled and whatever the data will be transferred to the output, Q becomes simply D, whatever the data here will be transferred to the output at the negative edge of the clock signal.

So, simply output is equal to input, this is called latching of the input information. So, the only condition for latching this data onto the output is the clock has to be changes from 1 to 0 which is negative edge. So, if I use 8 such I mean latches inside a IC which is called octal latch. There are some ICs available which will acts as a octal latch which is 8282. There is one IC 8282 will acts as octal latch.

(Refer Slide Time: 12:56)



As the name implies octal inside this there are D, there are say 8 D flip flops which will act as latches. So, I am not showing the 8 so, they will be having different 8 signals so, I will show these 8 signals with this symbol this represent this is multiple lines and this will be having clock which is common to all the flip flops. So, this is clock is common to all the 8 flip flops. So, we can use multiple latches here I am not showing this one so, total in 8086. Total there are 20 bus, 20 lines has to be demultiplexed. Total 20 lines are multiplexed in 8086 out of which 16 are address lines multiplexed with data and 4 address lines are multiplex with status.

This is AD15 to AD0 so, total 16 means so, I am not showing this one. There are two I mean 8282 chips here, but I am showing only single one only. So, here I will show this as 16 lines, means; this each line AD0 will be connected to D input of 1 latch, AD1 will be connected to D input of the second latch. So, like that there are 16 latches means if I use 2 I mean 2 sets of 8282 So, there are 16 D flip-flops so, 16 AD0 to AD15 will be connected to the D inputs of all the latches. Similarly so, if I use one more 8282 so, which will be having 8 latches out of which I will use 4 lines which are A19/ S6 to A16/ S 3.

So, totally there are 20 lines, means; inside this you should have 20 D flip flops so, each 8282 will be having 8 D flip flops so, we need 3 such 8282. I am not showing individually here single 8282 only I am showing. But in fact, here we require three

number of 8282. So, the clock is common to all the three 8282 and all the D flip flop inside that, but the inputs of D flip flops will be different. So, there are totally 20 different signals are there from 8086 so, one signal we have to connect to the one input of each D flip flop. That is the configuration here.

Then this clock will be applied to a signal called address latch enable. So, during the first clock cycle what happens is, I will place here address lines A15 to A0 and here address lines A19 to A6. So, that time I will make this signal ALE signal as a low to high signal this is actually practical clock, in this ALE signal.

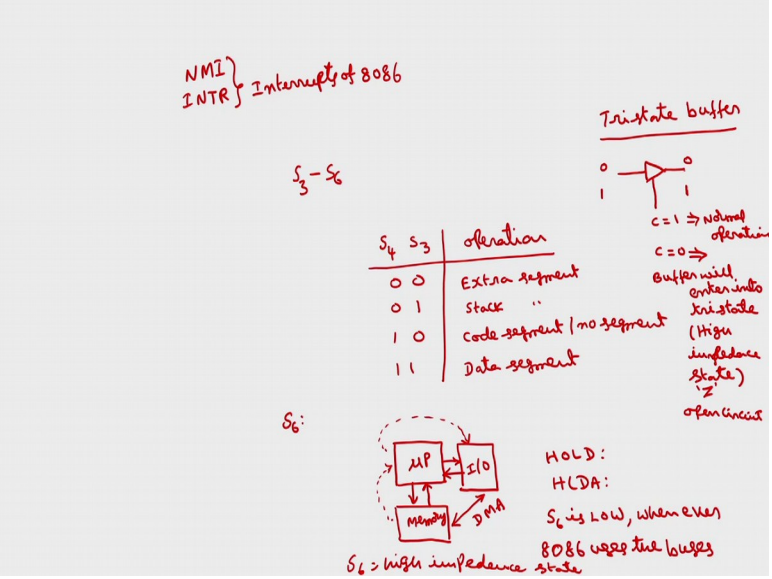
So, during this portion of the ALE signal, this is called negative edge. So, at this portion this is 8282 will be all enabled, there by whatever the input data will be transferred. That time what happens is this will be happen in the T 1 state; this will happen in the T 1 state. So, that time this multiplexed buses carries the address information. So, thereby the output will become only this will be only A15 to A0 and this will be A16 to A19. So, these total 20 bits of the address bus has latched onto the output.

