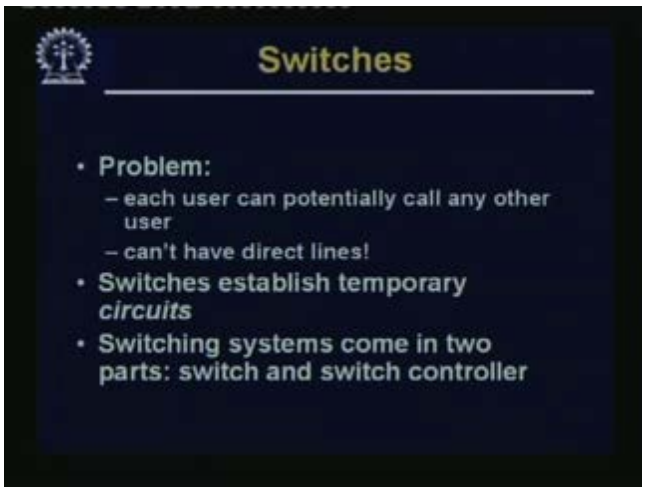
Computer Networks
Prof. Sujoy Ghosh
Department of Computer Science and Engineering
Indian Institute of Technology – Kharagpur

Lecture - 7
Switches-1

Good day. So today and in the next lecture we will be talking about switches. Switches are the most important part of a computer networks and we will look into some aspects of the different types of switches. In this lecture we will look at the kind of switches that are used in telecom networks. There is a special version of these, which we will cover in a later lecture and then in the next lecture we will be talking about packet switches, which are closer to the heart of data networks. First thing is why do we need switches?

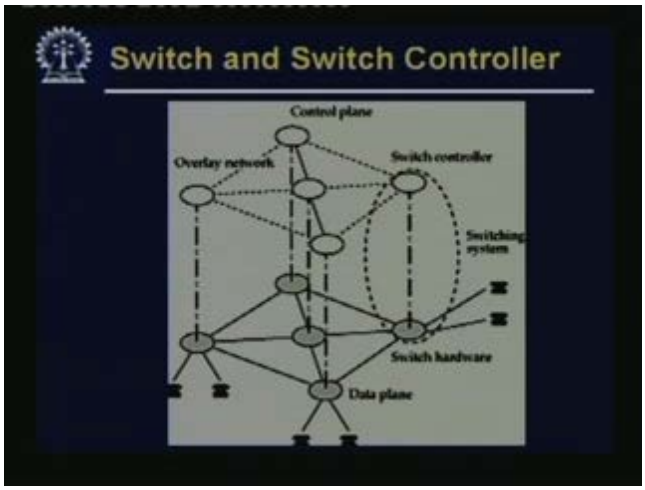
(Refer Slide Time: 01:38)



The problem is that each user can potentially call any other user, as we have seen in telephone networks and we cannot have direct lines. We cannot have an order n^2 kind of lines. So every user is connected to essentially a switch in the switching office. Switches establish temporary circuits; that means they establish a temporary connection between

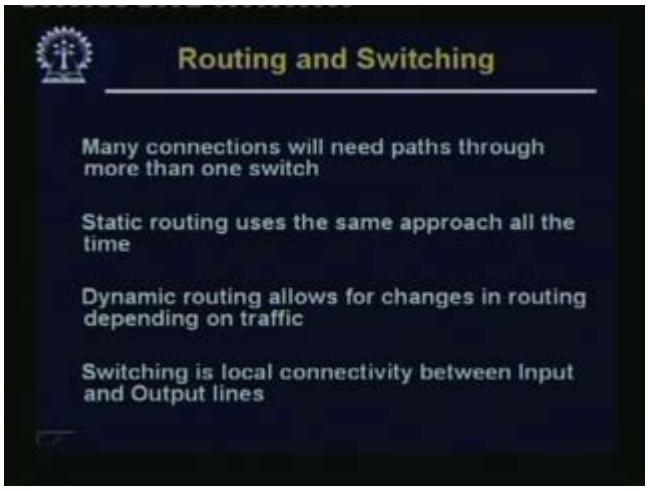
the caller and the callee at the switching station. Switching systems come in two parts: one is the switch and the other is the switch controller.

(Refer Slide Time: 02:16)



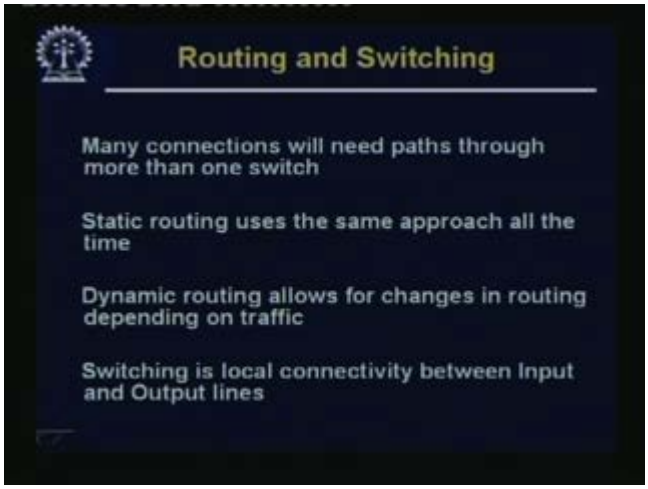
As you remember from our discussion about telecom networks, people want to talk to other subscribers, who are in maybe a different location, under a different switching station. The switching stations have to connect among themselves. For making this connection through the signaling that we talked about, there is a control plane over here. Control planes are some kind of computers, which are capable of sending some kind of signals or instructions to the individual switches. In the control plane, we set up the path and there is a corresponding signal from the controller to the corresponding switch for setting up a particular circuit or a particular connection, so that an end-to-end connection from one user to the other is set up. This control plane is sort of an overlay network over the actual switch, and once the connection is set up naturally the call and all the signals, etc., flow in this plane.

(Refer Slide Time: 03:42)



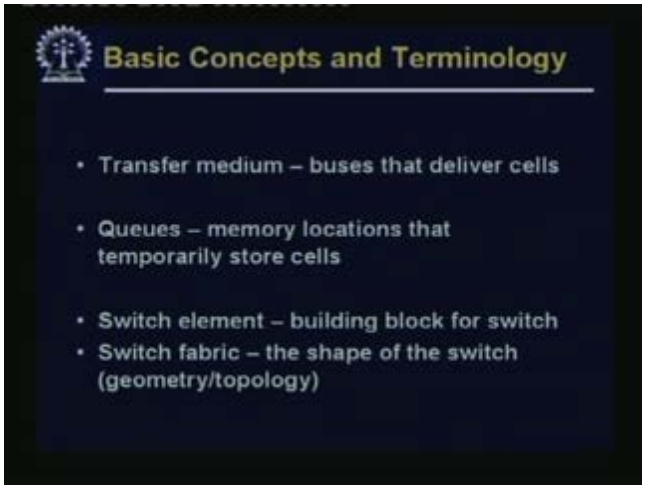
There is a network device called a router and there is routing, which is very close to switching. The only thing is that in routing, we decide about long-range paths. If you remember from our discussion about the seven-layer OSI protocols, you remember that there was a network layer and the router is a device in the network layer whose job it is to find out that in order to reach the ultimate destination station, what is the local link that I have to take here? For a switch it is slightly simpler; it is strictly a local decision but for the router there is a global implication – what is the local link on the output side that I must take for a signal coming from input line to a particular destination, so that I will finally go through all these connections through various other routers to reach my destination, whereas for a switch the job is a little simpler in the sense that when the signal is coming from one incoming link and is going to go out to a particular outgoing link, which is known at the switch, how to set up the connection? Actually, a router is a switch with some added functionality.

(Refer Slide Time: 05:34)



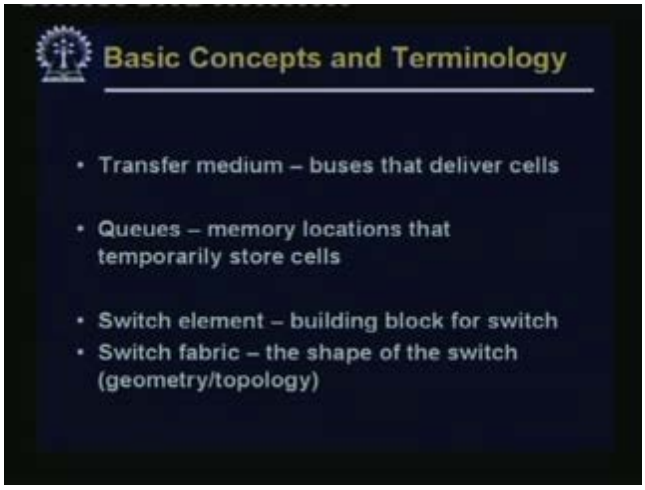
Many connections will need paths through more than one switch, so that is when we required routing. Static routing uses the same approach all the time; that means same destination is routed through the same path all the time. A dynamic routing may allow changes in routing depending on traffic; that means some part of the network may be congested, so it might want to avoid that path and go to the same destination through some other path. As we have seen at least in telephone networks, because of the way we distribute the telephone numbers to subscribers, it is easy to route. You can look at those incoming digits and just decide on the route. Switching is the local connectivity between input and output lines. Now we will discuss some basic concepts and terminology.

(Refer Slide Time: 06:31)



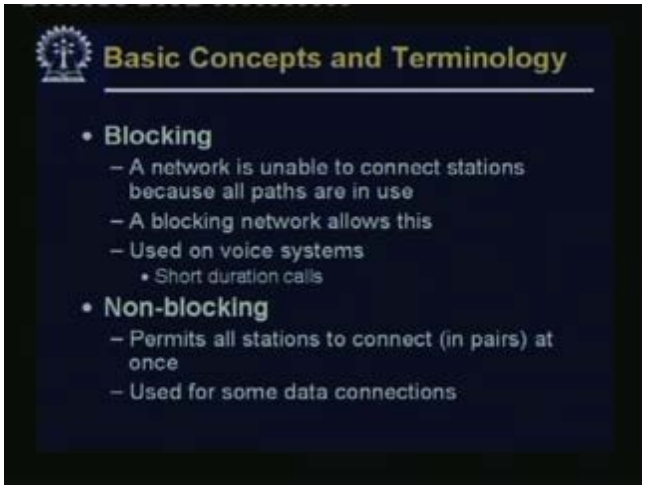
One is the transfer medium – transfer medium in a switch, could be buses, electronic buses, through which you can send a stream of cells. Queues are memory locations that temporarily store cells. Switches often require some queues; the queues may be on the input side or the queues may be on the output side; queues may be in various places. So we will look at all these different cases. But these are temporary memory locations, where we temporarily store some cells till we know what to do with it or forward it to its destination. A switching element is the building block for the switch. A switch may be either ON or OFF and if it is ON that connection is there or if it is OFF the connection is not there. Sometimes the switch is like this – at a particular point a switch may connect either to one point or to another point.

(Refer Slide Time: 07:58)



A switch is just a simple element and in a big switch, we have a lot of these switching elements. These switching elements form the so-called switching fabric; that means the shape of the switch: like, how the switches are organized so that all the incoming calls can be handled. There is a term called blocking and non-blocking.

