

Storage Systems
Dr. K. Gopinath
Department of Computer Science and Engineering
Indian Institute of Science, Bangalore

Storage media, Storage interfaces, Storage access mechanisms, Storage Protocols

Lecture - 06

SCSI interface & communication protocol, SCSI Terms, SCSI Bus Phases, SCSI Protocol, SCSI Examples, SCSI v3 Architecture, SCSI layers in Linux, SCSI Command Processing/Concurrency Models, Tagging request in SCSI v2, Tag ordering, SCSI Error Models

In the previous class we looked at disks and some aspects of scheduling. So, we will spend a bit more time looking at some our details correspond to this.

(Refer Slide Time: 00:31)

SCSI (Small Computer System Interface)

- Standard interface and communication protocol for computer peripherals
 - Allows device mgmt sw to be indep of storage device
- Based on the Client Server Architecture
- Clients called Initiators issue requests and Servers called Targets respond to initiator requests
- Arch may not be best when 3-party xfers needed
 - eg. backup betw tape and disk but control with sw
- SCSI commands sent in units called Command Descriptor Blocks (CDB)
- SCSI Transport Layer protocols such as iSCSI/TCP/IP, FCP govern the flow of SCSI command and data blocks

Particularly we have to look at the protocol called SCSI protocol or the SCSI base devices. And this is the first major disk interface that became popular. Previously all disk interfaces are proprietary. And once this particular interface became popular the disk industry relay took of quite a bit. One of the most important things about the SCSI device; SCSI devices is that it allows the computer software that manages distributes independent of the various types of peripheral devices.

So, basically what is happening is that, the software basically sent certain commands, and it is a business or the device itself to somehow execute those commands. And these commands are standardized. That means, that any number of guys can be disk vendors,

but they all have to satisfy this protocol. Of course, each vendor can provide some additional functionality, but they are considered optional facilities which are not to be assumed to be present other devices, other this devices.

So, in a sense it allows people to have a common set of protocol, let us say transfers that you can assume across all the disks. And a certain additional things which can be let us optional. And all the thing that is important about SCSI is that is based on a client server architecture. And this actually helps it to keep this particular protocol survey for a study for longer periods of time; I will mention what is this. So, in this client server architecture that clients called initiators who issue request and there are sorry is a mistake here there are servers called targets that respond to initiator request.

So, basically the way we understand this devices now is that there are parties called initiators that issue some requests to read something for example, and servers are those which actually respond to them. Since it is a client server architecture it turns out that sometimes you need something slightly different from a client server architecture. A good example is; suppose I am doing a backup between tape and disk. Now these are all the 2 entities, but I want to have control I want to control these transfers through some other entity it could be some piece of software. So, another 3 parties involved. Now in a client server architecture typically have usually only have 2 entities client and server. So, this kind of things may not be appropriate in some context.

So, you can you have to work around this clients architecture in case you need to do things like backup, and what is that? Backup is an important aspect of storage systems. So, I think it is nice that they came with a client server architecture, but sometimes this may not be exactly what is required. Now SCSI commands or sent in units called command descriptor blocks. So, these are the software which is managing these particular devices has to send all the commands in this command descriptor blocks. There is also some if you want to make SCSI adaptable not only that is useful in the context of a single desktop system, but also useful across longer distances. You need also a transport layer, and it turns out that there are been quite a few varieties of transport layer because that I have been devised, when good one is SCSI; it means you can transport the SCSI commands on internet.

iSCSI is basically SCSI over internet. There also other protocols like a FCP, this is a further channel protocol. And this also this transport layer protocols they govern the flow of SCSI commands and data blocks. This iSCSI for example, goes over TCP IP the internet, but FCP will go over typically over what is called a fiber channel protocol. Nowadays even this you can also use internet to transport this some these proprietary fiber channel related protocol unities, protocol units.

(Refer Slide Time: 05:37)

SCSI

- Early SCSI assumes a bus based architecture
 - not efficient in recovering from packet loss
 - not an issue in bus architecture
 - drivers still based on old SCSI standards and have been retrofitted for a network
 - applications designed to cope with the above
 - pipelining hardly used
- Applications need to commit to stable storage
 - When status OK sent, cannot lose data
- Storage response time
 - Few milliseconds for disks; sub-millisecond for caches

latency budget for interconnect should be less than storage response time

If you look at the SCSI system, the early SCSI assumes a bus based architecture again, going back to our previous classes. You notice that network became really let us say fast only post 1980s. So, against SCSI also came on the same time 1980s, but ethernet was not very common those days. So, they never seriously thought about coming with some network based architecture. And they assumed electrical electrically based let us say a system, and essentially a bus based system. About for example, you will have a bus on which you can put about 7 SCSI devices. So, that is the kind of stuff they came up with. And one thing about this kind of bus base architecture is that, if something goes wrong, you try to retry. And the retry it time also usually quite long. So, and basically if there are any bugs in the system, they have to be recovered from and it can take some time.

Now, if you are thinking about a bus based architecture for example, they are not really efficient in recovering from packet loss. Because you need to actually if you are not going for bus based architecture, and you want go for a network base architecture, you

can not have the same type of time outs that are there in the case of a bus based architecture. So, you need to have slightly different kinds of models. And all the same it turns out that a drivers curve built return for old SCSI standards, they have been modified so that can work for network also. And typically, also early SCSI would not try to do lot of parallelism, because it was slightly difficult to in those days it was there are not too many discs around to be put on single bus. So, usually the amount of storage that was being talked about was quite small.

So, there was hardly any serious attention paid to parallelism in the system the later versions of SCSI for example, SCSI 2 and SCSI 3. They systematically look into this more seriously. We will look at this some other thing that one has to think of SCSI is that; usually you can ask it to it transfer is need and only when the status sent the device has to commit the data and it cannot lose data after. That if it lose the data after that then it is considered a malfunction device.


So, essentially what it means is that the devices can do some caching. For example: I can have an electrical transfer from one buffer to another buffer. The buffer from the there is sending side; it can go to the buffer on the storage device. It can stay in the memory for some time, but once the status is sent; that means, it has to be committed to the persistent part of the storage system. For example, if it is magnetic device it has to be copied onto the magnetic portion of the system. It can not be sitting in volatile memory. It has to be in the persistent part of memory.

