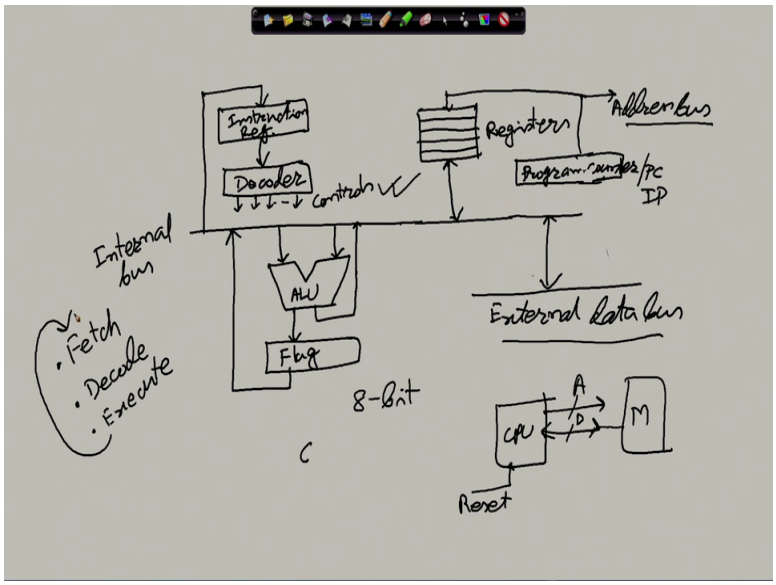
Microprocessors and Microcontrollers
Prof. Santanu Chattopadhyay
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 06
Basic Computer Organization (Contd.)

So, so far whatever we have seen: we have looked into the individual components and got some feeling like, how these internals of the microprocessors may be organized or for that matter any processor to be organized. For example, we have seen the design of registers; we have seen the design of functional modules like adder, comparator etcetera. Now how does it look like? Like any processor architecture. So, if you look into the structure of it how does it look like? So, we will start with that and then slowly go towards the microprocessors.

(Refer Slide Time: 00:50)



Now, any processor design, it will have something called some internal bus where from this bus all the components that are required for the processor they will hang. So, this is the inter. So, this is one common component that we will have in all the processors, you may have one such bus or you may have more than one bus, but essentially you will have a few buses on which this the components of the system will be hanging.

Another very common thing that we will find in any processor is the arithmetic logic unit or the ALU. So, this is the arithmetic logic unit and this arithmetic logic unit. So, this

will get inputs from the bus and it will produce the output to the bus. Now, apart from that we have got some apart from that there are some more modules like another important structure that we have are the set of registers. So, normally we will have a set of registers. So, we have seen the design of registers, now there in a processor there are a number of such registers say there are a say few registers here. So, these are the registers they are commonly known as CPU registers or simply registers.

Now these registers are connected to the bus. So, that the contents of these registers can be made available on the bus, and you can add the content of two registers using the ALU or you can compare them you can do any operation using the content of these registers and doing the operation by the ALU.

Apart from this, there are some other important components that we have in the system, one is known as the instruction register which is a special register. So, this is called instruction register. So, instruction register means what the processor will do next. So, that instruction should be available in this register. So, what when a processor starts working what it essentially does is that, it gets the instruction from the memory and loads it into the instruction register and then tries to understand what is the meaning of this instruction. And after understanding that, it will do that operation by getting the proper operands and all that.

So, this instruction register content it goes to another module which is commonly known as the decoder and this decoder it generates the control signals. It generates the control signals that will be required for doing the operation. So, these are the control signals that are generated. So, we will slowly understand; what does this control signal, what are these control signals etcetera.

Now, there is another special register in any processor, which is known as the program counter which tells like what is the next instruction that will be loaded into the processor. So, this is commonly known as program counter or PC. In some process some processors you will find this is called instruction pointer or IP essentially they are all same. So, what? So, this actually this PC or IP this is a special register and it points to the next instruction that should be loaded into the processor. As we know that we have got a memory connected to the processor and at the beginning of every cycle or every instruction cycle, the processor first gets the next instruction from memory and where is

this next instruction? So, next instruction is at the location pointed to by this program counter PC or the instruction pointer IP ok.

