Digital Switching

Prof. Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture – 23

We will continue from where we left last time. The way, actually, my whole plan is I will finish up with this particular part. Then, I will go to what we call buffered delta. So, the problem is that there are no close form solutions for that, but if you want to, for example, even simulate that; this is a complicated thing. If you will try to understand that thing now, you can build up the complete simulation by actually, understanding this state transition diagram of a 2 by 2 switch, which is going to contain one buffer per input. Because so far, in our switches either, the buffer only at their input or at the output, but remember, in this banyan network category, if there is one path being step up, it actually, blocks many other paths, which were sharing the same link. Those paths cannot be; all permutations are not possible. The only way you can get higher throughput is; because it is not strictly, a non blocking switch; it is a blocking switch, technically.

So, we need to put buffers. Buffers at input and output; we do understand, because it is a blocking switch. If it cannot go through, this stops at the output. Now, this is different than the speedup factor, and speedup is what at one single output, if you want to pass through something, then only, you require a speedup, more than one, but if the permutation is such that, that there is no conflict; crossbar will always permit this, but this particular switch will not permit that, actually. There is a map, which is possible when, there is no conflict for an output port, but still, you cannot step the connection; because that permutation is not permitted. So, people thought of putting adding into the buffer. So, we will look into that particular thing.

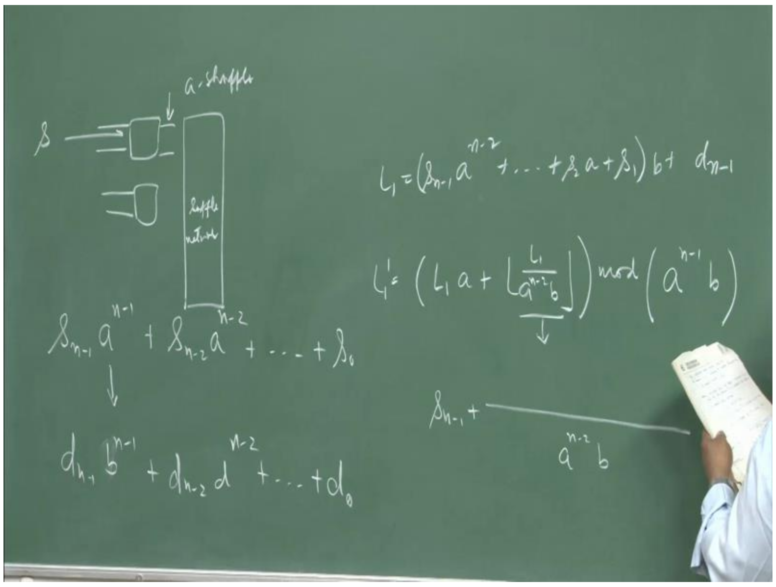
That is actually, from a paper; again, I will tell the title of the paper when I will come to that. Then, I will appropriately, move to voice over IP basically, SIP based system; how does it operate? So, that is the current day telephony structure. If I am able to finish SIP before the end of semester, then we will come back to again, switch architectures, and I will probably, do something more on these, which are basically, knockout switch and other kind of switches, which are being proposed in the literature. None of them are actually, implemented as such, in hardware, as of now; except, this particular one; the buffered delta is technically, implemented in almost, all packet switching systems. Even, if it is an IP switch, what we do is we do not actually; IP switch; remember, IP packets are variable length packets, and whatever we have been doing is only, for fixed length frames. So, people used to call them ATM switches, but technically, in the core, we always build these fixed length packet switches, and if you want to implement a variable length packet switch through this, at interface board, you actually, splice the packets into multiple fragments; put some identifier for them for reassembly; then, put the tag for each one of those fixed length fragments. Those fragments are routed to the output board. Only condition is now, you require an output interface board also, in the switch. You are encapsulating the fixed length packet switches and outside, the input and output interface boards; these will again, reassemble back the IP packet into its original size, and send it forward.

Student: (( )).