Now, if you think about if you think of interconnections, then you should look at the kinds of responsive expect from interconnect. Because it turns out that what this usually you have a few milliseconds for all access. And if you are using caches, it will be sub millisecond. So, the interconnect latency that you are using to connect this to for example, in the initiate when the target for example, should be less than typically much less than the let us storage response time for example, a few milliseconds for disk etcetera.

(Refer Slide Time: 09:53)

SCSI Terms

- "target": a collection of logical units, directly addressable on the network
- "initiator": creates and sends SCSI commands to the target
- "task": a linked set of SCSI commands
 - task attr: Untagged, Simple, Ordered, Head of Q, ACA
 - max one command in a task can be outstanding at any given time
- "task tag": used by target to distinguish between tasks
- "LBA": Logical Block Address
- "LUN": Logical Unit Number



Let us look at some of the terms. I should mention that most of this was devised the standard for device the industry. So, some of the terms are let us say, not exactly obvious what they mean, so will just go through it a bit. What is the target? Target is the as a mention there are 2 parts to the SCSI to the SCSI client. So, the architecture the client is basically the initiator. It creates and sends SCSI commands to the target. And the server is basically the target.

Now, the server also can be thought of as a collection of functionalities. And so, these are each functionalities considered as logically linked, and they can be directly addressable on in it on the network or the bus. What it means is that I can have a storage device, it can export multiple functionalities. And each of them can be independently accessed. A good example of this could be in a tape device often times the rewind function is taken to be a logical unit.

So, if you just want if you want to make the device rewind you just say is send it to that logical unit. The task is basically is a links set of SCSI commands. And basically, because if you look at storage devices, they often times do they have to seek to a particular place. So, I want I might want to have a set of commands is basically say seek there and read from there. So, this is an example of a link set of commands. Seek to that point and then execute read from their or write to that point.

This exam there other aspects for example, each task can also have some attributes, we will go into some detail about this later, and it can be tagged and tag; that means, that each IOP process might basically each activity that you are asking the target to do can be tagged so that you can essentially refer to it again later, or it can be untagged it has to be executed in the same order it came in. It can be simple ordered head of Q etcetera I will come to this later.

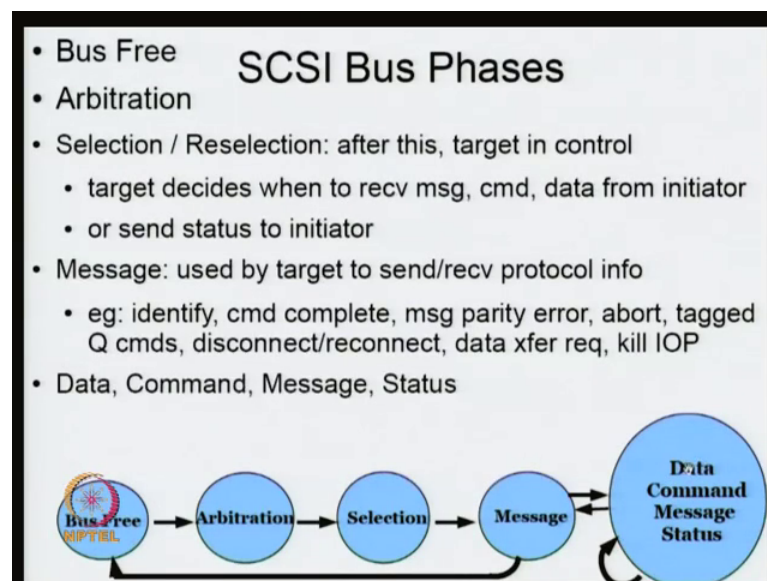
One important thing about SCSI is that; only one command in a task can be outstanding at any given time. And the tag as I mentioned used by target or distinguish between different tasks. What it can mean is that same disk can be accessed over this it can be servicing multiple initiators. So, I might want to distinguish between each of those initiators, and initiators themselves each initiator can actually send multiple commands. So, with in a each initiator target I might have multiple commands, and I want to be able to distinguish between them. I think the way to think about is similar to in the case of internet protocol IP, you will notice that there is something called the machine come up port.

So, the internet connection is given by machine one comma port one followed by machine 2 comma port 2 that same connection. Similar here also the connection is basically that of a initiator comma tag one with a target. And it is tag basically I want to be able to curve just the port numbers are used in the case of internet to distinguish in various services, I also need to be able to use where to distinguish between various io processes that are being initiated at the target. So, the tag is basically for that reason again the SCSI devices as I mentioned or the tried to make access to the devices independent with respect to the particular device.

So, the software does not have to know much about each disgusting things device. So, just like you have the notion of virtual memory, and you do not have done the detail. So, what is the memory of system here also you have an emotional logical bus address and what is I am sorry, I made a mistake this is logical block number I do not know why I have made is logical block address sorry I made a mistake here. So, this basically is some kind of a virtual address, and it is a business of the device to map it to whatever physical entities that there in the system for the particular piece of information.

So, logical block address is basically somewhat abstract notion about the way the user actually refers to some piece of information, and it can be it needs some kind of translation by the device before it can access again, this is again so that the software is independent of the particular x of the device similarly logical unit number I mention that a target is collection of logical units, and each functionality is given a number. Typical thing is that I can have what is say multiple partitions each partitions can be given logical unit number. Or I can have certain functionalities like rewind functionality as a logical unit number.

(Refer Slide Time: 15:25)



We will just quickly go through how the SCSI functions. Again all the terminology act you can as you can see is based on the notion of an electrical bus, but I think it historically that is how it is referred to. Bus free means that is currently on the bus there are no active requests. So, I get completely free if there are multiple accesses at the same time you know arbitrate again this is arbitration is it.

And then the initiator selects a particular target, and then after selection the target is in control every interesting model where the initiator once you selected the target then the target is complete in control of internet, and there is in why this is done. Because we discussed earlier it is, because the targets are extremely slow become the initiator could be a piece of softer running and a very fast machine, and it could be running at speeds of nanoseconds are tens of nanoseconds.

The target of our hand is a very slow device. So, when actually it finally, gets to seek to the particular point can take large amount of time. So, thing this I do not want to keep on pulling all this thing, you on pulling this an expensive of here. So, the target is in control and like it is basically the thing which decides how to communicate with the initiator after. That it is like the way interrupts are used. So, target decides when to receive message command data from initiator or send to him. So, every single thing whether went to get the message went to get the command from the initiator, went to get data from initiator or after the operation is done when to send status to initiator, all of it is initiated by the target.