So, if you after successive accesses to the instruction, this PC value has to be updated like if the current instruction is spanning over 4 bites, then after the instruction has been taken into the processor, this PC value should automatically be updated by 4. So, this is an automated feature and all the processors they increment this program counter value by the size of the instruction automatically.

So, these program counter value. So, this is actually available in the; this value will be available on to the address bus of the processor. So, this value is going to the address bus of the processor and this internal data bus in some way it is connected to the external data bus. So, this is the external data bus that we have.

So, your modules like the memory then the peripherals. So, they will see these external data bus and this address bus. They will not see the internal, they will see the external data bus and the internal data then and the address bus. In some cases what happens is that many a times we need to use these registers to provide the addresses. This is particularly true when we are trying to have pointer type of data, where the content of some register this tells the address from where the value should be loaded.

In that case I need to made make some of this register data to be available on to the address bus. So, for that type of situation many a times we will find that this address bus is also connected to this set of registers that we have. Now, so when the instruction is first coming. So, instruction has to be loaded on to the instruction register. So, this instruction register should have a connection like this from this data bus. So, the instruction is coming via this external data bus. So, that is coming going to the instruction register.

Next this content of this instruction register is decoded by the decoder, and this decoder will generate a set of control signals to control various operations like maybe the instruction is to add the content of two registers. So, what I need to do somehow this content of these registers should be available on to the bus, the ALU should do the addition. And then finally, the content of the ALU should be stored back on to some register or some memory location.

So, we need to generate a set of control signals. So, we will take an example and try to understand how these control signals are generated. But before that there is another very important register that we have in any processor design, which are known as the flag register. So, this flag register is the status of last operation arithmetic logic operation that is done by the ALU. So, if this flag if the content of this output after doing some operation becomes 0, the ALU content becomes 0. So, that may be there. So, that there is a specific bit in this flag register. So, this flag register suppose this is an 8 bit flag register. So, there are 8 such conditions that can be detected and status can be stored in the flag.

For example for arithmetic operation one positive one very common operation common situation that occurs is that there is an overflow. As a result a carry bit gets generated and the carry bit cannot be stored with the adder with the final value because of the size limitation of the register. So, the carry bit is stored in the as one of the flag bit in the flag register.

Similarly, if the result is say 0 doing some operation, then this 0 flag of this 8 bit register can be set to 1. So, this way different vendor, they have come up with different flag bits that incorporate into the system, and that helps in understanding the state status of this whole system

So, the major part that remains now is to understand; how does this control part operates; what are the control signals and how does this control part operate. So, to summarize we can say that the as far as the CPU is concerned. So, so from this is your CPU and this is the external world. So, we have got the address bus, we have got the data bus by which this the CPU is interfacing with the outside world. Now when you reset this there; there all the processors have a reset signal when you reset this processor, then what happens is that this address bus it gets a pre specified value, that pre specified value depends on the manufacturer.

So, that value is loaded into the program counter and the value becomes available on to the address bus. So, that value is put onto the address bus, it is expected that in the outside I will have a memory chip from where the memory the memory chip after getting this address like address line value, it will put the corresponding part into the this data bus, and it is expected that the value that is put is an instruction.

