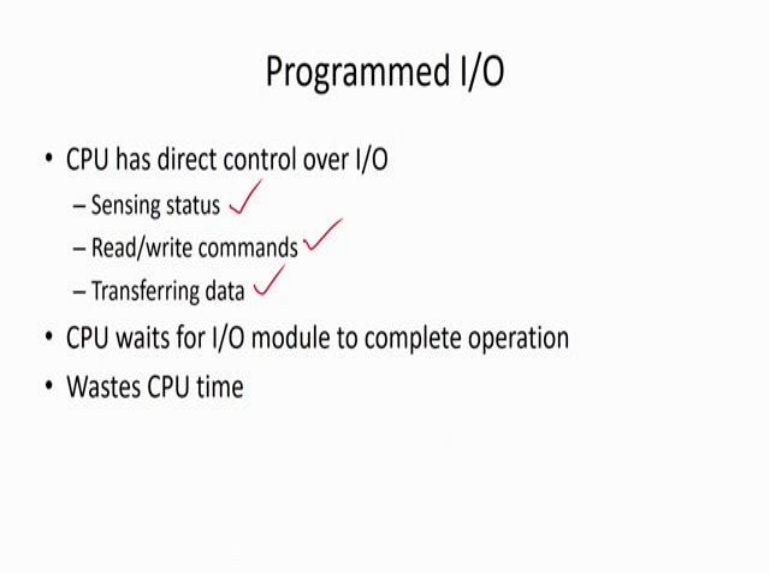
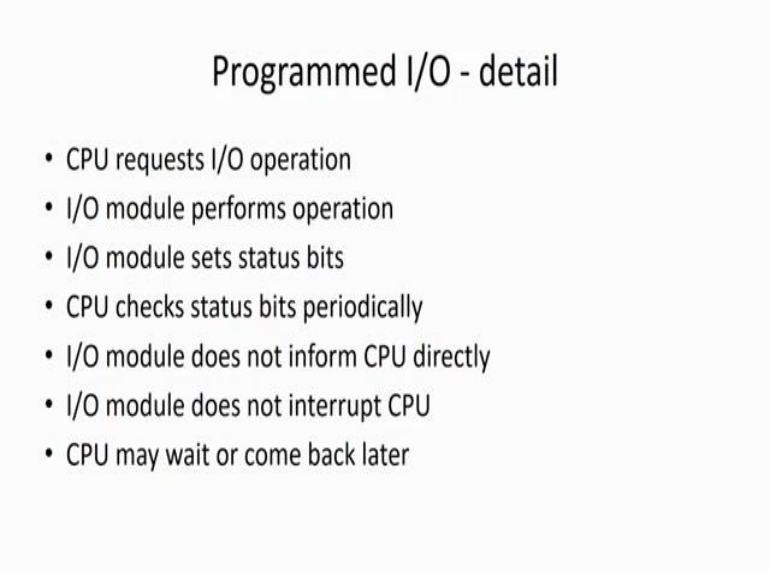
(Refer Slide Time: 43:23)



So, basically it is a sensing of status, then using the read write command, then transferring of data; this is the way that we are going to do, but here what is the problem that we have first? CPU needs to wait and poll, it is checking it continuously. So, it cannot do any other work. So, there is a wastage of CPU time.

So, basically you just see that if we are going to connect something like that this processor is going to, maybe 1 bit of the status register is going to continuously monitor and if it is set to 1; that means, device is ready, then it is going to perform the transfer operation. So, this is our programmed I/O.

