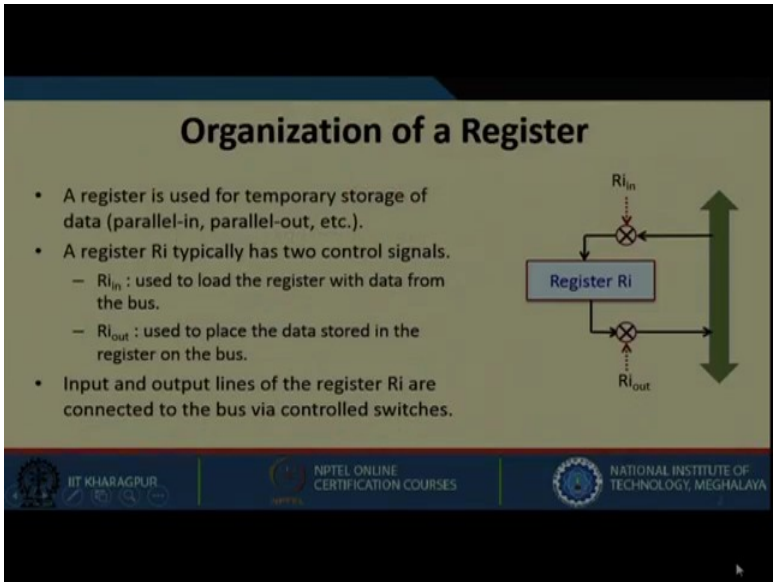
Computer Architecture and Organization
Prof. Kamalika Datta
Department of Computer Science and Engineering
National Institute of Technology, Meghalaya

Lecture – 18
Design Of Control Unit (Part 2)

Welcome to the next lecture on control unit design, design of control unit part 2. We will continue from where we left in the previous lecture.

(Refer Slide Time: 00:28)



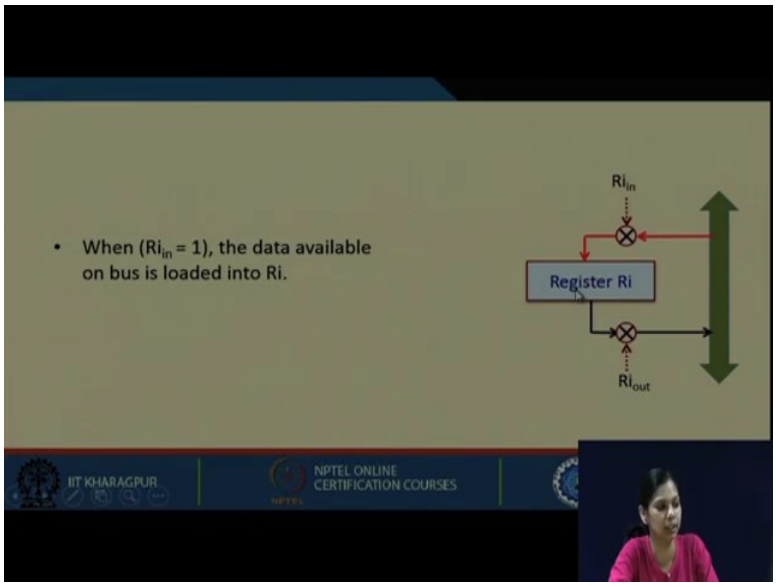
So, in the previous lecture we have seen a single-bus architecture, we have seen also a multi-bus architecture. Now, we will be specifically looking into how we can perform an operation. So, for performing an operation what is actually required? Let us see the organization of a register. A register is used for temporary storage of data. So, in this particular register, we actually store data. And now we are saying that we require the data when we are doing ADD R1,R2. We require the data from R1 to be available in ALU; we required the data R2 to be available in another input of the ALU. In such cases, the data from this register needs to be moved out through this bus and it reaches the ALU, and in the same way after the operation is performed the data must be brought into a particular register.

So, a register Ri typically has two control signals what is that Ri_{in} and Ri_{out}. Ri_{in} means something is coming into Ri. So, data available from the data bus will be available to

register Ri through the control signal Ri_{in}. Next we also sometime require the data from register to come out of this bus and then go to somewhere else. So, what is the control signal required for that? The control signal required for that is Ri_{out}. So, Ri_{out} will place the value from this particular register Ri into the bus. So, two signals are very much important Ri_{in} that will take the data from bus into this particular register, Ri_{out} that is used to place the data stored in the register on the bus. So, from the register, the data will be put into the bus. So, from the arrow you can easily make out.

