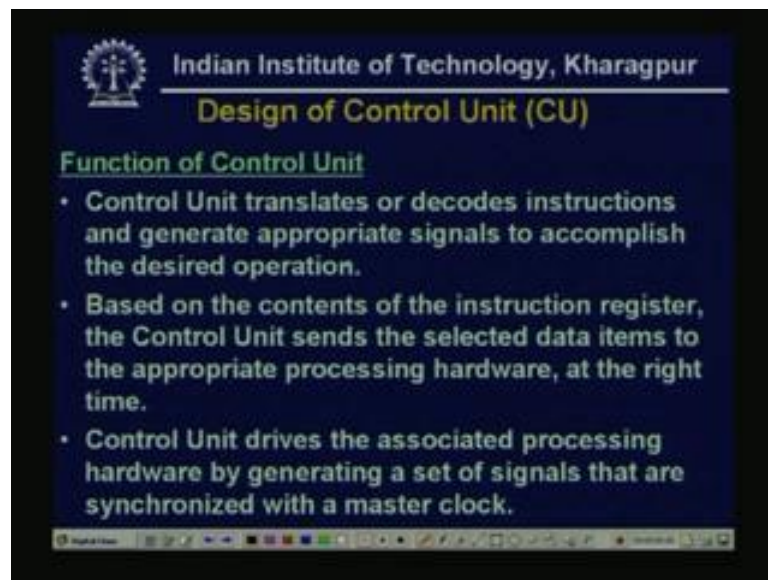


Digital System Design
Prof. D. Roychoudhury
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture - 36
Design of Computer Instruction Set and the CPU (Contd.)

We are discussing the design of a CPU, in the last class we have read the design of Arithmetic Logic Unit the ALU and today we will read the design of the control unit. Control unit as its name implies actually it is the unit of mastering the interfaces of all other units of a CPU, so first we see what is the function of control unit?

(Refer Slide Time: 01:27)



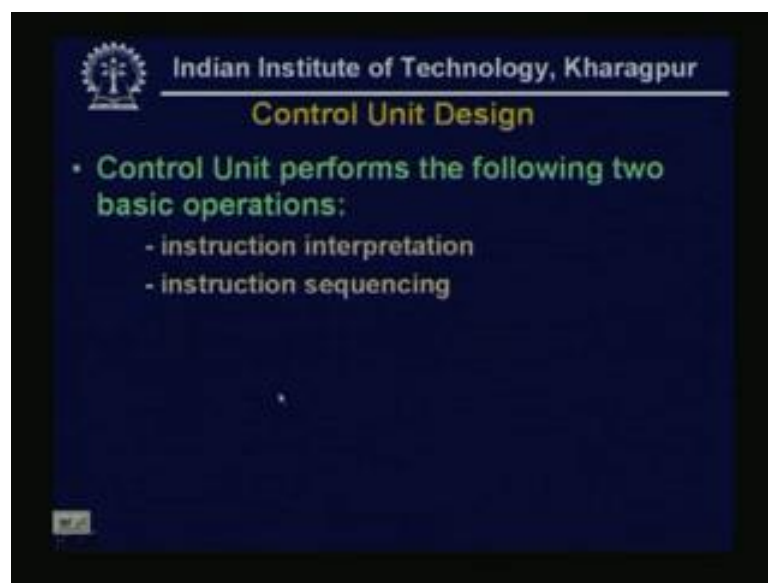
So, control unit translates or decodes instructions and generate appropriate signals to accomplish the desired operations. So, you have read the design of instruction set there the instruction is fetched and then it is decoded. So, after this decoding is done by the CU the control unit and then it generates appropriate signal for performing that particular operation defined by the instruction.

Now, based on the contents of the instruction register the control unit sends the selected data items to the appropriate processing hardware at the right time. To operate or to perform a operation we need some data, so what type of instruction, it is after decoding the by CU. Then, the data is accessed from that particular hardware and after operation

the result is again send to one specified hardware and all this task are supervised by the CU.

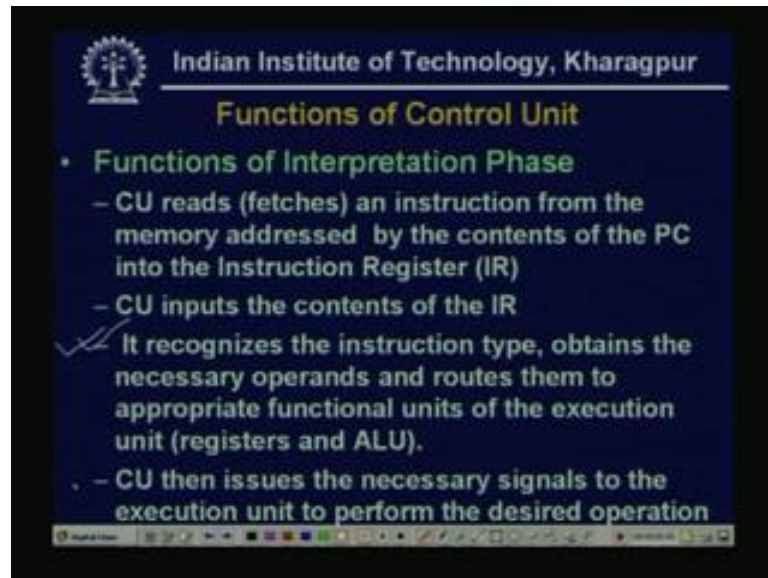
Now, control unit drives the associated processing hardware by generating a set of signals that are synchronized with a master clock. This is one of the important task of control unit that it generates the timing signal, normally for all digital systems there is a master clock and all other clocks are generated from this master clock in appropriate timing and control unit does this job.

(Refer Slide Time: 03:43)



Control unit performs the following two basic operation, actually it works in two phases rather one is called the instruction interpretation phase and another is instruction sequencing.

(Refer Slide Time: 04:02)



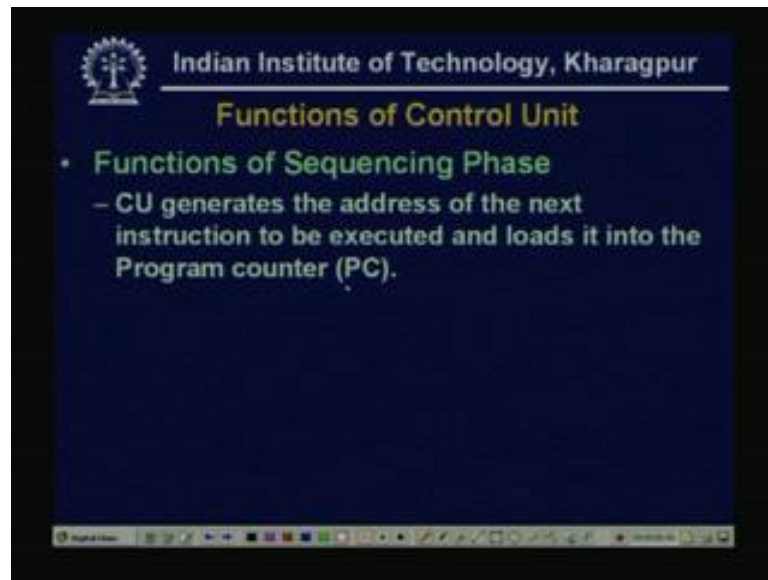
So, first we see the phase one or the instruction interpretation phase, so control unit reads or fetches an instruction from the memory addressed by the contents of the program counter into the instruction register. As already in the last class, we were we have read that the current address is always the content of the program counter. So, the addressed specified by the contents of program counter in to the instruction register, the control unit first fetches the instruction, then the in control unit inputs the contents of the instruction register.

