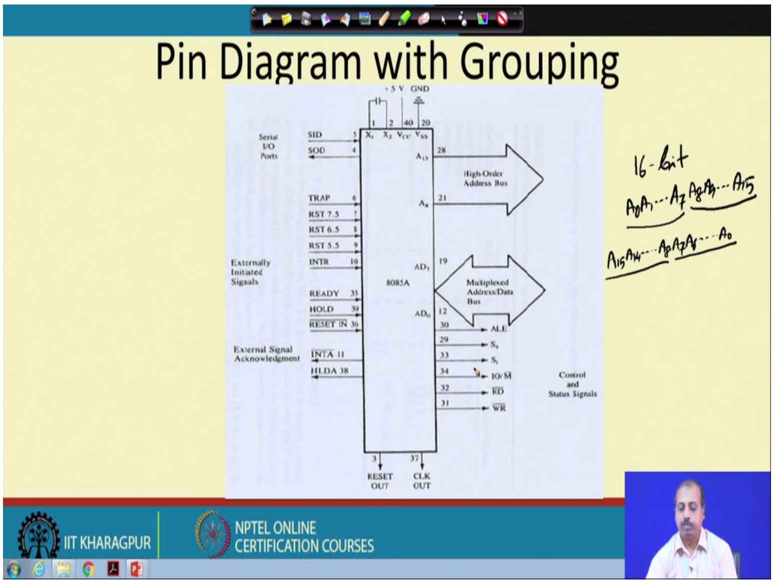
Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture -51
8085 Microprocessor
(Contd.)

So, 8085 pins, so if you look in terms of some grouping of them that is the related pins we keep them together; so, the pins which are doing a particular function.

(Refer Slide Time: 00:29)



So, if you see this in the first group of pins that we that will find corresponding to the address bus, and as I said that my address bus in case of 8085, so this is 16 bit. So, they are named, so this is A 0 A 1 A 7 A 8 A 9 up to A 15. So, out of that this A 0 to A 7 see this is lower order address bus conceptual lower order 8 bits and A 8 to A 15. So, this is the higher order address bus or higher order address bit.

So, you better the if I write this most significant bit in on the left side, so I should write it like this ok. So, A 7 A 6 up to A 0, so this is the higher order bus higher order address bits this is the lower order address bits. So, this higher order address bits they are coming from the higher order address bus.

So, pin number 21 to 28 they form this higher order address bus and they are this A 15 to A 8. So, A 15 is pin number 28 and A 8 is pin number 21 now this, the lower order address bus. So, A 7 to A 0, so that is multiplexed with the data bus so this they are so this is allocated pin number 19 to 12 ok, so this 8 bits. So, they are giving me address sometimes and also sometimes it will act as the data bus.

So, this pins 12 to 19 sometimes it will act as lower order address bus sometimes it will act as the data bus. Now, the address bus is going out from the processor to the memory and the data bus maybe bidirectional. So, as a result this higher order address bus is going out of the memory and this lower order address bus which is multiplex with the data bus, so this is bidirectional ok. So, apart from that we have got some other important pins like this ALE signals. So, this ALE signal will be used to tell the content of this multiplex address data bus.

So, when this ALE signal is high then the processor has got the address put on to this multiplex bus, and when the ALE signal is low then this bus may be used as the data bus. So, when the address is put this ALE signal is active, so when address is not there ALE is de active. So, it may in that times this data this is treated as a data bus. S 0 S 1 IOM bar already said that they are giving me the status of what the processor is doing now, then read bar write bar. So, they are actually telling what operation the processor is doing is a memory read operation on a memory write operation. And IOM bar so, this is actually is the pin that tells the operation that the processor is doing is it accessing a memory location or it is accessing some I O device.

So, if it is accessing an I O device, so in that case this IOM bar line will be equal to 1 and if it is accessing a memory location then the IOM bar line will be equal to 0. So, if IOM bar is 0 and this read bar is also 0, that means the overall the processor is doing a memory read operation. Similarly, if IOM bar is equal to 1 and this write bar is equal to 0 that means processor is doing some I O write operations. So, it is doing write operation on some output device it accessing some output device.