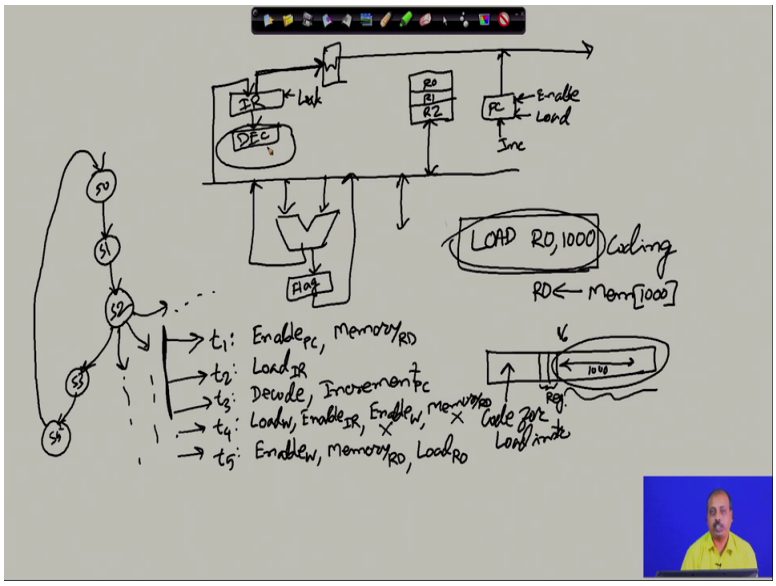
So, what the processor does is that whatever it gets onto the data bus. So, it takes it to the instruction register, and it goes into the decoder for decoding. So, first part when it is getting the instruction from the outside world. So, that stage is called the fetch stage that is called the fetch stage. It is in the fetch stage the location that the processor wants to access is put onto the address bus, the read signal is given to the memory. So, memory will pro get the content of the corresponding location, put it onto the data bus and the content will reach the instruction register. So, that this whole operation is the fetch operation.

The next stage of operation is the decode operation. So, in the decode operation what is done is that, this instruction that the processor has got in the instruction register. So, it tries to understand the meaning of that register. So, what is the stay instruction that it has? So, it will try to identify the meaning of that register meaning of that instruction. And now depending upon the instruction that it has, it goes to the third stage where it is the execute phase. Fetch decode and execute phase and in the execute phase it actually does the operation. And once the instruction execution is over it again goes to the fetch stage with the PC value the program counter value updated. So, that it points to the next instruction in the memory.

So, then it will fetch the next instruction and this way it goes on. So, starting with the initialization of this processor, it goes on doing operations in this fashion. So, this is the overall operation that is done and with the execute phase actually this ALU comes into picture. So, it does the operation and stores the result in register or it may store the result in some memory location, if the instruction has asked for something like that ok.

Now, we will look into another example to see like how this control part is working. This is a decoder part how is it generating the control signal, that part how is it working. So, for that purpose we will take a simple example of a CPU.

(Refer Slide Time: 13:40)



That has got this is a structure. So, it has the other parts remain same. So, it has got internal data bus, then it has got that ALU. So, these parts are same and this ALU has got the flag register.

Now we assume that there are 3 registers instead of having a for the sake of for the sake of understanding, we assume that there are 3 registers R 0, R 1 and R 2 there are only 3 registers and we have got otherwise we have got this program counter the PC, then we have got this instruction register or IR instruction register is written in short as IR, and there is another register W where we can store a part of the IR into the W register, and this w register content can be available on to the address bus.

So, this PC content is available on the address bus and W register content can also be made available to the address bus these R 0 R 1 R 2. So, they are accessible they are they can reach this internal bus, and they are contained may be available on to the internal bus and these you have got these connections that we had previously. So, they are remaining as it is.

So, then now on this suppose we want to execute some instructions. So, there is a decoder. So, this is the decoder which generates the control signals. Now suppose we have got an instruction which is something like this load R 0 comma 1000. So, as I said that when this processor is this PC content is put onto the address bus, from the external data bus the processor has got the instruction load R 0 1000. So, there is a there will be a

coding of this. So, that we will see later there will be a coding of this, and which is called the machine language coding of this instruction.

So, we assume that some code is there and that is entire instruction is available in the instruction register. Now how the decoder will generate control signals so that this instruction gets executed. Now we have got this, meaning of this instruction we take that R 0 should get the content of memory location 1000. R 0 register should get the content of memory location thousand. So, how to do this thing? So, we do it like this. So, we can say that first at time t 1, I should enable the program counter and I should give the memory read ok.

