

Microprocessors and Microcontrollers
Prof. Santanu Chattopadhyay
Department of E and EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 59
8086

We will next look into another microprocessor which is an advance version of 8085 which is known as 8086 microprocessor. So, like 8085, I said that this is one of the basic microprocessor that we have and for understanding any microprocessor architecture microprocessor microcontroller architecture it is better to start with 8085, but now a day's most of the systems that we see are based on some advanced microprocessors again those advanced microprocessors are quite complex in nature.

So, will try to see how this evolution has taken place from a 8085 towards the next higher level microprocessors and 8086 is the next step and from this 8086 the architecture remained more or less the theme remained same only the features have got modified, in the sense that the memory structure then the registers that we generally have their organisation. So, those things remain unaltered only thing the thing that changed is the may be the additional functionalities have been added to CPU like say some security features some phasing mechanism or memory management and all that. So, they have been added.

So, as a result the structure had become more complex, but essentially the access pattern or this design part. So, that remains the based on this 8086 processor only. So, that is the reason why we are looking into this 8086 processor. So, what we will do is that since we have already learnt 8085 so many a times. So, we will be just refering to 8085 and trying to see that what is the new feature that we have here ok. So, start with 8086.

(Refer Slide Time: 02:03)

Overview

First 16-bit processor released by INTEL in the year 1978

Originally HMOS, now manufactured using HMOS III technique

Approximately 29, 000 transistors, 40 pin DIP, 5V supply

Does not have internal clock; external asymmetric clock source with 33% duty cycle

20-bit address to access memory \Rightarrow can address up to $2^{20} = 1$ megabytes of memory space.

Addressable memory space is organized in to two banks of 512 kb each; **Even (or lower) bank** and **Odd (or higher) bank**. Address line A_0 is used to select even bank and control signal BHE is used to access odd bank

Uses a separate 16 bit address for I/O mapped devices \Rightarrow can generate $2^{16} = 64$ k addresses.

Operates in two modes: **minimum mode** and **maximum mode**, decided by the signal at MN and MX pins.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this is the first 16 bit processor released by INTEL in the year 1978 that is pretty old. So, first 16 bit processor. So, people thought that instead of having this 8 bit processor which is very small the capabilities may be less. So, why do not we go for 16 bit so, that our basic operation will be on 16 bit data. So, our operation will be faster and it is more powerful. So, and also you can accommodate more number of instructions.

So, if you have if you having 16 bit word size then you can accommodate more number of instructions. Like 8 bit word size we can have only 2^8 different instructions possible, but with 16 bit you can have 2^{16} . So, that is if your output size is 16 bit. So, you can have 2^{16} different instructions possible.

So, this was originally designed on HMOS technology and now it is HMOS 3 technology used. So, it use as the approximately 29000 transistors 40 pin dual in line package and 5 volt supply voltage does not have internal clock and external asymmetric clock source with 33 percent duty cycle. So, we have to give clock from the external word.

So, the from the external and then it has got 33 percent duty cycle. So, it is point so one third of the club duration, it should be in high state and the remaining time it should be in the low state, then the address is 20 bit address. So, compare to 8086 the 8085 that at 16 bit address. So, this has got 20 bit address and naturally it can access 2^{20} there is 1 megabyte of memories whereas, the 8085 at 16 bit address bus. So, it could access to 2^{16} or 64 k address bus, but this can access to 2^{20} or 1 megabyte of address

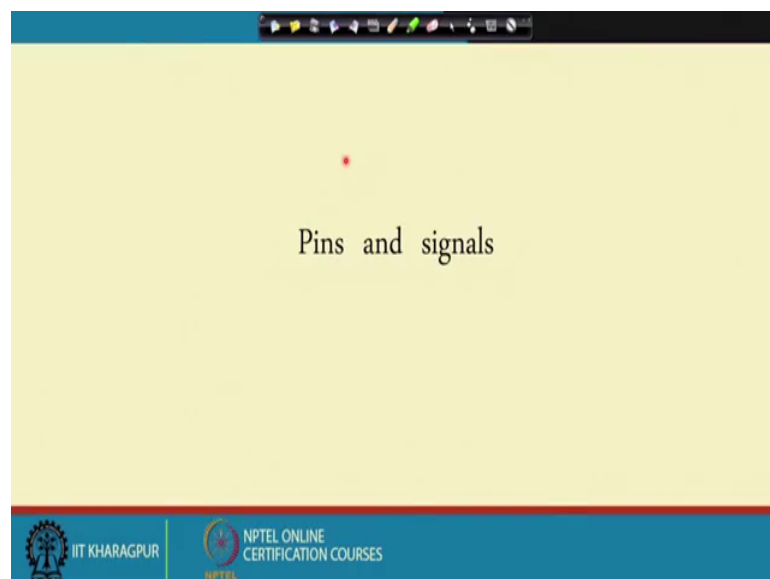
space and then addressable memory space is organized in 2 banks of 512 kb each. So, that is lower bank our even or lower bank and odd higher bank. So, address line 80 is used to select even bank and control signal BHE bar is used to select the access the odd bank.