So, this is way this means the IOM bar read bar write bar and S 0 S 1 etcetera. So, this S 0 S 1 you will see later. So, that gives us some in indication about what the processor is doing at present then we have got some additional pins like SID SOD lines have already said so, they are for serial input data and serial output data. So, if you, so this 8085 it has got the feature by which it can get data from outside world serially. So, this the SID pin and it can also output some data to the outside world by the SOD pin. So, that is serially the data will go, there are some ways by which you can interrupt the operation of this processor.

So, that is by means of this line TRAP, RST 7.5 6.5 5.5 and INTR. So, you look into this interrupt issues later, but they are actually to tell the processor to suspend the current operation whatever it is doing and do some special operation for the system. So, it may be that some something special has happened like user has paste some key. So, that has to be acknowledged that also that value has to be read and all, or maybe there is some emergency situation has occurred some fire allowance has gone up. So, that way that has to be detected and that has to be take a some special action has to be taken.

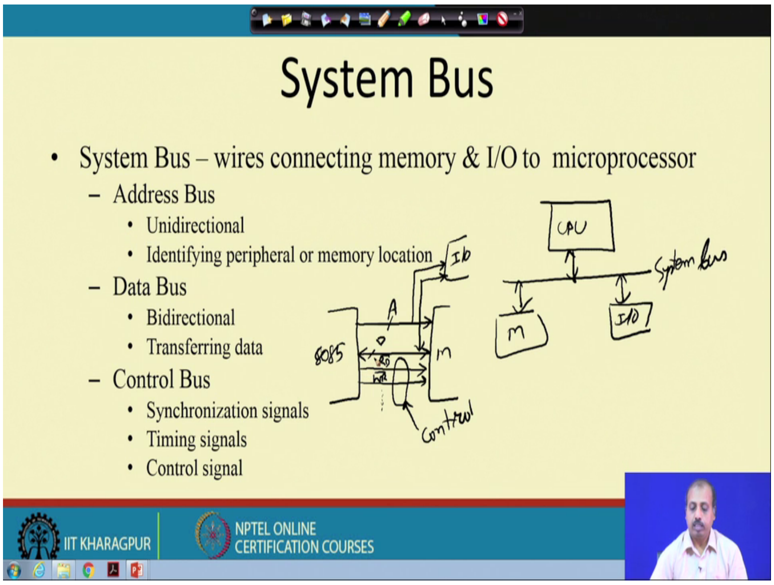
So, these are the interrupts that are coming then you what about the processor is doing. So, you can give it a reset signal by pin number 36 to activating this pin number 36 will reset the 8085 processer and the. So, whatever operation it was doing about that operation the whole system will be reset in is activated, so this 8085 apart from resetting its internal registers and all. So, it will also send output high on the reset outline. So, if my system consists of the 8085 processor and number of other devices. So, those devices can also be reset by this reset out pin.

Now, for getting the clock to the system, so we have to connect some crystal between pins 1 and 2 so crystal is connected. So, accordingly the 8085 generates the clock and through this pin number 37 this clock is also outputted. So, if you if you want to use this clock to some to some other circuitry. So, you can do that in using this pin number 37. So, most of the lines we are covered another important line that we have used hold and hold acknowledge. So, hold is actually telling the processor that, so that there is some other processor in the system like what happens is that if you connect says 2 such 8085 chips in a system and they are accessing the same memory ok.

So, at some time this address and data bus will be driven by the first 8085 chip sometimes it will be done by the second 8085 chip. So, if there is no coordination between the two then they will be accessing try to access simultaneously on the data corruption will occur. So, what is done is that we have these to hold pin. So, if whenever one processor wants to access the bus.

So, it will say it will give a hold to the other processor and as a result is other processor will generate hold acknowledge. And after getting the hold acknowledge only the first processor will understand that ok, the other processor has released this address bus and data bus for my usage and then it will proceed to use those address data bus lines. So, this way we can have multiple bus masters connected in a system and they will be, that will be controlled by these hold and hold acknowledge lines ok.