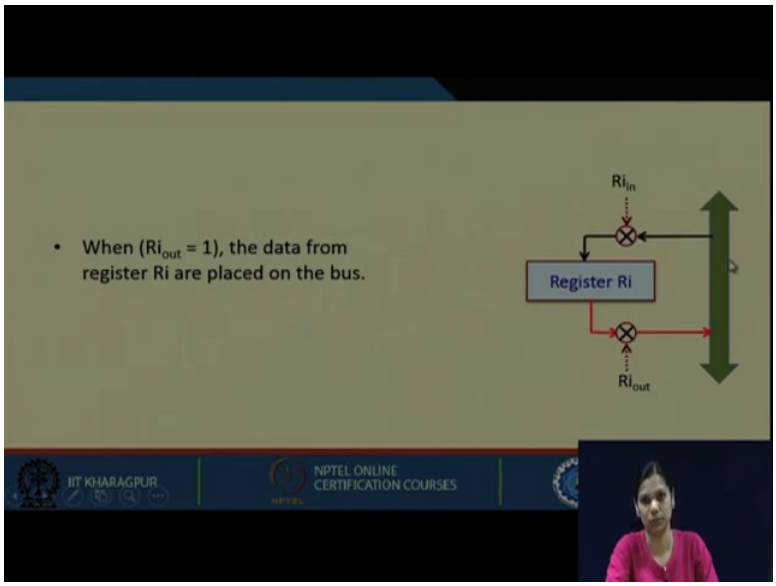
Input and output lines of the register Ri are connected to the bus via control switches. So, basically these are some switches. So, these switches are made on and off to do the particular function. So, what we have to do we have seen that Ri_{in} is used to load the value here, and Ri_{out} is used to take the value from the register and put it in this bus.

(Refer Slide Time: 03:43)



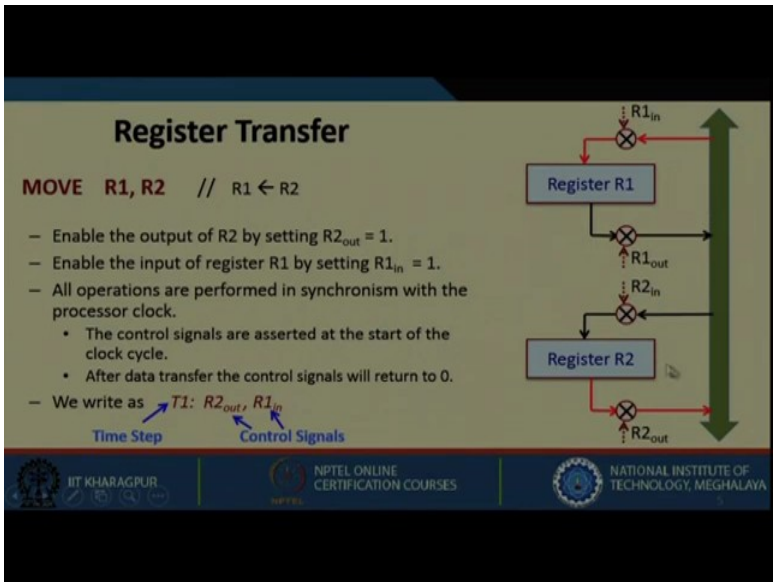
Now, let us see when Ri_{in} is 1; that means, Ri_{in} is active, the data from bus will be put into the register.

(Refer Slide Time: 04:04)



Similarly, what happens when Ri_{out} is 1? The data from register Ri will be made available in the bus. So, the data from registered Ri are placed into the bus. Now, you see what will happen when this is 0, there will be no change. So, this signal will not get activated, so nothing will happen neither in nor out.

(Refer Slide Time: 04:42)



Let us see this register transfer. So, similarly for individual registers we will be having some kind of signal like this, let say this is register R1 and this is register R2. So, for this register, one signal will be there $R1_{in}$, that will make the data available from bus to this

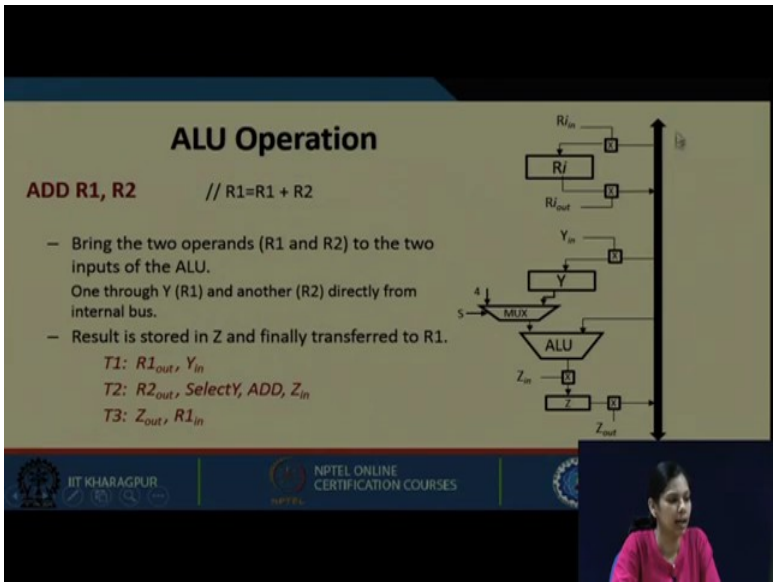
register; and they will be another signal $R1_{out}$ that will make available the data from register to the bus. Similarly, for R2 it is also there.