It recognizes the instruction type, first what type of instruction, it is obtains the necessary operands and routes them to appropriate functional units of the execution unit registers and ALU are the execution units. So, what it does, this is the important task it does the control unit, first it recognizes the instruction type with it is necessary, because how many data are needed or to perform that particular operation.

First the CU understands that thing, then from where the data or the operands will be available. So, it gives the signal or to obtain that particular operand from particular unit. After that it now, we have do, we have to perform that operation after getting the operands. Then, it the appropriate function will be functional units, say the arithmetic logic unit has to perform some operation or it as to be set into some registers some intermediate results or the final results.

Then, the CU gives a signal to registers and ALU to perform that particular operation, so it is the first job. CU then issues the necessary signals to the execution unit to perform the desired operation. So, actually this two are related that after getting that signals after getting the operands the CU issues the signal or ready signal to perform the operation.

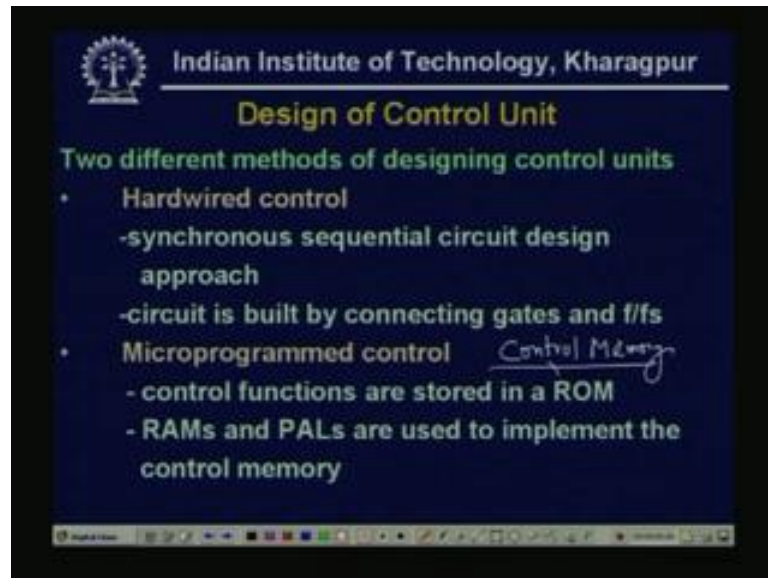
(Refer Slide Time: 07:10)



Now, functions of sequencing phase, the control unit generates the address of the next instruction to be executed and load it into the program counter. So, all we know that programs are normally the sequential programs, see it is the instructions are sequentially written, so during execution that it takes sequentially that one instruction after another.

So, when one instruction is executed or the execution is over, then the CU the control units, generates the address of the next instruction. And then the address that particular address is loaded into the program counter always the program counter contains the current address of the instruction or addresses of the current instruction to be executed.

(Refer Slide Time: 08:08)



Now, we see that how this control unit, so how this control units are designed can be implemented. We have seen that these are the functions mainly the instruction interpretation phase, some functions it has to be performed and the see instruction sequencing part.

Now, what type of hardware will be necessary, or to implement this control unit or to perform those operations. Normally two different methods of designing control units, one is the hardwired control, another is a microprogrammed control. Normally, the hardwired control is a synchronous sequential circuit design approach. The hardwired control evolved from the fact that the final circuit is built by physically connecting the components such as Gates and flip flops.

Now, in the micro program control unit, all control functions are stored in a ROM inside the control unit, now this memory is called the control memory. So, this memory is called this is very important ROM, this is called the control memory were the all control functions are stored, this is called the control memory.

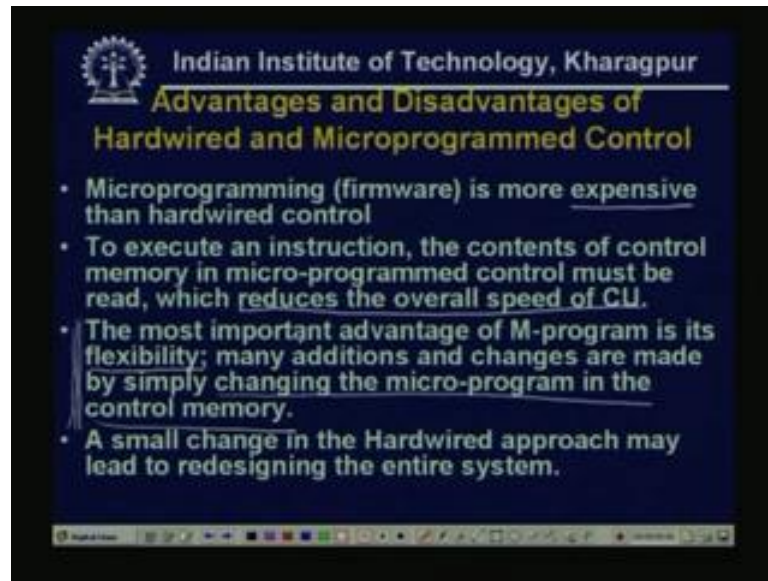
Normally RAMs and the PALs, the programmable array logic already we read are used to implement the control memory. Now, the words in this memory are called the control words and they specify the control functions to be performed by the control unit. Now, the control words are fetched from the control memory and the bits are routed to appropriate functional units to enable various gates.

The control unit is given the all enable signals to other units, the arithmetic logic unit or the registers and to start their operation. Now, these control words are stored in the control memory, so the control unit is taking that control words from the control memory. And then they are routed this bits are routed to appropriate function units like registers ALUs etcetera.

Now, this instruction is executed in this way, it is called the microprogramming and sometimes this microprogramming is called the firmware. As another approach is a hardware approach, actually physical connections are available, whereas the microprogramming we can tell as if this is the combination of software and hardware and that is why it is called, we are calling this is a firmware where the software... Because here the functional the functions or the words control words are stored in a memory. So, these are nothing but analogous to software.

And then, it is routed by the control unit to the appropriate functional units to perform the operation defined by one instruction, then this functional units are actually implemented by hardware. Moreover the signals routed to the units or the data routed these are all normally by hardware, we will see that thing that they actually the clock signals or the data bus these are all the hardware thing. So, actually that is why the microprogramming is sometimes called the firmware. Now, we see what are the advantages, or disadvantages of hardwired and the microprogrammed, control as there are two approaches of control unit design.

(Refer Slide Time: 11:24)



Now, microprogramming is more expensive than hardwired control, because it has to access the memory that control memory where all the functions are stored. To execute an instruction the contents of control memory in micro program control must be read from the memory which is a memory access which reduces the overall speed of CU. So, not only it is expensive not only it is expensive, but also that it reduces the overall speed of control unit. Because always, accessing a memory means it will take some time, so here the speed is reduced.

The most important advantage of micro program is its flexibility, because it is actually what is to be done the functions that are stored in a control memory. So, only changing the software or the functions in the memory we can change the instructions, so the most important advantage is its flexibility.