(Refer Slide Time: 07:54)



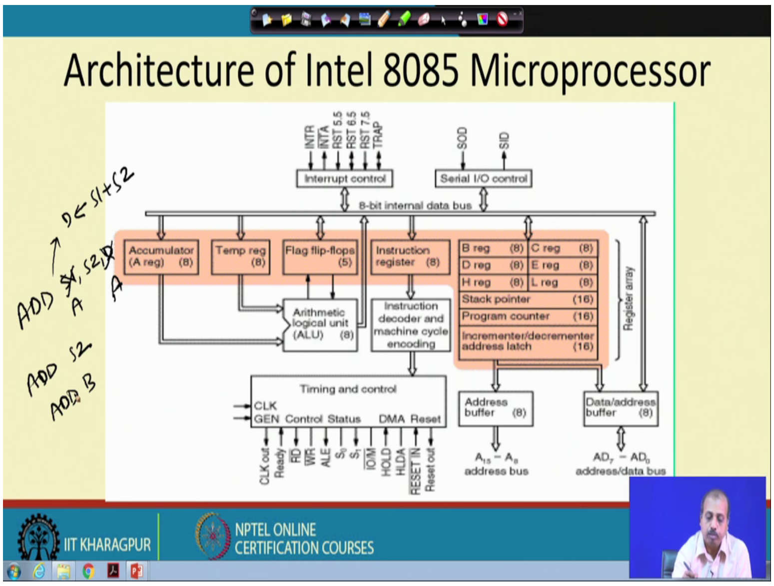
So, coming to the system bus, so this system bus, so these are the wires connecting memory and I O to the processor. So, you remember the diagram that we have shown previously, so if this is your CPU. So, we have got this memory and I O, so this is my system bus. So, CPU this memory and this I O they all hang from the system bus so, the wires connecting memory and I O to the microprocessor.

So, this is our system bus those system bus so though I have shown it as a single bus. So, in reality it consists of three different buses address bus, data bus and control bus. So, again if I look into this memory and CPU connection in more detail so, this is my 8085 processor and this is the memory now the address lines. So, they will go from 8085 to the memory chips, so this is these are the address line. So, address line is going to be unidirectional. So, this will also go to some I O devices this address lines also go to the I O devices to tell which I O devices which I O device it wants to access, so if it 8085 wants to access memory.

So, it will be giving address for memory location if it wants to access some I O device it will give the address to the I O device. Now address bus is unidirectional it goes out of 8085 and goes to the I O device for memory. And it will it is used to identify the memory location or the I O device location now there is data bus. So, data bus is bidirectional because I can have transfer from 8085 to memory or from memory to 8085 to read and write both operations are possible that the. So, this is this is going to be a bidirectional one so I it is also connected to the I O devices, so this is bidirectional connection.

So, this is data bus is bidirectional and for transferring data and we have got this the control bus consisting of other signal control signal synchronisation signals like that for example, in case of 8085. So, we have got a number of signals like say the read bar line, the write bar line ok. So, all these lines that we have so they are connected like this. So, there may be more such synchronisation lines as we look into more detail of this design, so will see. So, this my so my so this part to this part is known as the control bus. So, this is called control bus. So, this the first upper one is address bus and this one is the data bus. So, we have got address bus, data bus and control bus which constitutes the system bus ok.

(Refer Slide Time: 11:05)



So, let us look something inside, somewhat inside the 8085 microprocessor, how does it look like? So, this 8085 microprocessor if you look into so first you will find the number of registers ok. So, there is a special register called accumulator. So, which is as the name suggests it accumulates the value. So, whenever you are doing some arithmetic or logic operation in the processor, then one of the operands of that operation must be the A register and the result is also stored in the A register.

So, if suppose I am doing an addition operation, then this addition operation one of the operand must be in the A register and the other operand maybe in some other register or it may be some memory location, but the result will again be stored in the A registered. So, the advantage that it gives is that if I am doing some addition operation, then you see the there are three things to be mentioned the source 1, source 2 and destination fine. So, source 1 and source 2 are 2 registers, now our memory location and ultimate operation to be done is this D gets the value of S 1 plus S 2; Now, if do this then I need to specify three different operands S 1 S 2 and D.