For register transfer - MOVE R1,R2 that means, content of R2 should be available into R1. Firstly, what we need to do is that content of R2 must be available in the bus. So, if I want the data of R2 to be available in the bus, which signal is required; R2out. So, R2out is required to be 1. So, how do we do it, enable the output of R2 by setting R2out = 1. Now, enable the input of register R1 by setting R1in = 1. Now, we also want that the data from the bus should be made available in to register R1. In that case what we need to do the signal R1in should be 1. So, for outputting R2out, we require R2out for making available; inside R1, we require R1 in, R2out, R1in.

All operations are performed in synchronism with the processor clock. So, this particular register transfer is within the processor which is much more faster. The control signals are asserted at the start of the clock signal; after the data transfer the control signal will return to 0. So, when the control signal will return to 0, what will happen? When the control signal is 0, no data will be available either on the bus, or no data will be available in the register also. Only when that signals are activated the data from either bus comes in into a particular register, or from a particular register the data goes into bus. So, after the data transfer ,as I said the control signal will return to 0.

So, to perform this what control signals are required in a particular step? So, at T1 we can perform R2out, R1in; R2out will make available the data of R2 into the bus; and R1in at the same time will bring the data from the bus into R1 that is happening in a time step T1. These are called control signals and this is the time step where this particular control signals are activated.

(Refer Slide Time: 08:21)



Let us see an ALU operation. For performing an ALU operation ADD R1,R2 what needs to be done. We need to make available both R1 and R2 in these two places; one is directly connected to the bus and another value is coming from this Y register through this MUX. We need to add the two register values, we have those registers in place; one of the register output needs to be made available into one of the inputs of the ALU; another register output needs to be available in another ALU input. And we must understand one thing that we have a single bus. So, only one transfer can takes place at a time.

So, the first thing, we bring the two operands R1 and R2 to the two inputs of the ALU as I just now said. So, one will be coming through this Y, and another is directly coming from the internal bus, this is how the organization is made. So, this is how the data transfer will also be done. And finally, after this ALU operation, the result will be stored in Z, and then it will be made available through this control signal into the bus, and then it will move to R1.

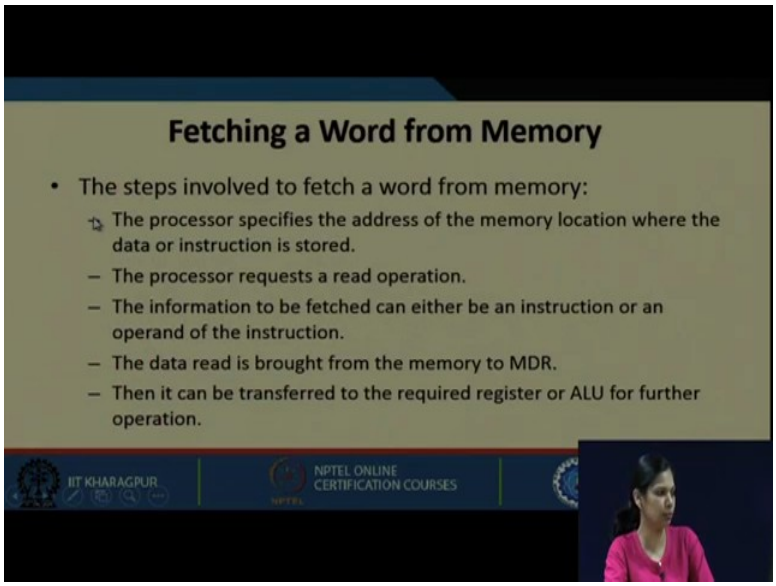
Now, let us see what are the various time steps and what control signals are required to be activated. So, in the first go let us make R1 available through Y. If we make R1out and Yin, the data will be available in Yin. So, data from R1 through this bus comes in and it goes to Y using the control signal R1out and Yin. Now, we have to make another signal available, that is R2out, if we just do R2out we see that it is directly connected, it

is not connected through any bus. So, at any point of the time whatever is out here is directly available into one input of the ALU. So, when we do R2out, this particular value is directly available here. And if we do select Y, then the Y value through this MUX will come in and will be available to this input of the ALU.

So, what we are doing here? We are doing R2out which is directly connected here, we are selecting Y using this MUX. So, the value which is there in Y, it was R1, will come in through another input of ALU and then we can perform the operation because both the inputs are now available. We do add and after adding what we do, we do Zin, because we can directly make this input into this through this particular signal that is directly connected to the ALU.

