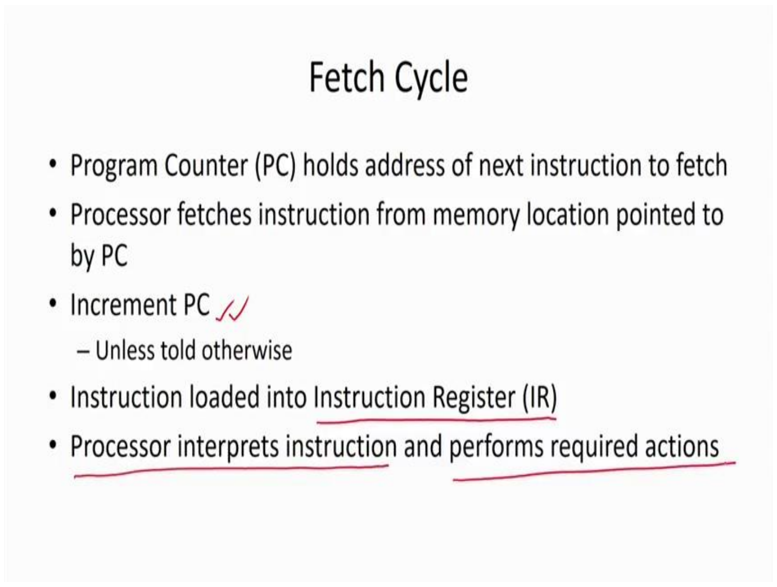
(Refer Slide Time: 43:01)



Fetch Cycle

- Program Counter (PC) holds address of next instruction to fetch
- Processor fetches instruction from memory location pointed to by PC
- Increment PC
  - Unless told otherwise
- Instruction loaded into Instruction Register (IR)
- Processor interprets instruction and performs required actions

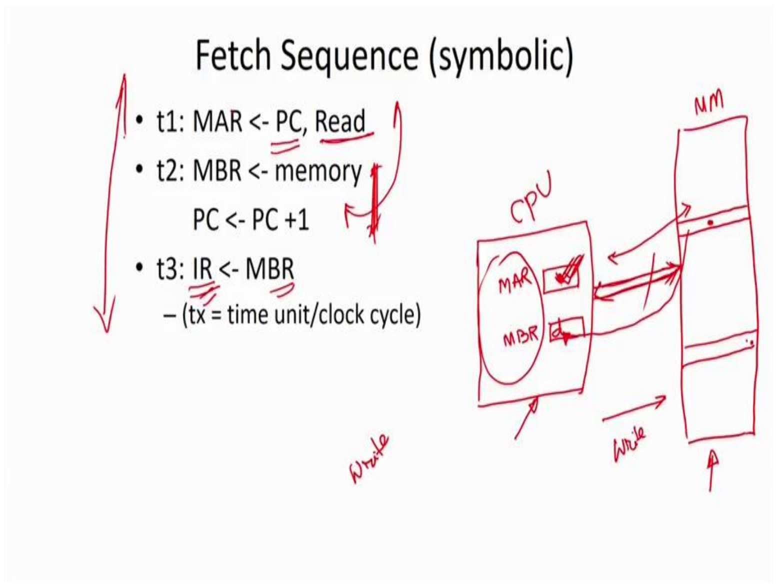
Now, basically what we are going to do in a fetch cycle. So, it is a fetching and information from memory to the processor. Now, what we must know when we are going to fetch an instruction, at least we have to know the memory location where we have the instruction. Now, where I am going to get this particular information. So, already I have mentioned that we are having a special purpose register are called program counter, PC - program counter. So, in that particular case, what will happen I am having a call register called program counter, and program counter will have the address of this particular memory location.

Thus say that it is a number, address you just think as a number say this is the zeroth address first location, second location like that we are having total n - one location. So, 0 to n - 1 total n location. So, we note that address is say 50, then program counter will have the value fifty over here that means, program counter is going to give us the information from where we need to fetch the instruction. So, we are having a register called program counter, and this program counter is going to give us the information from where from we have to fetch the instructions.