So, we will see this control lines later, there is a separate 16 bit address for IO mapped devices and naturally it can generate 2^{16} kilo byte of 2^{16} kilo bytes addresses. So, in case of 8085 so you remember that this input output statement. So, they were having only 8 bit address bus. So, you have that 8 bit address can be specified there. So, we could connect only 2^8 or 256 devices, but here the address bus for IO is also 16 bit. So, you can have to power 16 different IO devices connected to the processors. So, total 64 k addresses have been dedicated for the IO devices.

So, they can be connected to that, it operates in 2 modes minimum mode and maximum mode. So, there is a min max bar pin. So, this pin max bar pin so that will be deciding the operation of that. So, minimum mode means it will be in that mode the most of the time the processor will work and there the instruction set it is restricted, normally the user programs they will be executing the minimum mode and sometimes when you need the system mode of operation with unrestricted access for a every part of the processor. So, you can use this maximum mode.

So, that is min max mode. So, pins and signals.

(Refer Slide Time: 05:47)



(Refer Slide Time: 05:49)

Pins and Signals

Pin	Signal
1	GND
2	AD ₀
3	AD ₁
4	AD ₂
5	AD ₃
6	AD ₄
7	AD ₅
8	AD ₆
9	AD ₇
10	AD ₈
11	AD ₉
12	AD ₁₀
13	AD ₁₁
14	AD ₁₂
15	AD ₁₃
16	AD ₁₄
17	NMI
18	INTR
19	CLK
20	GND
21	RESET
22	READY
23	TEST
24	INTA
25	ALE (QS ₁)
26	DEN (S ₁)
27	DT/R (S ₂)
28	M/IO (S ₃)
29	WR (LOCK)
30	HLDA (RQ / GT ₁)
31	HOLD (RQ / GT ₂)
32	RD
33	MNV/MX
34	BHE/S ₃
35	AD ₁₆ /S ₄
36	AD ₁₇ /S ₅
37	AD ₁₈ /S ₆
38	AD ₁₉ /S ₇
39	V _{CC}
40	GND

**AD₀-AD₁₅ (Bidirectional)
Address/Data bus**

Low order address bus; these are multiplexed with data.

When AD lines are used to transmit memory address the symbol A is used instead of AD, for example A₀-A₁₅.

When data are transmitted over AD lines the symbol D is used in place of AD, for example D₀-D₇, D₈-D₁₅ or D₀-D₁₅.

A₁₆/S₃, A₁₇/S₄, A₁₈/S₅, A₁₉/S₆
High order address bus. These are multiplexed with status signals

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, you can you will be looking into different pins that we have like 8086 which has got this pins AD 0 to AD 14 then these are the 15 bit addresses and then we have got this AD 15 pin. So, that is also there. So, 16, 17, 18, 19. So, this is the 20 bit address bus and you see that address bus is also multiplexed with the data bus ok. So, this we have got 16 bit data bus AD 0 to AD 15 the D 0 to 15, but also we have got some more reliance. So, this a this should better though the names are given as 18, 16, 17 like that, but they are like that only address bus pins and they are multiplexed with some more status pins S 3, S 4, S 5, S S 6 and S 3, S 4, S 5 and S 6.

So, this 4 status lines are multiplexed with this address bus. So, that is why none of the address bits are free. So, they are coming as a 20 bit address and 16 bit data bus then other important pins that we have are the say this we have got this lines like this a non-miscible interrupt when interrupt line ok. So, we will explain them one after another. So, AD 0 to AD 15 so they are bidirectional bus address so, they are data bus.

So, this are low order and address bus. So, this these are multiplexed with data. So, when the AD line lines are used to transit to transmit memory address. So, then this the symbol a is used instead of AD. So, we call it a 0 to AD 15 and when they are used for transmitting the data. So, you will we will use the line D 0 to D 7, D 8 to D 15 or D 0 to D 15. So, for IO device so, they there will be D 0 2 the data bus lines will be there on a separate address (Refer Time: 07:48) a 0 to a 15 and for date for memory access. So, we

have got 20 bit address out of that this 16 bits are A 0 to A 15 then A 16 A S as I was telling that A 16, A 17, A 18 and A 19 they are the higher order address bus lines and they are multiplexed with status signals they are. So, they are also multiplexed, but they are multiplexed with the status bits S 3, S 4, S 5, S 6 then there are some.