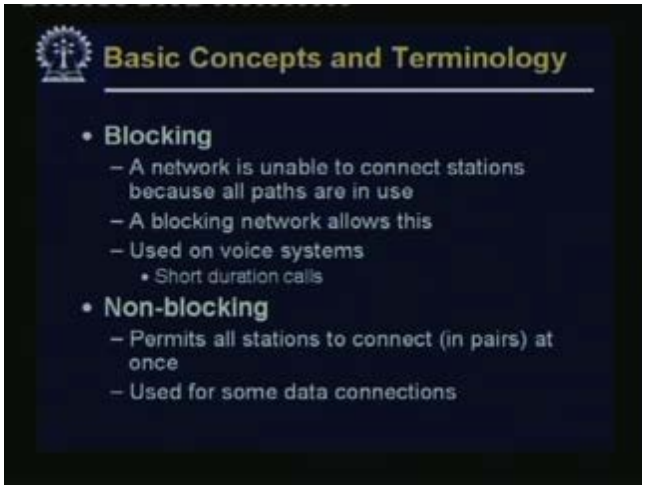
(Refer Slide Time: 08:18)



Blocking means a network is unable to connect stations because all paths are in use. A blocking network allows this. This is used in voice systems and the main reason is that

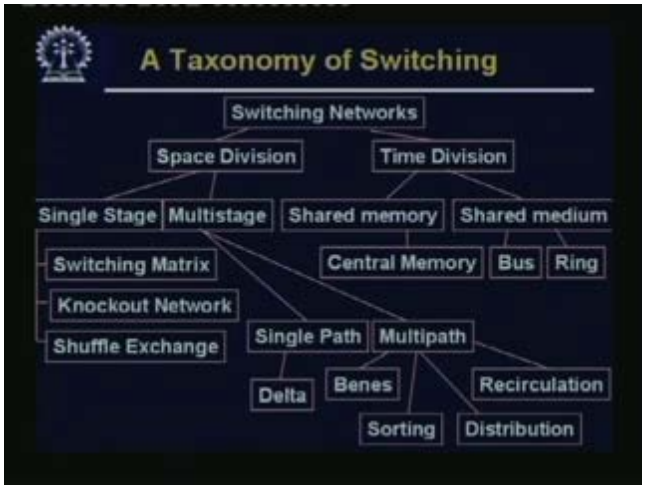
we have short-duration calls. The point is that there are so many subscribers, may be thousands of subscribers; if we want to provide the switching capability to the maximum possible extent, that means, if everybody is busy talking to somebody or the other, that means, if every subscriber is using this, then if we try to provision our equipment based on this assumption that everybody may be talking to everybody at the same time, then the switch will become very big and very expensive. So what the telephone people have done is that they have made an extensive study of how many people are likely to talk during a particular time of the day, especially Bell Labs in USA. They were the pioneers and a giant telephone company, which was the largest at one point of time, and later was broken up etc. During the early part of may be 1940s and 1950s, they made a very extensive study about people's calling habits. They found out that during the busy time or the peak hour when all the officers are open, how many people out of a subscriber base are likely to call. That is one kind of statistic they collected; and the other kind of statistic they collected is that if one person starts talking what is the distribution of the call duration; for how long is this particular subscriber likely to hold on to that line. They found the statistical distribution; and based on that they decided that if I provision this much in terms of our switching elements and switching capacity, then almost all the time everybody will be able to get the service, but exceptions may occur. This is based on some statistical study at some particular point of time. It may so happen that the number of subscribers who wanted to speak is just larger than the overall switching capacity that you have. In that case some of the calls will be blocked. I am just talking in terms of switching capacity. This kind of blocking may occur elsewhere in the network also. We provision based on some statistical averages but at exceptional times, for short durations of time, we may have to block some of the calls because we do not have the capacity to handle so many calls at the same time. So that is a situation where it is blocking and what people try to do is that they try to calculate the blocking probability, which may be 1%, 2% or 5% – 5% is quite high, as nowadays people do not even like that. That is the blocking probability that we wish to achieve – we are provisioning this much for these many subscribers.

(Refer Slide Time: 12:08)



Non-blocking permits all stations to connect in pairs at once. This may be used for some data connection where blocking is not allowed or blocking is not acceptable under any circumstances. If you want to make it fully non-blocking, that solution would be more expensive than if you allow a limited amount of blocking. Of course a high degree of blocking will not be acceptable to the users at all.

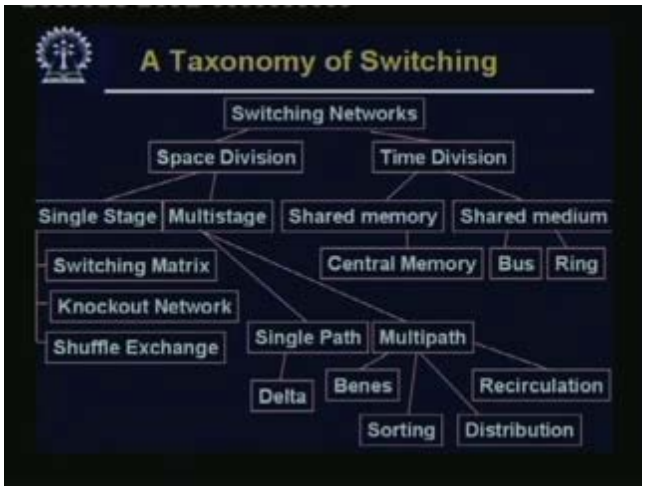
(Refer Slide Time: 12:39)



There are various kinds of switches. I am trying to present here a taxonomy of different types of switches – not that all actual switches neatly fit into the boxes. Some of them

take some portions of this and there is some kind of a hybrid or mixture. But this is one way in which the different types of switches can be classified. We have space division switching and time division switching. Like in multiplexing we have a time division multiplexing and frequency division multiplexing, here we have space division switching and time division switching. Time division switching is very close to time division multiplexing as we will see.

(Refer Slide Time: 13:37)

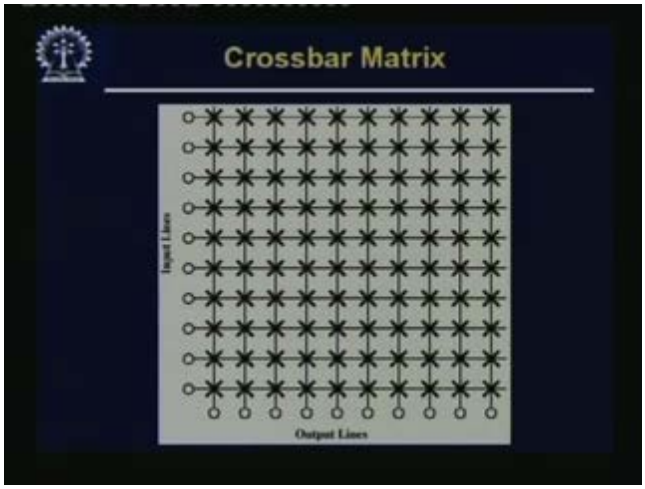


Now space division switches maybe single stage or multistage. In single stage we have a switching matrix; then we have a knockout network, shuffle exchange networks. In multistage, it may be a single path or multipath like delta networks or Benes networks, sorting networks, etc. In time division switches we have shared memory and shared medium. Shared memory could be a central memory or shared medium could be a bus or a ring etc.; so these are different types of switches. We will not have the time to look at all these in detail. So we will just look at some of the very common types of switches that are used.

(Refer Slide Time: 14:23)

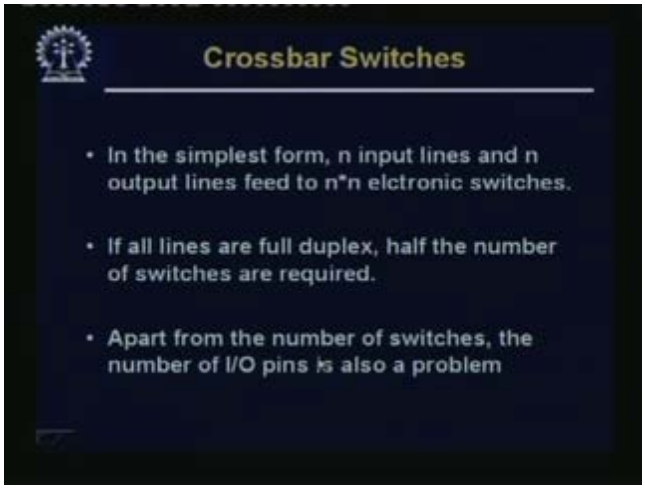
Let us first talk about space division switching. It was developed mainly for analog environment. Why do we call it is space division – because the different calls go through separate physical paths. That is why they are called space division switches, unlike time division, where different calls may be going through the same physical path at different time slots; we will come to that when we discuss time division switching. But in space division switching, different calls go through separate physical paths. The simplest of space division switches is the cross bar switch. The number of cross points grows as the square of number of stations; loss of cross point prevents connection; if a particular cross point becomes bad it is an inefficient use of cross points; all stations are connected but only a few cross points are actually in use; and this is fully non-blocking. Let us first have a quick look at a cross bar switch. A cross bar switch is very simple.

(Refer Slide Time: 15:32)



What do we have in a switch? In a switch you have n input lines coming on from one side and n output lines – you have to connect some of the line. Let us say I want to connect this particular line to let us say this particular output line, i.e., the fourth input line to be connected to the fifth output line. I want this particular switch to be ON and the other switch OFF. If this particular switch is ON, there is a connection between this line and this line. So I have a direct physical path between the fourth input line and the fifth output line. And each of these crosses that you see is a single switching element and as you can see, since there are n lines i.e., if there are n lines on the input side and n lines on the output side, we have n^2 switching elements, which actually is quite a lot.

(Refer Slide Time: 16:37)



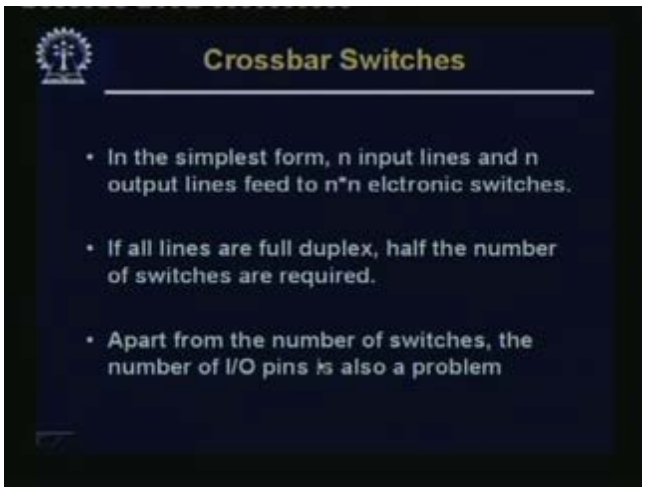
So if you go back the number of cross points grows as the square of the number of stations.

(Refer Slide Time: 16:38)

If a particular switch, which is at the crossing point between the fourth input line and the fifth output line, if that particular switching element, somehow has gone bad, then four and five cannot talk to each other ever. That is again another drawback of this kind of switches. Of course it is an inefficient use of cross point – at every cross point we have a

switch and all stations are connected but only a few cross points are actually in use. At most n/2 cross points can actually be in use, whereas we are using n^2 cross points, which is a really a poor utilization of the switches. One good thing about this kind of switch is that it is totally non-blocking; that means, all pairs of subscribers can talk to each other at the same time.