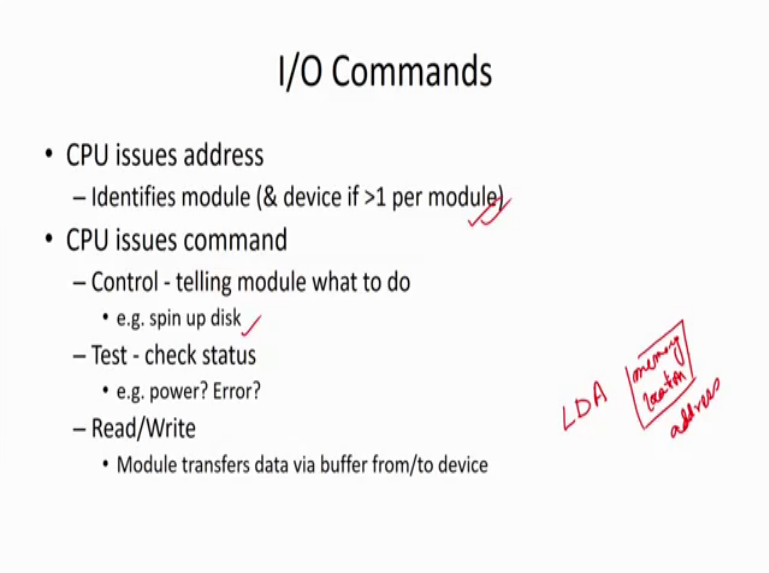
(Refer Slide Time: 44:01)



So, basically this is now programmed I/O details now you just see these are the simple steps ; first one CPU requests to I/O operation that in processor we have shown it, I/O module performs operation, then by looking into the state of the devices I/O module sets the status bit, CPU checks status bits periodically, I/O module does not inform CPU directly, you now see that here I/O module is not going to inform the CPU directly that device is ready, but in case of interrupt driven in somewhere, in somewhere I/O module is going to inform this particular things to the processor. So, I/O module does not inform the processor directly. So, processor is going to continuously check it, then I/O module does not interrupt CPU.

So, in that particular case I/O module doesn't intercept CPU, but now CPU may wait or come back later. This is unless you say after waiting for some more times, CPU may feel that ok it is now going to abandon this particular I/O operation, you say that I/O to come in at a later point of time. So, this is the way that I/O module is doing it. Now after that to perform work with the I/O module or input output devices what are the basic requirements.

(Refer Slide Time: 45:16)



Already I have mentioned that we need some commands I/O commands. So, first the issues for that I/O command is like that how to identify the module. So that means, we have to have device address so this is the one point how we are going to give the addresses of the I/O devices. Like that when we are discussing about some operation like that reading some information from memory or writing some information to the memory; say one example I think I have mentioned something like that LDA load accumulator from some memory location; that means, if it is an accumulator based machine then we are having a register called accumulator. So, we want to load something to that particular register accumulator from some memory location. So, we have to give the address of this particular memory location.

So, in the instruction we have to give the address like that, when we are going to perform the I/O operation. First of all we have to give the address of the devices from which we are going to take information or to who we are going to give the information. So, this is the addressing scheme, we have to see and will see how we are going to give it then issue some control commands.

So, basically some control commands we have to issue just to initiate the process like that already have explained. If we want to print something in the printer we should send some control signal to the printer to initiate it or maybe to bring the printer head to the appropriate position. Similarly when we are going to read something to the, from the hard disk then what will happen. We have to bring the appropriate location of the hard disk to the input output head,

we will elaborate these things. So, basically, so spin the disk we have to rotate the disk. So, those command we have to give like that check the status, whether it is ready or not, whether power is on or not, all those things status signals we are having and read and write.

So, one is this is the mode of transfer, whether read basically we say that we are taking something from the input device and write we are saying that we are going to put some information in to the output device. So, this is the read and write. So, basically we need some instruction to perform this read write instruction also. So so, the command related to control the I/O device can be now look into three different categories. One is your controlling, second one is your test and third one is your read and write. So, these are the commands that we have to design.