So, this output will be there, this 20 address binds until the next clock cycle where the address latch enable goes from high to low. So, till this point from this point to this point so, whatever the information that is latched will be available at the output of 8282.

Now, what happens is because already I have lathed this one in subsequent clock cycles. I can use this to carry only the status, I can use this to the carry only the data. This is data line D15 to D0; this is status lines S6 to S3, this you can demultiplex.

Now, address bus is available separately, databases is available separately, status signals are available separately. So, this will be the operation to demultiplex the address data bus and then address status bus, so for that we will use ALE signal. So, this is how we can demultiplex address in data buses. So, these are the common signals, but ALE is signal which is available for minimum modes configuration, but here because this function is required so, I have discussed this ALE here itself. So, one of the other common signals sort of this, I have discussed about the ground AD15 all this up to AD0 then, if I go in this order NMI INTR. This already have discussed it in the last class NMI and INTR. These are two interrupts of the 8086.

(Refer Slide Time: 18:43)



I have already explained NMI is non maskable interrupt and this will be a positive edge triggered; INTR is a positive level trigger and but in a INTR is maskable. So, this actually have explained already while discussing about the microprocessor operations. Now coming for the other signals clock, so we know that the microprocessor requires 5 megahertz clock in case of a simple 8086. There are two versions the other versions requires 8 megahertz and 10 megahertz. This clock has to be generated using the external crystal generator. There are different ways to generate the clock signal.

So, oscillators, basically we have Colpitts oscillator, Hartley oscillator. But normally for the microprocessor to generate the clock signal for the microprocessor, we will use the crystal oscillator because of the stability, crystal oscillators are most stable. So, that is why by using external crystal oscillator will generate the clock frequency of 8086 which will be for standard 8086, 5 megahertz and for different versions we have 8 megahertz and 10 megahertz.

So, this is nothing to do externally you have to generate a clock signal is in clock crystal oscillator and ground we have already discussed VCT. So, AD16 and AD15 is part of AD this one. So, then coming for this status of signals, we have discussed about the address data bus. Coming for the state of signals, we have status signals here S3, S4, S5, S6. So, what is the function of this S3 to S6? The status signals S3 to S6, they are multiplexed with address lines.

So, the function of S3, S2, S4, S3 will be operation. If they are 00, then this is extra segment. To access the extra segment, these two status signal should be 00, 01 this is for stack segment and this is 10 code segment or no segment. And 11 is for data segment. This is the purpose of these two status signals S4 and S3.

And coming for S6, S6 is a status signal so, during the HOLD operation. So, it is also explained in the microprocessor operations. So, HOLD be the operation where which will be used in direct memory access applications. As I have told if I take the micro processor and memory IO ok, the normal way of data transfer is so, whatever the data if I want to transmit from transfer from memory to IO so, memory first it will transfer to the microprocessor; the microprocessor will transfer to the IO. This is the normal conventional data transfer scheme.

You know this is time consuming so, for that we can use a direct memory access through between this memory and IO. This is called DMA, Direct Memory Access. But in order to perform this DMA operation, the microprocessor requires the control of the buses the microprocessor bus. For that the microprocessor memory will request through HOLD, HOLD signal is requesting for the buses of microprocessor and HOLD acknowledge HLDA, the microprocessor will acknowledge and it will transfer the control to the this I mean microprocessor memory. So, during this signal S6 will be normally low, S6 is normally low means logic 0 whenever the microprocessor 8086 uses the buses.

So, microprocessor can use the buses for performing its own operations, so if the memory request for this buses through the HOLD signal. So, whenever the microprocessor is performing using this buses then S signal S6 signal will be low. So, the memory will check the S6 signal, it will understand that microprocessor using the buses. So, whenever the purpose of this buses is over; if I mean; use of the per buses is over then what happens is.

So, the microprocessor is ready to give the buses to the memory. So, in that case S6 become high or in fact, this is not high; it will go into the high impedance state. So, once this microprocessor uses the buses and the microprocessor is ready to give the buses to the memory, then it makes S6 as high impedance state.