So, at T2 we are performing R2out, SelectY, ADD and Zin. So, now using this particular operation the value is available in Z. Now, Z contains the result, but we need to store the result in R1. So, for storing the result in R1, what we need to do? Now my value is available in Z, I do Zout. So, the data will be available in the bus and I do an R1in. So, it will going into R1. So, these are the three times steps that are required to perform this particular ALU operation.

(Refer Slide Time: 12:58)

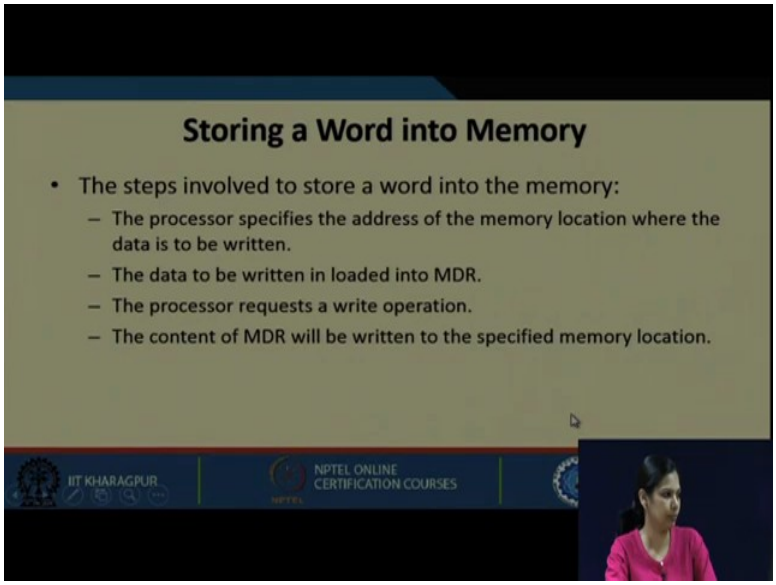


Let us move on. Now, we also need to fetch a word from the memory. So, for fetching a word what is required let us see that. These are the steps that are involved to fetch a word from memory. The processor specifies the address of the memory location where the data

or instruction is stored. The processor request for a read operation for fetching their reading actually; the information to be fetched can be either an instruction or it can be a data, accordingly it is fetched.

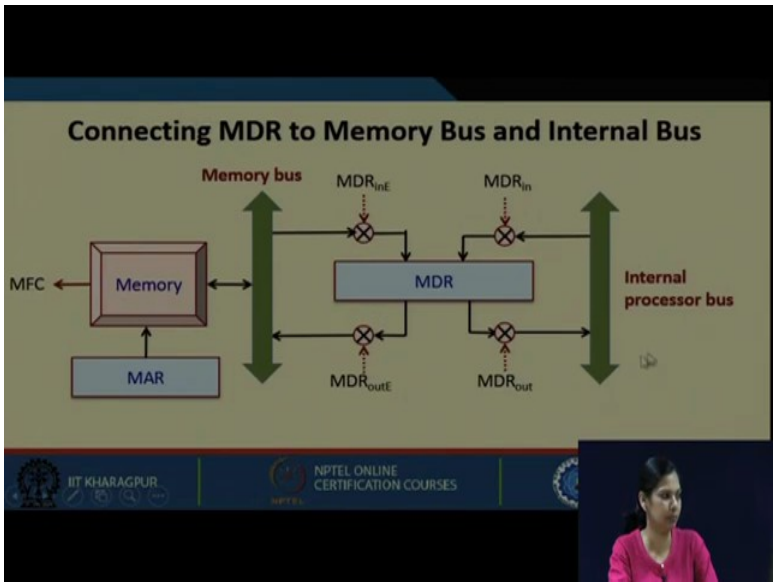
The data read is brought from memory to MDR. We already know all these things, we have already discussed in previous classes. So, when we fetch an instruction or data, we put the address in MAR, we activate the read control signal. Once it is read from the memory, it is brought it into MDR. And now from MDR that is an internal processor register, we can move the data or instruction wherever we want it. Then it can be transfer it to the required register or ALU. It can either go to IR if it is an instruction; or it can go to ALU for further processing if it is a data.

(Refer Slide Time: 14:18)