(Refer Slide Time: 08:18)

GND	1	40	V _{CC}
AD ₁₅	2	39	AD ₁₆
AD ₁₄	3	38	AD ₁₇ /S ₃
AD ₁₃	4	37	AD ₁₈ /S ₄
AD ₁₂	5	36	AD ₁₉ /S ₅
AD ₁₁	6	35	AD ₂₀ /S ₆
AD ₁₀	7	34	BHE/S ₇
AD ₉	8	33	MN/MX
AD ₈	9	32	RD
AD ₇	10	31	HOLD (RQ/GT ₂)
AD ₆	11	30	HLDA (RQ/GT ₁)
AD ₅	12	29	WR (LOCK)
AD ₄	13	28	M/IO (S ₂)
AD ₃	14	27	DT/R (S ₁)
AD ₂	15	26	DEN (S ₀)
AD ₁	16	25	ALE (QS ₁)
NMI	17	24	INTA (QS ₀)
INTR	18	23	TEST
CLK	19	22	READY
GND	20	21	RESET

BHE (Active Low)/S₇ (Output)
Bus High Enable/Status
 It is used to enable data onto the most significant half of data bus, D₈-D₁₅. 8-bit device connected to upper half of the data bus use BHE (Active Low) signal. It is multiplexed with status signal S₇.

MN/ MX
MINIMUM / MAXIMUM
 This pin signal indicates what mode the processor is to operate in.

RD (Read) (Active Low)
 The signal is used for read operation.
 It is an output signal.
 It is active when low.

Interesting lines this BHE bus low active as a it is active low signal. So, bus at bus high enable or status. So, it is used to enable data on to the most significant half of the data bus that is D 8 to D 15. So, if you are trying to get value on D 8 to D 15 then this BHE line should be remained low and 8 bit device connect to the upper half of data bus use this BHE signal it is multiplexed to the with the status line S 7.

So, this BHE signal for the lower order byte of the access. So, it will as you as I was said that the data bus is 16 bit. So, it may come from 2 8 bit locations like if I have got 2 memory banks then one memory bank may be activated by 1 signal the lower memory bank may be activated by some signal, the higher memory bank is activated by this BHE signal. So, BHE signal should be low for activating the higher part of the data bus, then this minimum maximum min max bar pin. So, this pin will the, indicate what more the processor will operate in there is a read signal which is active low. So, it is for read option output signal and it is active when output is low this is. So, this read bar line so, for reading the value from the outside world. So, we have got this test line.

(Refer Slide Time: 09:41)

The slide displays the pinout of the 8086 microprocessor. The TEST pin (pin 23) is highlighted in red. To the right, two text boxes describe the TEST and READY pins.

Pin	Signal	Function
1	GND	
2	AD ₁₆	
3	AD ₁₅	
4	AD ₁₄	
5	AD ₁₃	
6	AD ₁₂	
7	AD ₁₁	
8	AD ₁₀	
9	AD ₉	
10	AD ₈	
11	AD ₇	
12	AD ₆	
13	AD ₅	
14	AD ₄	
15	AD ₃	
16	AD ₂	
17	NMI	
18	INTR	
19	CLK	
20	GND	
21	RESET	
22	READY	
23	TEST	
24	INTA	(QS)
25	ALE	(QS)
26	DEN	(S ₁)
27	DT/R	(S ₂)
28	M/ \overline{IO}	(S ₃)
29	\overline{WR}	(LOCK)
30	HLD	(RQ / GT ₁)
31	HOLD	(RQ / GT ₂)
32	RD	
33	MN/ \overline{MEM}	
34	BHE/S ₄	
35	AD ₁ /S ₅	
36	AD ₀ /S ₆	
37	AD ₁₅ /S ₇	
38	AD ₁₆	
39	AD ₁₇	
40	V _{CC}	

TEST
TEST input is tested by the 'WAIT' instruction.
8086 will enter a wait state after execution of the WAIT instruction and will resume execution only when the TEST is made low by an active hardware.
This is used to synchronize an external activity to the processor internal operation.

READY
This is the acknowledgement from the slow device or memory that they have completed the data transfer.

So, this is this is again a control active low line this is test bar line. So, this is an active low line it is tested by the wait instruction. So, this is solved for connecting some output device. So, if the output device is asking the 8086 to wait it will it will be 8086 program may execute wait instruction and then the instruction will the 8086 will be put into an wait step and then it will be sensing the light test bar.

So, this test bar line is activated then only it will go to the 8086 will come back to its operation. So, it will be activated from that point onwards. So, this is used to synchronise an external activity to the processors internal operations like there may be many devices in my system and some device may be doing something at which the this activation this 8086 program will also running and may be the program needs some data from some device that device may not be ready yet.