Usually, I am looking at switching efficiency; within the given amount of hardware, how much I can pass through the output of the switch? We call it throughput; number of packets, going out of the switch per unit time, per unit slot.

Ideally, it should be there; when n output ports, it should be n packets per output slot; I should be able to push through, but that usually, cannot happen, because this statistical nature of the traffic. With buffering, something you try to improve, but it is getting almost, one is impossible. If you put a load of 1, means there is always a packet, available at the input. In every time slot, you will not get one packet per output slot coming out. Input output queuing; we have seen that particular case. For input queuing, this does not happen; we actually, limit to a certain value; you cannot exceed that. Well, that is again, statistical, because what will happen, once in a while, you get very large throughput and other times, you get very low, but average value will be maintained at that. We always compute these statistical averages.

(Refer Slide Time: 04:56)

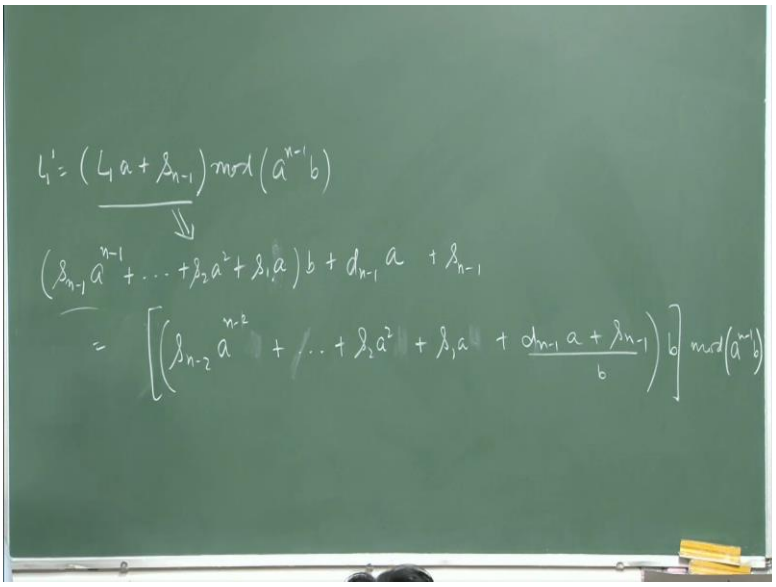

a-shuffle
s
shuffle network
$L_1 = (s_{n-1}a^{n-2} + \dots + s_2 a + s_1)b + d_{n-1}$
$L_1' = \left(L_1 a + \left\lfloor \frac{L_1}{a^{n-2}b} \right\rfloor\right) \bmod (a^{n-1}b)$
$s_{n-1}a^{n-1} + s_{n-2}a^{n-2} + \dots + s_0$
$d_{n-1}b^{n-1} + d_{n-2}d^{n-2} + \dots + d_0$
$s_{n-1} + \frac{\quad}{a^{n-2}b}$

Now, let us go back to where, we left. I think what I have done was I have taken this case of the switches. There was some switch, which was having some input address, and which was represented by; I think this is what was there. So, that was an input address, and this was to go to an output address; this was b. That was our objective. After the first switch, we know that this particular digit will be used in the first switch for routing purpose. So, before you do the shuffling, what will actually happen is I will find out the switch number from this switch. Finding out the switch number is you divide by a, and take the floor function. That will give the number; number is 0, 1, 2, 3, and so on. So, multiplied by total size, here; whatever is your current switch number, because it is actually, less 1; number itself, if you count. So, if this will come out to be 0, there is nobody. It was 0 into b plus, whatever, is d n minus 1; that is what we did.

That was a term known as L 1. Because I am dividing by a for floor function; this is what we got. This is nothing but the floor function part, plus d n minus 1, and this have to be multiplied by b; that was L 1. This was just before at this point. Now, you have shuffle; network shuffle interconnection, basically, and this is a a-shuffle. So, one sided a-shuffling; I used the formula that what will be L 1 prime. So, I simply used L 1 into a plus L 1; this has to be, if you remember; for a q shuffling, it has to be r; q into r is total number of elements. So, it has to be whatever, is total number of elements here, divided by a, because I am using a-shuffle. So, this a raise power n minus 1 into b; this will be a raise power n minus 2 into b modulo operation.