So, this actually, initially the processor puts the program counter value onto that on to the address bus and gives memory read as a result this load R 0 1000 whatever be the corresponding code, will be available on to the data bus the memory will put the content onto the data bus so that that will be coming. Then in time step t 2, what we do. So, by this time it is expected that the value is available on to the data bus so that value has to be loaded into the instruction register. So, this instruction register has got this connection. So, we give the load signal to the IR register.

So, this enable signal is actually the output enable. So, when I say enable PC so that in the when this is this PC it has got that control signals like enabled all the registers have got this type of control signal enable and load. Similarly this R 0, R 1, R 2 they have got individual control signals like enable R 0, enable R 1, enable R 2, load R 0, load R 1, load R 2 like that.

So, when I say that it is enabled a PC, then only the content of the PC will be available at the output. Similarly if I say enable R 0 then the content of R 0 register will be available in the internal data bus otherwise it is not. So, you see that the value is available here on the data bus and I have given load IR control signal to this instruction register. So, this has got the load signal load control. So, the load controls the decoder generates load controller at time t 2. So, the value this instruction will be loaded into the instruction register.

Then at time t 3 it will do the decode operation and also it will increment the program counter. So, the program counter register it has got another special control which is the

increment. So, it gives the increment signal to the program counter. So, that the program counter is incremented to the next value so that it can access the next instruction.

Then at time t 4 what is to be done. So, I have to give this see this in the instruction register I have got the content of this whole thing. So, if you look into the code of this, some part if this coding you say some say 16 bit coding out of that the first few bits, they will correspond to the keyword load like there can be other operation like add sub etcetera the first few bits may identify that this is the load operation.

So, this is in this part I will have the code for load instruction then the next few bits may identify the register on to which you want to load. Now in this case we have got 3 registers. So, 2 bits will be sufficient. So, these two bits will identify the register where you want to load and these remaining bits remaining bits. So, they will identify the address from where you want to load. So, this part is actually the value this will hold the value 1000.

Now, what is required? Now this entire thing is there in the instruction register now. So, from this instruction register, this part should be made available on to the address bus. So, that the value can the memory will be putting the value on to the data bus, and that the data the location content can come on to the data bus.

So, for that matter I have to give signal like load w. So, that the w register will be loaded with what value, we need to enable the IR register. So, that from the enable the output of the IR register content will be available as output. Now you see that here it is not shown explicitly, but you can understand that this connection from instruction register to IR to w. So, that is corresponding to only these bits. So, I do not need to connect all the bits to w. So, it is only those bits will be connected to w that correspond to the address part.

So, I give the enable IR signal then I also give the enable w. So, that the content is whatever value is loaded here is available on to this bus enable w, and also give the memory read signal. So, if I do this thing then what will happen the content of IR register will be loaded into this w register and the that value will be available onto the address bus and through this enable w, and this memory read control is given. So, whatever is the value that we have on the memory location that will be available onto the bus after some time.

Then at time t 5 I continue to keep this enable w, because it may be the case that putting this load w, enable w simultaneously. So, it may take some time for the value to get special stabilized. So, there can be options like some processor designer. So, they will not put enable w here, they will put enable w in the next stage only and accordingly this memory read will also not be put there, it will be put at the time step t 5 only.

However you can do this way also. So, you for some other design, it may be that it is enabled and it is also enabled in the next stage. So, enable w and this memory reads. So, they are continued here and we also give the load signal to the R 0 register.

So, this decoder it generates these control signals I enable PC memory, read load IR decode etcetera, etcetera. So, all these control signals are generated by the decoder, but you can understand that this is the all the signals are not generated at the same step. So, if I think in terms of the clock cycles. So, in the first clock these control signals are generated, in the second clock cycle these control signals are generated, like that it proceeds clock cycle wise it proceeds. So, I can say that this decoder that we are talking about here is not a simple combinational decoder that we have seen so far in our digital logic classes like if they are not like say simple 2 to 4 decoder or 3 to 8 decoder like that they are basically a finite state machine.