So, for example, started bus free arbitration, selection, and then once the target is selected the command is sent. And the target is not in control, it will say that I am ready to accept your command, and I am ready to have allocated buffers so that when you send information I can not pick it up. And again there is a big difference here, because you notice that the target is a slow device, and does not have too much memory. So, it has to completely control when it is ready to access things. Otherwise the fast CPU or fast devices fast memory, whatever can essentially, over run the buffers on the target side. So, this is and why the target is in complete control. They if decides then the initiator can send something.

So, every single thing has to be initiated by the target after this phase of the selection phase. What is in selection? Once the target has finally, seek to the particular place where the data is, then it I can reselect, and then afterward it tells this the initiator. I am ready now, I have allocated the sum of the buffer, or now you can send me this amount of data. And sometimes it turns out the initiator has to send some critical information to the target.

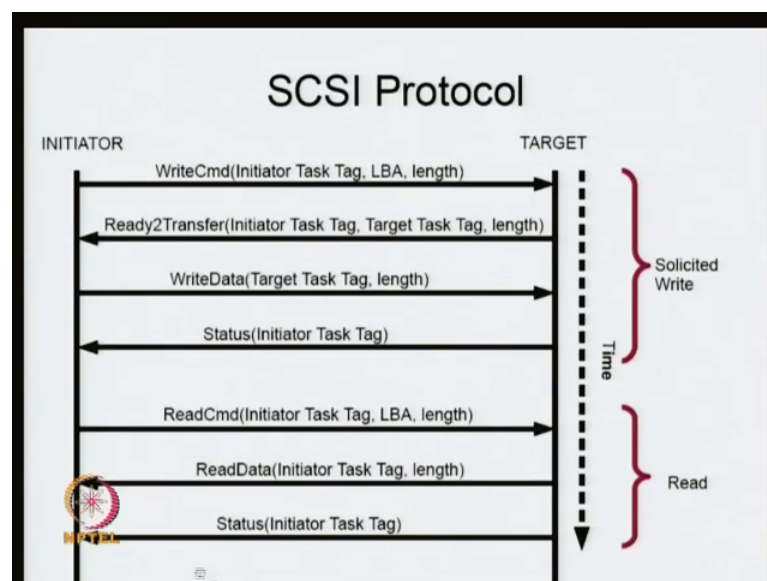
Again, the way it is done here is; just like a it is a interesting model. Here again the initiator essentially sort of bus gives an interrupt to the something like interrupt it is called attention. It gives attention signal to the target, and the target now this says, I got your attention signal please send the message. You want to send me again this target is in control it actually tells the initiator, I had noticed that he wanted to just make me respond to something I am no ready please send a message, etcetera. Something is status also. If it turns out that a particular operation is done, then the send the status information to the initiator, right? You want that basically the target basically it is sends and message status

that I am now ready to send a message, then you tell me what kind of stuff you want then I will send it to you.

So, this is an interesting model, I think the high level thing I want you to remember about this particular thing is that. Once the target is selected the target is in complete control because it is the slowest device it has to figure out what to do. It cannot be overrun it does not it has only a finite amount of memory. And therefore, it actually controls everything. And basically, because it turns out, again if you think carefully but there is a read and write operation. Read write read operation is going to be on the device. So, you do not need to do any flow control on that basically because the stuff is coming from the device the slow device.

So, there is no list to be done they no there is no lead for it to do any flow control on reads. But writes are coming from the fast memory or whatever right, flow control has to be done because some kind of a symmetric aspect is there.

(Refer Slide Time: 20:58)



Just let us the look at the protocol itself. This is initiator, the target.

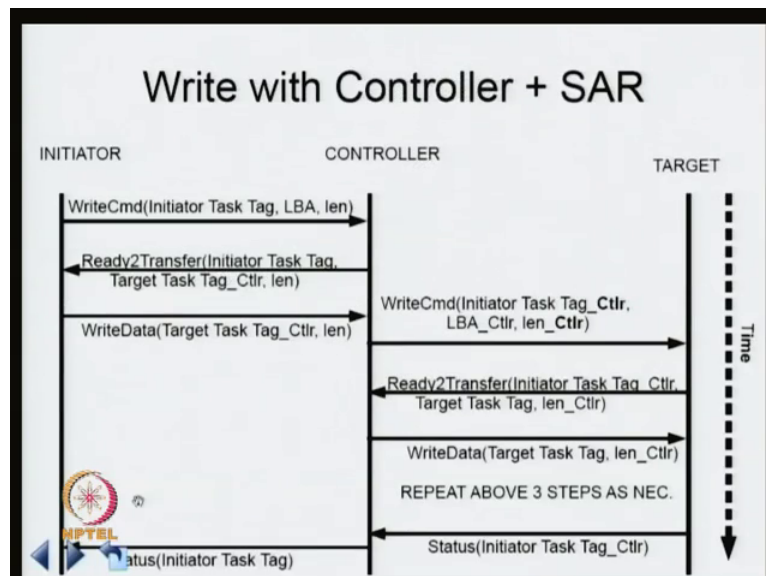
Now, I want to do a write. So, the initiator can have a tag for it. Saying that is a particular type of IOP process that is going on, some kind off and just like as I mentioned earlier in the case internet you have the notion of machine come a port number right. That identifies a connection. So, this is on the one part of the machine comma port number in

the internet connect is that is one part of it here. This is basically the logical block address, and this is the length. Now once this is given to the target; the target looks at it says that this guy wants this amount of memory to be sorry, data to be written. So, it looks at it and says it has to look at the buffer on the target side. Once it has done it then it says ready 2 transfer.

Again, it says with respect to the same with transcript sense the same tag back. And then it says that on my side I have a tag also. Just like the case of internet protocol you have port one on both sides, right. Is a port on the sending side and a port on the receiving side same thing here also and then after that you can do a write. And then once it is done you can send a status back. This is a simple example we are write similar thing with read, as I mentioned to you hear there has to be flow control. That is why the reduce transfer is there.