So, in case of 8085 the designers they have said that S 1 must be A and this destination must be A ok. So, this two are fixed so this two are accumulator. So, the instruction simplifies to add S 2. So, it may be for example, say add B, so that will add the v register content with the accumulator to get the value, now here you see that we have got this accumulator is A. So, these are special register so its output goes to the arithmetic logic unit and there is a temporary register.

So, where it holds the other operand for this arithmetic logic operations so whenever you are doing say add B. So what happens is that it is temporary register gets the content of B register and after that it is the operation is done by this arithmetic logic unit. So, this arithmetic logic unit as we know that is purely combinational unit combinational circuit. So, if its inputs are fluctuating then the result will also fluctuate. So, if it is coming from some register then definitely the inputs will not fluctuate. So, whenever it is giving it is told that you do the operation of addition.

So, it will this arithmetic logic unit will do that, so this is the temporary register then there is a set of flip flop or called flag flip flop. So, they are actually telling the status of the last operation that a last arithmetic logic operation that has been done. So, it may, so happen that after doing the operation. So, there was an overflow generated, so there is a particular bit in the flag register which will be set or it may so happened that the result becomes 0 ok.

So, in that case some again some flag bit flag flip flop is there which will be set and it will tell like what is the content of a, what was the status of the last operation, so that way it is used ok. So, these are the these are the special register plus there is set of general registers like this B, C, D, E, H and L. So, these are 8 registers that 6 registers that we have, so all this registers are 8 bit registers. So, they can be used as operands for these arithmetic logic operations as I said that add B. So, when say add B I am actually referring to this B register ok. So, that way I can have add C, add D like that ok.

So, they are these are the B, C, D, and H, L, so these are some registers. Now there are some special registers like there is something called an instruction register. So, this instruction register, so this is actually holding the instruction obtain from the memory. So, as I said that every it has so when this 8085 starts. So, in the first in the first instance it gets the first instruction from the memory. So, after getting the first instructions, so it needs to decode it. So, before decoding the instruction is when it is phase from the outside world. So, it is loaded into this instruction register.

So, in the successive phase this instruction register content it goes to this decoder, where it performs this instruction decoding and this instruction decoder output. So, that that actually identifies the instruction that it is going to do and then it is given to these timing and control module. So, which will generate all the control signals that will be needed at various time instants to do the operation? So, will see some timing diagrams latter which will tell you like how this is a control signals are to be generated and then.

So, this timing and control unit, so this generates the different control signal based upon this clock circuitry that we have. So, there was this clock signal that we are getting, now apart from that there are some special registers like there is a program counter. So, this program counter it holds the memory address from where the next instruction will be executed. So, as I said that when you reset the processor then this 8085 comes to a special address. And in case of this 8085 this address happens to be 0 the address location 0.

So, if you reset the processor then this program counter gets the value 0. So, that it will be accessing the next instruction from location g. Now, as the first instruction is faced this program counter value is automatically implemented and it points to the next instruction to be fetch. So, when the processor goes to the next fetch cycle, again the program counter output is put on to this address bus this the other has a higher order address bus and the lower order address bus. So, this program counter content come to this address bus. And then it is the instruction is the fetched from the memory and that comes again to the instruction register that goes into this decoding and the decoder will be generating will giving information to this timing and control that will give control for the execution.

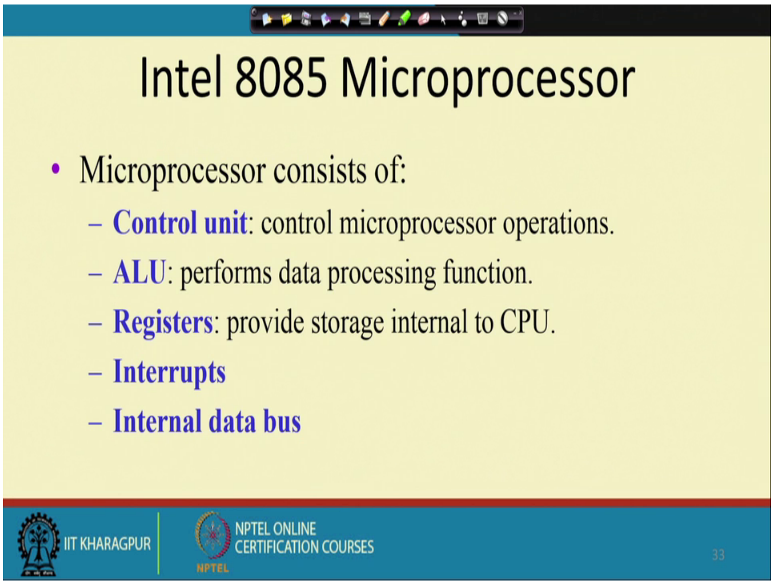
So, after this is over again this program counter output is put on to this address buffer and address data buffer. So, that way it will go, so this program counter is a special register who which tells like what is the next instruction the address what is the address of the next instruction to be executed, now for this program counter. So, we have got incremental and decremental address latch. So, sometimes, so automatically with the address register in the program counter content has to be incremented or decremented.