So, so, what the processor asks finding that is not ready it goes into an wait state executes wait state go to the wait state and in that wait state it will wait for this state bar line to become low. So, when this external device is ready. So, it will may send this test for line low as a result 8086 will come out from wait state and it will be continuing with the next instruction. So, this is the test instruction test pin and there is a ready pin. So, if the outside memory is slow the outside memory is slow in that case the once the address bus has been in copy has been getting the value of the address from where location from which location we want to read the value, then the it will the memory needs some time

for getting that data on the for putting the data on to the data bus, but the processor may be faster than this memory. So, the processor has to wait and for that purpose this ready signal is there.

So, what the memory module can do it can put this ready line low. So, that it will be the processor will be waiting and when this ready becomes high; that means, the memory has completed the data transfer. So, that way it can be now processor can read the value. So, this way this ready signal can be utilize. So, it will also true for 8085 two processor.

(Refer Slide Time: 12:10)

RESET (Input)
Causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles.

CLK
The clock input provides the basic timing for processor operation and bus control activity. Its an asymmetric square wave with 33% duty cycle.

INTR Interrupt Request
This is a triggered input. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending, the processor enters the interrupt acknowledge cycle.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

There also we had the ready line, then there is a reset pin. So, that will cause the processor to immediately terminate its present activity. So, it will be reset and then it will be going back to a predefined value of the program counter. So, it will start executing from that point and for the sake of see for the sake of security you have to make sure that this reset is really desired it is not a noise signal the signal must be active high for this 4 clock cycles.


So, if it is less that in the signal will not be taken as a valid one so, it will be ignored and if it is valid if it is high for 4 clock cycles then only the reset will take place, then we have the clock signal. So, clock will be that provides basic clocking basic timing feature for the processor and this bus control activity.

So, it is an asymmetric square wave with 33 percent duty cycle as I said that the high time should be 0.33 of the total clock period and low time will be 0.67 of the clock period then there is one interrupt request pin INTR. So, this is same as the similar to the INTR pin that we have got in 8085. So, this is a triggered input and it is sampled from the last clock cycles of each instruction to determine the availability of the request.

So, as similar to 8085 again that the interrupt line will be sampled on the last clock size so, if on that clock cycle you find that there is an interrupt that has come then only this interrupt line will be sensed ok, then the then the interrupt is pending then the processor will be go in to an interrupt acknowledge cycle activating that INTR signal and again the same thing device we put the address on to the data bus of this of the interrupt service routine and from that point the interrupt service routine should starts. So, these operations are similar to this operation of 8085 then we have got this min max pin.

(Refer Slide Time: 14:15)

Min/ Max Pin



8086

The 8086 microprocessor can work in two modes of operations : **Minimum mode** and **Maximum mode**.

In the minimum mode of operation the microprocessor do not associate with any co-processors and can not be used for multiprocessor systems.

In the maximum mode the 8086 can work in multi-processor or co-processor configuration.

Minimum or maximum mode operations are decided by the pin MN/ MX(Active low).

When this pin is high 8086 operates in minimum mode otherwise it operates in Maximum mode.

So, in min max pin, so this, have got. So, this has got 2 modes of operation 8086 processor it can operate in minimum mode or in maximum mode, in minimum mode of operation it microprocessor do not associate with any co processors and cannot be used for multiprocessor systems whereas, in maximum mode it can operate in the multiprocessor mode or the co processor mode. So, in minimum or maximum mode can be activated by this min max bar line.

So, when you are not using it with other processor. So, you should put it in the deactivated mode. So, this min max line should be made equal to 1 because putting it on a maximum mode. So, we have to put it as 0. So, this way we can go for this min max mode of operation, in minimum mode of operation it does not associate with any of the processors then in maximum mode it will work you know it can work in a multiprocessor environment.

So, you can put a number of processors and they will be cooperating with each other for doing some jobs done. So, that way you can put in to multiprocessor environment then, activated by this min max bar line and for minimum mode of operation. So, it should put it in the minimum mode otherwise a high put can high for minimum mode and low for maximum mode.

(Refer Slide Time: 15:44)

The slide displays the pin configuration for the 8086 Microprocessor, specifically pins 24 through 31. On the left, a pinout diagram shows the physical pins and their connections to internal signals. On the right, a table lists the functions of these pins. A yellow box at the top right states: "Pins 24 -31 For minimum mode operation, the MN/ MX is tied to VCC (logic high) 8086 itself generates all the bus control signals".