Now, after fetching one instruction then what will happen we have to after completion of this particular instruction, we have to fetch the instruction from next memory location, because it is in the sequence so that's how you can say that sometimes we have to increment the PC also. First it is going to have the address of an instruction processor fetch this information from memory to the processor, and along with that it will increment PC because after completion of this particular instruction, what will happen we have to go to fetch of the next instruction, and next instruction will be available in the next memory location. So, after fetching the information, generally we update this particular program counter, we just increment it.

After that whenever we are getting this particular instruction, this instruction will be loaded to instruction register. We have said that we are having a special register called instruction register. So, when we fetch an instruction, after fetching it, we are going to keep it in instruction register. Once we have the instruction in the instruction register, then processor will be knowing what operation we need to perform, so that information will be given to the control unit, and control unit is going to generate the appropriate signals. So, once it is having in that instruction register. So, now we are saying that processor interprets instruction and perform require action, so that means processor interprets instruction. So, after getting the instruction, we know that we have to perform say addition operation, then processor is going to perform the required actions; that means, control unit is going to generate the correct signal at correct time, so that operation can be performed in correct way. So, this is the fetch cycle.

(Refer Slide Time: 46:13)



Fetch Sequence (symbolic)

- t1: MAR <- PC, Read
- t2: MBR <- memory

  PC <- PC +1
- t3: IR <- MBR
  - (tx = time unit/clock cycle)

NM
CPU
MAR
MBR
Write
Write

And now, in a symbolic way, now I can say how we are going to do the fetch cycle. Now, already I have talked about PC that program counter that we are having. And we know IR instruction register after fetching it we are going to put it into the IR. Now, how we are going to do it. Again we are having two special purpose register, one is known as MAR - memory address register, and second one is your MBR - memory buffer register. So, these two register are basically the interfacing register of my processor.

So, now, what I can say this is my processor I am having a register called MAR and another register called MBR memory address register and memory buffer register. Just say that this processor is connected to main memory or maybe to I/O devices also. Now, how we are going to interact, how we are going to fetch it. You just see that somehow I have to give the address of the memory location, and whatever data we are having over here we are going to bring it into the processor. So, these two registers are going to act as an interfacing register.

So, if I want to read the information from a particular memory location then first that address we have to put it into the MAR. After that this address will go to this particular memory unit through this system bus ok then we are going to read the information, so that's why we are saying that first I am going to place the information from PC what we have in PC. In PC, we have the address of the memory location from where we need to fetch the instruction. So, first that information of PC's transfer to MAR now in MAR we are getting the address of the memory location. Now, I will generate the read signal. So, control unit will generate read signal what it will say that now we want to read the contents of this particular memory location. And once we are getting this particular information then reading it one read is complete, then what will happen the information will come to your MBR.

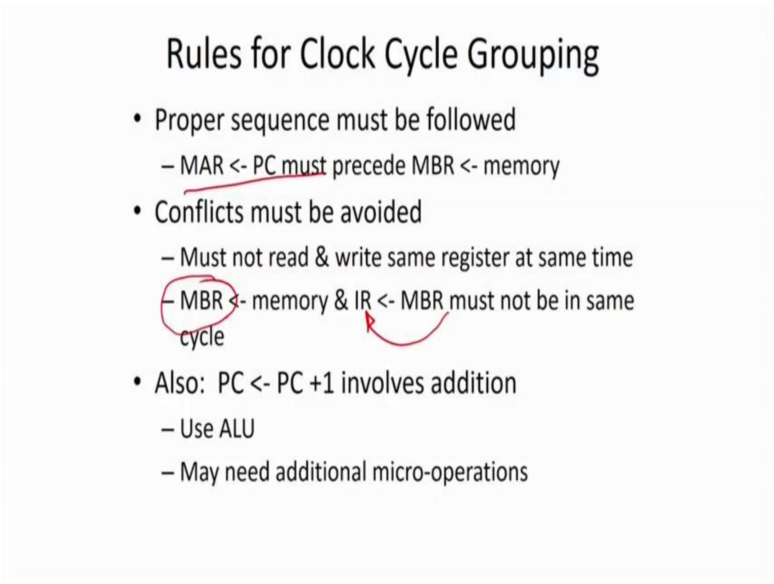
So, now say first cycle clock we are placing the PC to MAR then giving a read signal we are going to read it. Now, in second cycle what we are going to do, we are going to take the information from memory to MBR. And along with that what we are doing we are incrementing the program counter $PC = PC + 1$. Now, once this time cycle two is over then what we are going to do now that information is inside my MAR. Now, what we are fetching now we are fetching the instruction; that means, we have to execute this particular instruction. So, in the third clock cycle we are transferring the information from MBR to IR instruction register. So, now, I know the instruction what instruction we need to execute, now accordingly now control unit is going to behave control unit it is going to act and going to execute this particular instruction