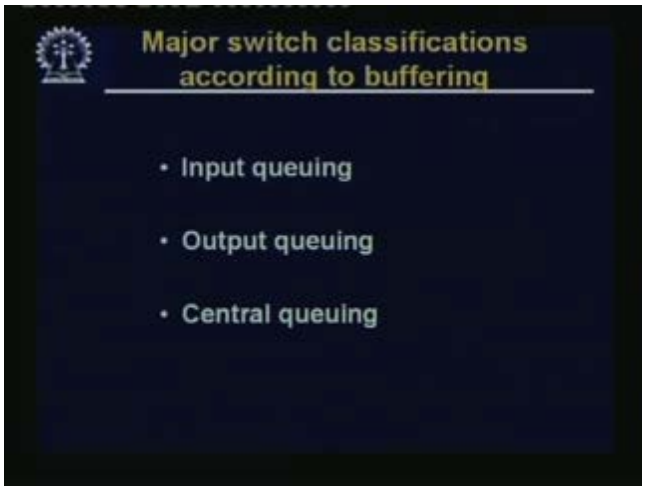
(Refer Slide Time: 17:50)



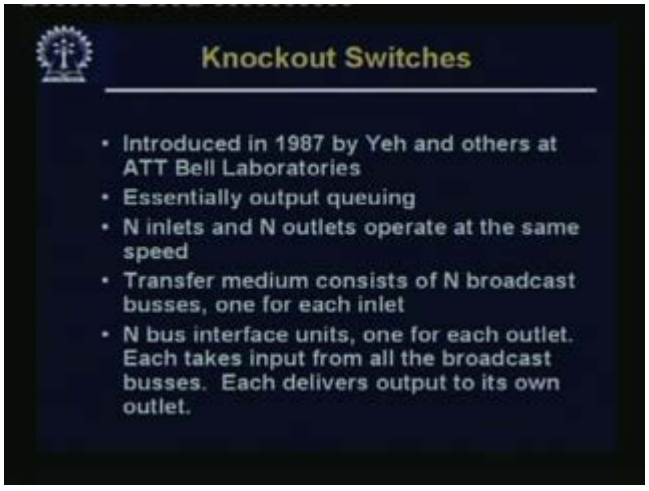
The cross bar switch is the simplest form of n input lines and n output lines that feed to n^2electronic switches. If all lines are full duplex, that means all lines allow communication in both the directions, then we require only half the number of switches. We possibly do not require the diagonal switches, the switches which are in the diagonal. I do not want to talk to myself; the other thing is that only either the top half or the bottom half of the switches – top half means top half of the principal diagonal or the bottom half of the principal diagonal – only those switches are required . In general $n^2/ 2$ switches would be required if you have full duplex lines. The other problem about the cross bar switches is that apart from the number of switches, the number of I/O pins is also a problem. If you want to a make let us say 1000/1000 line switches for 1000 subscribers, you may have 1000 incoming lines, 1000 outgoing lines, but how do you put all these 2000 pins on the chip? It also becomes a problem – people have gone to more

sophisticated design for switches. They are classified in various ways: through input queuing, output queuing or central queuing. We will look at these one by one later on.

(Refer Slide Time: 19:32)



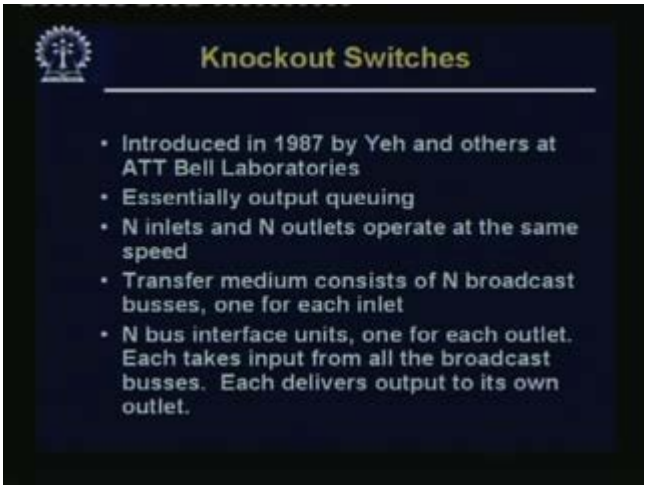
(Refer Slide Time: 19:35)



Let us look at another kind of a space division switches known as knockout switches, which were introduced in 1987 by AT&T Bell Labs. Essentially we require using output queuing over here. We will look at output queuing and the issues in detail later on. If the switch is trying to pump in more number of bits or more number of bits to a particular output line than that an output line can handle, you have to either drop them or you have

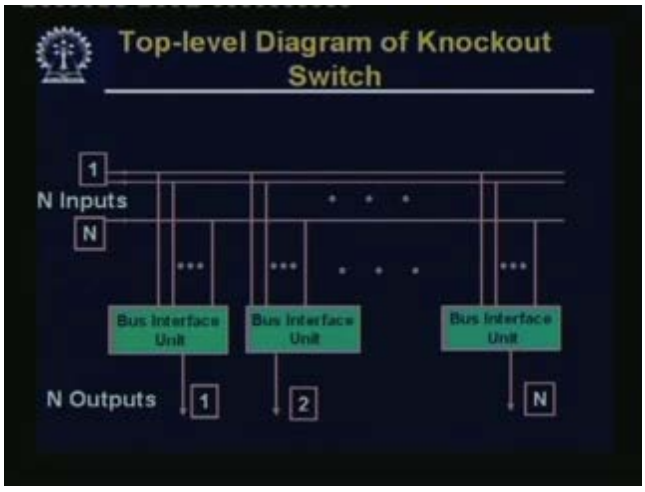
to store them somewhere, maybe in a queue, and that queuing is associated with the output ports. That is what we mean by output queuing. So N inlets

(Refer Slide Time: 20:26 – 20:44)



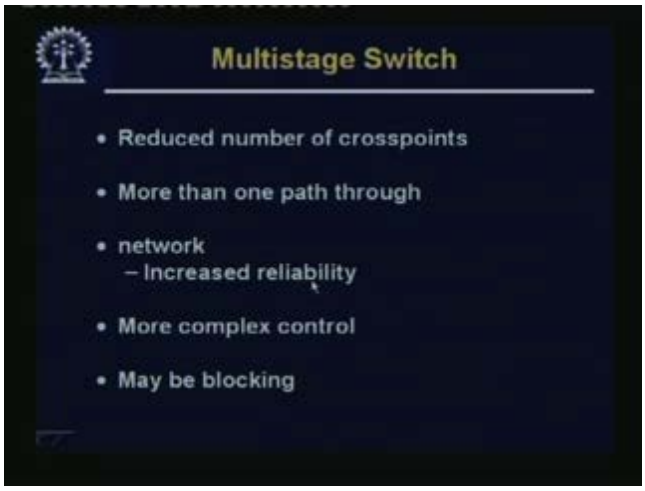
and N outlets operate at the same speed. Transfer medium consists of N broadcast buses, one for each inlet. N bus interface units, one for each outlet, takes input from all the broadcast buses. Each delivers an output to its own outlet. Let us take a look at the figure.

(Refer Slide Time: 20:47)



So we have the N input lines on one side. A call may come from anywhere, let us say subscriber 1 is trying to call to subscriber 2. So what subscriber 1 will do is that it will broadcast the signal over this bus. There are N buses over here and all these N bus interface units are connected to all the buses, each of the bus interface units – 1, 2, 3 up to N – will receive that signal from N but this bus interface unit 2 knows that this is supposed be meant for me so he will allow this signal to transfer to the output line, whereas others will simply ignore it. This is again another fairly simple scheme. We have, of course, reduced the number of switching elements that are required.

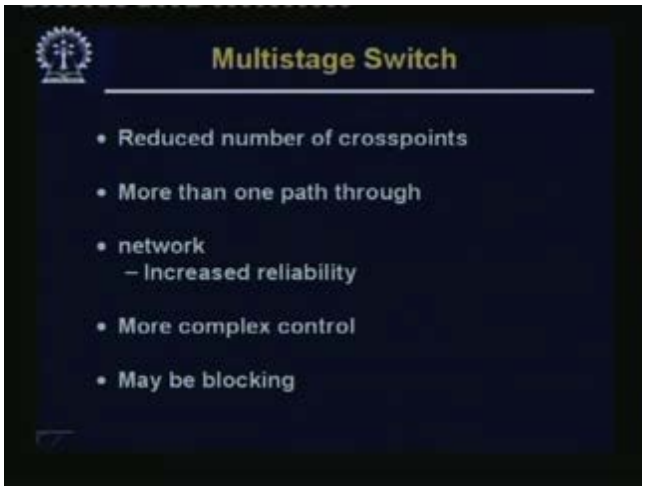
(Refer Slide Time: 21:53 - 22:01)



It is not using a bus, and you will not get a very fast switching, but this works fairly well. If you want to go beyond this, that means, you do not want so many lines, etc., as I said and even if you have 1000 subscribers you want to service from the same switch, even in knockout switch you have 1000 buses inside, which is not technically feasible. So we go to the other kind of switches which I showed earlier; namely, multistage switches and multistage switches have one of the most commonly used switching architecture even today, so far as space division is concerned. In multistage switches we may actually provision less number of switching elements than are strictly required for the highest use,

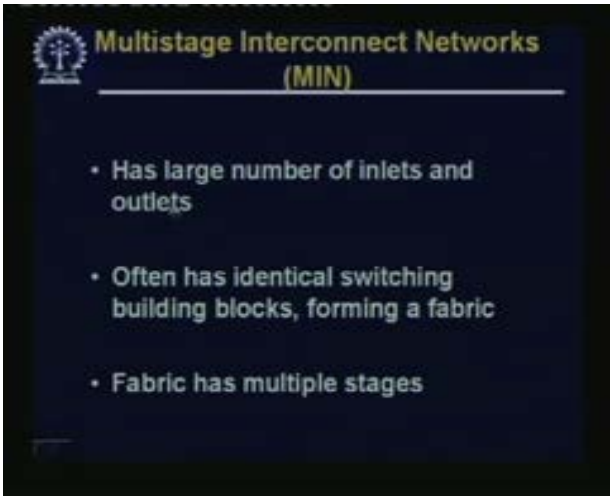
as we want to reduce the cost of the switch. All the input lines can go to all the output lines and it is broken up into several stages as we will see.

(Refer Slide Time: 23:19)



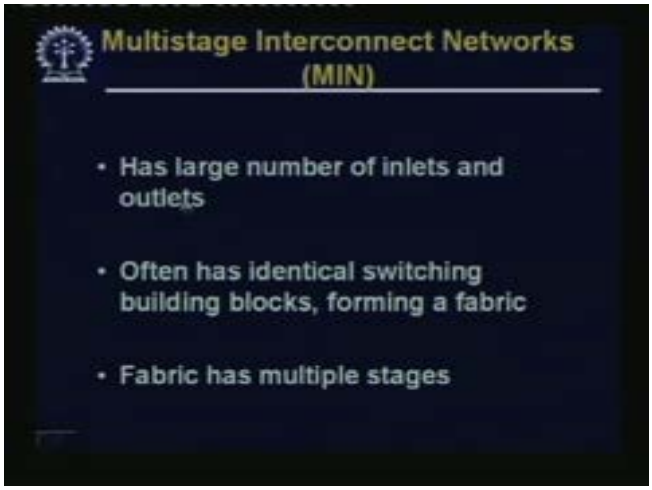
The advantage of a multistage switch is that it has reduced number of cross points, more than one path through the network; this increases reliability. As you remember, in our simple cross bar switch, if one particular switch is bad that particular pair of users cannot talk to each other; that is not really desirable. Since there are more than one path through the network, even if one particular switching element becomes bad they are still able to communicate. The control becomes more complex and it may be blocking. So, as I said, in most of the cases it will not block, but in some cases, it may be blocking.