Here there is no need for flow control, that is why you will see that you just say read command and then the data is sent out like this and then a status association dot. I hope this pattern is clear while there are 4 parts creates in the 3 parts, create it is basically because there is no need for a flow control on this part.

(Refer Slide Time: 22:55)



So, the simple model of a SCSI execution; now we can try to make it slightly more interesting. Now, let us say that instead of I initiated talking to a target directly. We also have controller in between. And this may be needed for many reasons. It is in could be

that the target is sending too many interrupts with the initiator, example your CPU, and the CPU does not want to be bothered by the interrupts that the target is sending out the reason could be that. Initiator wants they write or let us say 64 megabytes, and the target can only accept let us say every some smaller amount.

Let us say 4 kilobytes or 16 kilobyte something (Refer Time: 23:38); that means, that for every 16 kilowatt that has to keep on interrupting initiator. So, you do not want that to happen. So, you want to put some agent a controller often called a host bus adapter. That is what is (Refer Time: 23:51) now how does this particular thing? That actually turns out that all these guys are have to speak this SCSI protocol all of them. How does it work? Just same as before right command initiator task tag will be a length. And the controller can respond to the initiator because this 2 guys are speaking SCSI now.

So, because the write, again the controller should not be overrun therefore, the controller will basically say I will look at it from space on my host bus adapter card; there is some what a memory there. And I am willing to accept that particular length. So, and then finally, it says right and then it is a tend to the controller. Typically, these transfers are electronic transfers. Because on this side there is some memory, on this side there is some kind of electronic memory. Now once this particular initiator as return to is controller, for the controller in turn will again this SCSI protocol to write to the target.

Now, here if the controller has enough memory to write to this, right. It can do it one shot. Typically, the target might not have assets memory. For example, I remember once upon a time in the 1980s, this target might have only as little as 64 kilobytes of memory in the target. And initiator might want to do a write of let us say one megabyte, and typically what would happen is a controller would have something like one 28 kilobytes at 2 fifties kilobytes of memory; that means, that initiator would write in chunks of 250 is kilobyte, and this itself again is going to be split up and written in terms of 64 kilobytes to this curve.

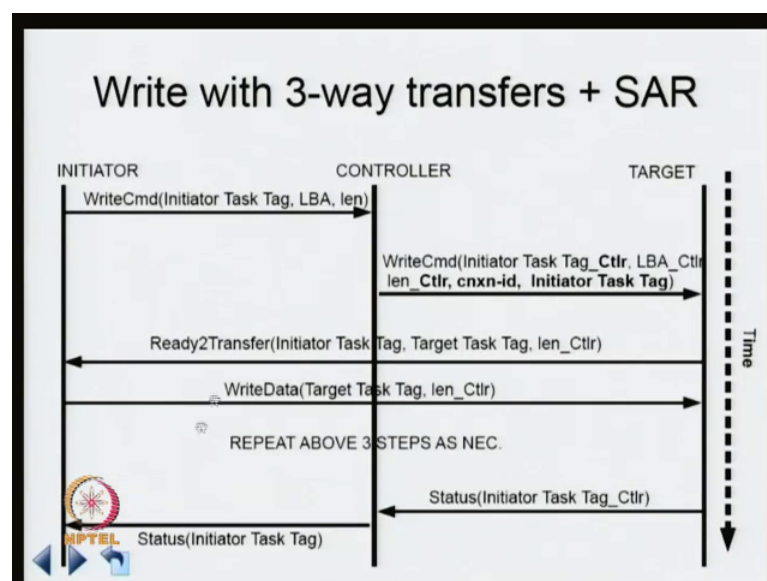
So that means, that this writes command you have a initiator task for the controller, this again tag for the controller itself. And then there is a LBA that the controller has to come off with, because I am going to use this is going to be a long valid system 4 megabytes. So, then this party has to say that it is at 256 bytes, then 512 kilobytes, then one megabyte then 1.4 megabytes, that is what this is it basically saying at this offset please

do it on for this length. Again, these to talk to protocol this to as talk thus SCSI protocol again. So, this target will now tell I am ready to accept your write let us say data. And I have allocated my buffers. Therefore, please go ahead that is what this is this particular message is block.

So, this length that this target accepts is basically what it is memory can handle. And then this is written and you repeat this might as many times as necessary to the whole transfer is done. And then this practice is it I am done. And this actually taken with the control is send back right. Now this is perfectly fine as long as single disc of system a single disc of system, your electrical interfaces are quite fast. For example, in disc of systems, you have once upon a time started with 5 megabyte perfect 5 megabyte per second then we came 10 then became then they came 20 then became forty 80 320 megabytes and 640 megabyte per second. Since is a quite fast this electrical interfaces, 640 megabyte per second is equivalent about 5 gigabit per second.

So, these takes coming quite fast, but suppose I am doing the same kind of idea, but an internet. You can do internet it turns out that sending it back and forth between 2 3 parties initiator controller and target can be a bit let us say wasteful. So, might want to do something better. So, here we are already we taken care of segmentation assembly and reassembly here. That is what we have done here, but you likely would also do something more efficient.

(Refer Slide Time: 27:55)

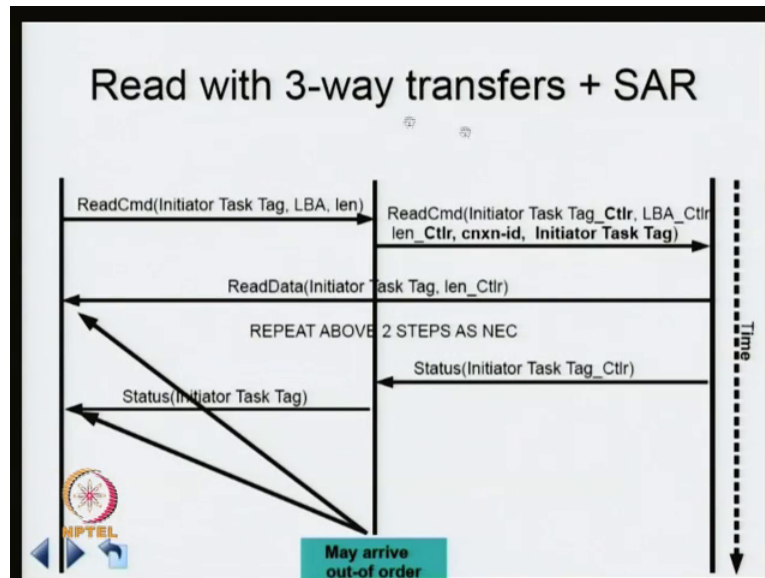


How is this to be done? That say wait do this also; what we can do is you can have essentially equivalent of 3 way transpose. This initiator controller and target and the target initially the initiator also sense simulate of information, except that there is an additional (Refer Time: 28:11) information about the connection id that is kept, and the idea basically is the target should directly send the information to the initiator. And it is repeated this 3 stags as necessary. The idea is basically instead of if you look at the previous one, right? You had to transfer it from here to here and then finally, there is trans from here to here from here to here.