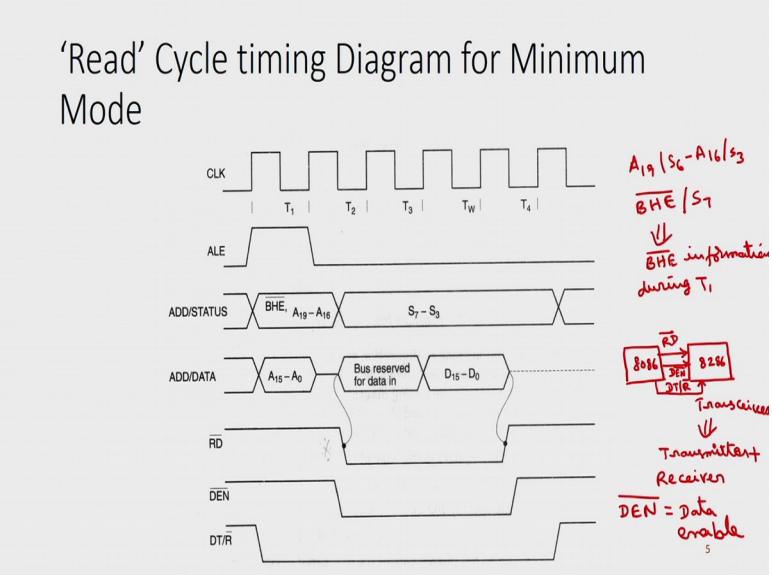
So, what is meant by high impedance state is say in digital technology, we know logic 0 and logic 1. If I take a Tri state buffer so, buffer is basically current amplifier, but voltage will be same input and output voltage will be same. So, if I take 0 if I apply 0 here output is also be 0; if I apply 1 here, output is also 1. There are two states logic 0 and logic 1. This is called two state buffer. If I want tri state, you have to incorporate one more control signal C here, this is tri state. So, if C is equal to 1 normal operation, that if input is 0 output is also 0; if input is 1, then output is 1.

So, if C is equal to 0, then the buffer will enter into tri state. Tri state in the sense is also called as high impedance state, also called as Z. High impedance means here the impedance will be high. If I take the high impedance, what happens? This almost will act as an open circuit, means; it will not draw the current from the device which is connected to this output. So, if it is not high impedance state even if it is logic 0 also it will draw some current from the I mean gate which is connected to the output of this buffer. If it is high impedance state almost this is open circuit, it will not draw any current it will not load the device which is connected to the output of this output of this buffer, that is meant by high impedance state.

High impedance means almost its open circuit; it will not draw any current so, it is not going to load the device which is connected to the output of the buffer. That is meant by high impedance state. So, here also what happens is, so after the microprocessor uses this buses during the use, whenever the microprocessor uses the bus S6 is low. After the use of buses is over 8086 is ready to give the buses to the peripheral device.

That time what it makes is S6 will become high impedance state, then the control will given to the memory. That is the status bus S6 and, then next is there is one more status signal which is called S7 or bus enable, I will come to that later. So, now coming for this the other signals such as DEN bar DT /R bar.

(Refer Slide Time: 27:39)



So, before going for this function of the other pins so, I would like to I mean like to explain about the bus cycles of 8086. As we have already discuss a bus cycle, it will take 4 T states; T1, T2, T3, T4, but there is one wait state. So, if the external device is not ready as we have discussed in the last class also. So, whenever this microprocessor is want to read the data from the peripheral, we know that the peripheral is slower when compare with the microprocessor. If the peripheral device is not ready, what the microprocessor did it will enter into the wait state until the data is ready.

So, what basically if this external peripheral device is ready, then it will take only 4 T states. As I have discussed to demultiplex address and date buses status and address buses, ALE will goes high in the T state. So, during this point the address bus and data bus will be demultiplexed, address bus and status bus will be demultiplexed, as we have discussed just now. Similarly there are two more signals called bus enable and S7 signal, S7 to S3. So, as I have told so, like the status signals, A19 to S6, A16/S3 they are multiplexed.