(Refer Slide Time: 24:05)

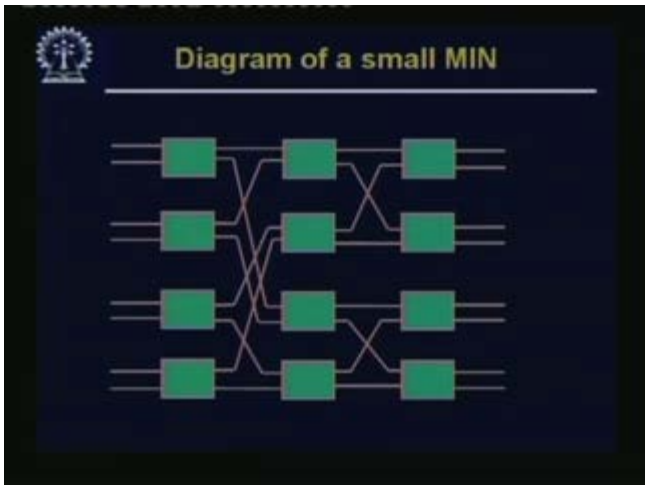


They have a large number of inlets and outlets which are, sort of, grouped into small groups so that these problems with pins and input lines are not there. Often they have identical switching building blocks, forming a fabric. So there are many switching building blocks consisting of some number of switches and such modules are used all over the switch. As you can guess, in a multistage switch, we will have different stages. The input will come from one side and will go to one stage, then may be another stage, and a third stage, and then may be out. So there are multiple stages and each stage is broken up into small modules. These modules may or may not be identical, so it is sort of cheaper to manufacture and it also handles all the pins, etc., in this fashion. So it often has identical switching building blocks forming a fabric, and a fabric has multiple stages.

(Refer Slide Time: 25:11)

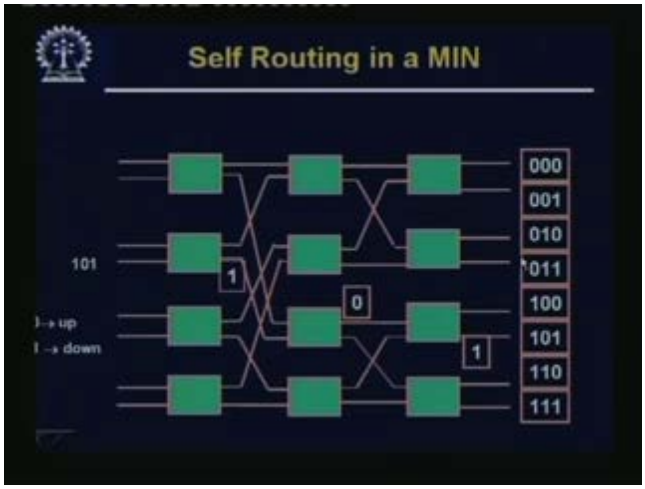


(Refer Slide Time: 25:16)



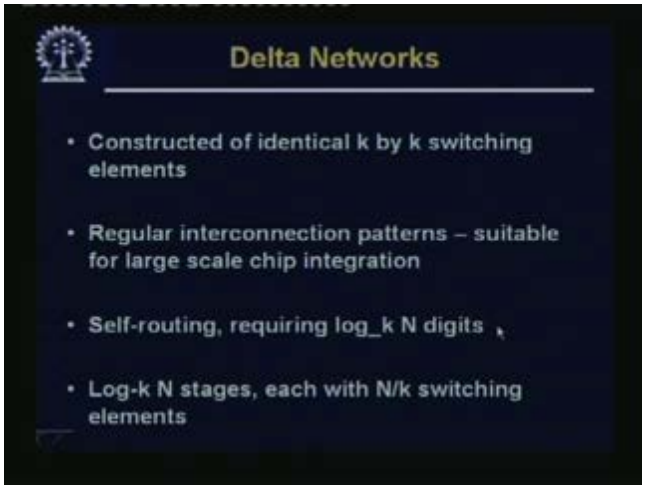
This is just one example of a small multistage interconnection network. So as you see, this is one stage, this is another stage and this is the third stage. There are three stages over here. Each stage is broken up into small blocks; so this is an 8 × 8 sort of switch. If you remember to have 8 × 8 switches in our cross bar switch, we require 64, or even if we take half of it, maybe 30 or above 30 switches. Here each is a 2 × 2 switch; so actually you can handle it with only one switch. The number of switches has come down drastically but there are problems, as we will see, that it may be blocking kind of system.

(Refer Slide Time: 26:17)



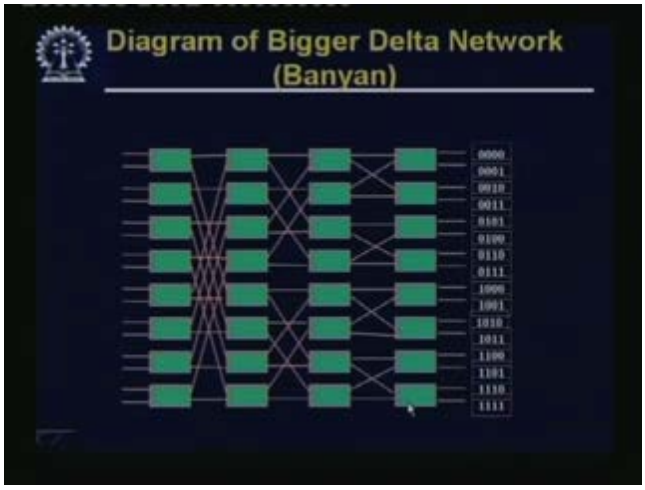
So how do you know that something, which is coming over from an input line, would reach the particular output line? Suppose the output lines are numbered and since there are eight output lines, we can code them using these three bits, from 000 to 111 in a binary fashion. This particular line, which really is the input line number 011, wants to have a call to the output line number 101. If you know to which output line you want to go to, 1, 0, and 1 really become control signals for these switches on the way. So 1 means go to the lower output line; 0 means go to the upper output line. And you can see that those that are starting with 0 come at the top, those that are starting with 1 come over here and that is the same binary number order that we are maintaining. The switch is simply a rule that if it looks at bit, if it is 0 it connects to the upper line and if it is 1 it is connected to the lower line. So 1 will connect to the lower line, 0 it will connect it to the upper line. 1 over in this stage will be connected to the lower line and we will reach 101. This is almost self-routing through the switching fabric and because of this, routing is easy.

(Refer Slide Time: 28:02)



Delta network is one kind of a multistage interconnection networks. It is constructed of identical k × k switching elements. It has regular interconnection patterns suitable for a large-scale chip integration. It is self routing as we have shown, requiring log_k N number of digits and a log-k N number of stages each, with N/k switching elements.

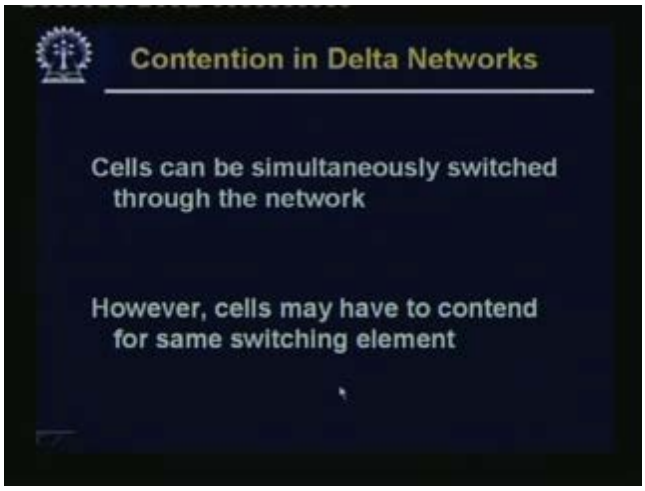
(Refer Slide Time: 28:34)



So this is the picture that we had seen earlier; this is the diagram of a bigger delta network, may be with 16 input lines and 16 output lines. As you can see, this is a four-stage switch, because in order to code for 16 lines, it will require four bits from 0000 up

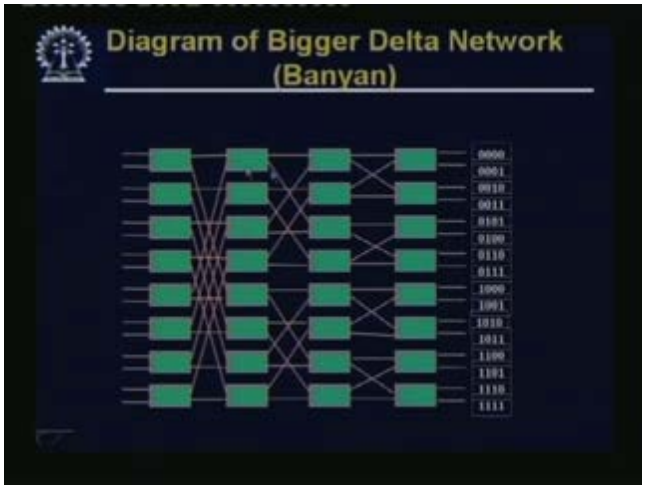
to 1111. Each bit decides whether that particular switch connects you to the upper output line or the lower output line and from any of these input lines there is a path to any of the other output line. This is the 16 × 16 Banyan network. As you can see, by doing it in this multistage fashion in a regular cross bar switch, while we require some thing like 125 switching elements here, you require only, say, 32 switching elements. So the number of switching elements is much less.

(Refer Slide Time: 29:54)



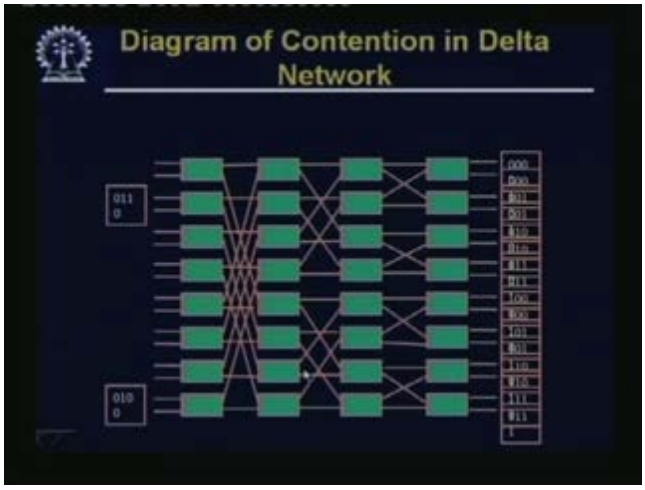
But there may be a problem of contention in a delta network. Cells can be simultaneously switched through the network. However cells may have to contend for the same switching element. Let us go back to the earlier diagram

(Refer Slide Time: 30:13)



You see this, say, 0000 wants to talk to 0001, so he will simply go like this and similarly another line over here, may want to go straight down to somewhere over here. So many calls can go through together but this is not always the case. There may be a contention for the same switching element as we will see in this diagram.