So, we want avoid this expense. So, if it is possible you might want to do it directly between initiator and target. Of course, the number of interrupts etcetera will be high in this case, but if you are talking about at the level of internet speeds. Probably the number of interrupts that are coming in or manageably, it is not a serious problem because the rate at which your sending stuff is not very high.

So, this is one example of a 3-way transpose that can be done. Where the interesting about, this is that each of this parties are essentially speaking SCSI protocol. Because is a both are SCSI protocol is client server it is feasible for us to keep introducing intermediate agents. And that is even why the SCSI protocol since 1980s it is still going strong. Whatever new technologies are coming in as I storage their networks, or there is any other type of networks are coming in, they can essentially modify things here and there a bit and the same SCSI protocol can be still used.

(Refer Slide Time: 29:45)

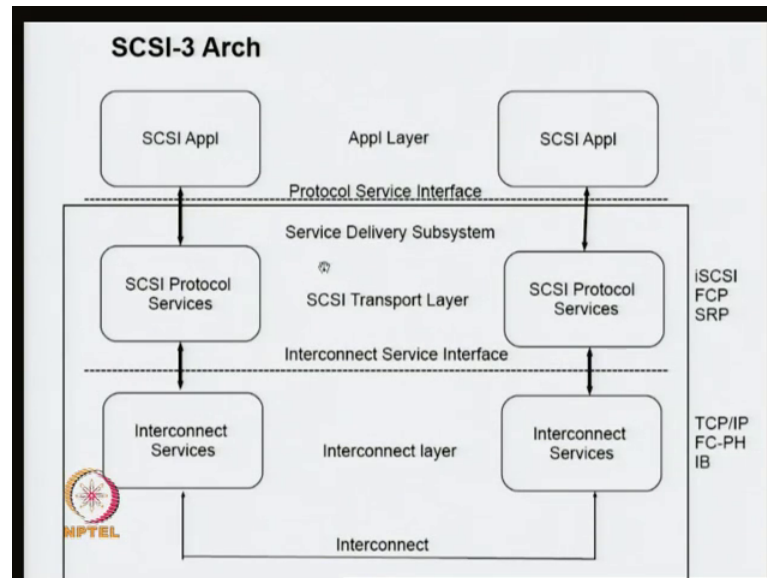


I can for read fuzz it come to the same things again what I showed you earlier was for writes basically it is coming from a initiator down to the target. And this required some kind of flow control whereas, in the case of the 3-way transfers is slightly simpler they can directly send it as fast as I can. And again same the similar, except that sometimes there could be some interesting issues with respect to some of this messages coming out of all right.

So, is something to be take a (Refer Time: 30:18) again you notice that whatever we talking about right now, is actually similar things you have to worry about whether you are doing it at this at this low level. But SCSI level even if you do it right straight higher levels for example, if there are 2 file systems talking here talking to each other. They may also have to do some of these things flow control, and also to worry about how to segment things, how to reassemble things. All this issues, what are we talking at this level are also applicable even if you try to go at higher levels almost every single issue.

Say for example, one of the things that this particular SCSI protocol does not worry about is security. So, there is no question of any encryption etcetera here. And SCSI has not been divine has not been defined using security in mind, but iSCSI has been defined assuming security in mind. So, it uses the internet IPSEC protocol if any security aspects had to be handled.

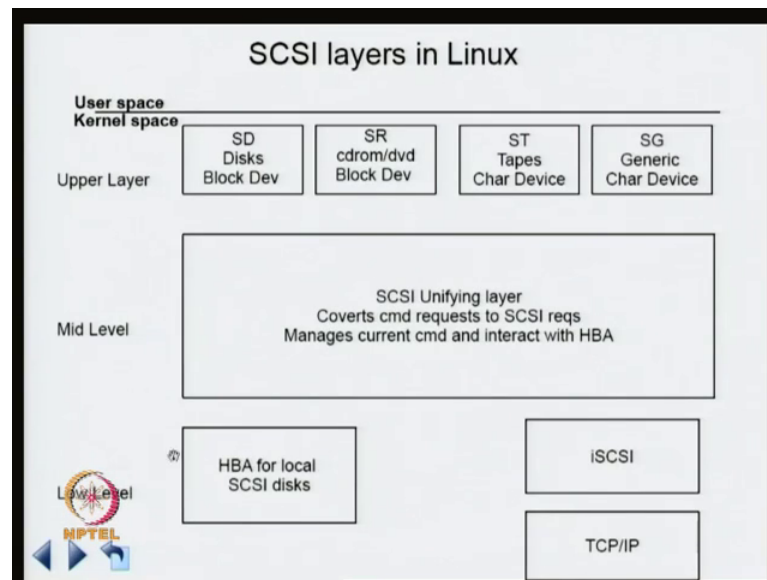
(Refer Slide Time: 31:24)



Again, to think about this SCSI architecture a bit more in, and the traditional style with respect to how networks are described. You will see that there is application layer, there is a transport layer this interconnect layer. This is closer to your data link layer, this is closer to your TCP slash (Refer Time: 31:43) TCP level it is basically the actual protocol services here, and this is the application layer for example, if the tape or a CD ROM or DVD it has specific commands. Those things are a handle clear.

So, think to do is that this is basically the thing which follows you the interconnections. This thing which is connected with just the layer above the transport layer, you have some presentation or the protocol aspects of that is what being handled at this point, and this is by the application layer. And you will see that SCSI and some of other further channel kind of protocols operate at this level.

(Refer Slide Time: 32:38)



Whereas the TCP IP and the physical layer of fiber channel or infiniband, all this since operate this level. If you look at from the lenox point of view, you will notice that the upper layer I some mention then the previous class, you have 3 levels; upper layer, middle level and lower level.