Pin	Signal	Function
31	HOLD	Output signal from the processor to control the direction of data flow through the data transceivers
30	HLDA	(Data Enable) Output signal from the processor used as output enable for the transceivers
29	WR	(Address Latch Enable) Used to demultiplex the address and data lines using external latches
28	M/IO	Used to differentiate memory access and I/O access. For memory reference instructions, it is high. For IN and OUT instructions, it is low.
27	DT/R	(Data Transmit/ Receive) Output signal from the processor to control the direction of data flow through the data transceivers
26	DEN	(Data Enable) Output signal from the processor used as output enable for the transceivers
25	ALE	(Address Latch Enable) Used to demultiplex the address and data lines using external latches
24	INTA	Write control signal; asserted low Whenever processor writes data to memory or I/O port
23	TEST	(Interrupt Acknowledge) When the interrupt request is accepted by the processor, the output is low on this line.
22	READY	
21	RESET	

So, for minimum mode of operation so, this min max bar line it should be tied high ok, this min max line so it should be tied high and then we have got this other lines like this 8086 will generate all the control signals for this operation for the operation because that is the that is the only master for the bus that we have here in this mode. So, this DT R bar line. So, this is for data transmit receipt.

So, this is the signal from processor coming out of the processor to controller direction of data flow through the data transceiver. So, if you are transmitting the data then it is data transmit data transmit and you are receiving the data receive bar. So, in this line is 0 it is

basically that \overline{DR} that is data receipt bar and if it is transmitting then \overline{DT} equal to this, then we have got this \overline{DEN} bar data enable bar lines. So, output signal from the processor used to as output enable for the transceiver. So, \overline{DEN} bar if you have connecting some transceivers into this with this 8086 processor. So, this \overline{DEN} bar line can connected to the corresponding enable line of this transceiver.

So, that its activating as for output signal, then we have got the \overline{ALE} signal. So, this address latch in enable. So, for de multiplexing address and data lines using external latches. So, this same as the 8085 we have the \overline{ALE} line then there is \overline{M} by I/O. So, this \overline{M} by I/O bar. So, this is used to differentiate in memory access and IO access. So, just like 8085. So, for memory access instruction is \overline{M} I/O bar line should be made high and for and for instruction and for IO access this \overline{M} I/O bar line should be made low, then there is a write bar line for write control. So, if you are putting it as low then it will be it will be putting doing a write operation ok.

So, we are going to do a write operation and then we have got this \overline{INTR} bar line that is the interrupt acknowledge. So, if you are asking for an interrupt, if you if you are if you find that there is an interrupt at the end of the current is instruction then the processor will enter into the interrupt acknowledge cycle and do the operation.

(Refer Slide Time: 18:11)

Minimum mode signals

Pins 24 -31

For minimum mode operation, the $\overline{MN}/\overline{MX}$ is tied to VCC (logic high)

8086 itself generates all the bus control signals

HOLD	Input signal to the processor from the bus masters as a request to grant the control of the bus. Usually used by the DMA controller to get the control of the bus.
HLDA	(Hold Acknowledge) Acknowledge signal by the processor to the bus master requesting the control of the bus through HOLD. The acknowledge is asserted high, when the processor accepts HOLD.

8086 Pinout:

- Pin 1: \overline{RD}
- Pin 2: \overline{AD}_{16}
- Pin 3: \overline{AD}_{15}
- Pin 4: \overline{AD}_{14}
- Pin 5: \overline{AD}_{13}
- Pin 6: \overline{AD}_{12}
- Pin 7: \overline{AD}_{11}
- Pin 8: \overline{AD}_{10}
- Pin 9: \overline{AD}_9
- Pin 10: \overline{AD}_8
- Pin 11: \overline{AD}_7
- Pin 12: \overline{AD}_6
- Pin 13: \overline{AD}_5
- Pin 14: \overline{AD}_4
- Pin 15: \overline{AD}_3
- Pin 16: \overline{AD}_2
- Pin 17: \overline{NMI}
- Pin 18: \overline{INTR}
- Pin 19: CLK
- Pin 20: \overline{RD}
- Pin 21: RESET
- Pin 22: READY
- Pin 23: TEST
- Pin 24: \overline{INTA}
- Pin 25: \overline{ALE}
- Pin 26: \overline{DEN}
- Pin 27: $\overline{DT}/\overline{R}$
- Pin 28: $\overline{M}/\overline{IO}$
- Pin 29: \overline{WR}
- Pin 30: HLDA
- Pin 31: HOLD
- Pin 32: \overline{RD}
- Pin 33: $\overline{MN}/\overline{MX}$
- Pin 34: $\overline{B}/\overline{S}$
- Pin 35: $\overline{AD}_{16}/\overline{S}$
- Pin 36: $\overline{AD}_{15}/\overline{S}$
- Pin 37: $\overline{AD}_{14}/\overline{S}$
- Pin 38: $\overline{AD}_{13}/\overline{S}$
- Pin 39: $\overline{AD}_{12}/\overline{S}$
- Pin 40: $\overline{AD}_{11}/\overline{S}$
- Pin 41: $\overline{AD}_{10}/\overline{S}$
- Pin 42: $\overline{AD}_9/\overline{S}$
- Pin 43: $\overline{AD}_8/\overline{S}$
- Pin 44: $\overline{AD}_7/\overline{S}$
- Pin 45: $\overline{AD}_6/\overline{S}$
- Pin 46: $\overline{AD}_5/\overline{S}$
- Pin 47: $\overline{AD}_4/\overline{S}$
- Pin 48: $\overline{AD}_3/\overline{S}$
- Pin 49: $\overline{AD}_2/\overline{S}$
- Pin 50: $\overline{AD}_1/\overline{S}$
- Pin 51: $\overline{AD}_0/\overline{S}$
- Pin 52: \overline{AD}_{16}
- Pin 53: \overline{AD}_{15}
- Pin 54: \overline{AD}_{14}
- Pin 55: \overline{AD}_{13}
- Pin 56: \overline{AD}_{12}
- Pin 57: \overline{AD}_{11}
- Pin 58: \overline{AD}_{10}
- Pin 59: \overline{AD}_9
- Pin 60: \overline{AD}_8
- Pin 61: \overline{AD}_7
- Pin 62: \overline{AD}_6
- Pin 63: \overline{AD}_5
- Pin 64: \overline{AD}_4
- Pin 65: \overline{AD}_3
- Pin 66: \overline{AD}_2
- Pin 67: \overline{AD}_1
- Pin 68: \overline{AD}_0