Similarly, there is one more status signal S7 which is multiplexed with bus high enable signal; I will come to the function of this bus high enable. So, this is multiplexed with S7. So, this multiplexed bus carries the address during the first clock cycle, you see here in the first clock cycle A19 to A16. Similarly this bus I enable will be this BHE bar/S7 is also multiplexed signal so, this will carry BHE bar information, during T1 and in subsequent clock cycle S7. Here also in subsequent clock cycle S6 to S3 which we have

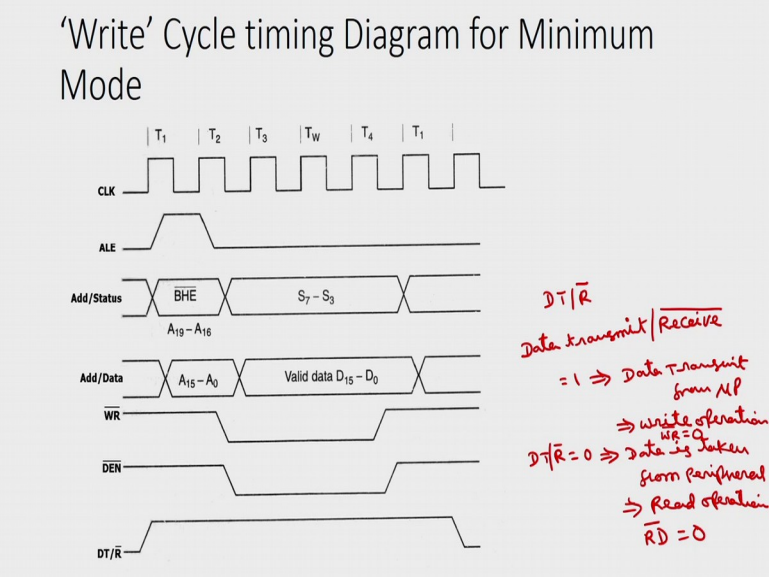
already discussed this S7 to S3 including S3 to S6 and the S7 so, in subsequent clock cycles.

So, here this address bus will be latched as we have already discussed. So there are two more signals called READ bar DEN bar DT/R bar. So, in order to read the data from external devices, so the microprocessor can read the data from external device such as transceiver. There is one device called 8286 which is called transceiver. Transceiver is the combination of transmitter and receiver plus receiver. So, in the transmitter, you have taken trans; in the receiver you have taken ceiver. So, it is called transceiver, transceiver is combination of transmitter and receiver.

So, using this transceiver this 8286 will acts as a transceiver. So, the microprocessor can read the data from 8286 or microprocessor can write the data into 8286. So, for that it uses READ bar signal, and then there are two extra signals which are called DEN bar, DT bar, DEN bar then DT / R bar.

So, now, to perform any operation whether it is a read operation or write operation this is read operation so, DEN bar should be low, but this goes the difference between the read and write is; in case of read operation DEN bar goes low. So, during this T2 cycle, it will be low up to T4 cycle ok.

(Refer Slide Time: 31:47)



So, whereas, in case of write operation so, this DEN bar is low during T2 whereas, here is just earlier than this one. So, why because in the READ operations, what happens is the peripheral device is slow device so, the microprocessor has to wait it cannot directly read. In case of write means microprocessor is writing something to sending some data to the transceiver. So, because already microprocessor is a fast device so, it will send immediately. That is why this DEN bar will be low in case of write operation, early when compare with DEN bar of read operation. I think you understand the difference between DEN bar in read cycle and write cycle.

In case of read cycle, the microprocessor has to read the data from the external device which is lower one so, the operation will be a little bit delayed. So, whereas in case of write operation, the microprocessor will send the data to the transceiver. So, because microprocessor is already fast device so, this DEN bar will be low a bit earlier than that of in write cycle sorry, read cycle.

Similarly in case of write bar write bar will be there this all this operations are common ALE. This entire thing the only difference between the read and write cycles is so, here in place of WRITE bar there READ bar will be there DEN bar will be goes low earlier when compare with the DEN bar of read cycle because the microprocessor is the fast device and DT/R bar is.

So, in case of write cycle DT/ R bar will be, DT/ R bar represents data transmit/ receiver. As the name implies this is one implies data transmit, data transmit from microprocessor which is write operation. So, in case of write operation, this DT by R bar should be high whereas, in case of read operation it has to receive, microprocessor has to receive.