So, for example, this have a particular type of software met call application level software for handling disks. There is some specific disc for handling CD ROMs, these one for tapes. And this is for a generic device without any structure like disks or CD ROM tapes. And basically, here is a mid-level, where all the commands are converted to SCSI requests. These were basically the I mentioned to you about the discuss SCSI about we did some ready to transfer etcetera.


Those kind of (Refer Time: 33:33) those things have done here and this also manages the interaction with the host bus adapter. If you are thinking in terms of interconnect, for low level is basically through HBA, if it is for the if you are using internet for example, it will be iSCSI followed by TCP IP, this got basic interconnect. It usually basically interconnect part of it. So, again the important thing to remember is that you try to ensure as much as possible to reuse code if you can, and the rotary is the code is basically to take out interconnect part which is quite different for local disk versus internet. And then all disk for example, have this particularity to software. And CD ROMs DVD select

travel is a slightly different application layer in a call it. And because there are some settle difference with each of these things we had had list specifically.

So, again there are lots of issues here also. For example parallelism, suppose this is a SCSI layer indirect right. So, upper layer is handling this how many is there are multiple disk. Can I pump all that this at the same time? Again, this particular device this particular software has to take into account how many colonel threads are there which are actually pumping requests into this part of the layer whether this is (Refer Time: 35:00) let us say paralyzed. There will lot of locking another can a considerations all considerations should be here. So, he said this looks I mean safe with look think a bit more carefully there are lot of issues that if it is handled before you have a working system.

(Refer Slide Time: 35:18)

More SCSI Terms

- SCSI-1: Linked commands: Eg: seek+read
 - Many cmds executed in seq in 1 I/O process (IOP)
 - - One entry in IOP Q
- SCSI-2: IOP Qing:
 - Untagged: allows a target to accept one IOP from each initiator for each logical unit or target routine
 - target can accept cmds from an initiator while executing IOPs from another
 - Tagged: target can accept a series of IOPs from the same or diff initiators. Executed according to a Q mgmt alg or in specified order
 -  - Tagged queuing allows drive to accept multiple commands from each initiator

Again, you just basically well continue a bit more on looking at SCSI, because there is an doing so much taking so much time in talking about SCSI is that everything that we talk about SCSI here, you will find that this is similar things had to be done at even higher levels. The same issues keep copying up again and again.

So, this is look at SCSI 1. SCSI 1 was one of the earliest SCSI standards. And they already had provided something called linked commands, as I mentioned earlier if you. Are talking to your tape? You want to sit your place and then had did something right. So, that is why you would like insta sending took insta sending 2 separate commands,

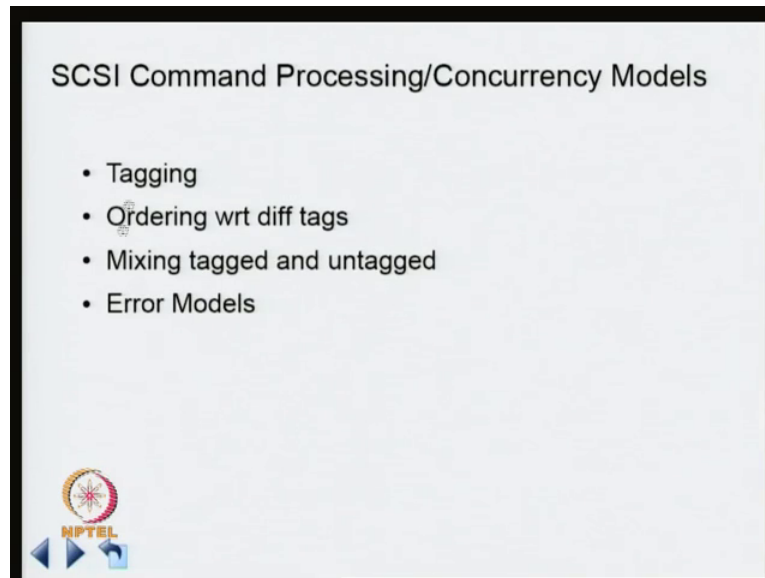
you would like to send a seek plus read. That is why this is important is because for atomicity divisions for example, the tape can be used by multiple initiators, 1 by 6 and then before the read command has made it is way somewhere person also sick somewhere else right. There can be some problems, that is why you need to ensure that link commands represent.

So, then there is also something called IOP process qing basically IOP process is basically one unit of work that you want the target to do. So, there are they could be a Q in the target, and each IOP process can be given one entry unit. So, the commands are execute in sequence. Suppose we look at the qing in the IOP. This was provided in SCSI 2. So, there are varieties of models this untagged model. In the tag model in untagged model it allows a target except one IOP process from each initiator for each logical unit or target routine. So, target can accept commands on from an initiator while executing IOP process from another ok.

So, the idea is to follow some kind of parallelism and it is not as much as this one. We have in the tag case which can accept a series IOPs from this same or different initiator; that means, I have there is initiative one initiator 2 initiator 1 can send 5 or 6 requests initiator 2 can send 3 of request or whatever number. And the requests sent by initiator 1 or initiator 2 they can be again use elevator algorithm like what I saw before is (Refer Time: 38:00) during that can be also done.

So, in a sense what I can do is with tagged, I can because I have information about each of the requests. And I know which io which initiative is doing it. I can actually execute them in the order that is best for the device, and they can respond to it to the actually initiator did without any difficulty, the do not have tagged ones like I am not ready in a position to do it. So, basically the tagged queuing allows drive to accept multiple commands from each initiator.

(Refer Slide Time: 38:35)




So, I think basically once you have notion of tagging it turns out you need to start thinking about how do you order things, can you mix tagged and untagged things? What are the error models? I can think a bit about someone of the issues are similar to what has to be done in the case of a multi threaded system right. What is ordering between request if I have accessing memory? And can I have certain consistency models right.

For example, is it possible for me to say that this particular memory operation has to be committed before I look at any were memory operation, similar things. For example, is an error is are multiple populations that I am doing in a memory based system? How they report the errors?