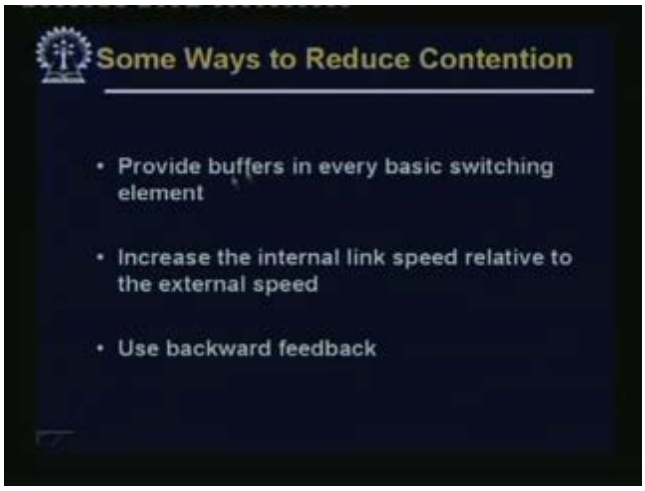
(Refer Slide Time: 30:42)



Let us say 0 1 1 0: so 0, 1, and then 1,0; 0 will come here, then 1 will go straight; 0 1 1 0; and then 0 1 0 0. So 0, upper 1, 1, and you see that we have potentially got a contention

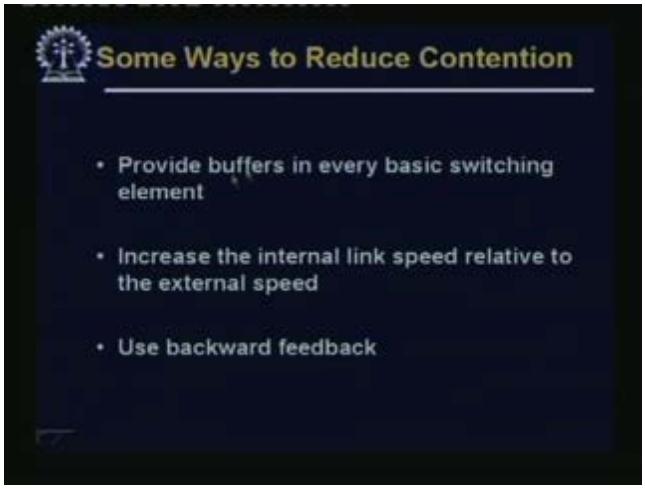
for the same switching element from two different points in the network; it is sort of converging to the same switching element before it can proceed. So there is going to be a contention over there. If there is a contention, obviously both of them cannot be handled at the same time; either one of them has to wait. You have to make it wait by putting it in some buffer, etc., or one of them will have to be blocked or dropped. Although we save on the number of switching elements, sometimes we get a contention.

(Refer Slide Time: 32:26)



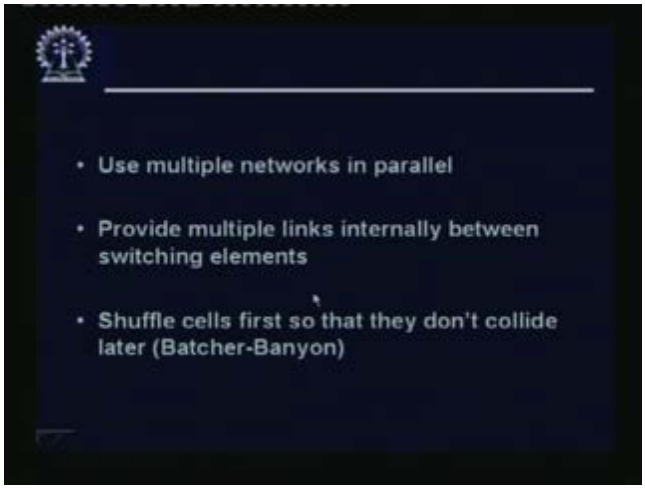
Some ways to reduce contention is to provide buffers in every basic switching element, increase the internal link speed relative to the external speed, and use a backward feedback. These are different sort of techniques to reduce contention. One is of course providing buffers as we have already discussed; you could increase the internal link speed relative to the external speed. That means if the internal link is much faster than the external speed at which you are pumping in, then it may be able to handle both of them.

(Refer Slide Time: 33:04)



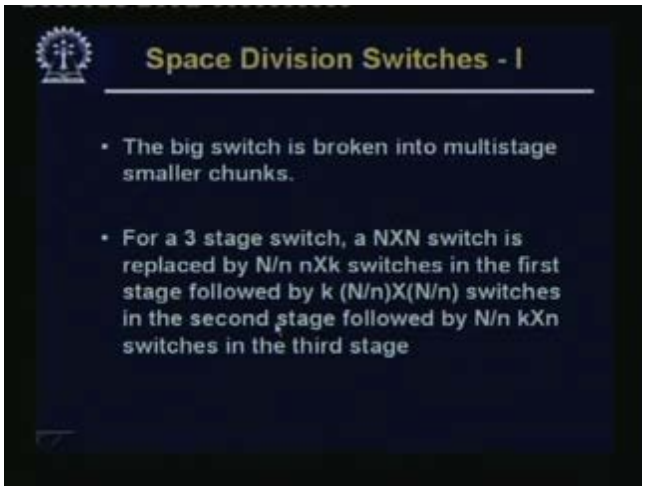
So far as the input streams or the output streams are concerned, they will not notice the faster switching that is going on within the switch. So that they will think that there is no contention. You can use backward feedback; that means, at some later point in the main if you find that there is a contention then give one of them feedback so that it sort of re-circulates.

(Refer Slide Time: 33:32)



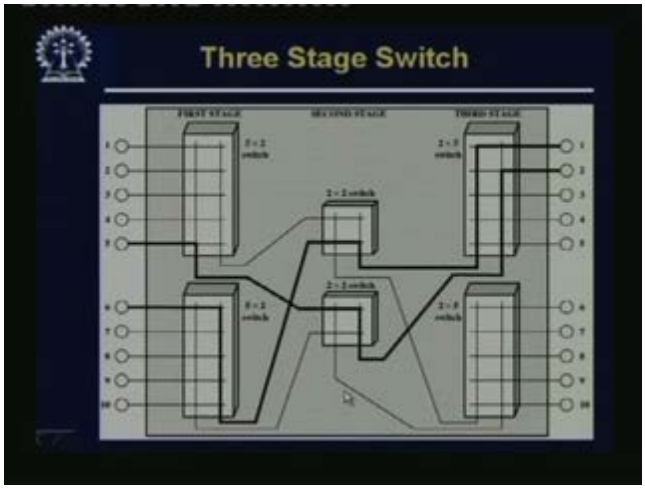
The other ways are to use multiple networks in parallel; provide multiple links internally between switching elements; and shuffle cells first so that they do not collide later. This is also another kind of switch that is quite common; it is known as the Batcher Banyan switch. We have seen the Banyan switch part; in Batcher part there is some shuffling of the input cells so that in the Banyan part they do not collide by contending for the same switching element.

(Refer Slide Time: 34:08)



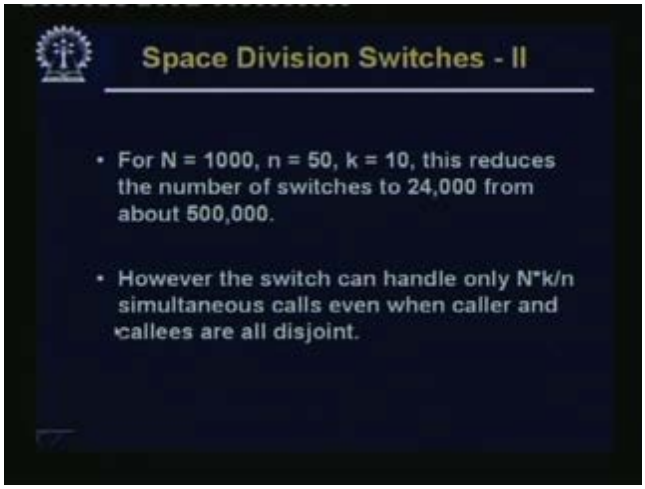
In the space division switch, the big switch is broken into multistage smaller chunks. I will show a three-stage switch. Suppose I want an N/n switch; suppose it were a cross bar, then it would be an N/n switch. This is replaced by N/n number of n × k switches in the first stage, followed by k (N/n) × (N/n) switches in the second stage, followed by N/n number of k × n switches in the third stage. So let us take a first look at a simple example.

(Refer Slide Time: 34:58)



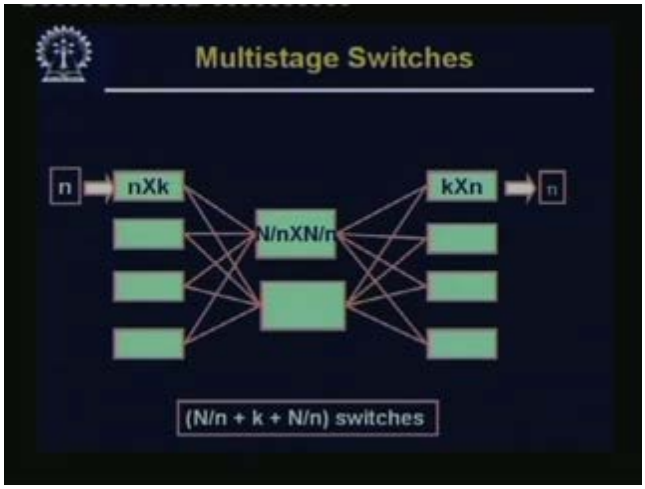
So this is a three-stage switch; in the first stage, we have 10 of them. So N is 10; and k is 2. So we have 5 × 2 switches. We have 1 2 3 4 5 at the input stage, so N/n, that is, two of them or 5× 2 switches and then two 2 × 2 switches followed by 2 × 5 switches. Let us say 5 wants to speak to 2, so 5 comes here; gets switched in this fashion to 2; and 6 wants to talk to 1. So 6 goes to this switching fabric in this fashion in the third stage.

(Refer Slide Time: 35:58)



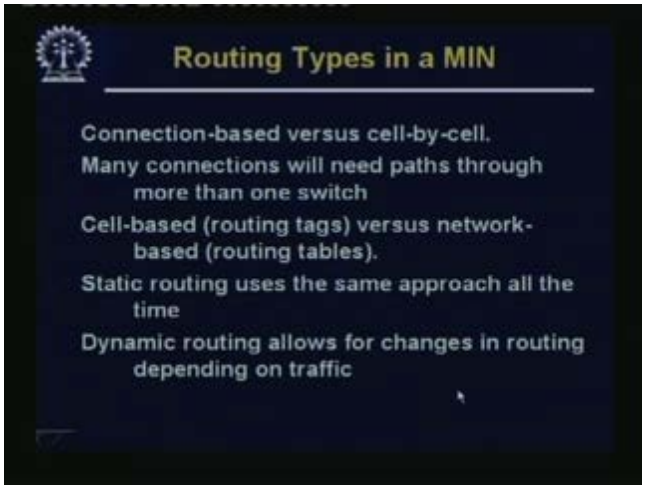
If you take the general case of N = 1000, n = 50, k = 10, this reduces the number of switches to 24,000 from about 500,000 because 1000 × 1000/2 is 500,000; that is, the number of switching elements you would require for a simple cross bar kind of switch, whereas by breaking it up into three stages in this fashion taking this particular value of n = 50 and k = 10, you reduce the number of switches to 24,000. However, the switch can handle only N*k/n. So if N = 1000 and k = 10 and n = 50, so N * k/n would be 200. So only 200 simultaneous calls, so only 400 users can use it at the same time. As I mentioned, it may be fairly alright because you have a phone but you are not talking on that phone all the time; most of the time actually it is just lying idle. If the telephone company makes a good study and finds out that 400 is the maximum number that is sufficient for a very low blocking probability even during busy hour, then that is fine.