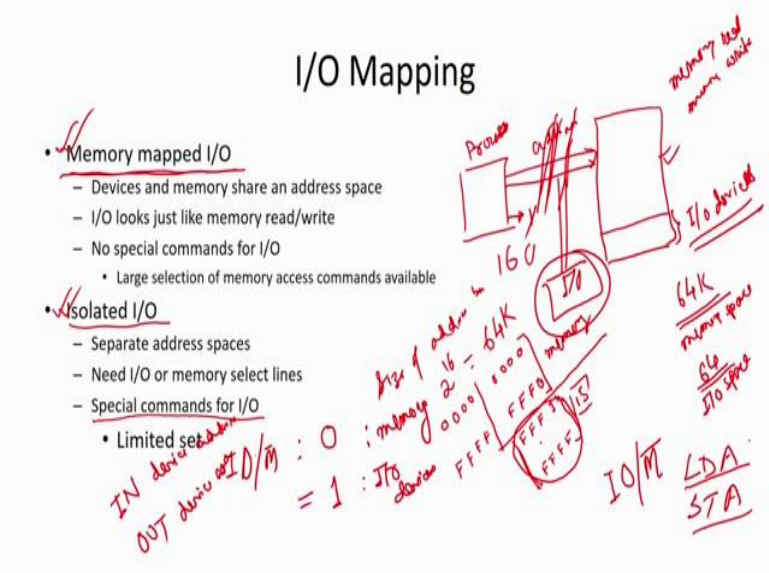
That means in the instruction set we have to put some commands to carry out this particular work. Now how to address an I/O devices. So, in that particular case, it is basically now, already I said that we have to give a unique identification to the processor or to the devices. So, for that we have to give an address. So, again this address is nothing but a bit stream of 0s and 1, because we are going to work with the 0s and 1 only.

So, basically like that, like memory address which is a bit stream of 0s and 1, then what will happen in that particular case. We are assigning a particular code to the memory unit or memory location. Similarly we are going to assign a particular code to the input output devices which will be treated as an address for that particular device and that address has to be unique, because we have to identify device uniquely.

When we are going to use the printer you have to give a unique code for a printer or a unique address to the printer, when we are going to use your hard disk you have to give a unique address to the hard disk. So, basically this is the addressing scheme. So, what is the address? It is very much similar to the address of a memory location which contains 0s and 1s ok; that is all.

So, in binary we can say 0 and 1 or you can say it is having a particular number. Now, how we are going to give this particular address, how we are going to map this particular input output devices. So, for that we have to look I/O mapping, so how we are going to map the I/O devices to the processor. So, that we can identify that particular device correctly.

(Refer Slide Time: 49:27)



So, in that particular case we are having two different ways of doing it. One is your memory mapped I/O and second one is your isolated I/O. So, I will just simply explain it with the help of small example. Now you just see that we are having the processor and I am going to say this is the address bus, we are connecting the memory.

Now, in memory mapped I/O it says that same address space is paired by a, shared by my memory and input output devices. Now you consider that size of my address bus is your 16, then what is the memory capacity with my size of address bus is 16, you should know it now. Say, this is your 2^{16}, which is your 64k. So, we can have 64 k memory location.

Now address will go from 0000 to all FFFF. Now out of this particular 64 k addresses what will happen. Some of addresses are reserved for my, your I/O devices. So, total address space is my 64 k. Now I am not going to use 64 k memory location, we will use less memory location, but some of the memory location will be your going to identify the I/O devices. Now say now I am going to consider that I am going to look for the memory location from 00 to FFF0 say these are the addresses that we are going to use for memory. Then what are the addresses that are remaining here this is your FFF1 to FFFF. So that means, 15 addresses are remaining left. So, these 15 addresses can be used to identify my input output devices; that means, if I am separating the address space this particular way, I can connect 15 different I/O devices to the processor ok.

That means, in entire memory space I am having address space 64 k, out of that some of the addresses are used to connect the memory; that that many memory locations you can use and 15 addresses are kept reserved for my I/O devices and whenever I am going to give this particular addresses or addresses from this particular range we are going to identify input output devices.

So, if this is the way we are going to reserve the address of I/O devices, we will say this is your memory mapped I/O ok. Now another one is your isolated I/O. So, in case of isolated I/O the memory space and I/O space are different. So, now, if you consider the same example then what will happen? We are having 16 as size of address bus is 16. So, we can go for 64 k. So, when I am going to have this isolated I/O then we can have memory space 64 k.

Similarly, we can have 64 k I/O space; that means, we can connect 64 k that means, 64 kilo which is equal to 2^{16}, which is equal to 65000 something you can calculate it; that means, more than 65000 devices can be connected to the processor ok. So, this is the isolated I/O we are having different address space.