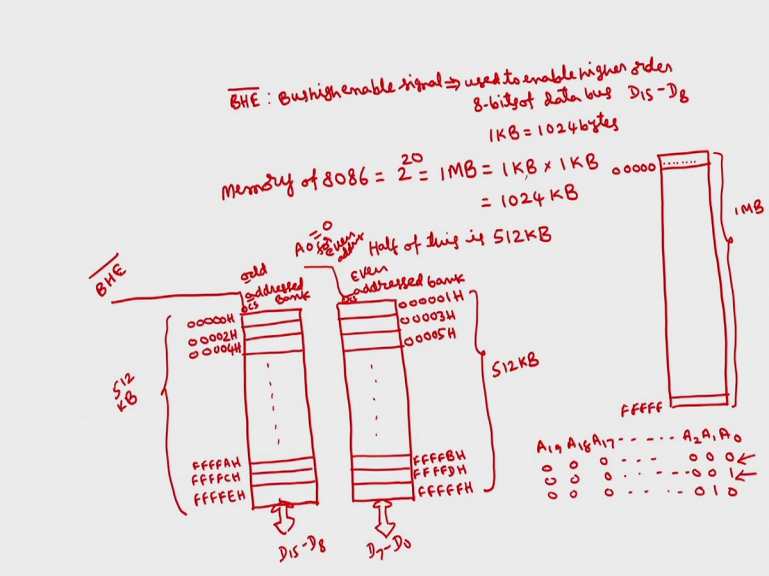
So, DT /R bar should be low implies data is taken from peripheral so; that means, it is a read operation. That is why in case of read operation, we can see the DT /R bar is low in case of write operation DT /R bar is high. And then DEN bar here you have WRITE bar signal whereas, in this case we have READ bar signal. Another difference is DEN bar will be goes slow in both the cases because data enable, DEN bar is stands for data enable; stands for data enable D for data EN for enable.

So, in any case whether you want to perform the read operation or write operation data enable should be 0. But only difference is in case of read operation because of the

peripheral device is slow, then DEN goes to low a bit delayed when compared with the DEN bar of write cycle. That is the only difference in D and DEN, but both are lows; here READ bar will be there DT/R bar will be low. So, in case of write cycle DT /R bar is high, this is WRITE bar signal whereas, DEN bar goes to low a bit earlier than that of read cycle. This is about the functions of DT/R bar, DEN bar, READ bar, WRITE bar.

So, for write operation; for write operation what happen is, for read operation READ bar should be 0. For write operation WR bar will be 0. So, this will complete the functions of DT /R bar, DEN bar, read bar and write bar four signals of the microprocessor.

(Refer Slide Time: 36:13)



Then S7 and BHE bar so, this is bus high enable which is going to enable the so, higher order bus of this signal. So, now, coming for the function of BHE bar and S7; BHE bar as the name implies bus high enable, the full form of this one is bus high enable. So, which enables the higher order 8 bits of the bus.

So, what are the higher order 8 bits of the data bus, is D15 to D8, lower order is D7 to D0. How this will be enabled? So, for that we have to discuss about the memory bank concept of 8086. So, we know that in 8086, the memory will be divided into banks. Memory of 8086 is maximum memory capacity is 2 raised to the power of 20, because 20 address lines are there relation between the maximum memory that can be connected to 8086 or in any microprocessor. And the number of address lines is 2 raised to the

power of number of address lines is the maximum memory capacity, this is nothing but 1 megabyte.

So, in fact this is in terms of the kilobyte if you want to write, this as I have told already 1 kilo is not equal to 1000 in microprocessor terminology 1 kilobyte means it is 1024 bytes. So, this can be written as 1 kilo x 1 kilo, 1 kilobyte x 1 kilobyte. If I substitute this 1 kilobyte this is 1024 kilobytes, 1 mega byte is equal to 1024 kilobytes. So, what is the half of this one? Is 512 kilobytes, half of this is 512 kilobytes. So, the total 1 megabyte which is 1024 kilobytes of the memory will be divided into, if I take this total megabyte of the memory.

So, we have 20 bit address line, the starting and ending address of this one will be, this will be FFFFF H and this will be 00000 H. So, this is total 1 megabyte memory which is 1024 kilobytes. In 8086 what happens is; so now, to enable a 16 bit data from the memory because each location is capable of storing only 8 bit of the information. This will contain 8 bits of the information, remaining of 1024 or 1 megabytes is we have 1024 x 1024 locations and each location is capable of storing 8 bits of the information. But 8086 is a 16 bit microprocessor; in some applications we need 16 bit data also. So, which we cannot read further we require two clock cycles. At a time, we can read only the 8 bits of the data so, to read the 16 bit of the data; we required two clock cycle.