(Refer Slide Time: 37:34)



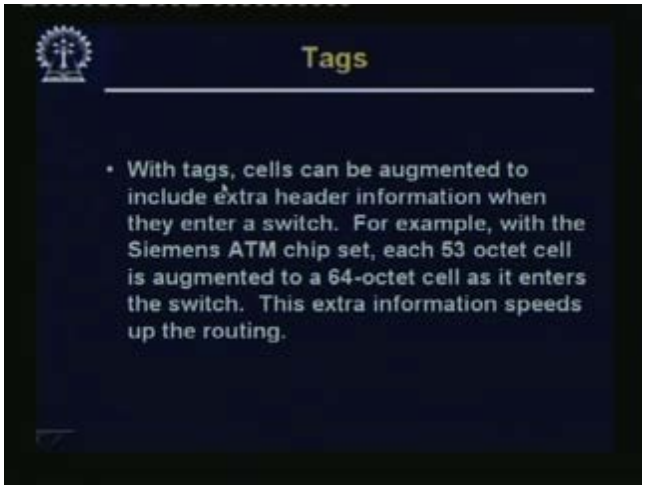
The general scheme of this is that you have N lines fitting into each of these N/n modules at the first stage. So you have a total of N lines fitting into one module. So you have n × k kind of switches so that this switch can be directed to any of the switches on this side. So there are k of them here there are N/n switch modules in this first stage, k modules in the second stage, and again N/n modules in the last stage, third stage. So you have (N/n + k + N/n) modules.

(Refer Slide Time: 38:25)



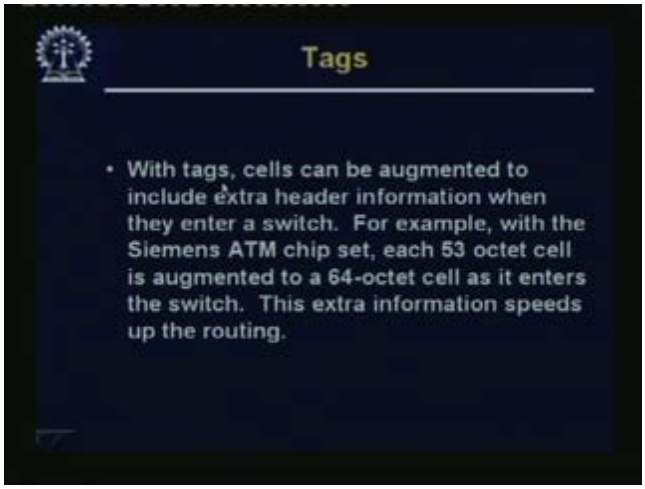
In the multistage interconnect network, you may either have a connection based kind of system, that means, once a connection is made it is permanent till the call is held, as we have in a usual telephone network, or it may be a cell-by-cell, that means, although the user sort of goes on having the conversation, his stream of bits is divided into small cells and each cell is individually routed through the main. Many connections will need paths through more than one switch – cell-based means routing tag versus network-based, which is routing tables. So if you have tags it is faster; if it is table it is slightly slower and as I said, the static routing uses the same approach all the time; dynamic routing allows for change in routing depending on the traffic. So what is a tag?

(Refer Slide Time: 39:32)



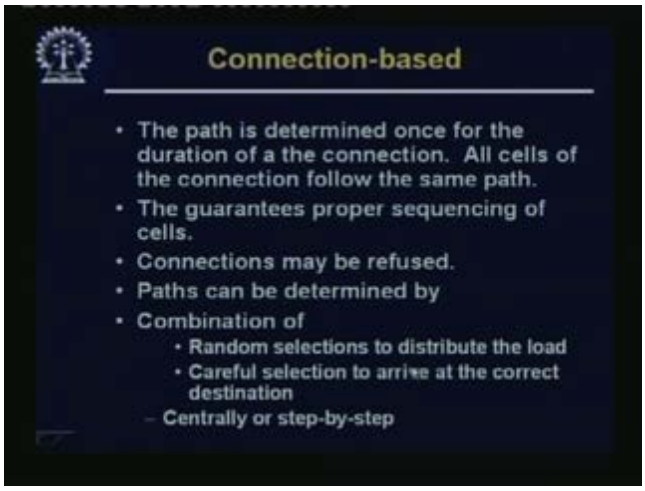
You remember that in a delta network we were putting 001 etc.; we were encoding the input or the output, specifically the output lines, and then using that address so that the cell can route itself through the network. That is an example of a tag, which means that within a switch each of the lines has got some kind of code in it; you might call it tag. This tag will be used for routing. Now if it is self-routing, like the one shown in delta network, which is very simple, it will be very fast.

(Refer Slide Time: 40:27)



If it is some other kind of tag we are using where such a simple scheme does not hold, then maybe we have to look up to some table to find out where I want to go. With tags, cells can be augmented to include extra header information when they enter a switch. That means you look at which is the output line it wants to go to and then add the code for that output line on to this cell, which sort of becomes the tag. For example, with the Siemens ATM chip set, each 53-byte octet or byte cell is augmented to a 64-byte cell as it enters the switch so these extra bytes are for putting in this tags, etc. This extra information speeds up the routing process.

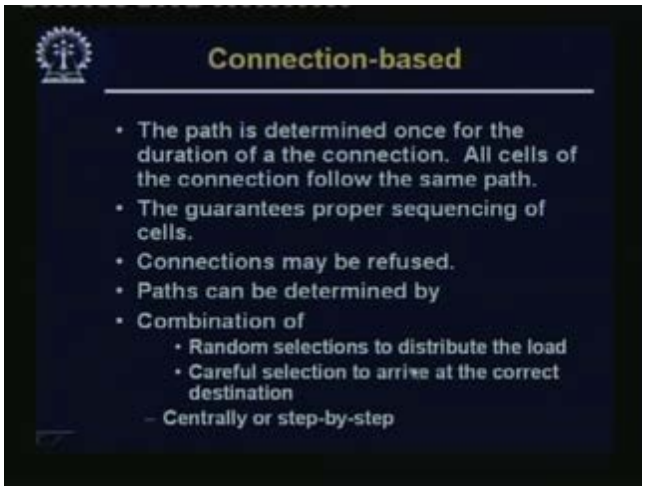
(Refer Slide Time: 40:59)



In a connection-based system – we discussed this earlier, but once again – the path is determined once for the duration of the connection. That means, once the connection is made, for the entire duration of the connection the path remains the same. All cells of the connection follow the same path. Now it may be a connection-based system with cells; that means, once the connection is set up, a particular path is designated that this is the path all the cells, which are originating for this particular call, will take. So, all cells of the connection follow the same path. This guarantees proper sequencing of cells; of course, they will now go in an order. The connection, of course, may be refused: that means when you are trying to set up a connection you find that in an intermediate stage

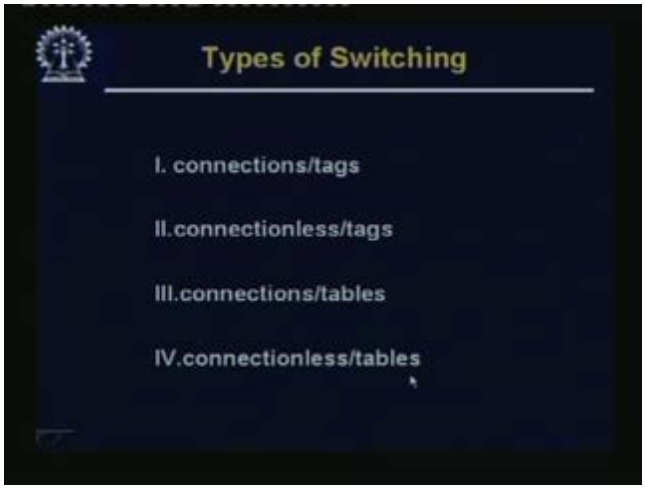
there is not enough resource. There is too much contention for that, so the connection may be refused. Paths can be determined by a combination of random selection to distribute the load. If you remember, in a multistage switch specially, there are various paths from the input to the output; there are various possible paths, so that if one path is not alright because of some defect or one path is not very good because at that particular point of time that path is loaded, you may route your path through some other path of the switch.

(Refer Slide Time: 42:32)



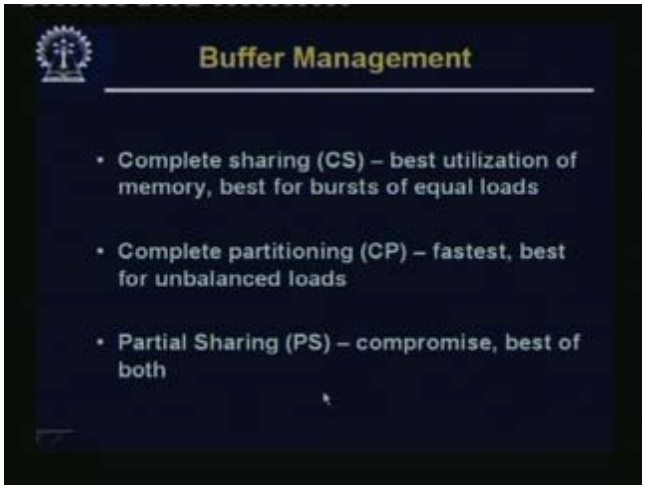
You might do a random selection amongst the paths to distribute the load, do a careful selection to arrive at the correct destination – if this and this selection can be done either centrally or in a step-by-step fashion as is done in a delta network.

(Refer Slide Time: 42:50)



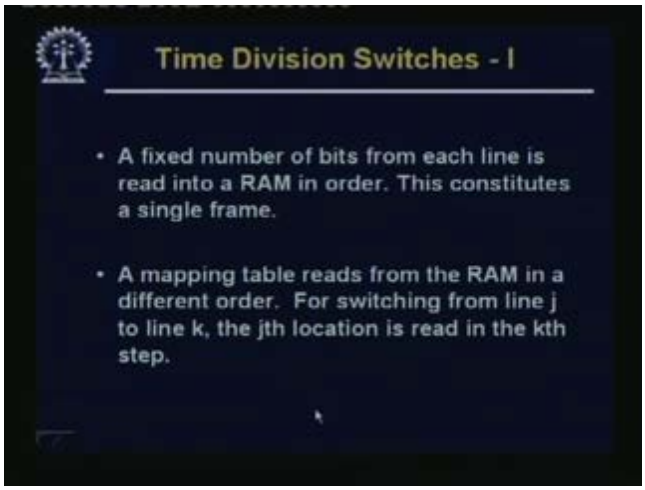
We have different types of switching – we have connections plus tags, you have connectionless, which are tags. Of course we will look at connectionless switches in more detail when we discuss packet switches. So the same discussion that we have now, part of the discussion that we are having here, could also be useful in the next lecture. We have connectionless plus tables, that means, if tags are not possible you have to do some table lookup because this table lookup will always be slightly costly in terms of time. The kind of speed that you can achieve with a table lookup becomes limited because of that; but any way if this is again a possibility that you can have connections-oriented system by using tables, you can have connectionless systems using tables; this is also used in routers specially.