Then we have got 2 more pins hold and hold acknowledge. So, which is similar to 8085 again so, this is input signal to the processor hold is an input signal to the processor and

that way it can act it can require. So, other bus masters that we have in the system they can request for this holding. So, they can tell the 8086 processor do hold and the. So, that it would be release the address and data bus controls. So, that this, those devices they can take control of this pins of this address and data bus and do the operation and there is hold acknowledge. So, hold acknowledge will acknowledge it is an acknowledgement signal. So, that is given by the bus master to the controlling the 8086 processor to which this hold has come. So, when it is when it is done with releasing the buses ok, it had released the control of the address and data bus, then this hold acknowledge line will be activated.

So, that the requesting processor or requesting master will know that now the buses are free for my use, the other 8086 processor it is released the bus. So, it is now for used by us. So, this hold acknowledgement is an acknowledge signal by the processor to the bus master requesting the control of the bus through the hold pin, the acknowledge is asserted high when the processor accepts hold, accepts hold means if it has released the address and data bus controls then in the maximum mode of operation.

(Refer Slide Time: 19:45)

Maximum mode signals

During maximum mode operation, the $\overline{MN}/\overline{MX}$ is grounded (logic low)

$\overline{S_2}, \overline{S_1}, \overline{S_0}$ Status signals; used by the 8086 bus controller to generate bus timing and control signals. These are decoded as shown.

Status Signal			Machine Cycle
$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	
0	0	0	Interrupt acknowledge
0	0	1	Read I/O port
0	1	0	Write I/O port
0	1	1	Halt
1	0	0	Code access
1	0	1	Read memory
1	1	0	Write memory
1	1	1	Passive/Inactive

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, we have got this status signals like S 0, S 1 this S 0, S 1 and S 2 during maximum mode min max bar line is grounded and this S 0, S 1 S 2. So, they are they are the status signals and they are doing it like this that when it when it is 0 0 0 it is an inter interrupt acknowledge cycle 0 0 1. So, this is read IO port 0 1 0 is write IO port 0 1 1 is called. So,

these are different types of access, then this part is interesting like you see that this 1 0 0, 1 0 1 and 1 1 0. So, these are 3 instructions, 3 different status modes. So, where it will try to differentiate between code access and data access so, if it is 1 0 0; that means, it is doing the off code phase 1 0 0 1 is reading from memory and 1 1 0 is writing on to the memory and 1 1 1 means the processor is not doing anything.

So, it is in the passive or inactive mode.

(Refer Slide Time: 20:52)

Maximum mode signals

GND	← 1	40	← V _{cc}
AD ₁₆	← 2	39	← AD ₁₅
AD ₁₅	← 3	38	← AD ₁₄ / S ₁
AD ₁₄	← 4	37	← AD ₁₃ / S ₁
AD ₁₃	← 5	36	← AD ₁₂ / S ₁
AD ₁₂	← 6	35	← AD ₁₁ / S ₁
AD ₁₁	← 7	34	← BHE / S ₀
AD ₁₀	← 8	33	← MN / BKK
AD ₉	← 9	32	← RD
AD ₈	← 10	31	← (RD / GT ₁)
AD ₇	← 11	30	← (RD / GT ₀)
AD ₆	← 12	29	← (LOCK)
AD ₅	← 13	28	← (S ₁)
AD ₄	← 14	27	← (S ₁)
AD ₃	← 15	26	← (S ₁)
AD ₂	← 16	25	← (QS ₁)
NMI	← 17	24	← (QS ₀)
NTR	← 18	23	← TEST
CLK	← 19	22	← READY
GND	← 20	21	← RESET

8086

Queue Status (QS₀, QS₁) The processor provides the status of queue in these lines.