So, they go through a set of states. So, initially it is in state 0 s 0 in the state s 0, it outputs the controls for t 1. So, when the next clock comes, it transits to the state s one and in the state s 1 it gives this load IR signal. So, that the instruction register is loaded with the value. In the next clock signal it comes to the state s 2 where it gives this decode and increment PC.

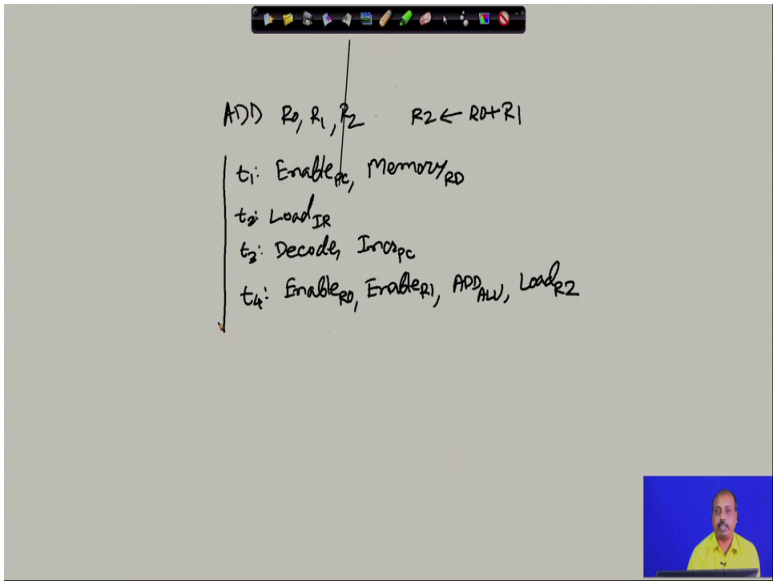
Now, see up to this much is common for any instruction. Any instruction you have up to this much has to be done. From this point onwards the operation depends on the instruction that you are executing. So, you can expect that from this point onward there will be differentiation there will be classification or diversion depending upon the instruction that you are executing.

In our particular case, this comes to the state s 3. So, it comes to the state s 3 where it outputs these control signals this t 4 control signal load w etcetera. Then it comes to the state s 5 where it outputs also this enable w memory read etcetera, and then it will go back to the state s 0 where it will be loading the next instruction.

So, this way this similarly these parts can also be developed; so this decoder that we have told. So, this is not a combinational decoder, but it is a sequential decoder. So, it is there is a there it is more of a controller rather than a decoder. Now there are issues like with designing this controller so that you can find in some architecture class. So, we will not go into that.

We will rather look into another example like how the add operation will be done.

(Refer Slide Time: 26:06)



Suppose I have got the next instruction add as add R 0, R 1, R 2; meaning that R 0, R 2 will get the content of. So, suppose we have got R we have got this instruction add R 0, R 1, R 2 which means that R 2 will get the content of R 0 plus R 1 ok.

Now, how to do this? So, t 1 is same as the previous case, I have to give enable PC enable program counter, and I have to give this memory read. So, these two are to be given. In t 2 I have to give load IR signal, in t 2 I have to give load IR signal, then at t 3 I have to give decode and increment PC these two signals, and at t 4 I have to do this actual operation. So, I have to enable R 0 or to enable R 1 and we have to give the add signal to the ALU and I have to give the load signal for R 2.

So, here actually the difference comes. So, in this way you can for different instructions, we can decide like how this instruction will get executed over the number of cycles. So, later on when we go to the microprocessor details like when you look into the timing

behavior of the microprocessors. You will see that these types of control signals are generated over the number of cycles to get the operations done by the microprocessors.