Now, many additions and changes are made by simply changing the micro program in the control memory. So, the program that is stored in the control memory that we can easily change and that will change the overall design procedure or actually the processing. So, this is this flexibility is here.

On the other hand, if it is a hardwired approach, then actually it is a fix thing, as hardwired already we have define the hardwired approach is actually physically connecting the gates already the design is fixed. Now, if we want to change, then actually we have redesign the entire systems which will be very expensive as well as

time consuming and there is no flexibility at all. So, this is the one important advantage of the micro program control design, whereas the some other disadvantages as already I mentioned is expensive and overall it is very slow process.

(Refer Slide Time: 16:20)

Indian Institute of Technology, Kharagpur
Implementation of Register Transfer of Control Unit

Register Transfer is the fundamental concept in the design of Control Unit

$R1 \leftarrow R0$; $R0, R1$ — 16-bit register

- It denotes a register transfer
- but it does not indicate the number of bits to be transferred

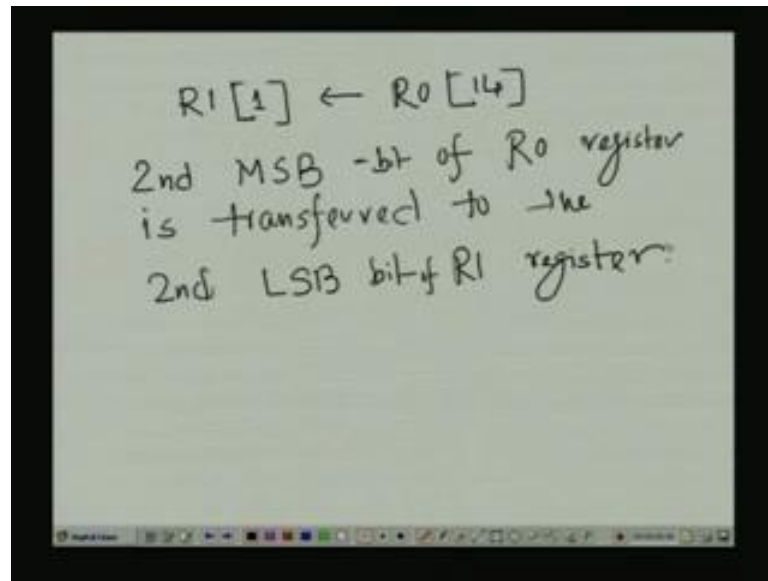
- Declare registers $R0 [16], R1 [16]$; $R0, R1$ are 16-bit register
- $R1[1] \leftarrow R0[14]$
- $E: R1 \leftarrow R0, E$ is enable signal
 - enable signal controls the transfer
 - E may be a function of more than one variable

Now, implementation of registers, transfer of control unit, so register transfer is the fundamental concept in the design of control unit. See, how we implement it or how we declare that thing, because it is a firmware for micro program. So, how we declare the register transfers, see we have two registers assume we have two registers R 1 and R 0, so R 0 R 1 are two registers. Now, I want to transfer the content of register R 0 to R 1, so normally this is the notation is used that R 0 to R 1.

So, by this notation it denotes a register transfer, but it does not indicate the number of bits to be transferred. Say whether, this register is a 8 bit register or whether it is 32 bit register that is never specified by this notation or by this notation it is not understood that how many bits are transferred.

So, for this purposes we have to declare the two registers in advance, that how many bits or what is the construction of that registers how many bits are involved. So, for that we use declare registers see here declare registers $R0 16 R1 16$, this means $R0 R1$ are of 16 bit registers. Now, by this second notation say $R0 14$ to $R1 1$; that means, here one only particular bit is transferred.

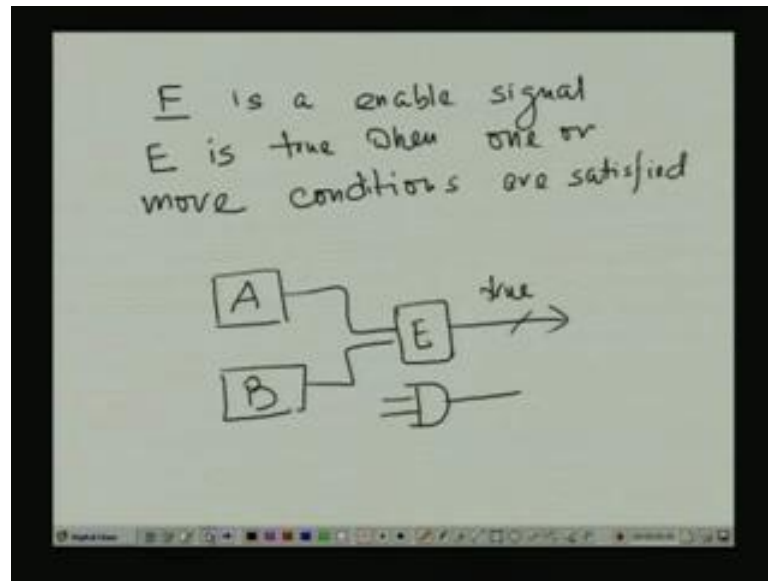
(Refer Slide Time: 18:54)



So, R 1 1 by R 1 1 is assigned to R 0 14, this means the second MSB bit most significant bit MSB bit of R 0 register is transferred to the second LSB bit the least significant bit of register of R 1 register. So, not only the whole, all the bits of the register value are transferred in this way we can one particular bit can be transferred.

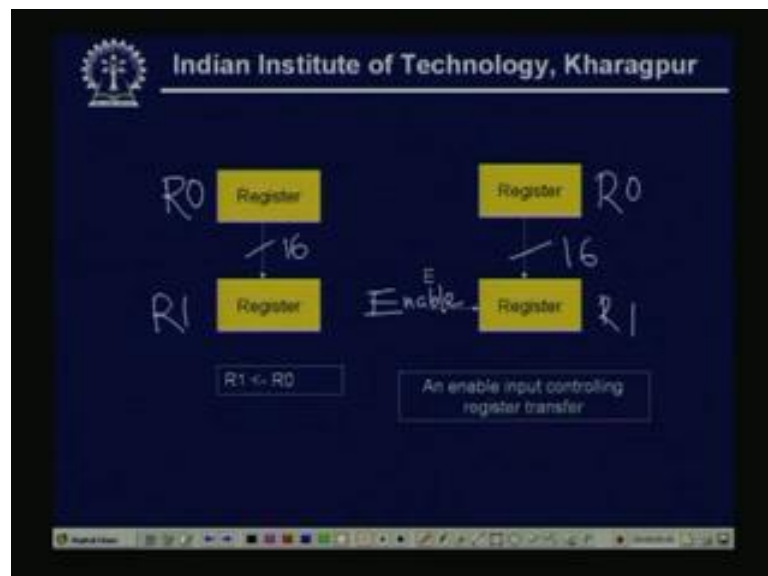
Now, this is actually unconditional transfer this is R 0 14 into R 1 1 or simply R 1 simply that a whole R 1 bit all the bits of R 0 is transferred to R 0 R 1 that also we can do. Now, this is the condition say E R 0 to R 1, here E is a enable signal. That means, enable signal controls the transfer. So, if or E may be function of more than one variable.