You just see here that means, to fetch itself we need three clock cycles or three step. Now, again see in the second step, what we are doing we are covering two operation memory to MBR and PC + one adding the PC and keeping it up updating the program counter value. Now, why it is possible basically just I want to mention it say this is a processor CPU, it is an electronic device, this is a memory unit, this is also a semiconductor device, but the speed of the processor is not same with the speed of the memory, memory is always slower than the processor.

So, for that while I am transferring this particular information, it takes more time. So, since it is taking more time, so during the time inside the processor also I can carry out some work. So, that's why I am doing $PC = PC + 1$. Ok so, this is the way we can do it. Again this is a read we are reading it from memory to the processor, similarly we can have the write operation also; in write operation what happens we are going to write the information from a register to the memory.

So, in that particular case, what will happen first we will give the address to MAR that means, we are going to identify the memory location, where we are going to write or store the information. Then we will put our data in to the MBR - memory buffer register then we will give the write signal. Then what will happen whatever information we have in our MBR that will be stored in this particular memory location. So, we are having two operation, one is called read operation and write operation. And while we are interacting or interfacing with external to the processor mainly that MAR and MBR is going to act as my interfacing register.

(Refer Slide Time: 51:34)



Rules for Clock Cycle Grouping

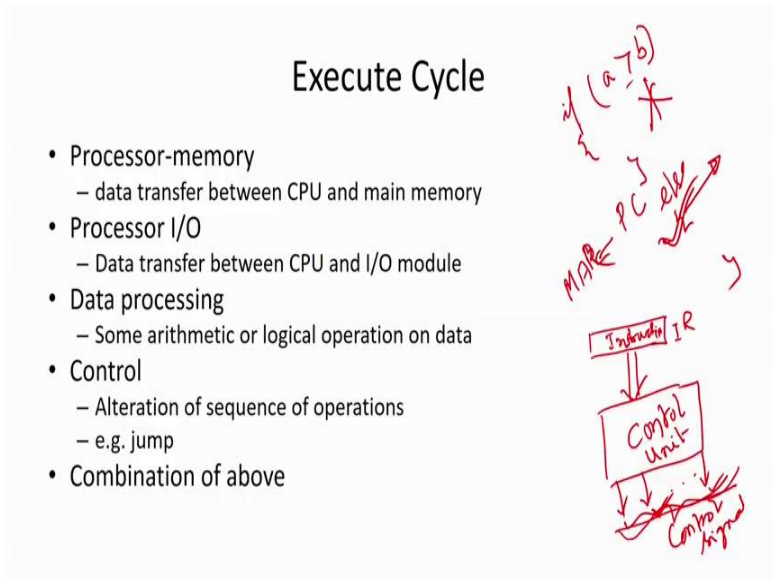
- Proper sequence must be followed
  - MAR <- PC must precede MBR <- memory
- Conflicts must be avoided
  - Must not read & write same register at same time
  - MBR <- memory & IR <- MBR must not be in same cycle
- Also: PC <- PC +1 involves addition
  - Use ALU
  - May need additional micro-operations

Now, here what will happen in this particular case say we are doing some operation parallely or sometimes you can say that we are performing this step then only coming to new step, why I cannot perform both together. So, there is some resource conflict that's why everything cannot be done in one clock cycle we have to perform in different step. So, for that there we have to follow some rules what are the signals that can be grouped together and we say this is a clock grouping first thing is that we have to maintain a proper sequence.

So, what will happen PC to MAR always precede memory to MBR. So, basically it's associated after knowing the address only I can get the information from memory to MBR. So, these operation cannot be performed before the first operation. So, this is the proper sequence. So, first I have to place the program counter to MAR then only I will be knowing. So, PC to MAR must precede the memory to MBR. Again some conflict must be avoided why I am saying that everything cannot be done it one go because they are some conflict may occur. So, we should not must or we should not read and write the same register at the same time, because if we are reading from some memory location or reading from a register, at the same time we should not write the same in same information to the register because it will change the contents of the register. So, read and write should be in two different clock cycle.