So, this is what was there. In fact, what we did is we actually, took this particular thing out. We wrote the whole expression. It turns out to be something like s n minus 1 plus, then everything divided by a raise to power n minus 2 b. What we did is in the numerator, we put the all possible maximum values, and we figured out after doing all summations, that values are less than this. So, this is always going to be a fraction. When I am taking the floor function, if this is a fraction, and this is the integer; this is what I will get, actually. So, I can replace this whole thing by s n minus 1. So, you will get, I think that is the point where, we finished last time when I closed.

(Refer Slide Time: 09:01)



$$L_1' = (\underline{L_1 a + s_{n-1}}) \bmod (a^{n-1} b)$$

$$\Downarrow$$

$$(s_{n-1} a^{n-1} + \dots + s_2 a^2 + s_1 a) b + d_{n-1} a + s_{n-1}$$

$$= \left[\left(s_{n-2} a^{n-2} + \dots + s_2 a^2 + s_1 a + \frac{d_{n-1} a + s_{n-1}}{b} \right) b \right] \bmod (a^{n-1} b)$$

So, you will end up in L 1 prime, which is given by L 1 into a plus; this is the only thing, which will be left here. I think that is the point where, I ended. So, you have now, solve this modulo thing. Let us find out what is L 1 a plus, s 1. We expand this. As an expression, we try to understand; if I do a modulo operation whatever, is a component, which is integer multiple of this factor, will be removed and whatever, leftover; that will be the outcome of this modulo operation, and that whatever, is leftover, has to be less than this value. So, we will do that. So, L 1 a plus, s 1; this will be; now, what is your L 1? L 1, you have to understand, is this. So, I have to multiply this by a. This will become s n minus 1 a raise to power n minus 1, and I have multiplied by a, already. Now, s 2 a square plus, s 1; there is already b in front of it.

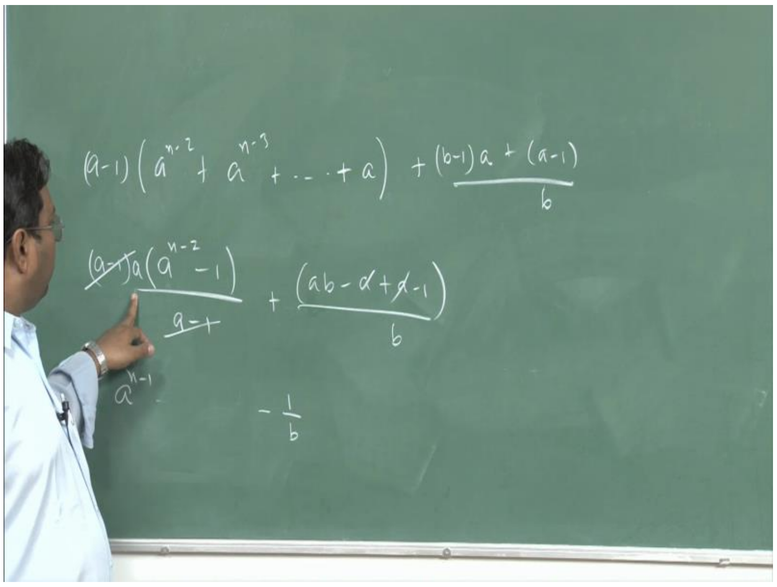
Student: Sir, s1a (( ))?

s 1 a, right.

This is also fine, and ultimately, this s n minus 1 will be there; that is the term, which will get for this. You have to essentially, handle this,, and then by using a similarity principle and following the same thing, you come to the proof. Remember, there is one term, which will go out, and this is another term, which will come in; that is what going to happen. At the input of this second stage, I want to find out a switch is here. What happens here? That is what; I am interested in, actually. So, every stage, what happens? This is an one stage transition, or you call it; at this stage, what happens; and after the next interconnect, what is the number here; keep on doing it, but ultimately I am interested in; there are totally, how many? This is L 1. I am interested in their total n stages; I am interested in L n, actually.