Now in that particular case what will happen we are giving the addresses through this particular address bus, because you are having one address bus only which is a part of my system bus and through this particular system bus we are connecting the I/O module also. Now how to identify whatever addresses that we have put in this particular address bus. It is an address of a memory location or it is an address of an I/O devices. So, to identify these things so, we are going to use one more control signal, there will several control signal will come through this particular control bus. So, processor is having a control signal and in most of the cases this control signal name is given as $IO/\bar{M}$; that means, input output memory. So, by looking into it what will happen if the signal of this particular signal $IO/\overline{M}$ is 0.

So, M bar; that means, you are taking the complement of it; that means, if the control signal available in this particular $IO/\bar{M}$ is 0 it will say that basically it is going to look for a memory; that means, whatever address we have over here, it will be an address of a memory location and when this $IO/\bar{M}$ is 1, this is your I/O devices or maybe I/O module.

So, when the control signal value is 1, then whatever address we have kept in this particular address bus, this will be treated as an address of an I/O module or I/O devices; that means, I/O module is going to now use this particular address, but if the value of this address signal is 0

then this address will be the address of this particular memory location. Now you just see that we are having two ways of mapping this particular I/O devices; one is your memory mapped I/O and second one is your isolated I/O all right.

So, in case of isolated I/O we are going to get a bigger I/O space, because equal number of memory location and equal number of I/O devices, but in case of memory mapped we are going to reserve some of the memory addresses for identifying the I/O devices. So, I/O space is limited over here. So, this is the way that we can look into it and for that now, again I can say that since in case of your memory mapped I/O what will happen in that particular case, this is nothing but an address of a memory location, but we are attesting to an I/O devices. So, whatever memory operation that we are having; say memory read and memory write, then same operation can be used for those particular I/O devices.

So, in most of the cases we are talking about your LDA load accumulator or say SPA store accumulator; one is your read command, another is write command. So, for that load accumulator we are going to load something to the accumulator. So, in that particular case what will happen? Now we are giving address of the memory location.

So, if the address happen to be an address of I/O device; that means, we are going to take the information from I/O device and going to put into the accumulator. So that means, same instruction can be used for memory as well as I/O devices, if we are using memory mapped I/O, but if we are using isolated I/O then we have to have separate instruction to control this particular I/O devices. So, in most of the cases we are going to get some I/O commands like that.

So, that's why we are saying that some special I/O command. So, these are the I/O commands that what we are having in most of the cases we can get like that; one command is your in, another command is your out, in device address and out device address ok. So, in case of in whatever address device we are giving from that particular address we are going to get the information and going to bring it to the processor.

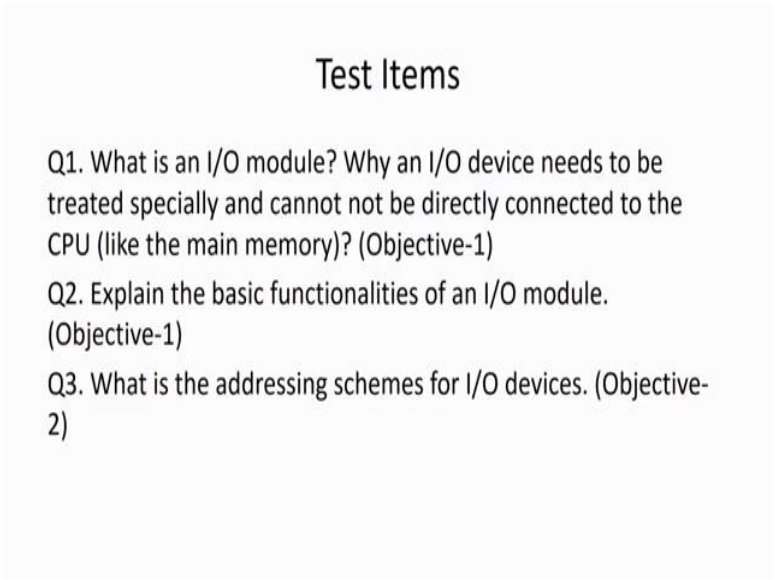
And with the help of out instruction, the information of the processor also can be sent to the output devices. So, these are the issues, basically we have one is your addressing scheme, second one is what are the commands that we need to handle this particular input output devices and along with that I think we have already used those particular program I/O details.