(Refer Slide Time: 21:15)



So, E can be generated as E is an enable signal, so that will be on E is true, when one or more conditions are satisfied. So, this is a control condition that is satisfied. Say I have two such conditions, say condition is A and B if A and B, then only E is true, so actually in that case E is actually one, say one AND gate this type of gate, so this is an enable signal that is true. So, an enable signal controls the transfer E may be a function of more than one variable that there can be more than one input of the condition.

(Refer Slide Time: 22:43)

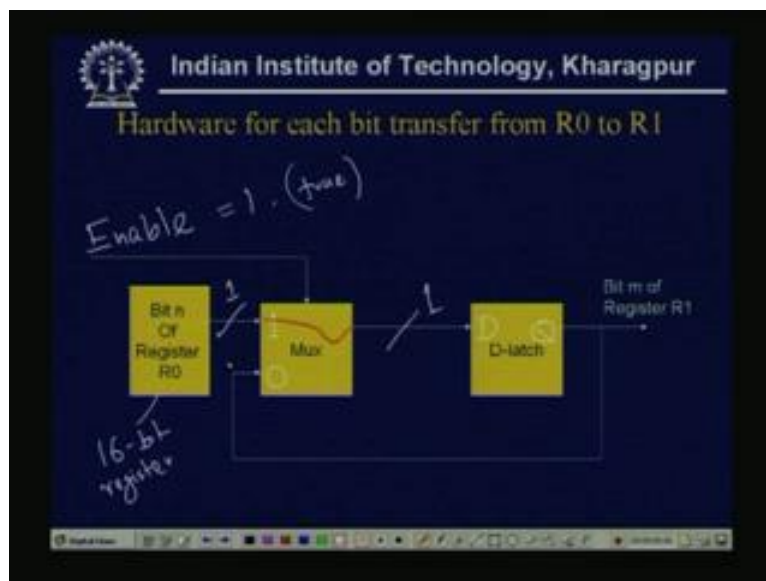


So, now if we denote the 16 bit register transfer or the enable input controlling register transfer. Say this is a register R 0 this is my register R 0 and this is register R 1. Now, to denote that this is a 16 bit register the bus is denoted as 16, so this is a 16 bit register transfer from R 0 to R 1.

Now, an enable input controlling register transfer, so again this is a register R 0 register R 1 16 bit and this is my enable signal E. And actually here all the conditions is the output of the conditions after checked that means, whether E is true or E is E is false that will be after processing the conditions. It will give that because this is a control register transfer.

Now, if we that how actually what will the hardware for each bit transfer, because earlier slide what we have seen that, this is the notation that hardware in block diagram level. The hardware notation that all the that all the 16 bits are transferred, now actually when physically it is being transferred how that bit wise it is being transferred.

(Refer Slide Time: 24:13)

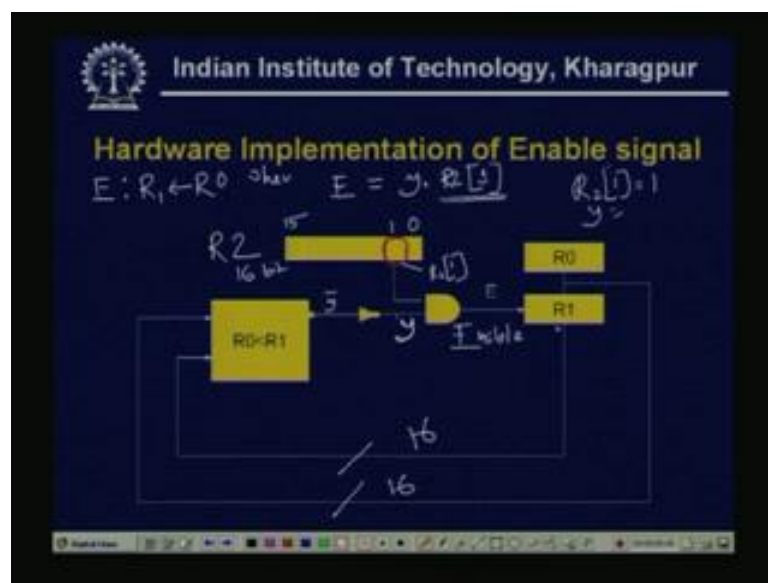


So, each bit transfer, now this is a bit n of register R 0; that means, this is a register R 0 from their 1 bit is coming here. So, this is this is actually 1 bit by bit transfer. So, this is 1 bit this is this, R 0 is a 16 bit register. Now, see the register can be implemented by our flip flops, so these are the output of the flip flop that is only 1 bit. So, this 1 bit is coming the this multiplexer, so multiplexer of 1 and 0. So, again this is my enable line, so in this case if the enable is 1; that means, if it true it is true, then this register bit OR 1 bit from

the register will be output at to the MUX output this is also 1 bit. Then it is latched, because it is a D latch. So, this is my D input, this is the Q output and it will go to bit m of register R 1.

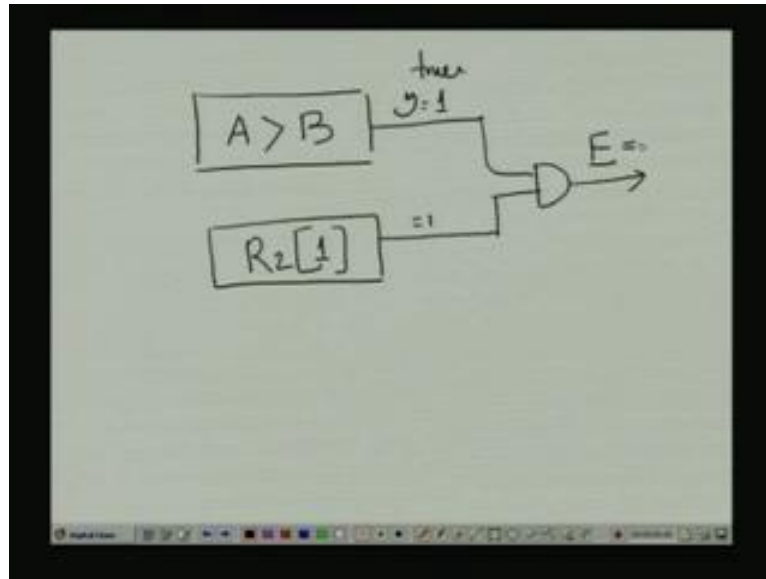
And, the other is the zero input of the multiplexer, when it is enable is 0, then it will be actually the same value is again latched. So, in this can be the actual bit by bit, how the bit transfer is occurred from a n bit register to another n bit register that is this is the hardware for that.

(Refer Slide Time: 27:27)



Now, the hardware of the enable signal, see here we are considering an enable signal E like that E say my E controlled the R 0 to R 1, where E is some y dot R 2 1. That means, here the enable signal is a function of the one variable. See this is a R 2 is a register and the second LSB bit of R 2 if it is 1; that means if R 2 1 equal to 1 R 2 1 equal to 1 and sum y value again that y can be the output of some other circuitry that actually implementing the condition. So, if we draw the hardware diagram say I have some condition to be checked.