Now, solving this; expand. You just observe this. This will be n minus 2 b. Now, this particular factor is divisible by a raise to power n minus 1 b. When I will do a modulo, when I divide by some these things, then I will get an integer. So, when I do a modulo operation, this has to be removed. So, I can forget this, and live with the remaining stuff. I have done the modulo. I have to only now, check, if this is greater than a raise to power n minus 1 b. So, whatever is this, can be probably, written into some integer multiplied by a raise to power n minus 1 b plus some k. So, I just want to find out that k, actually. Doing that, let us find out what is the maximum value of this particular thing. In fact, still the better way, I can do is I can take this b out, because anyway, this does not matter. Now, the modulo of this is what will be the answer to me. So, I am now looking, focusing, going to focus only, in this thing, which is there in the inside brackets. So, what is the maximum value, which I can achieve?

(Refer Slide Time: 14:38)


$(a-1)\left(a^{n-2}+a^{n-3}+\dots+a\right)+\frac{(b-1)a+(a-1)}{b}$
$\frac{\cancel{(a-1)}a\left(a^{n-2}-1\right)}{\cancel{a-1}}+\frac{\left(ab-\cancel{a}+\cancel{a}-1\right)}{b}$
a^{n-1}
$-\frac{1}{b}$

Put all the maximum values of s n d, and whatever it is. So, we will end up in; let me do it here, inside. So, maximum value of s n minus 1, s n minus 2, and so on is a n minus 1, actually. I can put all that value. So, the maximum value of this thing will be this; this is what will be the maximum. So, I can solve it. Totally, how many terms; n minus 2 terms, obviously, and first term is a. This, a will cancel with this, and because I have taken b out of it. So, if I multiplied by b, this will be, this whole thing now, technically, the maximum value will be a raise to power n minus 1 b, minus 1. Conclusion from this actually, what you get is this whole thing can never be greater than a raise to power n minus 1 b. If I take a modulo operation, I will get the same thing, which is here in the argument. So, the modulo operation ultimately, will give me nothing but .

Student: (( ))?

That is the maximum value of d n minus 1. This goes from 0 to b minus 1; s n minus 1 also, goes from 0 to a minus 1. So, I put the maximum value. I am trying to find out what maximum can be achieved by this argument.

Student: Sir, middle term is also there a also there sir in between these two a will also be there coming from that movements right sir right a b by b sir right sir next term sir.

This one?

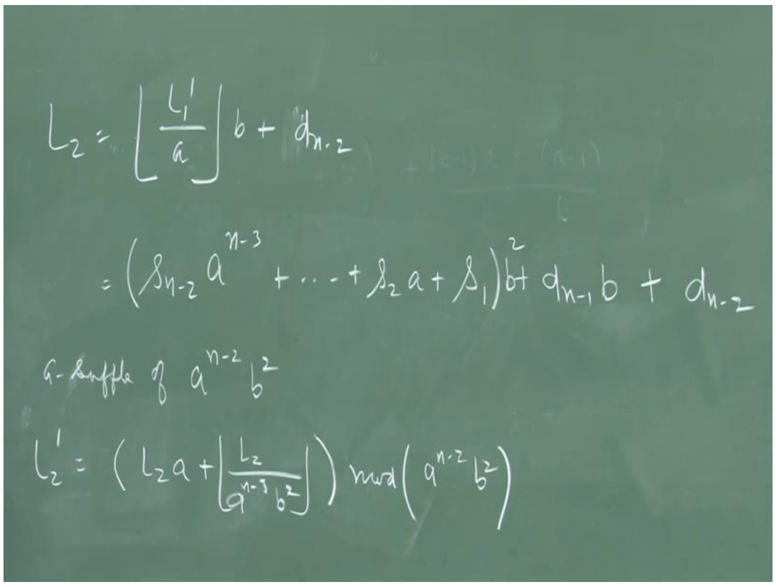
Student: Next term, next, yes, this one.