(Refer Slide Time: 39:31)

SCSI-2 Models

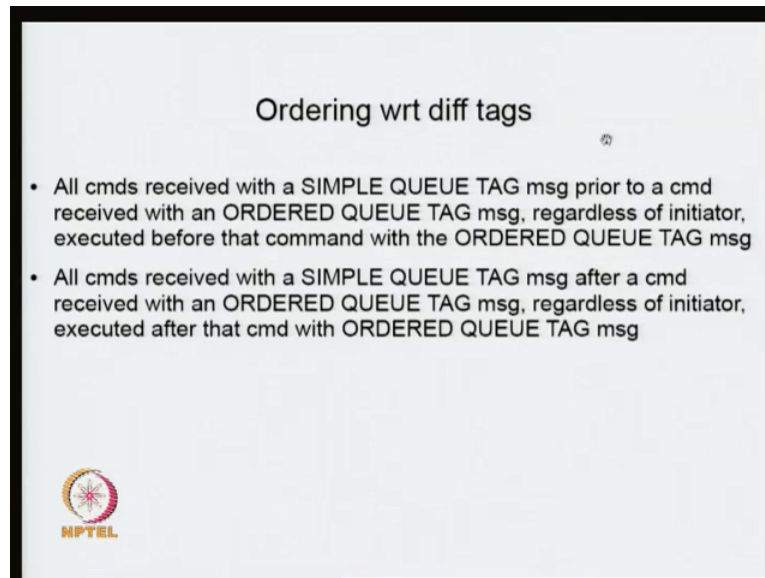
- Simple Q tag
 - IOP added to cmd Q; target decides when to execute (disks can use elevator alg)
 - Commands from other initiators also executed in an order selected in the same manner
- Head of Q tag
 - Add to beginning of Q
 - Multiple Head of Q tags in LIFO order
- Ordered Q tag
 - In order of entry except Head of Q tag processes



So, you and SCSI also has to look into same similar issues. So, we briefly look at this. So, if you look at simple there is something called simple Q tags. So, all it is saying is that, initiator wants a send some new requests, and it is just added to the command q, and the target can actually reorder it is up to it. Then basically if a simple Q to add it means that the target is completely free to (Refer Time: 39:57) giving it once. Nobody is talking whether there is something head of Q tag. What this means is; that there is something very critical has to be done. If there is an absolutely critical this is the thing has to be done for the sanity of the system, I sent head of Q tag; that means, it is going to be executed immediately after the current one is finished.


So, I am ordered Q tag, it is the same except that in order of (Refer Time: 40:29) except it is not too far on the head of Q tags.

(Refer Slide Time: 40:35)



Ordering wrt diff tags

- All cmds received with a SIMPLE QUEUE TAG msg prior to a cmd received with an ORDERED QUEUE TAG msg, regardless of initiator, executed before that command with the ORDERED QUEUE TAG msg
- All cmds received with a SIMPLE QUEUE TAG msg after a cmd received with an ORDERED QUEUE TAG msg, regardless of initiator, executed after that cmd with ORDERED QUEUE TAG msg

 NPTEL


And if you look at SCSI manuals you will find a lot of details about this and because storage is a critical part of the infrastructure. So, if you get any somethings wrong you are going to lose data. So, that is why there is a lot of effort in specifying exactly what is happened here. I will just read out some out one or to things just to get an idea. If you get a command with a simple Q tag prior to the command with the ordered Q tag, then you have to execute that command before you get the ordered queue tag. It is something similar to you if you are looked at memory subsystems; you have the notion of what is called a barrier right.

So, if you get a barrier you have to execute all of those things before, then only you have to do commands after the barrier. Now the ordered queue tags message basically some kind of barrier. You know, it self for somebodys telling you that I want this to be done now. I do not want you to reorder things. Because if you reorder it probably my semantics of what the storage operation you can doing are going to be not the same as I want essentially. It is say I am sending you a command to make sure it is done then in there do not reorder my request for. This thing the same thing about here also all commands will see with a simple Q tag message after a command received ordered Q tag message regards, initiator executed after that command, so the test to be executed only after I finished this part.

(Refer Slide Time: 42:10)

Mixing tagged and untagged

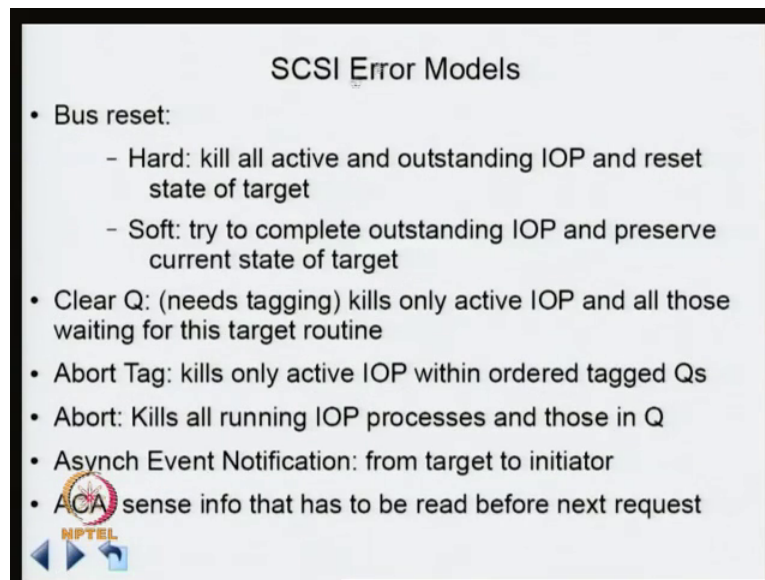
- An I/O process received from an initiator without a queue tag message while there are any tagged I/O commands in the command queue from that initiator is an incorrect initiator connection, unless there is a contingent allegiance condition.
- An I/O process received from an initiator with a queue tag message while there is an untagged command in the command queue from that initiator is also an incorrect initiator connection.
- In either of these cases the drive removes all commands in the queue from that initiator, aborts the command in process if it is from that initiator, and sets the Sense Key to Aborted Command and the Sense Code to Overlapped Commands Attempted.



Again, there are issues with respect to mixed, tagged and untagged requests. And it turns out that most of this mixing is usually considered an error. Again, if you just read one of them and have a process received from initiator without a queue tag, message while there are any tag io commands in the command queue from that initiator is incorrect initiator connection. So, basically if you try to mix tagged and untagged there are going to be some problems. And that is also the pre solution of these errors.