(Refer Slide Time: 43:50)



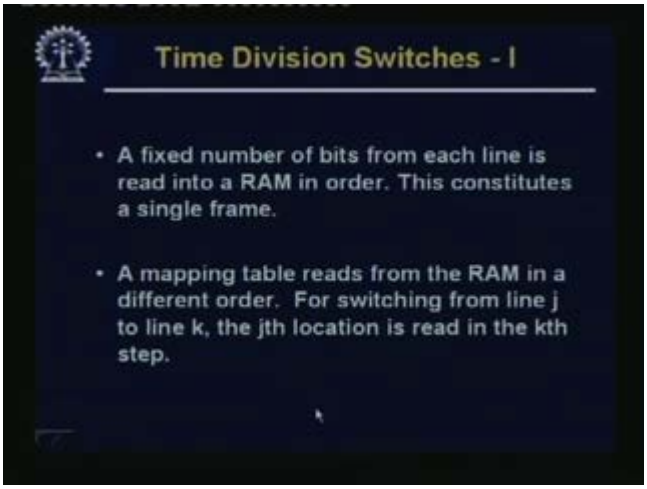
We will look into buffers in more detail when we discuss packet switches. Buffers may be complete sharing, which is the best utilization of memory and best for bursts of equal loads. Complete sharing means everybody can share all, so there is a central memory kind of thing, which is shared. It may be a complete partitioning; naturally complete partitioning would be faster and best for unbalanced load. Partial sharing means some of it may be centralized, some of it may be distributed, etc. There may be various strategies regarding buffers.

(Refer Slide Time: 44:41)



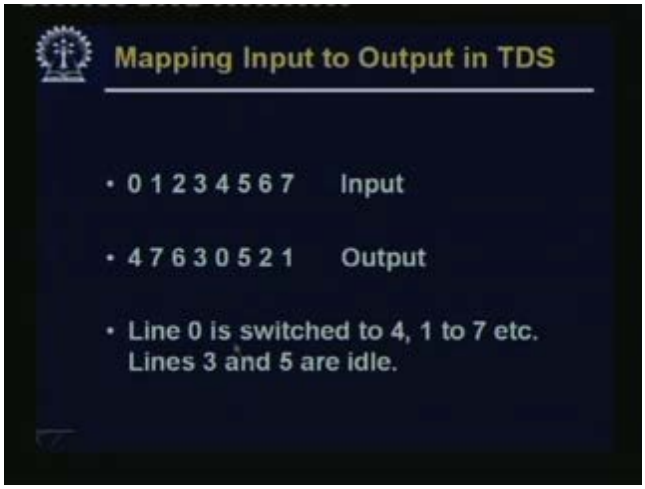
Next we come to time division switches – time division switches follow almost the same principle as time division multiplexing, in which the same line was being used for communication by different channels; what we did was that we kept a specific time slot for one channel and these time slots go in a round robin fashion. So after a particular frame is over, that means, all the input lines have had their time slots then one particular channel will get back its second time slot, and so on. So it will go on in a round robin fashion. We use the same principle in time division switches; in time division switches the bits or bit streams may follow the same path, that means, different calls are following the same path but they are using different time slots. So a fixed

(Refer Slide Time: 45:55)



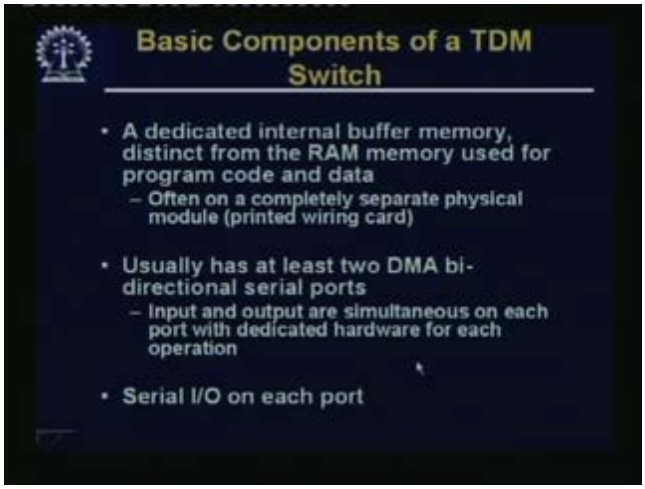
The number of bits from each line is read into a RAM in order – this constitutes a single frame. That means a fixed number of bits from each frame, each line. A mapping table reads from the RAM in a different order for switching from line j to line k, the j th location is read in the k th step.

(Refer Slide Time: 46:18)



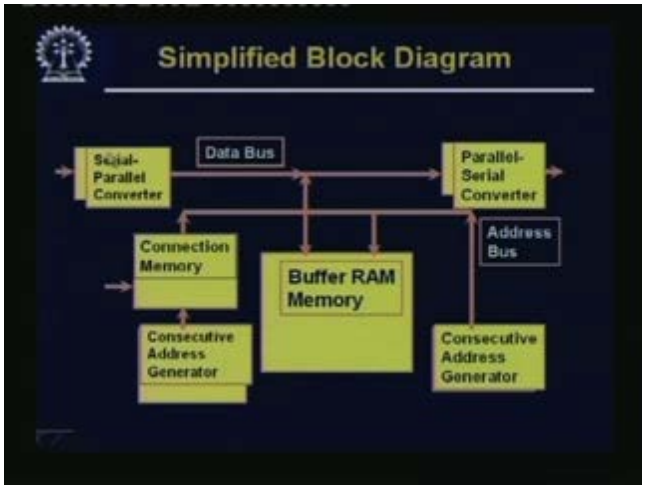
This is shown here. Suppose these are the input lines 0 1 2 3 4 5 6 7. The output is read in a somewhat different fashion 4 7 6 3 0 5 2 1. That means actually whatever the 0, its line 0, puts in this slot would be read by line 4, because line 4 is reading in that time slot. So line 0 is, in effect, connected to line 4. Similarly, line 1 is connected to line 7, line 3 is not connected to anybody. It's idle at the moment, so 3 3; similarly a 5 5; so it is idle. Similarly, just as we have 0 4 we 4 0; so this is a bidirectional communication, duplex communication going on. So in the first slot or, let's say, the 0 th slot, the line number 4 reads it and in the 4 th time slot, line number 0 reads it. So line 0 is switched to line 4, 1 to 7, etc. Lines 3 and 5 are idle.

(Refer Slide Time: 47:28)



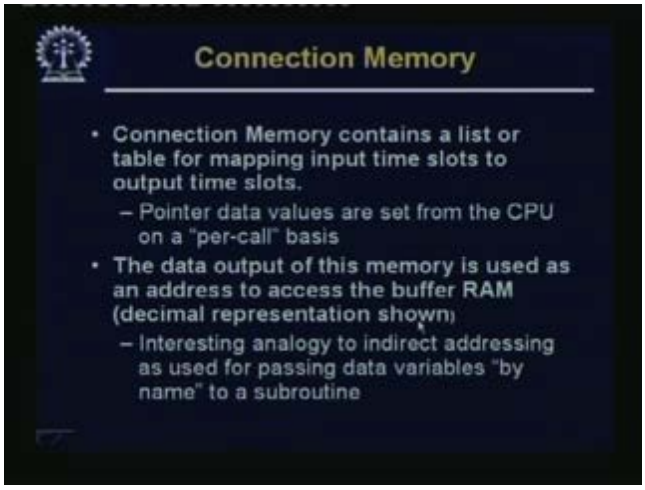
Basic components of a TDM switch –we will not have the time to go into the details of this. It has a dedicated internal buffer memory, distinct from the RAM memory used for program code and data. And this buffer will sort of contain the bits which are to be written into and read often on a completely separate physical module, that means, printed writing card. Usually it has at least two DMA bidirectional serial ports. If you have studied operating system, that is direct memory access, which is used for fast communication without the intervention of CPU, that is DMA. Of course we want this as fast as possible; so DMA bidirectional serial ports, input and output, are simultaneous on each port with dedicated hardware for each operation. So our memories are very special; that means, they have multiple port memories and serial I/O on each port. This is a very simplified block diagram.

(Refer Slide Time: 48:32)



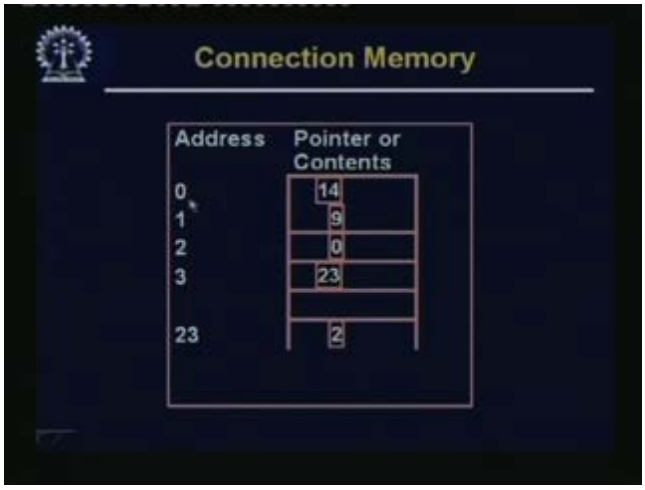
Usually in your line, the bit stream will come serially but on the bus internally we want to put them together, maybe 1 byte at a time. So we have a serial to parallel converter for getting 8 bits and converting it to a byte, which will be fed in parallel. So database goes into the buffer RAM memory – there is a connection memory which is a mapping table that we have talked about. So when you are setting up a connection, you will make entries into the connection memory. We will just have a look at the connection memory, that is, what we have already seen and then it gets consecutive address generated here and then from the buffer memory once again it goes to the output side and then there is a parallel to serial converter for feeding into the individual lines. So this is the data bus and this is the address bus.

(Refer Slide Time: 49:26)



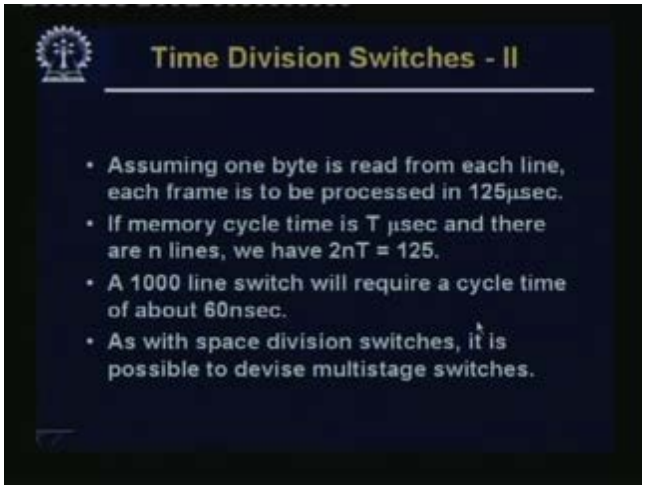
Connection memory contains a list or table for mapping input time slots to output time slots. When a call is set up, you make an increase in this connection memory. The pointer data values are set from the CPU on a per call basis. The data output of this memory is used as an address to access the buffer RAM; decimal representation is shown.

(Refer Slide Time: 49:48)



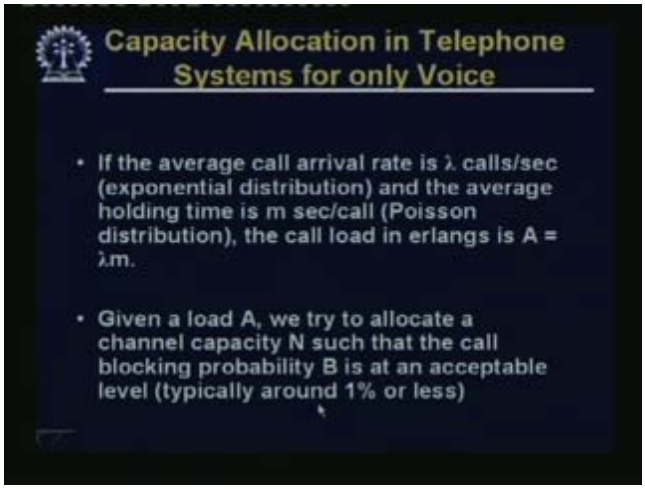
This is another example. So in address 0, we have pointer or content is 14; that means line number 0 may be connected into line number 14 in this example.