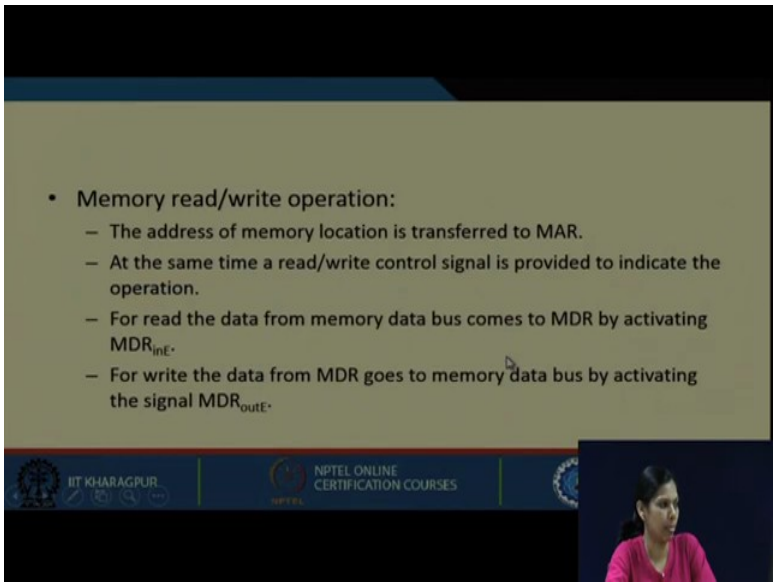
Now, what happens for storing a word into memory? For storing the word the steps involved is first the processor specifies the address of the memory location where the data is to be written. Now, here the thing is different; you have to load the particular address of the memory where the data is to be written into MAR, and then you have to load the data that you want to write in that particular location into MDR. So, you have the data in MDR you have the address where to be written in MAR and the processor request for a write operation. So, the content of MDR will be written to the specified memory location.

(Refer Slide Time: 15:06)



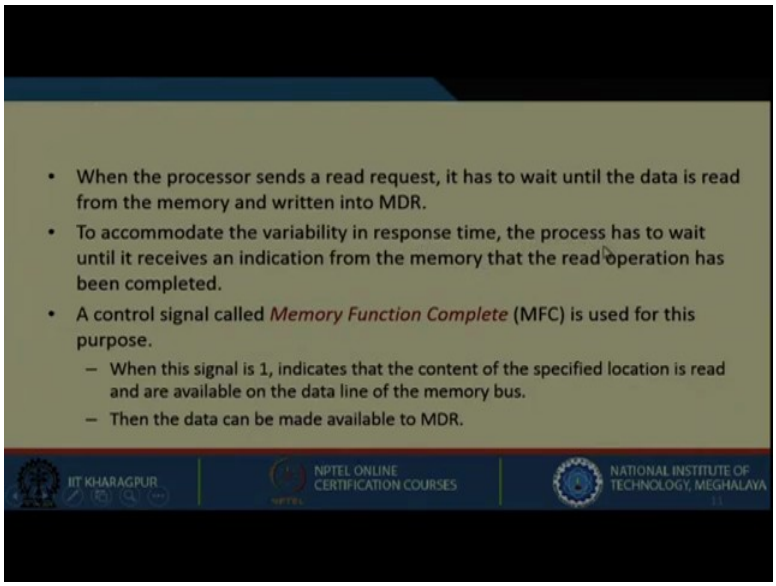
Now, let us see this is the connection of MDR. Now, you already know that MDR is a register that is connected to the internal processor bus and it is also connected to the memory bus of the system. So, in this particular case, we require four switches, one as it is connected to the internal processor bus when we do MDRout the value available from MDR will be made available through MDRout into this internal processor bus. And when we are doing MDRin then the data from internal processor bus will be made available to MDR. As MDR is also connected to an external memory bus we require two more signals MDRinE, MDRoutE, E stands for external. So, MDRinE means data available in the memory bus will be made available to MDR, and the data available in MDR can be transferred to the memory bus using MDRoutE. So, whenever we are doing MDRoutE then the data from MDR may be available in the memory bus. So, these are the control signals that are required.

(Refer Slide Time: 16:53)



Memory read write operation. What happens here? The address of the memory location is transferred to MAR, as I have already said at the same time the read or write control signal is provided to indicate the operation. Once we put the address in MAR we need to also specify whether it is a read or write. For read MDR is not coming into picture initially, but for write you also need to write that particular data and for the read the data from the memory will come into MDR. So, for read the data, from memory data bus comes to MDR by activating MDRinE as I just now said. For write the data from MDR goes to data bus by activating the signal MDRoutE. So, these are the signals that are required for communicating with the external memory bus.

(Refer Slide Time: 17:56)



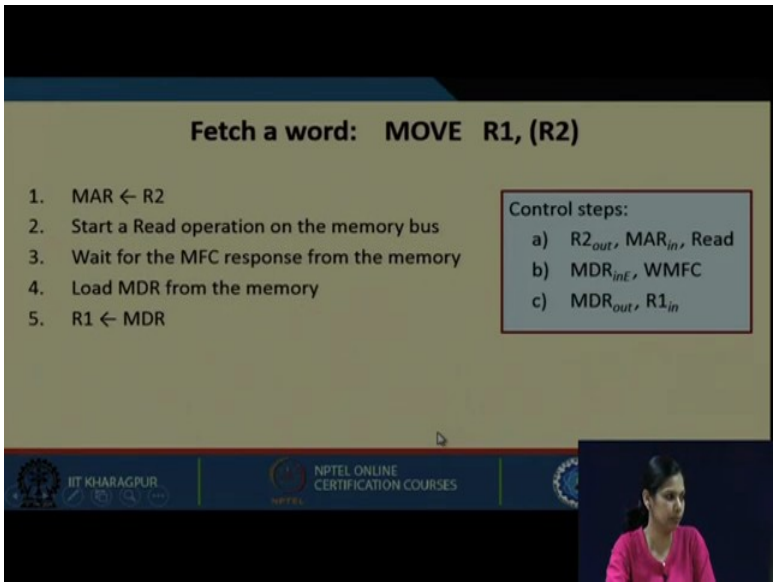
So, when the processor sends a read request it has to wait until the data is read from the memory and return into MDR. It is pretty obvious that the memory is slower than processor. Once the processor sends a read request it has to wait for some time till the data is available in MDR. To accommodate this variability in response time as there is no fixed response time that how much time it will be required, the processor has to wait until it receives an indication from memory that the read operation has been completed.