So, instead of that, if you want to I mean parallelly read total 16 bit simultaneously then, memory is divided into two banks which is called as odd bank and even bank. Now what is the capacity of each bank is 512 kilobytes. So, this is 512 kilobytes, this is 512 kilobytes, the address of this one will be; this is odd address this is even address. So, what are the addresses total addresses 000? So, 00000 H, this is the starting address of the total 512 kilobytes of the memory that will goes to even bank. The second location which is 00001 will goes to the odd bank alternatively. And then the next one will be here 00002 H and here this will be 00003 H, this will be 00004 H, 00005 H and so on.

And what about the last address? FFFFF H this is odd address so, odd address will be here starting because with here it has to end here it has to end with this ok. So, this is odd address 15 say, FFFFF H will be the odd address, the one address which is before this one is FFFFE H and the one which is before this one is FFFFD H and this one will be FFFFC H, this one will be FFFFB H and this one will be FFFFA H and so on. So, this

will be contains only the contents of the even addressed locations, this will contain odd addressed locations. So, what is the capacity of this? This is 512 kilobytes, and here also this is another 512 kilobytes.

So, each memory signal will be having some chip enable signals in order to select this particular chip at all. So, we require some chip select signal. After selecting the chip as a whole, we will select the particular location that is how the memory read or write operation has to be performed. To perform with a read operation or write operation there will be some chip select signal corresponding to each memory. Normally these are active low signals, this is called chip select signal CS bar.

So, in order to select any location the first requirement is you have to select the chip. If I apply a logic 0 on this particular chip select signal that particular chip is selected after that you select the location.

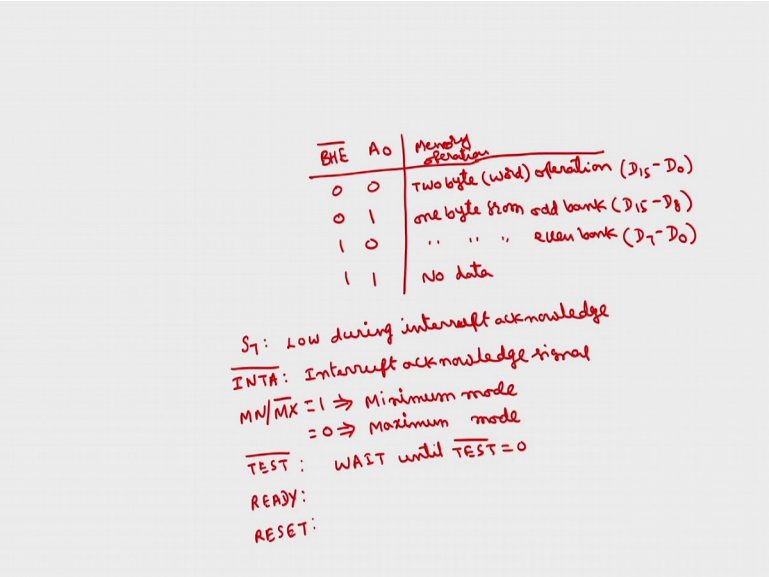
So, here this higher order bus will be supplied by this one, this will you total 8 bits this will give total 8 bit totally we have 16 bit data bus. So, this will give this total 8 bits of the data, this will give another 8 bit of the data, this will give D15 to D8, D7 to D0 overall this will be D0 to D15, 16 bits. So, in between this one this is higher order data bus. So, this will be connected to bus high enable signal BHE bar as the name implies bus high enable signal so, this will be connected to the even order even addressed bank and this will be covering connected to; so in order to select this for only the odd address.

So, you can simply give A0. Because, so we know that total 20 address lines are there A19, A18, A17 so on up to A2, A1, A0. For all odd addresses even address A if I take the starting address all 0s, the next address is all 0s; last one is 1. The next address is 0s, then this is 2 and so on. So, for even addresses A0 is 0, for odd addresses A0 is 1 ok. So, when does this will be selected? So, A0 is equal to 0 for even addressed; A0 is 0 for even addressed.

So, for A0 is 0 even address this will select so, this is this should be even addressed line; this is even addressed and this should be odd addressed, understand? So, this is odd addressed because this is even address, because A0 is connected A0 is 0 for only even addressed lines. So, for even addressed and A0 is 0 so, chip select is 0; this chip will be

selected. Whereas higher order bus this odd address so, BHE bar will be used for selecting this chip select.