(Refer Slide Time: 28:44)



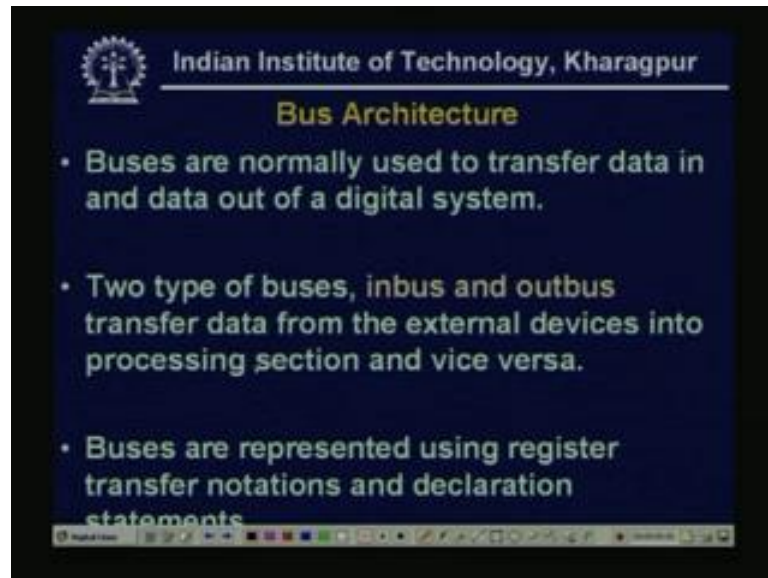
Say I have two values A and B, if A greater than B, then only y is y is 1 means true say this is my condition. And another is say this is the R 2 registers the second LSB bit R 2 1, if it is 1 if this value is 1. Then this two are added say then it will generate a E equal to 1; that means, true. So, this is my enable signal generation E is 1.

So, there are y dot R 2 1, then E equal to 1, say this is a R 2 is a 16 bit registers say this is my R 2 this is 16 bit register and this is 0 bit 1 bit 15 bit say this is my R 2 1 this is my R 2 this is my R 2 1. The second LSB and this is my y, y is the output of this thing and say it is coming from some register output, where it is the condition is whether R 1 R 0 is less than R 1 or what just I have show A less than B or this type of condition as to be satisfied.

Now this is y then E equal to 1, when R 2 1 is one any y is equal to 1, then E is 1, so this is my enable line this is my enable signal. Now, when enable signal is on then only that R 0 is transferred to R 1, so R 0 to R 1 that will and these are all 16 bit transfer these are 16 bit and in this way it will occur.

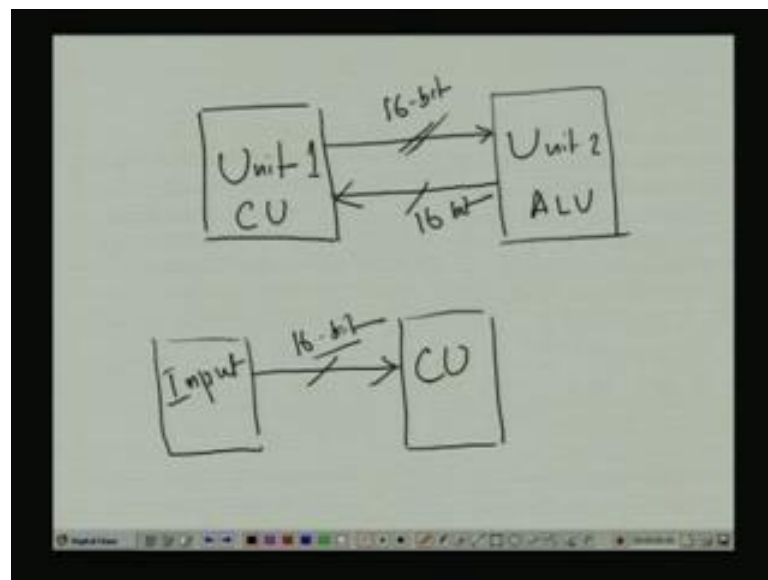
So, mainly the main point is that actually R 1, the register has another input and this is actually enable signal or that this is a control line that whether the register transfer will occur or not that is controlled by this E line that enable line.

(Refer Slide Time: 32:14)



Now, another very important thing is there in control unit design and that is called the bus architecture. So, what is a bus? Buses are normally used to transfer data in and out of a digital system.

(Refer Slide Time: 32:40)



So, say we can tell that this is actually there are two units say I have two units inside a CPU say unit one that can be my control unit say and say this is my ALU the arithmetic logic unit that is my unit two. Say ALU or it can be any other say this can be one input device and this can be my the whole CPU, I can write or the say the CU.

Now, here the control unit say is accessing something from here this can be of this type normally this is a. So, this is a 16 bit line, then we are calling this is 16 bit bus. Similarly, here also this can be 16 bit interface line. So, this is a bus, buses are normally used to transfer data in and data out of a digital system.

Now, two type of buses, inbus and outbus transfers data from the external devices into processing section and vice versa. The external devices are normally the input device and output device and the internal device are our different units the arithmetic logic unit the control unit or the whole CPU we can think.

And then, the inbus and outbus, and normally the inbus called the transfers data from the external devices into the processing section and the and vice versa. So, buses are represented using register transfer notations and declaration statements, so again how we represent the bus, so this is bus notations.

(Refer Slide Time: 35:14)

Indian Institute of Technology, Kharagpur

Bus Notations

- **Declare inbus[16], outbus[16]**
 - indicates that the digital system has two 16-bit wide data buses
- **R0 ← inbus** *transfer*
 - means data on the inbus is transferred into register R0 in the next clock
- **Outbus = R1[15:8]**
 - means the high order 8 bits of the 16 bit register R1 are made available on the outbus for one clock period

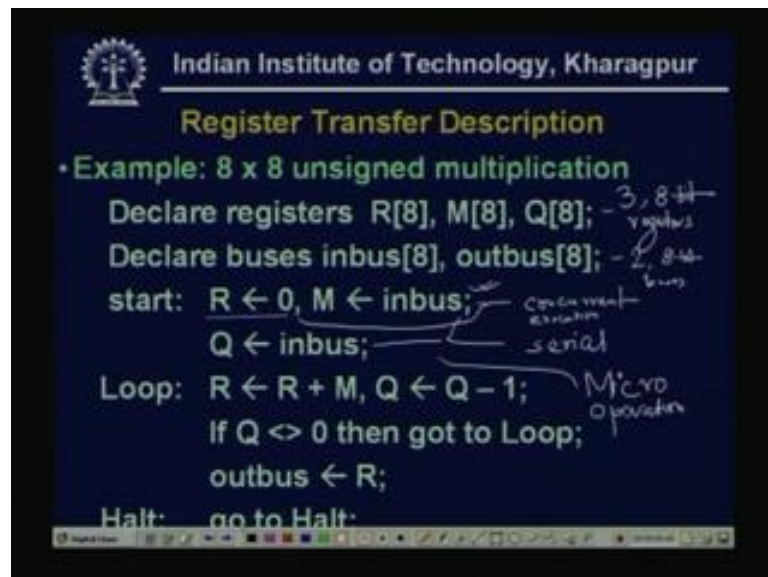
Say I have just now I mentioned we have two type of bus inbus and outbus. So, declare again here it is just like the register transfer declare inbus 16 outbus 16, it indicates that the digital system has 2 16 bit wide data buses.