B cancels, so there is a into this a raise to power n minus 1, minus a; a b, b cancels. So, it is a plus a minus 1 over b. So, this actually, cancels. So, final result will be of this modulo operation, obviously, it is nothing but whatever I am writing here; that is what it will be; that will be L 1 prime.

Student: First term, sir?

First term will not; this would not exist; it is n minus 2. So, I can keep on doing this and I can find out L 2 and so on.

(Refer Slide Time: 19:08)



$$L_2 = \left\lfloor \frac{L_1'}{a} \right\rfloor b + d_{n-2}$$

$$= \left(s_{n-2} a^{n-3} + \dots + s_2 a + s_1\right) b^2 + d_{n-1} b + d_{n-2}$$

a-shuffle of $a^{n-2} b^2$

$$L_2' = \left(L_2 a + \left\lfloor \frac{L_2}{a^{n-3} b^2} \right\rfloor \right) \bmod \left(a^{n-2} b^2 \right)$$

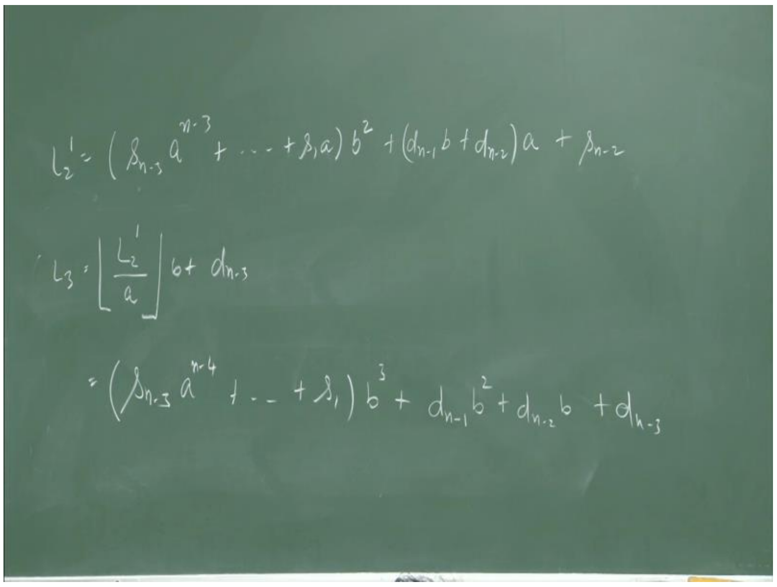
But only now, I think I have got this structure. Now, what is L 2. L 2 will be nothing but L 1 prime divided by a into b plus d n minus 2; that is operation at the second stage. I will find out which switch, the input is going to, by doing this floor function, dividing by a, doing the floor function, or in the output side, how many was already covered by all earlier switches; in my current switch, I am going to go to this particular port number; that is the second stage. So, I can write down that whole expression here, and this will turn out to be s n minus 2. I have divided by a. So, again, the powers will reduce. That will be the floor function plus, sorry, there will be b also, which will be L 1 prime; you divided this whole thing by a, actually; this whole terms can be divided by a. This term can be divided by a. This is what will be the fraction; s n minus 1 divided by a will be the fraction. When you will take the floor function, this will go away. So, you divide everything by a, and multiplied by b and that is actually has to come.

So, I can just multiply this thing by b. This can be multiplied by b. Of course, this is d n minus 2, will come. So, remember, that I am actually, removing one term is being removed here, and another term getting added here. That is what is happening now, at every stage. Ultimately, all these terms will die away with time in every stage, and only these d n minus 1 terms will come. Whatever is the output, will tell what is the port number to which, it is mapping. If that port number is what is being governed by d n minus 1 raise to power b n minus 1, plus d n minus 2 raise to power b into n minus 2 and so on, till plus d 0. So, I am actually, going to an output port address. So, it gets proved actually, in that process.