(Refer Slide Time: 46:12)



Now, we can form the table BHE bar versus A0 and what is the type of memory operations. If both are 0s, what happens? This is also 0, this chip select bar is 0 means this will be selected, this chip also will selected, this will give 8 bit information, this will give 8 bit information. At a time we can read 2 bytes which is called one word, this is 2 byte operation. 2 byte is normally called as word, this is word operation because both the banks will be enabled. Because A0 is connected chip select of even bank, BHE bar is connected chip select of odd bank. So, because both are 0, both banks will be selected; one bank will give 8 bit another bank will give 8 bit overall it will be 2 byte or it will be a one word.

Now, is 01 so only 1 byte from, which bank is selected? From so, this BHE bar is higher odder which is odd from odd bank, 10, 1 byte from even bank. Both are 1s; none of the chips will be selected so, no data; entire data bus will not be enabled. So, see here BHE bar is 0 only the odd bank will be will give one data which is higher order bits, this will be lower bits. So, this if bus enable bar is 0, this will be 1 byte from the odd bank which will be higher order 8 bits, D15 to D8. And in this case D7 to D0; in this case the entire thing D15 to D0, here no bus will be selected. This is the purpose of BHE bar and A0.

Coming for one more status signal which is called S7, there is one more status signal, but this is not multiplexed with any other signal. So, this normally this will be low during interrupt acknowledged signal, interrupt acknowledge. So, whenever the microprocessor is executing some program as we have discussed in the last class, so any external device can interrupt and it can ask the microprocessor to execute some emergency task.

So, during that time, it will request the microprocessor through INTR or NMI. So, whenever the microprocessor receives the signal, interrupt signal; it completes the execution of the current instruction, then it will send the interrupt acknowledge signal. There is one more signal called INTA bar which is called interrupt acknowledge signal. So, whenever the microprocessor puts a logic 0 on this; this active low signal; that means, the external device understands that the microprocessor is going to in service the interrupt the routine of that particular interrupt ok.

So, during this time interrupt acknowledge cycle, this S7 will be kept low; otherwise it can be tri stated, so that it will not carry, it will not I mean draw any current, it will not the overload the device which is connected to the S7 pin. So, these are the different pins. So, now what are the pins that are left? So, here we have discussed about these pins S3 to S6, BHE bar/S7, then there is one pin call MN /MX; MN /MX bar. This as I have told there are two modes: minimum or maximum mode; MN stands for minimum, MX stands for maximum.

So, this is equal to 1 implies minimum mode, single user. To operate the 8086 in minimum mode, you have to set this pin to 1. If you set this pin to 0, the microprocessor operates in maximum mode which is multiprocessor mode. So, till now we have discussion about the all common signals and there are some signals used in minimum mode. So, we have discussed about the all the common signals here and all the common signals here up to this point.

So, I will come to this later. Then the remaining three common signals are TEST bar, READY and RESET. So, the other signals are the common signals test bar. So, test bar signal; coming for the test bar signal. So, there is a one instruction called WAIT instruction, I will discuss this while discussing about the instruction set. So, whenever the microprocessor execute this instruction as the name implies, microprocessor simply it

will wait ok. So, till which time it will be WAIT? So this microprocessor will WAIT until test bar is equal to 0.

So, as long as TEST bar is equal to 1, it will WAIT only. So, once if there TEST bar is 0, it will stop waiting; it will come out of the WAIT cycle and will perform the various tasks at a simple purpose of test signal. And READY (Refer Time: 52:01) signal we have already discussed. READY signal so, this will be used by the external device. So, whenever this READY signal is 1, the microprocessor read the data from the that device otherwise simply it will enter into the WAIT state. This I have already discussed while discussing about the externally initiated operations.

Similarly RESET whenever the microprocessor is RESET, when the microprocessor will suspend all the operations internal operation. This also already discussed. It will makes stack segment, data segment, extra segment instruction pointer to 0, then code segment is FFFFH ok. So, whenever this reset is removed, then next instruction that is to be fetched will be FFFF0 H which we have already discussed in the last class.

So, it is all about the common signals and some minimum mode signals. With functions of this maximum mode signal and remaining minimum mode signals will discuss in the next class.

Thank you.