Again memory to MBR, and MBR to IR cannot be done in the same cycle because here we are storing some information, it will take some time after getting the proper result only I can transfer it to IR. So, they cannot be done in the same time. So, like that we have to see what are the things, what are the signals can be generate at the same times what other subtasks can be done in the same time, they will be done in one clock cycle. If there is some resource conflict then surely we have to go into two different step, so that's why by looking into all those particular scenario, we are saying that fetch cycle is going to take three different clock step, it cannot be done in one clock step or two clock step.

(Refer Slide Time: 53:56)



Execute Cycle

- Processor-memory
  - data transfer between CPU and main memory
- Processor I/O
  - Data transfer between CPU and I/O module
- Data processing
  - Some arithmetic or logical operation on data
- Control
  - Alteration of sequence of operations
  - e.g. jump
- Combination of above

if (a>b)
MAR PC etc
Instruction
IR
Control Unit
Control signal

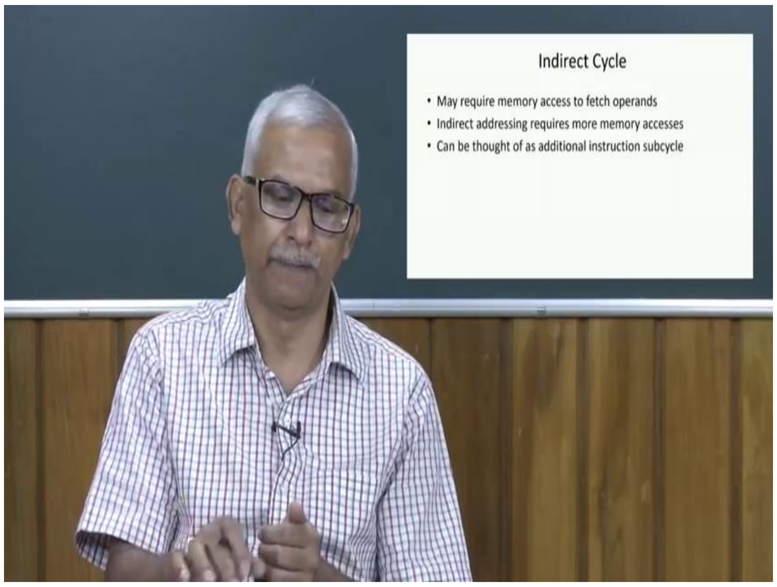
Now, what is an execution cycle? Now, say once we are getting the information in IR in instruction register, now I am having that instruction, some instruction we have fetched and we are having it. Now, according to this instruction what will happen we have to perform our operation, so this information will basically go as an input to your control unit. So, we are having a control unit inside the processor. Now, this control unit is going to generate those particular different control signal. And those control signal is going to control different components or basically going to control perform different task. Like that so first in the fetch cycle itself we are talking about say PC is transferred to MAR.

So, in that particular case control unit is going to generate some signal. It will indicate that whatever value we have in the PC has to read the contents of the PC and it will generate another signal, it will say that now whatever information we have read from the PC now write it into the MAR or store it into the MAR. So, these are the control signal that will come out from the control unit. So, like that after getting the instruction we are going to get giving it to the control unit, now control unit will generate a control signal.

So, we have what are the operation basically we are going to do, one is your processor to memory; that means, data transfer between processor and main memory. So, this is basically transferring the information from processor to memory or memory to processor this is one kind of operation that we can have. Another class of operation we may have processor I/O that means, information will be transferred from processor register to some output device or maybe from some input device we are going to take the information to the processor register. So, these are another class of instruction. So, depending on that we have to perform what task. So, in execution cycle basically we are going to perform those operation.

Third categories we are talking about the data processing, basically say if I am having my data then what will happen if it is an add operation now we have to perform the add operation. So, we are going to use the ALU, the adder of the ALU to perform the operation. So, control unit is going to generate those particular appropriate signal to trigger those particular device. And another one it is control or say sequence control basically in some of the cases we are having jump instruction or jump on condition. So, such type of operation we have to do for that we need appropriate instruction. Like that I have already mentioned that if you are going to write a program like that $a > b$ then perform this instruction else perform the other one. So, if it is false then what will happen we are not going to execute this part of code, but somehow we have to come to this particular instruction. So, for that we need those particular control instruction or some instruction may be combination of some of those particular operation. So, we have to execution cycle or execute cycle we are going to perform those particular tasks.