Student: This will b square; b square first term and bracket and multiplied by b square.

It is b square; agreed. This should be b square, because this b has to be multiplied by that. From here, you will get a-shuffle. Now, number of output switch will be shuffled, are these. Earlier, it was a raise to power n minus 1 into b outputs. Now, these many outputs will be now, be a-shuffled. So, this is a second shuffling interconnection. This will give me nothing but L 2 prime. L 2 prime is L 2 a plus L 2. Now, what will be the numerator? This divided by a; q is divided a; r, you have to find out. So, a raise to power, and this has to be done a modulo operation over this.

(Refer Slide Time: 23:02)



$$L_2' = (\delta_{n-3} a^{n-3} + \dots + \delta_1 a) b^2 + (d_{n-1} b + d_{n-2}) a + \delta_{n-2}$$

$$L_3 = \left\lfloor \frac{L_2'}{a} \right\rfloor b + d_{n-3}$$

$$= (\delta_{n-3} a^{n-4} + \dots + \delta_1) b^3 + d_{n-1} b^2 + d_{n-2} b + d_{n-3}$$

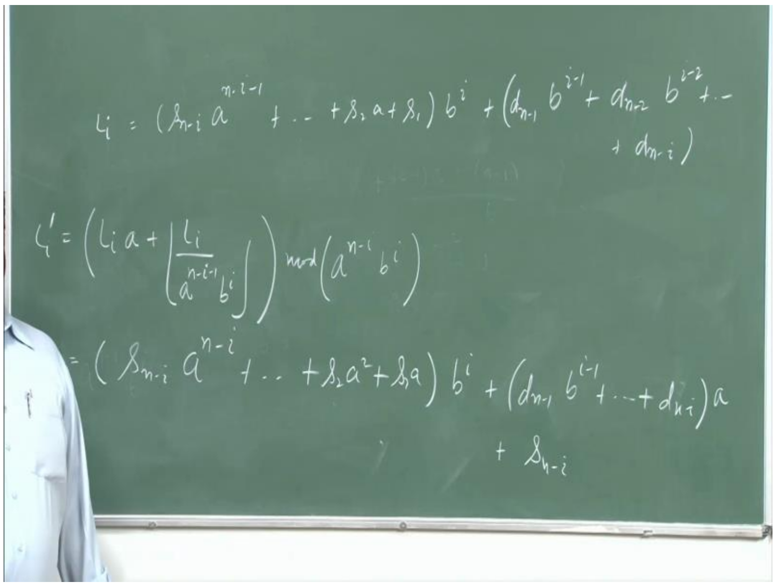
So, you will get L 2 prime as once you solve for this. You have to just remember the symmetry now; the structure of the earlier expression; this actually, comes from there.

So, L 2 prime is; one term has actually, again, gone from here; n minus 2 s n minus 2 has gone now; it is no more there; plus, you multiply that whole term by a. So, it is a modulo operation, has to be taken. So, we now know, which term will go away, during modulo operations or during floor functions. Yes, L 2 will give this; d n minus 1 b plus d n minus 2 b into a; this will come from your floor function. Earlier, it was s n minus 1. This time, it has to be s n minus 2. You take the floor function of that; you divide that whole thing by a raise to power n minus 3 into b square. So, this term will be completely, deleted. Rest, everything you take care, and you will find out that does not; that is the fraction; that is less than 1. So, that can be removed. So, s n minus 2 will be the only one, which is left out on that side from here.

You can now, just use those clues, or if you want, you can actually, put the actual values, compute the series and figure out. Yes, it will be always, less than 1,, and then of course, from here, once you know this L 2 prime, you can find out what is the L 3. L 2 prime divided by a into b plus d n minus 3; you divide this whole thing again, by a, which can be done. All these will reduce by that value. This can be divided by a; this will be lost, because when I am taking this floor function; n plus d n minus 3. So, what you will get? You can actually, directly, get that thing; s n minus 3. You will multiply by b, so it will be b cube. This thing will be divided by a.