So, that is done by this one this address latch. So, is doing the increment decrement and there is a stack pointer. So, will come to its usage latter that is there are some special operations that are needed to be done in the in any processor for storing the return address from subroutines and the stack pointer is used for that purpose, so will come to that later.

So, apart from that, so you will find a module which is known as the interrupt control. So, this interrupt control actually handles all this interrupt plans ok, accordingly it will tell the process have to suspend current operation do something else and that will be coming to this timing and control unit also to tell to tell it like what to do. And there is a serial I O control input control line SID and SOD. So, that module will be doing this serial communication. So, this is more or less the internal view of 8085 processor, so as a user of the system.

So, we know that there is a special register A and there are some general purpose register B, C, D, E and H, L. So, these are the 7, these are the 7 register that we have in our depository. So, I can use my these registers for writing programs for the processor. And apart from that there are some flip flops which are the status flags and from the install from the manual of the processor. So, will know what are the arithmetic logic operations that the processer can do and accordingly we can that that will tell you: what is the functionality of the ALU.

(Refer Slide Time: 19:43)

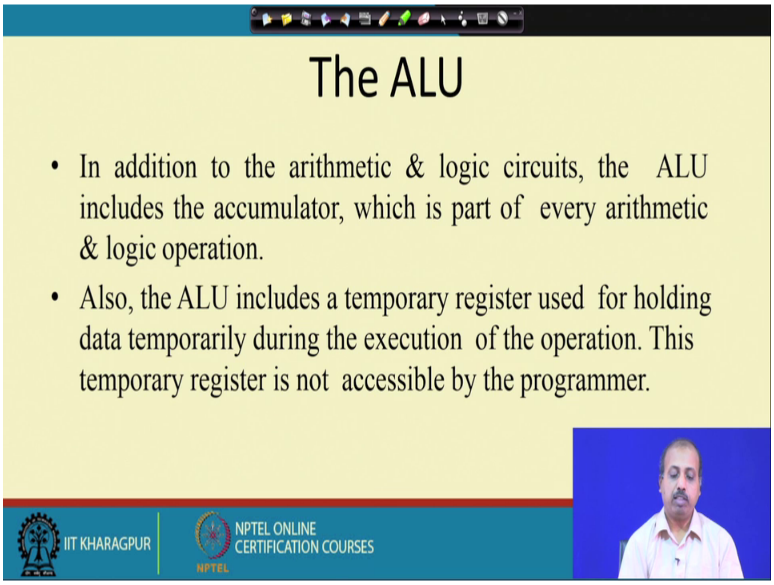


So, to summarise this 8085 microprocessor it consists of control unit ALU registers interrupts and internal data bus. So, control unit it will control the processor operation ALU will perform a data processing function then this registers they provide storage internal to the CPU. So, this registers like if you have got some data.

So, normal way to have the data is that have it in the memory from there fetch it, but normally is in case of 8085 this memory happens to be a chip which is outside which is the memory is a location which is in a chip outside the processor. So, to access it, so it is much slower compare to on chip axis line. So, if you have if you are having communication within the chip. So, that is much much faster than some communication which is of chip.

So, if I know that some of the variables are needed very often in my program. So, I may decide that I will put it in one of those registers B, C, D, E, H, L and use that. So, that for that particular variable I don’t need to access the memory. So, that way the access will be faster. So, this registered they can provides storage internal to the CPU and there are interrupt facilities and there is internal data bus. So, these are the major component that we have in the 8085 processor.