Please try to understand this. For a read operation, what we do is we put the address from where we need to read into MAR, and we activate the read control signal. Now, we really do not know at what time the data will be available, but the data will be available after sometime. So, till that time the process has to wait, but when the processor will know when a signal will be available from the memory to the processor that, the memory function has been completed. So, a signal is required to be sent from the memory to the processor to know that the data is now available in the particular register in that particular MDR. So, to accommodate this variability in response time the processor waits until it receives an indication from the memory. This indication is provided by a control signals known as memory function complete (MFC).

Whenever MFC is provided then we understand that now the data is available in MDR either way; for a write you have put the data into MDR, the address into MAR, and you have activated the write control signal; once the write is performed, it will inform. So,

when this signal is 1, it indicates that the content of specified location is read and are available on the data lines of the memory bus then the data can be made available to MDR because it is directly connected to MDR from the data lines of memory.

(Refer Slide Time: 20:42)



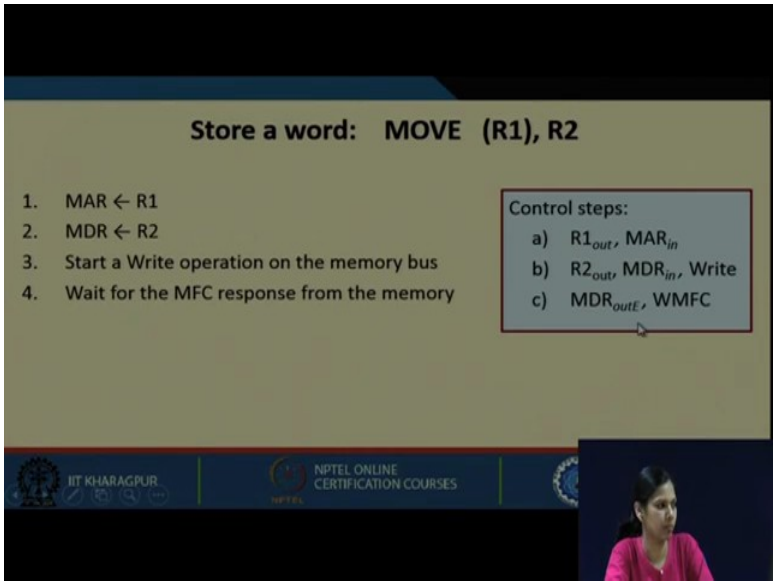
So, let us see what are the control steps or what are the time steps that are required for this particular operation. Fetch a word MOVE R1,(R2). (R2) is a memory location. So, what we need to do we have to put this particular memory address R2 value into MAR, and then we have to activate the read control signal. Once that is done then we need to wait for a signal that is MFC. Once that signal is available then from MDR the value can be move to R1.

So, let us see what are the steps that are followed. R2 goes to MAR we start a read operation on the memory bus, wait for the MFC response from the memory, load MDR from the memory and then from MDR it is transferred to R1. So, let us see the steps. First we do R2out and MARin, because in MAR the address should be present, and we activate the Read control signal. We perform all these things in one go, in one time step. After it is read the data will be available in MDR, how it will be available, it will be available through this signal MDRinE and we have to wait for MFC. Or the other way around if you think that when wait for MFC, this particular signal will get activated from the memory. Then automatically this MDRinE should be on made available such that the

data comes into MDR. And once the data is available in MDR it can be transferred to R2 through MDRout and R1in.