Now, inbus to R 0 means data on the inbus is transferred into the register R 0 in the next clock. See this is a transfer, this notation is normally this is a transfer, here it is a equal. So, R 1 15 colon 8 is assigned to outbus means the high order 8 bits, because 8 to 15

means the higher or the 8 MSBs, so high order 8 bits of the 16 bit register R 1 are made available on the outbus R 1 clock period.

See here it is a transfer, this notation that by this arrow sign we are meaning that the data available in the inbus, that is being transferred to the register R 0. Here, it is a assign an equal; that means, the high order 8 bits that are available as a as the outbus for R 1 clock period. So, that these are the different notations normally used for bus. Now, what will be the register transfer description, when we use the bus or the registers together we take one example.

(Refer Slide Time: 37:13)



See this is a 8 by 8 unsigned multiplication, now we declare registers R 8 M 8 Q 8; that means, we have 3 8 bit registers say we have 3 8 bit registers declare buses inbus 8 outbus 8. That means, we have two 8 bit buses, now start see 0 to R inbus to M means that the operations can be performed this two operations can be performed simultaneously. That means it is a reset the register R is set to 0 this is a reset register and inbus; that means, data available to inbus is transferred to the register M and these two are parallely performed.

Now, inbus to Q means the data available to inbus that is also transferred to the register Q as a general rule a comma inserted between operation can be executed concurrently. That means, this is this comma means this is a concurrent execution or parallel

execution; that means, these two transfers are being performed parallelly, this is a concurrent execution.

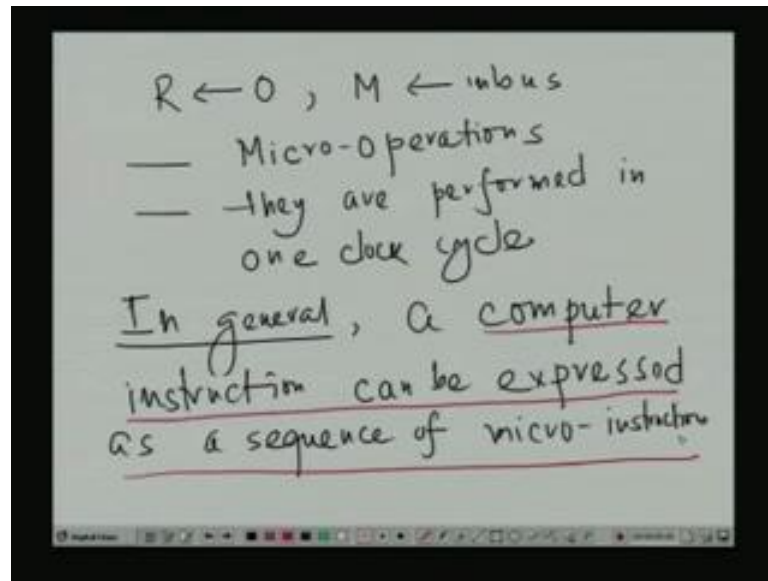
And, a semicolon between two transfers normally they are performed serially, so semicolon and this semicolon mean this semicolon means it is a serial execution. After this thing has been performed then only this transfer will occur that is that data on the inbus will be transferred to register Q.

Now, this restriction is primarily due to the data path provided in the hardware, now as there is only one input bus. So, the operation inbus to M and inbus to Q cannot be performed simultaneously, because there are only one 8 bit inbus, so after these data these data is transferred to M. Then in the next clock cycle the inbus is transferred to the register Q. So, these two operations are carried out serially.

However, one of these operations may be overlapped with the operations 0 to R, because this is a separate register is involved that the register R is reset to 0 or the content of the register value should be 0. So, that we can these transfer can be performed concurrently to any one of these transfer inbus to M or inbus to Q, because it will not affect the other one.

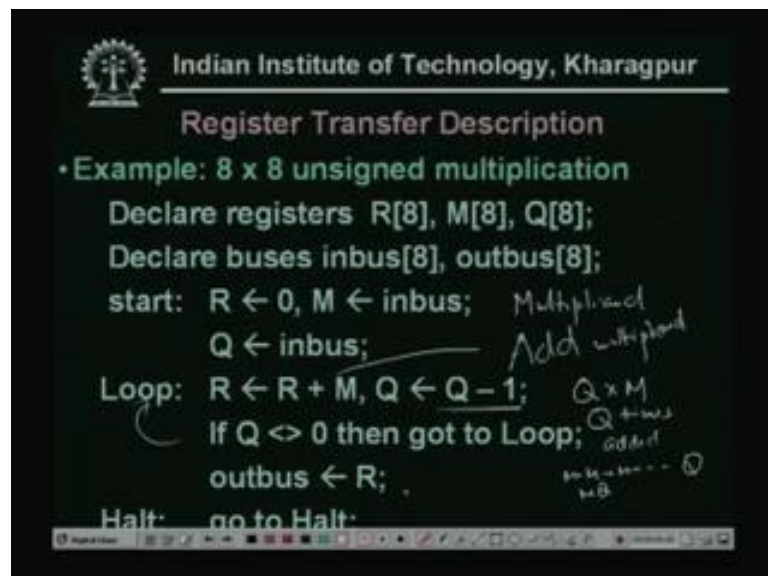
But here as inbus is common, so we can concurrently we cannot do this transfer inbus to M or inbus to Q these are this must be performed serially. And, these operations are called the micro-operations all these transfers these are called the micro-operations because they can be completed in one clock cycle.

(Refer Slide Time: 42:27)



So, this micro-operations, so first thing is 0 to R or say M to inbus these are called the micro-operations and they are performed in one clock cycle. Now, in general we can tell that a computer instruction can be expressed instruction can be expressed as a sequence of micro-instructions. So, this is one very important concepts that a computer a instruction it can be expressed as a sequence of micro-instructions and micro-instructions are nothing but that this type of register transfer or the bus transfer.

(Refer Slide Time: 44:23)



Now, another continuation of the program is that now register the two registers are added. So, actually this is a clear register and inbus to M means see the multiplicand goes to the register M the multiplicand transferred and then transfer the multiplier in the next clock cycle another operand goes to Q. Now inbus to Q, R plus M to R, so add multiplicand this is the add multiplicand and the result is kept to the register R, because initially it was 0. So, as if that is that contains the result.

Now, multiplier is decremented by one because multiplication is nothing but that how many times we will add say we are actually doing Q into M Q into M, so this is a Q times M will be added. So, what we are doing, initial that one addition after one addition Q should be decremented the multiplier, then it is decremented.

And, we have to continue this up to Q times, because this is Q into M is nothing but Q times Q times addition M should be added Q times M plus M plus M plus up to Q times that is MQ. So, mainly the normal concept of our multiplication it has been that if Q naught equal to 0. Then go to rule and that means, Q times M should be added and the result will kept in R and then R to outbus. That means, the result is transferred to the outbus and halt and go to halt. So, this is the micro sequence of micro-instructions for the program of a 8 by 8 unsigned multiplication.

(Refer Slide Time: 47:16)

Indian Institute of Technology, Kharagpur

Addition in Single Bus Architecture

• Example: Addition $R0 \leftarrow R1 + R2$

$B1 \leftarrow R1$; $B2 \leftarrow R2$; $R0 \leftarrow R1 + R2$

1st Clock cycle : The contents of R1 are moved to buffer register B1 of ALU.