(Refer Slide Time: 57:11)

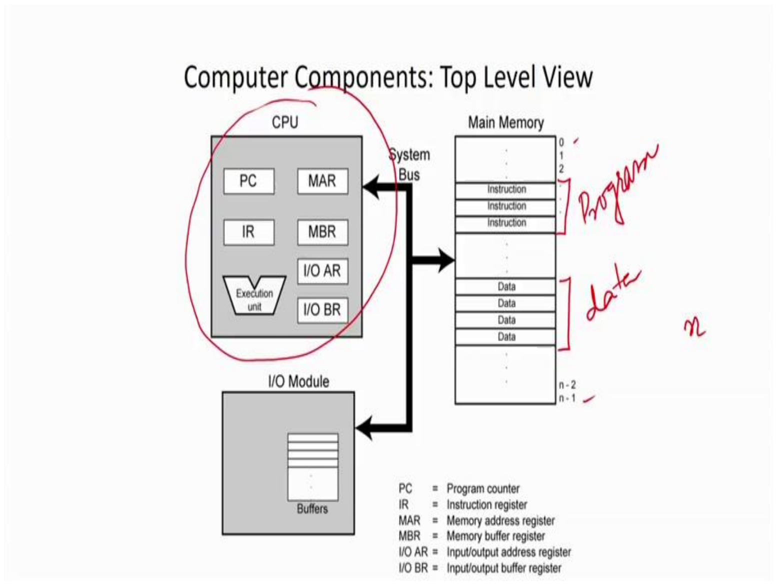


Indirect Cycle

- May require memory access to fetch operands
- Indirect addressing requires more memory accesses
- Can be thought of as additional instruction subcycle

So, in that cycle I have already mentioned so, if we need data for my instruction we will go to the in that cycle and we are going to fetch the data and keep it in some register inside the processor which are nothing but the temporary storage inside the processor and accordingly we are going to use those things.

(Refer Slide Time: 57:30)



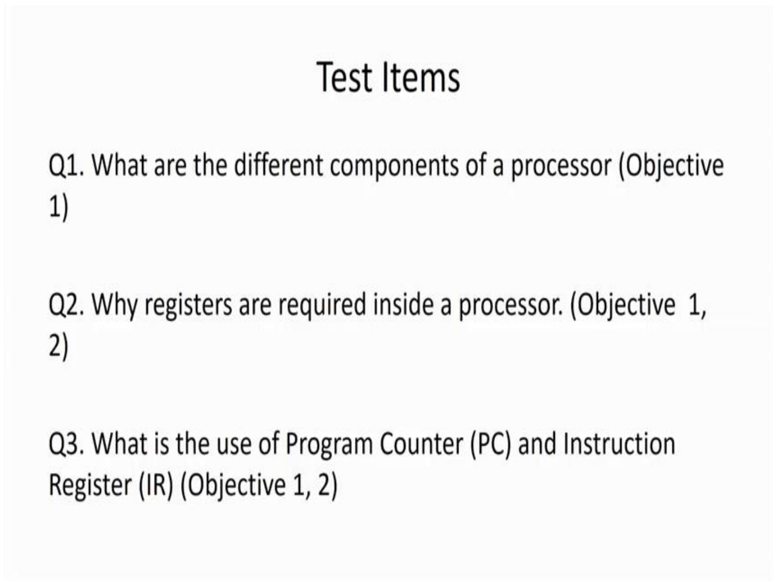
Computer Components: Top Level View

PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

Now, this is the component that we have discussed here today. So, processor is the main component central processing unit it is having a processing element and this processor is connected to main memory or I/O devices through system bus and works on Von Neumann's stored program principle. And now we have seen what are the basic components that we have inside this particular processor, till now we have not discussed anything about the main memory, we will see the construction of the main memory while we are going to discuss about the memory module. Secondly, we are going to see the complete design aspect of this particular processor when we are going to discuss about the central processing unit or the processor design, but here just in top level we have seen how we are going to perform an operation and what are the task that we are going to perform.