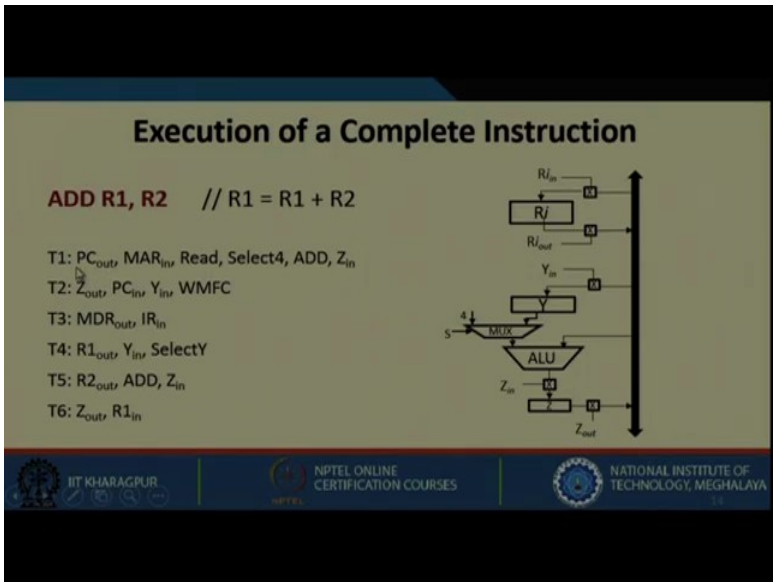
So, these are the following steps which will be required first we are making R2out putting it in MAR, using signal MARin, we activate the Read control signal. Then we make MDRinE, and we wait for MFC, when this signal comes and this is already on; that means, the data will be available in MDR. Once the data is available in MDR in this particular step, then we perform MDRout and R1in.

(Refer Slide Time: 23:23)



For storing a word, two things are required; one is that particular address where we need to store should be in MAR, and the data what we need to store should be in MDR. So, we are doing that. Here in this particular location R2 must be stored. So, R1 should be in a MAR and R2 should be in MDR because R2 is the data that needs to be stored by the address pointed by R1. So, R1 goes to MAR, R2 goes to MDR, start the Write operation on the memory bus, and wait for the MFC. So, in this case, we do R1out, MARin, R2out, MDRin. So, in MAR the address is present; in MDR the data is present, we perform write operation and then we do MDRoutE and wait for MFC.

(Refer Slide Time: 24:34)



Now, let us see the execution of a complete instruction. So, for executing a complete instruction what is required? We need to first fetch that instruction. In the same way we need to put the content of PC into MAR, we need to activate the Read control signal at the same time we have to increment the PC value, and then we have to wait for MFC because we are performing a Read. Once it is done and then the data that is present will be brought in through MDR, and then from MDR it will go to IR. So, we have to fetch the instruction in this particular case.

So, for fetching the instruction let us see what steps are required. So, we do PCout content of the PC stores the address of the next location that is to be executed. We do PCout, MARin, we activate the Read control signal. At the same time we Select4 because once you do PCout it is available in the B input. So, this particular diagram is not entire single bus architecture, but let say you have PC here in that. So, when you do PCout; that means, it is available in this bus and from this bus it is directly connected to one input of the ALU. So, it is available here. Once you do PCout, it is automatically available here and then you are putting that address in MAR activating the Read control signal.

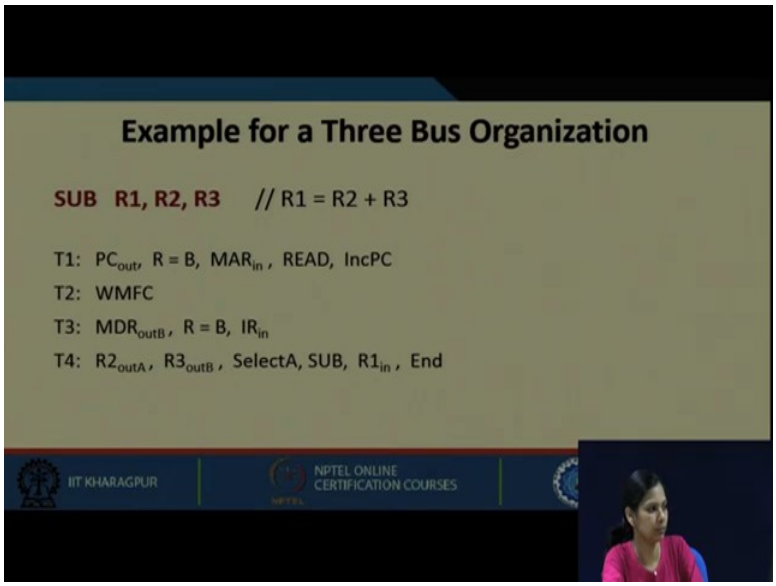
And at the same time, we are doing Select4. If we select 4, then this particular input will have 4. So, this is having PCout and this is having 4, and we do an ADD and then after adding we put the value in Zin, so that means in Z now you have the incremented PC

value. What needs to be done to this PC value, this PC value needs to be transferred to PC again that we will be doing the next step. So, in this particular step we are putting the value of the particular PC MAR, we are activating the Read control signal, through the select we are selecting 4. Now, we put it in Z, and next we do Zout and PCin. Now, PC contains the increment value, that is PC + 4, we have done through ALU.