2nd Clock cycle : Contents of R2 are moved to buffer register B2 of ALU.

3rd Clock cycle : The sum generated by ALU is loaded into R0.

Now, we see that bus structure again we are we are taking one example to explain the different type of bus structure available that we can that is normally implemented. So,

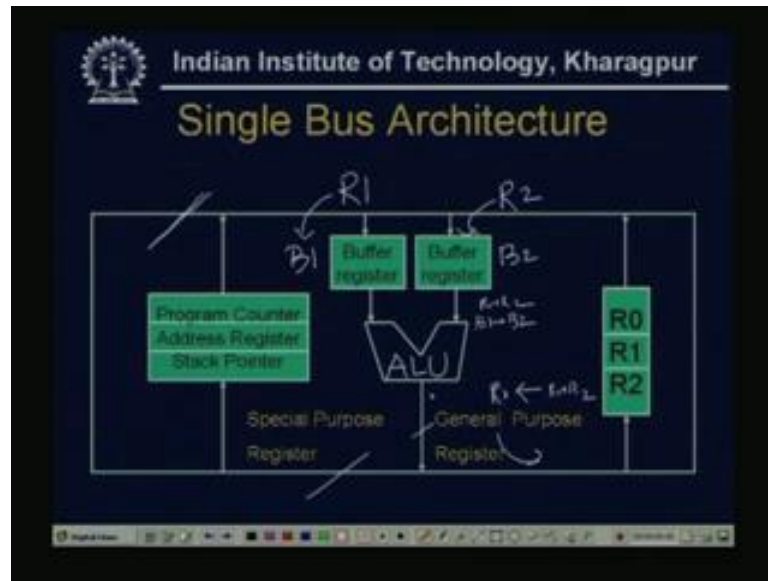
one simple addition say the two register contains R 1 and R 2 is added and the result is transferred to R 0, that means, R 1 plus R 0 R 1 R 2 is transferred to R 0.

Now, if we see that just to perform these addition using three registers, what will be the step by step by procedure. So, in the first clock cycle, see in the first time step we can tell the contents of R 1 are moved to buffer register B 1 of the ALU. So, I have to add R 1 and R 2, so what we are doing first R 1 is transferred to B 1. So, B 1 is my buffer register of ALU, because the addition must be performed by ALU. So, we have to transfer all this thing to ALU, so first this is transferred.

In the second clock cycle, so it will it must be separated by a semicolon, because it is serial. So, B 2 contains the R 2 as it is single bus structure that is why we need two clock cycle to load the operand in the buffers of the ALU. So, that is why this is R 1 to B 1, R 2 to B 2 separated by a semicolon, so contains of R 2 are moved to buffer register B 2 of ALU.

And, in the third clock cycle the addition is performed the sum generated by ALU is loaded into R 0. That means, again that R 1 plus R 2 now already ALU has done this addition and that sum is transferred to the register R 0. So, we have taken only the registers no bus involved only the buses are for the transferring the operands and the results. So, if we use a single bus architecture, we need at least three clock cycle, because we have to do the serially we have to transfer the operands and the results. So, this is the hardware for the single bus architecture.

(Refer Slide Time: 50:42)

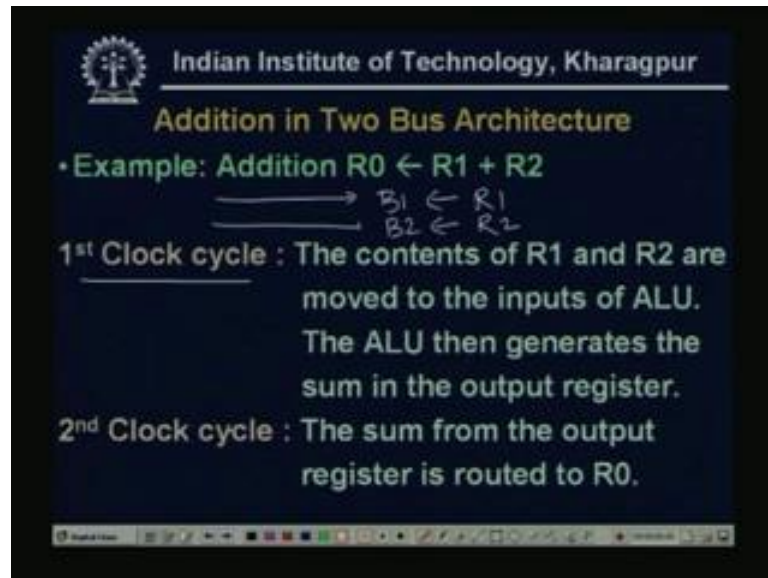


Say I have a program counter address register stack pointer, see this is my ALU and here actually this is the general purpose registers I have denoted only 3 actually it can be many more. So, R 0 R 1 R 2 are my register pool and these are buffer registers again buffer register can be many here only two we have shown.

So, first that R 1 should come to buffer one if this is my B 1, then in the first clock cycle R 1 will come here, so this is a R 1 to B 1. Similarly, in the next clock cycle R 2 will come the value of R 2, so this is R 2 to B 2. And then, that two values will be added so, actually this is B 1 plus B 2 that same thing as R 1 plus R 2 and then it will go to R 0.

So, actually this value will go to this will be a R 0 to R 1 plus R 2, so this is a single bus same bus, whether it is a this time it is a taking the operand from the register to the buffer register of ALU. And after the addition, the result is transferred to another register R 0. So, by the same bus it is doing that thing.

(Refer Slide Time: 53:10)



Indian Institute of Technology, Kharagpur

Addition in Two Bus Architecture

• Example: Addition $R0 \leftarrow R1 + R2$

$B1 \leftarrow R1$
 $B2 \leftarrow R2$

1st Clock cycle : The contents of R1 and R2 are moved to the inputs of ALU. The ALU then generates the sum in the output register.

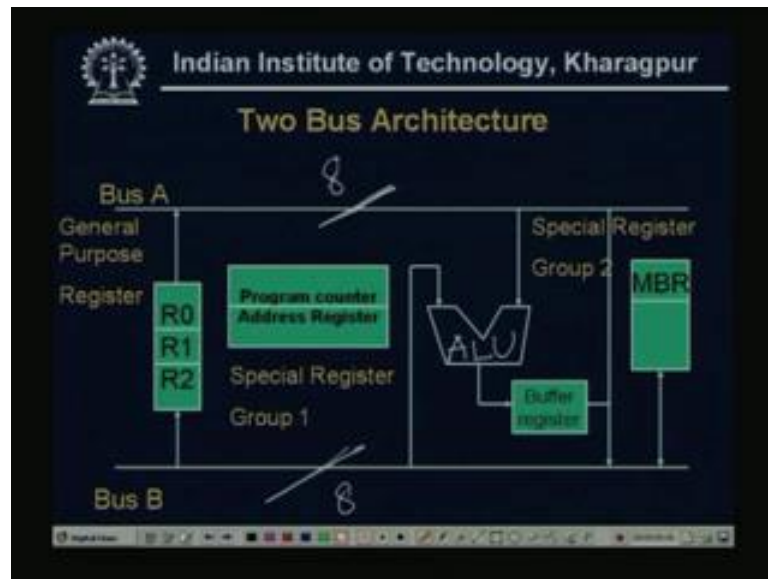
2nd Clock cycle : The sum from the output register is routed to R0.