So, with this we are giving an having a brief idea about what we have inside the processor, why memory need to be connected to the processor because it works on Von Neumann stored program principle and how a program is executes basically it is having basically two part I can say that fetching of instruction and execution of instruction. Fetching is basically done from the fetching information from memory. So, here in this case you just see that we are having 0 to n - 1, total n memory location. So, in this particular memory location we are having those instruction. So, we can say this is my program and while I am going to execute this program it needs some data. So, this is the data that we have here in the main memory. So, processor need to take those things. So, this is the basic ideas about the components of a computer and also we have mentioned about the components that we have inside a processor and see what is a program and what how we are going to execute a problem.

(Refer Slide Time: 59:27)


Test Items

Q1. What are the different components of a processor (Objective 1)

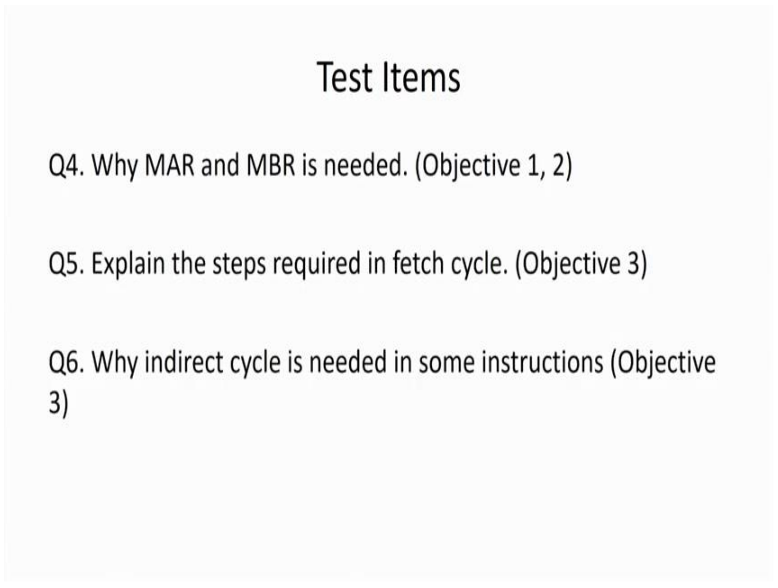
Q2. Why registers are required inside a processor. (Objective 1, 2)

Q3. What is the use of Program Counter (PC) and Instruction Register (IR) (Objective 1, 2)

Now, just look for some test item that whatever we have discussed today. So, first question I have given here like that what are the different components of a processor. I think by now you know all those things, but you don’t know about the details how we are going to design it, but in top level you know it. So, we have identified some objective for this particular unit and this question is basically related to objective 1. So, if you fulfil the objective 1 then you will be able to answer this particular question.

So, basic three components we are having ALU, registers and control unit, and they are connected to a inter connection network. Why registers are required inside the processor, again it is going to meet the objective 1 and objected 2. So, this is basically temporary storage. We have to have our information inside a processor, computer works on Von Neumann’s stored program principle. So, information is in your main memory. While we bring this information from main memory to the processor, we need some storage element we have to keep those information in some place. So, these are nothing but the registers. What is the use of program counter and instruction registers, I think already I have mentioned about it, what we are going to keep in program counter and what we use to keep in our instruction register.

(Refer Slide Time: 60:51)



Test Items

Q4. Why MAR and MBR is needed. (Objective 1, 2)

Q5. Explain the steps required in fetch cycle. (Objective 3)

Q6. Why indirect cycle is needed in some instructions (Objective 3)

Why MAR and MBR is needed? Again this going to fulfil objective 1 and 2, I think in my lecture I have mentioned that these are the interfacing register for processor. So, we have to give the address of a memory location, we will give it in the MAR. And whatever data we are going to use either I am going to take it from memory or we are going to write it in the memory, it will pass through this particular MBR. So, explain the steps required in fetch cycle. So, instruction execution is having two cycle fetch and execute, fetching is nothing but taking the information from main memory to processor. So, we have to perform a specific task for complete this particular fetching of the instruction. And this is same for all the instruction, and already we have discussed how we are going to do it.

Now, question 6, why indirect cycle is needed in some instruction this is objective 3, because if instruction needs the data, but data is not available inside the processor then we have to take it from memory. So, with this I wind up this particular lectures, hope you have enjoyed the lecture.

Thank you very much.