We are also doing Yin. Why we are doing Yin we will be seeing it for the case of branch instruction that it is required. And then as we have done Read operation here, in the next we need to wait for MFC. Once this is asserted and this is performed in T2, then the data is available in MDRout. Now, you see we are not giving MDRinE here. We wait for MFC. Once this is performed, MDRout we perform because the instruction pointed by PC is now available in MDR, we are transferring that to IR. We are doing MDRout and IRin, that means in these three steps what we have done we have just fetched the instructions from memory.

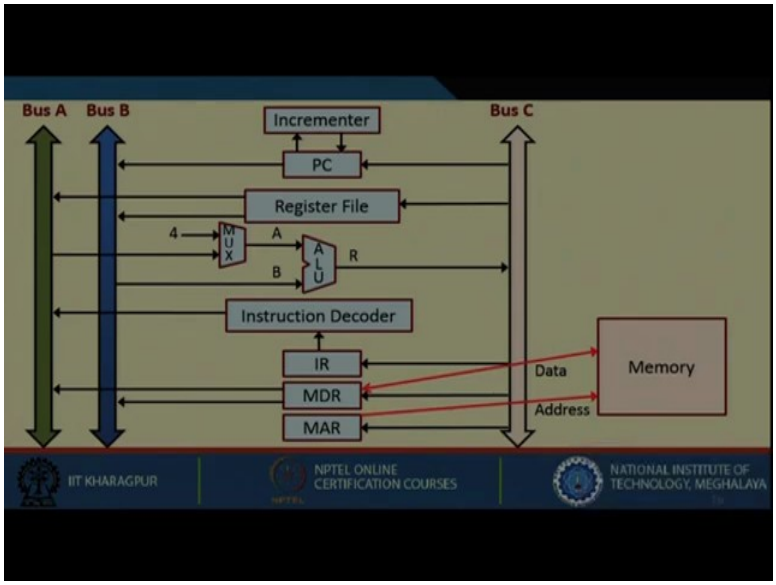
Now, what we need to do, we need to execute that instruction. For executing that instruction, in the same way, you have to bring R1 into one of the inputs of the ALU, and R2 into another input of the ALU. So, we have already performed these steps for an example previously. We do R1out, Yin. So, R1 is made available through the bus to Y and we do selectY. Similarly, we do R2out that is directly connected to ALU, and then we do ADD. As this is already selected, so we perform add here and then we perform Zin. So, the output of R1 + R2 is present in Z finally, it has to be brought into R1. So, we do Zout and R1in. So, for executing a complete instruction part of which we already knew it, so this is only fetch phase. So, every instruction will have these three steps in common for single bus architecture.

(Refer Slide Time: 30:08)



Let us move on with the three-bus architecture where we will be performing with operation SUB R1,R2,R3. So, R1 will be R2 minus R3. So, in this particular case same way we need to fetch from memory, and this is the bus organization.

(Refer Slide Time: 30:35)



Now, PC has got an incrementer, and from this register one input goes to A another input goes to B. The input which goes to A has to come through this MUX and made available to the A input of the ALU. And another input is directly connected, so one input is directly connected to the B input of ALU. So, with this architecture, now we will execute

this particular instruction. So, first we do PCout, R=B. So, R is a signal, we are making R equal to B. We do MARin. So, basically PCout when we do it is available through B bus. So, as it is available through this B input. So, R=B, MARin, Read and IncPC. So, we are performing it in one go.

In any case, if we have performed MARin, we have to wait for MFC. Once the signal is available to the processor, then in MDR the data will be available. And we do MDRout R=B and IRin. Let us see this. We do MDRoutB. So, MDRout can be made available to A also to B, but we are making MDRoutB then R=B and through this R, it is going to IRin. So, this is the path which it is following; we are doing MDRoutB and then we are doing R=B. Now, R=B means the B value through this R will be made available to C bus; and from this C bus, the data will go into IR.

Now, these three steps are required to fetch the instruction. Now to execute the instruction, as we already know that two registers can be read simultaneously and one register can be written simultaneously, so two register read and one write can be performed simultaneously. Similarly, for this instruction we require something like this; we need to read the value from R2 and R3 and we have to write the value back in R1. So, what we are doing R2outA. So, R2out will be available to A bus, R3outB it is available to B bus. We need to selectA, because it is connected the A bus, then we provide the operation that needs to be performed that is SUB. And then at the same time we do R1in; two different reads and one write is performed in one particular step and finally End.

Let us see this. What we are doing we are doing R2out we are doing R2outA. So, from this register file we are doing R2outA. Then R3outB, B is directly connected no problem. As A is connected to this MUX we are doing selectA, so it is connected, we are performing SUB and we are doing R1 in that is what we are doing. R2outA, R3outB, selectA, SUB, and R1in and finally, we End. So, what we can see that for performing the same operation earlier we were requiring three steps, that is now reduced to one single step; in one single step, we are able to perform this SUB operation or any ALU operation because at the same time we are reading from two registers and writing into one register.

So, we have come to the end of this lecture. And in the next lecture, we will be looking forward to how we can execute various other instructions.

Thank you.