The queue status can be used by external device to track the internal status of the queue in 8086.

The output on QS₀ and QS₁ can be interpreted as shown in the table.

Queue status		Queue operation
QS ₁	QS ₀	
0	0	No operation
0	1	First byte of an opcode from queue
1	0	Empty the queue
1	1	Subsequent byte from queue

Then there are 2 more pins QS 0, QS 1 which has known as queue status the processor provides the status of queue in these lines. So, this queue basically what happens is that 8086 if you look into the internal block diagram you will find that there is a there is a instruction queue, instead of having a single instruction register then it is an instruction queue. So, it is the, it can tell like how many instructions are there in the queue the queue is by default 6 by 2 8. So, this is it can it can tell like what is the condition of the queue. So, like when it is 0 0 there is no operation, if 0 1 the first byte of an off code is from the queue then 1 0 is a is empty the queue is empty and 1 1 subsequent byte from the queue has to be.

So, this queue status that the processor provides status of queue on these lines the queue status can be used as external device drag the internal status of the, internal status of the queue in 8086. So, what is happening inside? So, you want to know at that instruction phase decode execute cycles. So, you can keep a watch on this QS 0 bar and QS 1 bar

line, the outputs can be is outputs are interpreted as shown in this table that is no operation or first byte off code from queue. So, like that then we have got some more pin started RQ GT 0 and RQ RQ GT 0 and RQ bar RQ 1 1 GT 1.

(Refer Slide Time: 22:22)

Maximum mode signals

GND	← 1	40	→ V _{CC}
AD ₁₆	← 2	39	→ AD ₁₆
AD ₁₅	← 3	38	→ AD ₁₅ / S ₁₆
AD ₁₄	← 4	37	→ AD ₁₄ / S ₁₅
AD ₁₃	← 5	36	→ AD ₁₃ / S ₁₄
AD ₁₂	← 6	35	→ AD ₁₂ / S ₁₃
AD ₁₁	← 7	34	→ AD ₁₁ / S ₁₂
AD ₁₀	← 8	33	→ MN ₁ / R ₁₀
AD ₉	← 9	32	→ R ₉
AD ₈	← 10	31	→ (RQ / GT ₁)
AD ₇	← 11	30	→ (RQ / GT ₀)
AD ₆	← 12	29	→ (LOCK)
AD ₅	← 13	28	→ (S ₁)
AD ₄	← 14	27	→ (S ₂)
AD ₃	← 15	26	→ (S ₃)
AD ₂	← 16	25	→ (DS ₁)
NMI	← 17	24	→ (DS ₂)
CLK	← 18	23	→ TEST
CLK	← 19	22	→ READY
GND	← 20	21	→ RESET

(Bus Request/ Bus Grant) These requests are used by other local bus masters to force the processor to release the local bus at the end of the processor's current bus cycle.

These pins are bidirectional.

The request on \overline{GT}_0 will have higher priority than \overline{GT}_1 .

LOCK

An output signal activated by the LOCK prefix instruction.

Remains active until the completion of the instruction prefixed by LOCK.

The 8086 output low on the LOCK pin while executing an instruction prefixed by LOCK to prevent other bus masters from gaining control of the system bus.

So, this is a bus request grants. So, these requests are used by other local masters to force the processors to release the local bus at end if the processors current base mass cycle. So, in minimum mode of operation so we have called hold acknowledgement, for maximum mode of operation we have got this RQ GT lines ok.

So, they are other processors can till the 8086 that you release the bus at end of the current cycle. So, that way these pins are useful, that this pins are bidirectional. So, it is like this processor can also send a request to another processor that is why it has bi relational and it is also there is a priority that is GT 0 will have higher priority than Gt 1. So, if there are request on 2 different masters for this release of the bus then this whichever is requesting on line 0. So, that will be taken and that will be regretting the bus, then there is a another line lock. So, these output signal activated by the lock prefix instructions so, you see that some instructions are got lock prefix and this remains active until the completion of the instruction prefixed by lock.