(Refer Slide Time: 21:12)

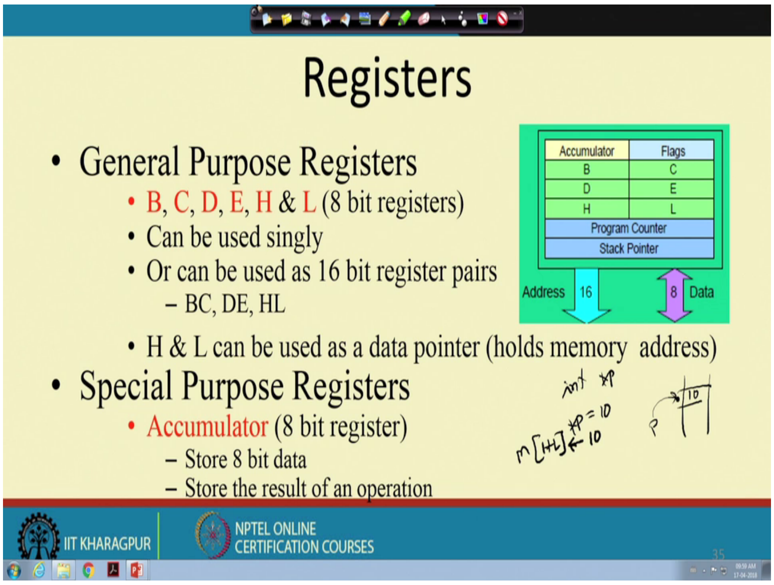


So, will start with the ALU so, this ALU it as the name suggest arithmetic logic unit. So, this is doing the arithmetic and logic operation. So, apart from this logic operation so, it also has the accumulator which is part of every arithmetic logic operations though in the diagram we have shown this the accumulator separately. So, for all practical purposes, so you can think that the accumulator is a part of the ALU because ALU always gets one of the operands from the accumulator and it produces result on to the accumulator and it also has a temporary registers.

So, like in this diagram you see that we have got this ALU we have got the accumulator and the temporary register. So, they are very much integrated with the ALU. So, for all practical purposes I can say that these three units are together for and becoming an integrated part of the arithmetic logic unit now internally how is it implemented, so that is not known.

So, that is not diverged by 8085 designers, but conceptually it we can say that they are all part of the accumulator ok. So, this ALU it holds a temporary register for holding the data temporarily for the during execution of the program and this temporary registered is not accessible by the programmers. So, as per as the programmer is concerned there is no such temporary registered available. So, it has got the register A it has got the registers B, C, D, E, H, L.

(Refer Slide Time: 22:45)



Now, if you look into the registers. So, registers can broadly we classified into two categories one is called general purpose register and other is called special purpose register. So, general purpose registers, so we have got the registers B, C, D, E, H and L. So, these are the general purpose registers. So, they can be used singly or as a 16 bit pairs like you can say that also, so you can sometimes.

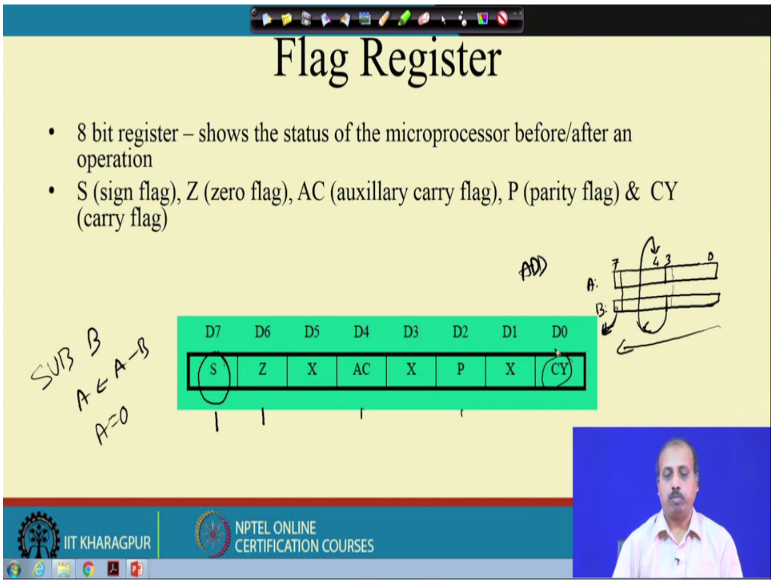
So, this for example, this BC pair, so BC pair so you can load some value want to the b register or you can load some value on to the C register. So, you can load some values there or you can say that I will load a value into this 16 bit pair ok. So, this is treated as a 16 bit registers B and C taken together considered as a 16 bit registers, so that is possible.