Now, if we do the same thing that same addition by using a two bus architecture now instead of one bus say as if in the architecture there are two buses available. So, now we see that what will be the advantage or disadvantage, see there are two buses.

So, the contents of R 1 and R 2 are moved to the inputs of ALU concurrently as there are two buses available. So, one bus will take the R 1 buffer B 1 R 1 to B 1, another bus will take R 2 to B 2. So, this will be in the same clock cycle concurrently I can do this thing concurrently and the ALU, then generates the sum in the output register. In the second clock cycle, because the two buses are busy for transferring these content of R 1 and R 2 to B 1 to B 2.

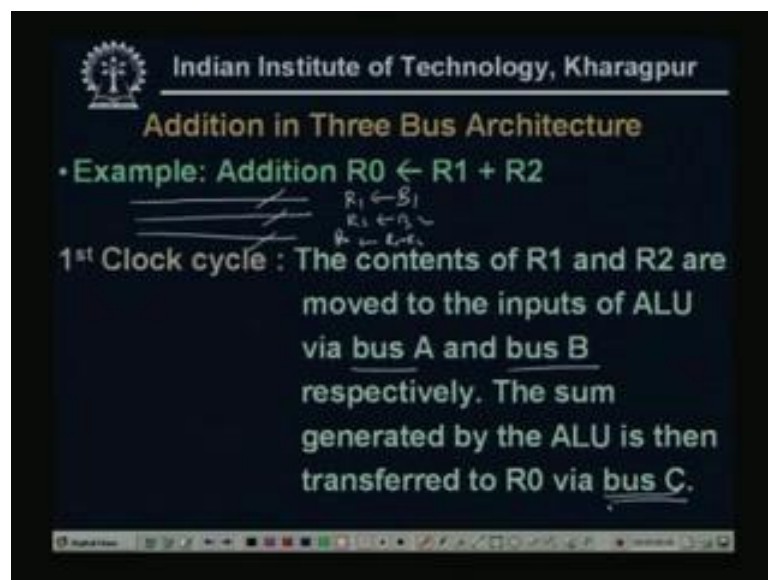
So, in the second clock cycle the sum from the output register is routed to R 0, so we need here two clock cycle. So, what will be the advantage, the advantage is that for the single bus architecture we need three clock cycles to do this addition represented by this micro-instructions or that execution of this micro-instruction whereas for two bus architecture we need only two clock cycles. But parallelly, as it is a two bus so; obviously, it needs the more hardware, so its hardware is more, but it is a fast thing. Now, the performs of two bus architecture can be improved by adding a three bus architecture, so we see that if it is a three bus architecture

(Refer Slide Time: 55:43)



So, this is a architecture of a two bus, the same thing there is a register pool again the special register group program counter address register. And, that this is a memory buffer register again this is a pool and actually these are two 8 buses, so these are the two 8 bit buses used.

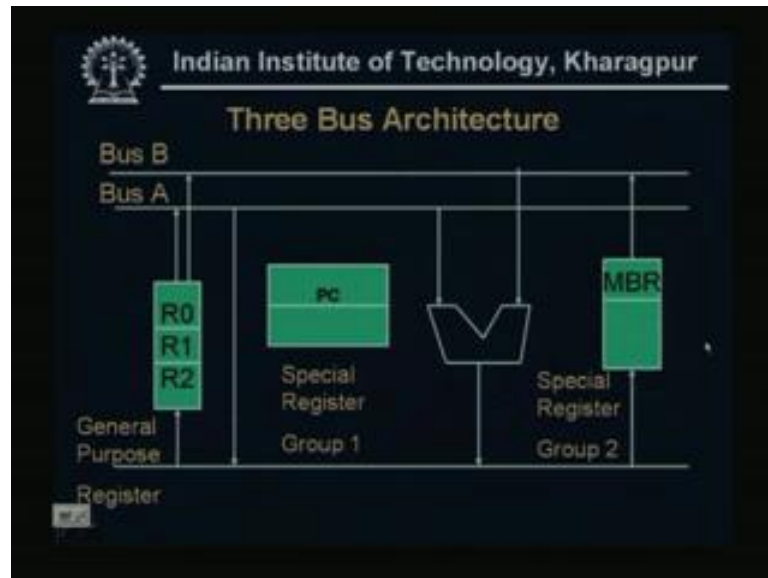
(Refer Slide Time: 56:17)



So, now it can be further improved by three bus architecture, say if it is a three buses are available. So, what we will be do, doing three buses. So, one will do that transfer R 1 to R 0 R 1 to B 1. Second will do R 2 to B 2 and third will do the result R 1 plus R 2 to R 0, so in the same clock cycle the contents of R 1 and R 2 are moved to the inputs of ALU via bus A and bus B respectively.

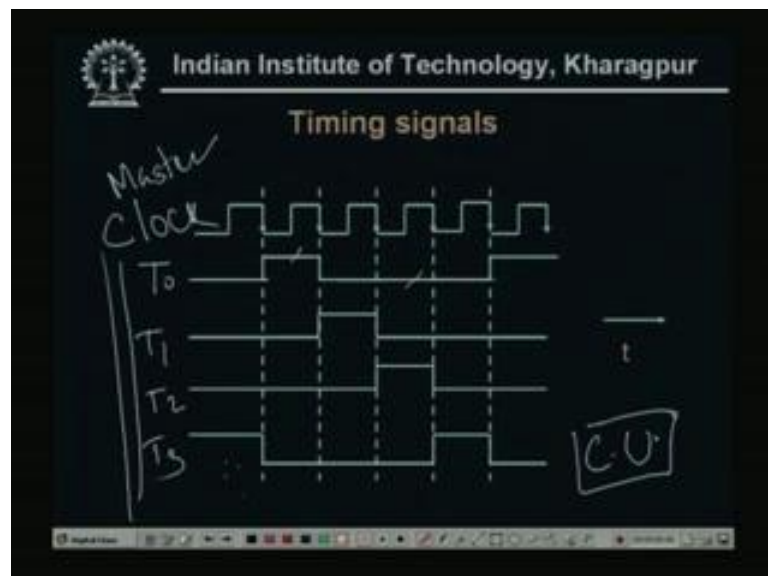
The sum generated by the ALU is then transferred to R 0 via bus C, so as now three buses available. So, in the only in one single clock the whole operations will be the addition operation will be performed. But these three buses architecture, obviously, the system cost will increase and the complexity of the control unit design will also increase, but it will be very fast, because only one clock cycle is needed.

(Refer Slide Time: 57:34)



Now, another important feature, so this is a three bus architecture of the three buses are available.

(Refer Slide Time: 57:39)



Now, another important architecture or another important function of the control unit is that timing signals. See that control unit actually one of the main task control unit is properly sequence a set up operations such as sequence of N consecutive clock signals.

Now, to carry out operations the time signals are generated from a master clock and that has been shown in this picture. So, this is my actual clock and some four timing signals say T 0 T 1 T 2 T 3 are generated, so this is one type of signals that one clock this is the one period and the whole that actually the three one periods of is of period.

Similarly, T 1 is a different T 2 is another type and T 3 as different type, so from these master clock it will be generated and this is the task of control unit. So, we have read the functions of CU and the design of CU here.

Thank you.