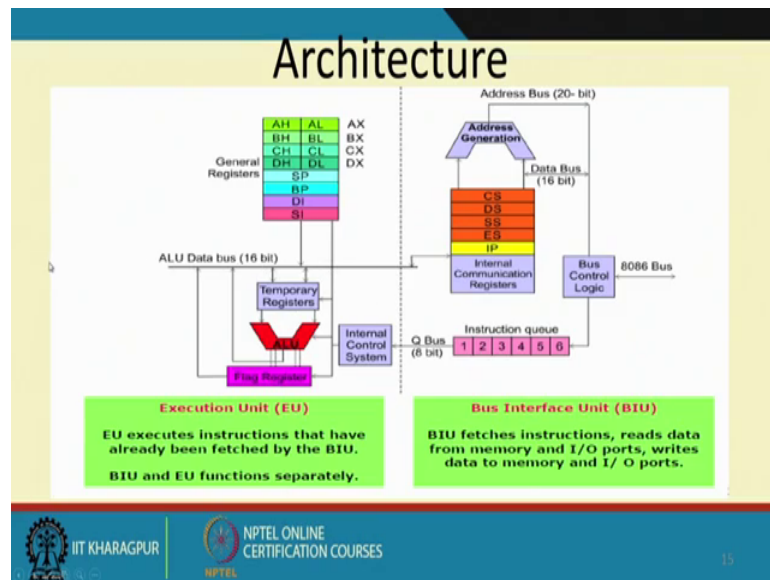
So, this is basically the it will the output is will be made low while executing instructions you will have to prevent other bus master from the any control of the system, like if you if some processor if the 808 is in maximum mode it is there. So, we have got a number of

such masters connected to the device. So, number of 8086 chips. So, each of them can act as a master, now for some of the instruction what can happen is that it is not possible to release the bus. So, those instructions while executing they will have a lock prefix in them. So, that this lock signal will be activated and then other processor they will know that now the instructions that are the, this particular processor is executing it is in lock mode. So, it is it will not release the bus. So, even if I tell it if I send it a request to that RQ 0 GT 0 line, that that I release the bus at the end of current instruction so that will not happen. So, that way this facility lock has provided for locking the buses for the processor, next will look into the architecture of this 8086 processor.

(Refer Slide Time: 24:46)



(Refer Slide Time: 24:50)



So, this is the architecture you see that it can be broadly divided into 2 parts or 2 regions one is called bus interface unit, another is called execution unit. So, in the bus interface unit we have got the features by which it can access bus, the address bus and the data bus.

So, this as you know that in case of 8086 we do not have 2 separate buses. So, it is multiplex 80 to 80 15 or and then we have got this we have also got this status lines multiplex with the bus. So, ultimately it is a 20 bit bus that is going out of 8086 ok. So, this is the 20 bit. So, address 20 bit address bus is going and out of that lower order 16 bits will be will be taken as data bus when it is doing the data operation and there is a bus control logic. So, bus control logic from which this bus will come out. So, ultimately one bus is coming out of the 8086 processor, unlike 8085.

So, there are 2 buses. So, 1 bus was the 16 bit address bus and another bus and out of that 8 bits were for the higher order 8 bits were not multiplex with anything, but the lower order 8 bit are multiplex with data base, but here entire address bus and data bus they have multiplex then ultimately you get a 20 bit bus from the processor, then if you look into the address generation you need you will see that there is an instruction queue. So, in case of 8085 when the this bus that they when the data, data bus was coming inside the processor from there it was going some instruction register followed by decoder, but here

it does not happen like that instruction register has been replaced by a 6 byte queue so that we can have a 6 different bytes loaded was.

So, it depending upon the instruction size. So, this 6 byte may constitute a single instruction or they may continue a they may constitute say 6 different instructions. So, both the extents are possible. So, whatever it is so whenever these external bus is free 8086 will request the memory to get the next instruction from there and put it on to the instruction queue and of course, if the some instruction is a jumped or branch type of instructions your fall type of instructions then it will ignore the content of current queue. So, it will flush out the queue and it will start loading from the new location average. So, that way this queue is handled.

So, it will be it will getting the instructions from this bus and put on to this queue, plus there is a set of registers which are known as segment registers CS DS ES and SS. So, they are called this CS stands for code segment register, DS stands for data segment register, SS stands for stag segment register, ES is called extra segment register and there is one instruction point are IP.

So, each of those each of these registers 16 bit register and this 16 bit register and 1 of this 16 bit register and this IP value. So, they are put on to this address generation unit accordingly it generates a 20 bit address, from these to 16 bit data it generates a 20 bit address and this 20 bit address goes to this 8086 bus to the outside world as the address bus and sometimes we also need to connect to this data bus is also connected here because we may need to do some operations on the data that is coming from the data bus. So, some at computation related to the address.

So, that is why it is connected there, otherwise this instruction is from this instruction queue. So, these instructions they come to the internal control systems are other part of this architecture which is the execution unit. So, in the execution unit so it will be having a number of general purpose registers AX, AX, CX, DX each of which is basically consisting of 2 8 bit registers for example, AX consists of and AL, BH consists of BH and BL, CH consists of CH and CL. So, like that it consists of a number of registers and then we have got this SP, DP, SI and DI.

So, those register, so they will be consisting of they are some special registers for doing the purpose. So, other control will be other portions that we have in this portion of this

execution unit. So, they have there is a ALU this ALU is doing the actual operation some temporary registers for doing some holding some temporary data flag register just like that PSW register we have in 8085. So, here also you have got flag register and that way this whole control takes place.