In that particular scheme, it is that in out or input and output will be controlled by a program and what this program is going to do we are going to execute this particular program, that in a loop it is going to check the status of that particular status bit; once status bit is ready, then this program is going to execute the other part; that means, going to transfer the information.

So, with the help of one simple program and we are doing it which is known as your device service routine in it is basically to control that particular device, we need a program. We say this is the device service routine, we are going to execute it, instead of programming cell we are going to check the status. Once the status is ready then we are going to carry out the input output operation.

So, this is the way we are going to transfer information in programmed I/O and for that we need the addressing of the devices and we need the command. So, these are the things that we have discussed in this particular unit. Now, look for the some test items and here I am giving the first test item.

(Refer Slide Time: 58:43)

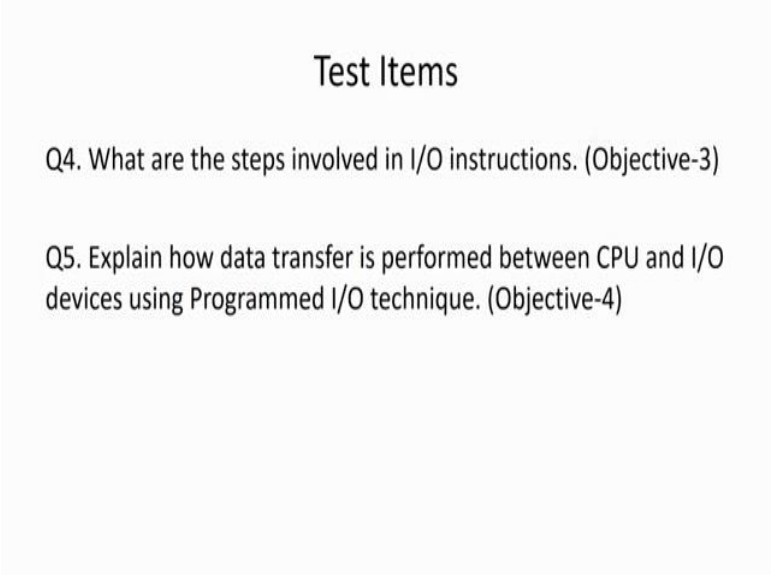


Like that question on, what is an I/O module, why an I/O device needs to be treated specially and cannot be directly connected to the CPU. So, like memory we are directly connecting to the CPU, but in I/O device you are not directly connecting the CPU, because already I have mentioned it that integration of the processor will increase, so that is why we are transferring the responsibility to I/O modules and for that we need the I/O modules.

So, basically here we are addressing this particular question and we are meeting the objective 1 that we have cited for the objective of this particular unit. Second one I am saying that, explain the basic functionalities of I/O module, again this is in the some knowledge level only which is the objective 1.

Again we are meeting it and I think I have explained it what are the functionalities that we are having for the I/O module. What is the addressing scheme of I/O devices? This is basically objective 2 we have mentioned and we are meeting this thing and I think now you know how we are going to give the address of an I/O devices, this is similar to the memory address only.

(Refer Slide Time: 59:50)



Question 4: What are the steps involved in I/O instruction. So, we are meeting the objective 3 and we just said that already mentioned the step. This is basically processor is going to give the information to the I/O modules, then I/O module check the status, it will report to the processor, then processor is going to start the transfer. So, this is how basically we have to see what are the step involved for this input and output instruction. Question 5 explain how data transfer is performed between CPU and I/O device using programmed I/O techniques, and this is in the design level and that means we have just meeting the objective 4 if we can perform this particular or you can solve this particular problem.

So, already I have explained, it is nothing but a program we are writing it and we are having a loop, it will be in that particular loop, until that device is ready, once device is ready, then it is going to execute the remaining part of the program and that program will be known as my

device service routine ok. So, these are the things that we have discussed over here and I think that at least you can have idea now, how we are going to transfer information from the input and output devices to the processor. So, with that I am going to wind up the lecture today

Thank you very much.