(Refer Slide Time: 50:03)



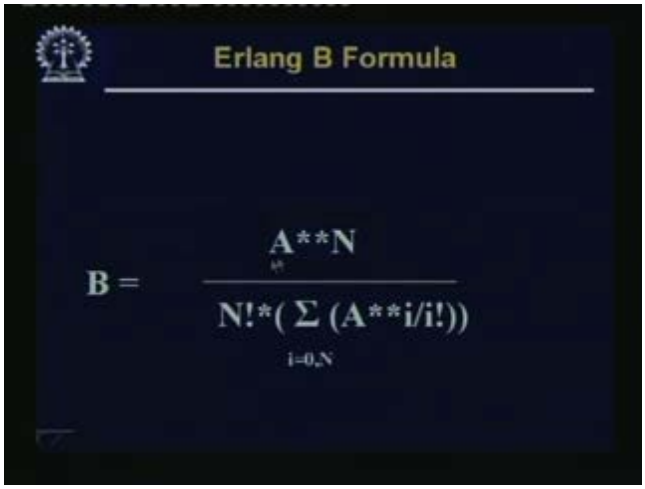
Now just one more thing – that assuming 1 byte is read from each line, each frame is to be processed in 125 micro seconds. You remember how this 125 micro second comes about? Remember we talked about the voice channel, and each voice channel requires a capacity of 64 kbps as we mentioned. So 64 kbps means that what is the gap between every bit in 1 second – there are 64,000 of them. So it would be 64 kbps. If we handle 1 byte at a time, it will be 8 kbps and 1/8 kbps would be 125 micro seconds. So this is the basic time for frame. And in the entire time division hierarchy, as we will see in time division multiplexing, this 125 micro second is a very sacred figure; this is what everybody sort of sticks to so that a voice call can smoothly fit into this. Of course if you want to have a higher speed channel then what you might do is in the same 125 micro seconds' slot, you may pack in a large number of bits. That is alright but we keep the frame size to 125 micro second. If the memory cycle time is T micro second and there are n lines we have 2nT, because each of them has two operations: one read and one write. So 2nT would be 125. So a 1000 line switch will require a cycle time of about 60 nanoseconds. That is the kind of memory speed, and as we will see, as the number of lines goes up or as the speed of switches goes up, this memory speed becomes a constraint as with space to division switches, it is possible to device multistage switches.

(Refer Slide Time: 52:07)



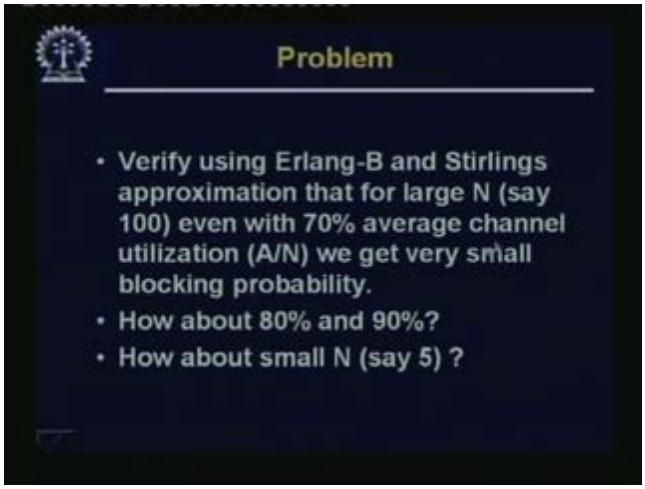
Now just one last point about capacity allocation; I mentioned that the service providers actually want to provision less than the maximum possible. But what would be the blocking probability if you provision it in a certain manner, let us say, in the C stage switch? So I will just give you the formula. If the average call arrival rate is, say, λ calls per second, which is an exponential distribution as people have seen for a voice network – for data network it is different; we will come to that in the next lecture – the average holding time is m seconds per call, which is Poisson distribution. The call load in erlangs is $A = \lambda m$. This is called the load and given a load A, we try to allocate a channel capacity N such that the call blocking probability B is at an acceptable level, typically around 1% or less. This is the typical figure that we try to achieve. The difference is that within the busy hour, we get not more than 1% call blocking probability and the formula for calculating this is this

(Refer Slide Time: 53:16)



B is equal to A ** N / N ! *(∑(A**i/i!)) – so you can use this formula or may be write a small program using this formula to calculate the call blocking probability giving values of A, N, i, etc.

(Refer Slide Time: 53:40)



So you can verify this. This is one last this thing – verify using Erlang-B and Stirlings approximation for handling the factorial, that for large N, say 100, even with 70 % average channel utilization, that is A/N, we get very small blocking probability. But of

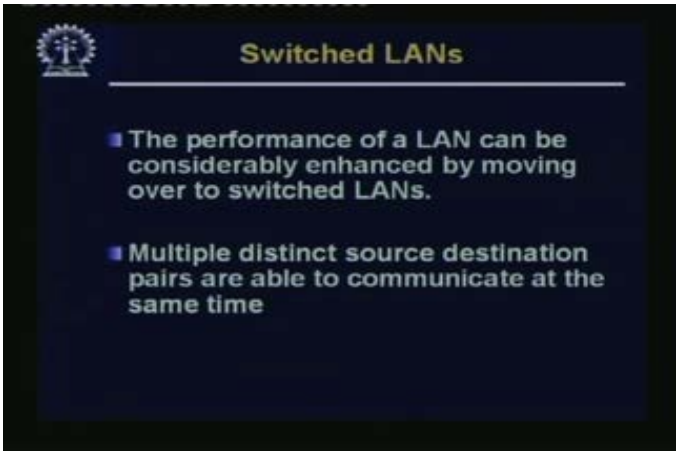
course, if the channel utilization or A/N becomes 80% and then 90%, what will happen is that you will find that your channel probability is going up and then you can do the same thing for a small n and try to see what happens. Thank you.

(Refer Slide Time: 54:22)

good day so in this lecture we will discuss about packet switches that means packet switches are switches which as specifically designed for handling packets we have seen the telecom switches like space division time division switches etc ok we have also discuss something about multistage switches which are used quit often in packet switches

also but in this lecture we will specifically discuss about a packet switches which is at the heart of all most all your computer networks ok.

(Refer Slide Time: 55:11)

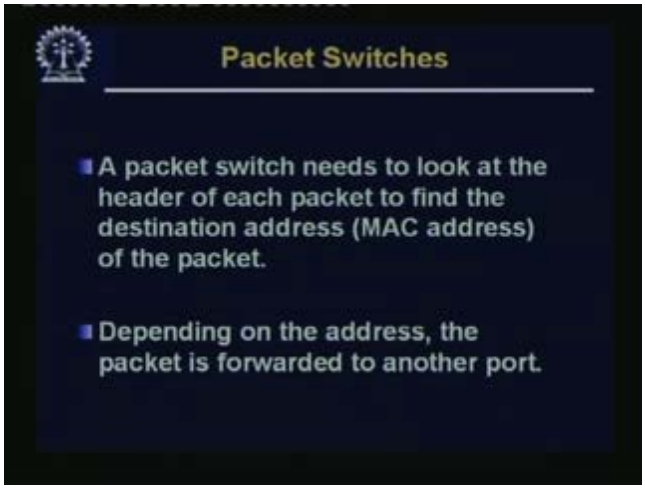


So first just to give you some motivation that why do we require switches the performance of a LAN can be considerably enhanced by moving over to switched LANs moving over from what if you remember that some of the earliest kinds of LANs that we have just seen a brief glance of some of them used shared memory like a hub or a bus etc now in a if you are using a shared medium what is going to happen is that only two nodes connected to this LAN can communicate with each other at a particular point of time where as if you have a switch multiple pairs of users or multiple pairs of nodes can communicate with each other at the same time so naturally you get a very good boost in performance of the overall LAN so that is one reason why we require switches of course we require switches in other place also.

(Refer Slide Time: 56:18)

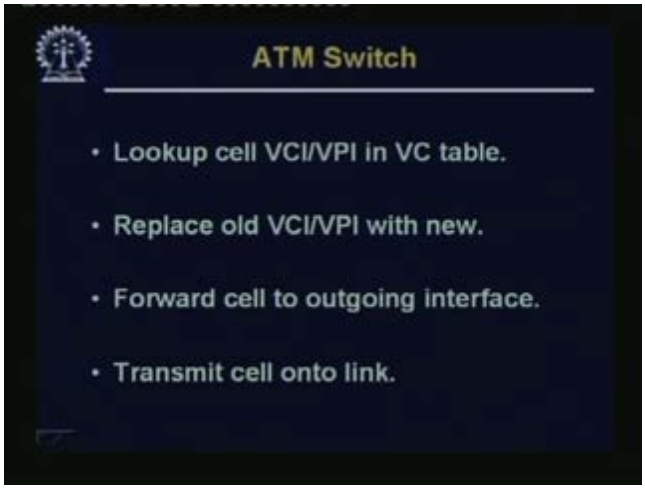
Multiple distinct source destination pairs are able to communicate at the same time thorough a switch so that is why we require a switch in packet networks.

(Refer Slide Time: 56:24)



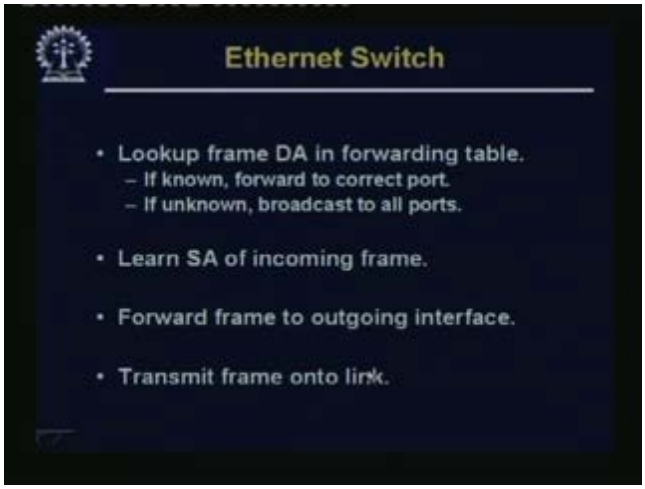
A packet switch needs to look at the header of each packet to find the destination address of the packet so you do not have to do a full scale sort of lookup of a very big table a

(Refer Slide Time: 56:43)



so you have a you lookup only a very small table of V C I and V P I in the V C table replace but of course virtual connection just as their setup they may be taken down so replace old V C I V P I with new ones forward cell to outgoing interface transmit cell onto link so this is the this is how an a t m switch works so if you so you see that the basic concept of the cell remains the same that means we have we have some we have to find out the output port from the address and the here the address once it is setup after that the output the address is simply a small table of V C I V P I which makes this lookup process much faster so this together with the fact that the cells are of same size so you can develop your hardware in such a way so that this a t m switches will work very fast otherwise the rest of the switch and the switching principle remains the same as we have discussed already

(Refer Slide Time: 57:53)



next comes comes Ethernet switch which is perhaps the common most common type of switch that you see almost everywhere as a matter of fact this number of Ethernet switches which are being used and which hare being put to use the almost regularly is so high that the cost of this switches have also come down you know because if if you produce something in very large number you can amountize the cost of is development etc over a large number of switches so this cost of the switch comes down so Ethernet switches have become very cheap these days ok switch which are used to cost may be a lack or so may be ten fifteen years back cause only a few thousand rupees today so this is the most common type of switch which we see everywhere.