So, for pairing, so the pairing is in terms of B, C, D, E and H, L. So, will see some instructions later that will show you how to access this B, C, D, E, H, L are 16 bit pair. And out of that pairs it is H and L, they are also used as data pointer. Now, pointer you know that that is the very interesting data structure that is there in many programming languages. So, where we do indirect addressing, so in case of say for example, the language like C. So, I can define an integer pointer i n t star p and sometime later I write star p equal to 10. So, whatever be the location at which p is pointing to so that location gets the value 10.

So, how this thing is going to happen internal, so for this pointer to be supported or this indirect addressing to be supported the underline processor must support this indirect addressing. So, in case of 8085, so this is done by this H and L pair. So, there are instructions by which you can tell that I want to store the value 10 at the location pointed to by the H, L pair ok. So, all memory addresses are 16 bit, so this H, L pair is a 16 bit value. So, if you say that way so this memory location which is pointed 2 by H, L pair will be getting the value 10. So, they that actually, so if I my if the if the address of p address of this location is loaded onto the H, L pair, and after that I will do this M H L gets 10.

So, essentially it implements the pointer as a pointer operation. So, this H and L, where pair can be used as data pointer for holding the memory address. Now, apart from that there are some special purpose registers accumulators that we have already set. So, this is a source and destination of all arithmetic logic operation, then we have got this store the result on is also store the result of this register. So, there are some flag registers there are some flag registers which will tell you like the status of the last operation

(Refer Slide Time: 25:48)



So, we have got this flag register which is also an 8 bit register and it shows the status of the microprocessor before or after an operation. So, out of these 8 bits all bits are not used, but the bits that are used are the sign 0, auxiliary carry parity and this carry. So, this is the may if you look into this register. So, then this bit D 7 D 7 is the sign. So, if you do the last operation and after doing the operation the result becomes negative for example, suppose I have gotten instructions subtract B.

So, the essential it will do A gets A minus B and after doing the operation if the value of A becomes negative, then the sign flag will be set to 1. Otherwise this sign flag is 0 then the next bit number D 6 corresponds to the 0 flag, like if you do this operation if it happens that A and B they were same the values are same then after this A will become equal to 0 and whenever A becomes equal to 0 this 0 flag is set to 1.

So, many a times we need to compare between some two or more operant and depending upon the comparison result we want to take some action. So, this sign and 0 flags they will help that way bit number D 5 is not used bit number D 4 is for auxiliary carry. So, auxiliary carry is like this, so if you are doing some addition this is the first 8 bit this the this is the A register and this is the B register this is the B register. So, so this is bit number 0 to 3 and 4 to 7, now if I the instruction is a add. And when doing this operation if some carry was generated, when I have added say this bit 3 is it some carry is generated and this carry is taking is going to the next one.

So, if these carry is generated, so this is known as auxiliary carry. So, it is at the middle this is sometimes useful in this your B, C, D, E addition and all; So, will see that so this is the auxiliary carry bit then this parity bit. So, we have already seen that whether the, you want even parity or odd parity. So, based on that this bit may be set to 0 or 1 to make it either even or odd and this bit number 0 is the carry bit. So, carry is after doing this whole addition if some carry is generated at this position.

So, after you have done this whole addition if some carry is generated then this that you carry is stored in the carry flag. So, this way this flag register it uses different bits to represent the status of the last operation and based on the value that we have. So, we can we can write program that will exploit the values and accordingly take some conditional decisions within the program to continue with some part of the program or to go to some other part of the program.