The basically says drive removes all commands in the queue from that initiator aborts the command in progress. And sets the sense key sense key means basically it is saying that if the initiator wash know what happened. That sense key is basically the information that will be send to the target to the initiator. Saying that something that happened error condition I am recording it you can look at it later.

(Refer Slide Time: 43:11)



The slide is titled "SCSI Error Models" and contains a list of error types. At the bottom left, there is a logo for NPTEL (National Programme on Technology Enhanced Learning) featuring a globe and the text "NPTEL".

- Bus reset:
 - Hard: kill all active and outstanding IOP and reset state of target
 - Soft: try to complete outstanding IOP and preserve current state of target
- Clear Q: (needs tagging) kills only active IOP and all those waiting for this target routine
- Abort Tag: kills only active IOP within ordered tagged Qs
- Abort: Kills all running IOP processes and those in Q
- Asynch Event Notification: from target to initiator
- ACA sense info that has to be read before next request

Again, razor script look at SCSI error models. I can there is an I am going through in some detail as because if you are designing a store system. Whatever level I am working it you have to really worry about all these kind of situations. Example if I look at nfs I has a same situation, let me let me go through this analytical we are related to the nfs situation. You have some promise called bus reset. Why does it happen? It sometimes turns out an electrical system, there could be unexpected behavior because of some malfunction of a particular device. And it may turn out that they might be what I call excess a reflections on the bus. You may want to reset the bus.

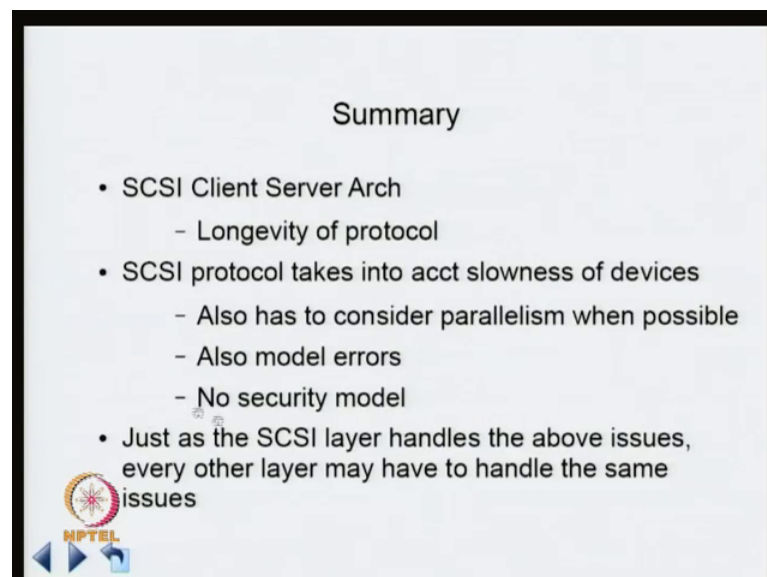
So now there has to be semantics about this be sitting. There can be one type called hard and soft. In hard case you basically kill all active and outstanding never processes. And this set the state of the target; that means, the disk or whatever it is it can forget about whatever state it has accumulated during the processing so far. Essentially you reset it to the some previous known well good state. Then the case of soft you basically saying that you believe something to be transient error, I wanted to try to complete the outstanding IOP process, and please do not change any state that are accumulated so far, it with respect to the device.

So, again let us look at nfs. In nfs also you have something like I connect to a particular remote file system. I can have what is called a hard mount and or a soft limo or a soft mount. Now of course, then the semantics are slightly different. In the soft mount case, I

try to do it a few number of times. And then if I do not succeed I stop. In a case or hard mount, I keep on trying it till I get access rate. Because is not the same as the way the hard and soft the words are used here, I just want to mention that there will be similar issues at a level there whatever, and even application about say might have similar issues. So, there are because there are queues also, you may want to say that I need a situation where only one initiator related thing is a problem. Or there is some issue with respect to one particular target routine. I want to only kill those things.



So, clear Q basically says and want to selectively kill certain things. So, there are various types of selective killing, and I think it is best studied by yourself, if you want to look into exactly all this is what they saw. And there is one thing which is called aca, it is call auto contingent allegiance or something. And basically, what it says is that him there is similar. It is basically saying that please do not eliminate that error information, right. Before the next request basically it has to be late. So, there is something important I do not want anybody to proceed to the next request before it has been write. So, is some kind important error information?

(Refer Slide Time: 46:37)



Summary

- SCSI Client Server Arch
 - Longevity of protocol
- SCSI protocol takes into acct slowness of devices
 - Also has to consider parallelism when possible
 - Also model errors
 - No security model
- Just as the SCSI layer handles the above issues, every other layer may have to handle the same issues

So, basically if you look at SCSI. It is a client server architecture. And because this client server architecture you can see that it has survered for quite some time. And the SCSI protocol also takes into account slowness of devices. Again, whenever whatever protocol you are doing, whatever level given an application level you need to worry about what

things you are dealing with; so against SCSI protocol at the lowest level talks to slow devices like disc and tapes and what not.

So, it actually takes into account the slowness of devices. So, for example, it tries to do some flow control with x at a protocol level itself there is something called (Refer Time: 47:19) transfer commands. So, that it the targan can target can indicate to the initiator that it is ready it has buffer. The thing it has got buffers ready to access the data it also has to take into account parallelism where possible, because this is a slow devices as much parallelism that you can exploit it is better. There are errors that also can crop up, and you have to figure out how to handle the errors. One thing about SCSI is that does not have any security model.

Nowadays when you have disks with encryption then this is as far as I know is outside as SCSI model. So, again to repeat on more time; it is SCSI layer handles are many of the issues. And it is own way any other layer above you will have to handle similar issues, in it is own particular base. And it is a good protocol if it knows the entities it is dealing with. So, having an accurate model or is it a good model about the kind of devices that you are dealing with or entities, you are dealing with that actually has a significant impact on your design and SCSI has many of the things. So, if you are familiar with some other protocols SCSI might look a bit strange approxily, because it is handling certain things which are not common with other protocols metal update.

I think I will stop at this point. And we will continue in the next class on the; I think, I am not clear what I am going to talk about, but I think I will talk about fiber channel possibly.

Thank you.