So, this is being, wholly divided by a; that does not matter. Last one will not remain there,, and then I multiplied by b. So, I will get this value, plus d n minus 3; it is very similar to that expression, you can see. Every time, I am adding now, and forming a series at every step. So, I think, so far, we do understand how it is happening. So, by intuition, you know that is going to work, but formally, if you know, i, you have to prove for i plus 1, and henceforth, you will say, because it was working for 0; it was working for 1; it was working for 2; and if it is works for i, it works for i plus 1. So, it works for everything for all integers. So, what is it for n, and get the estimate; that is what has to be done.

(Refer Slide Time: 27:20)



$$L_i = (s_{n-i}\, a^{n-i-1} + \dots + s_2 a + s_1)\, b^i + (d_{n-1}\, b^{i-1} + d_{n-2}\, b^{i-2} + \dots + d_{n-i})$$

$$L_i' = \left(L_i a + \left\lfloor \frac{L_i}{a^{n-i-1} b^i} \right\rfloor\right) \bmod (a^{n-i} b^i)$$

$$= (s_{n-i}\, a^{n-i} + \dots + s_2 a^2 + s_1 a)\, b^i + (d_{n-1}\, b^{i-1} + \dots + d_{n-i})\, a + s_{n-i}$$

So, at L i, is the line number just before the stage i. So, that L i is (( )).

Student: How to state it, because L 1, we are doing after first stage, and before a first this.

Right, after stage; agreed. It has to be after stage i. Look at the similarity of whatever was the earlier expression; from there it comes. This is what will be there, after stage i. From there, you can actually, find out what will happen, after stage shuffle interconnection,, and then what happens after stage i plus 1? Again, following the same principle; first of all, you have to do the shuffling. Shuffling again, says that it has to be L i, whatever shuffling you are doing, plus L i divided by whatever are the number of lines in that, which are being shuffled; this will be a raise to power n minus i plus b raise to power i. I think that is what it will be.

Student: Minus 1, sir a raise minus.

i minus 1, right. I hope it is right, yes. No.

Student: a raise to power of n minus i minus 1, sir, a raise to power.

b raise to power i; total sum of these two, is always n.

Student: Yes, sir, this is the totaled objects and this we divided by a. So, this will be a raise to power of n minus.

That is for r actually, when I am taking in this base.

Student: This will be n minus 1, i minus 1 b, yes, sir.

This is, I think what it should be.

So, for i plus 1, I can now find out what is L i prime and from there, I can get L i plus 1, because that is a step, which is required for induction. So, this will give me nothing but L i prime. I can just multiply this thing, and I will get. It is b raise to power i plus, I have to find out this operation. So, you divided this whole thing, same L i by a raise to power n minus i minus 1 by b i. Only this thing will be the whole thing, which will be available; rest, everything will become a fraction. So, what you will get is s of n minus i. Now, when you take a modulo operation, only this, a raise to power n minus i into b i is complete; rest, everything will still remain a fraction. So, what I will get is L 1, L i plus 1. First of all, L i prime only, I have to solve this.

(Refer Slide Time: 31:56)

Because L i prime will come after modulo operation. So, I have to do modulo on this. Once I do a modulo, s n minus i 1 will actually, go away; rest, everything I have to write. So, this one will be, yes, right. These things also, come here; it is intact. That is what it will be L i 1 prime. So, L i plus 1; you can again, use same formula, you divided by a; take the floor function, plus whatever is the next digit, which is d raise to power n minus i minus 1, now. So, n minus whole bracket of i minus 1; that is what it will be. So, when you divide this whole thing by a, what goes out? Yes, this thing is wholly divisible. This thing is wholly divisible; s n minus i, which what comes out; s n minus i divided by a will be the fraction. So, that has to be removed; rest, everything will be there. So, it will be s n minus i minus 1, because everything has to be multiplied by b to get L i plus 1, and of course, the last one. So, I think this fits; if it is for i, it is for i plus 1; it is proven.

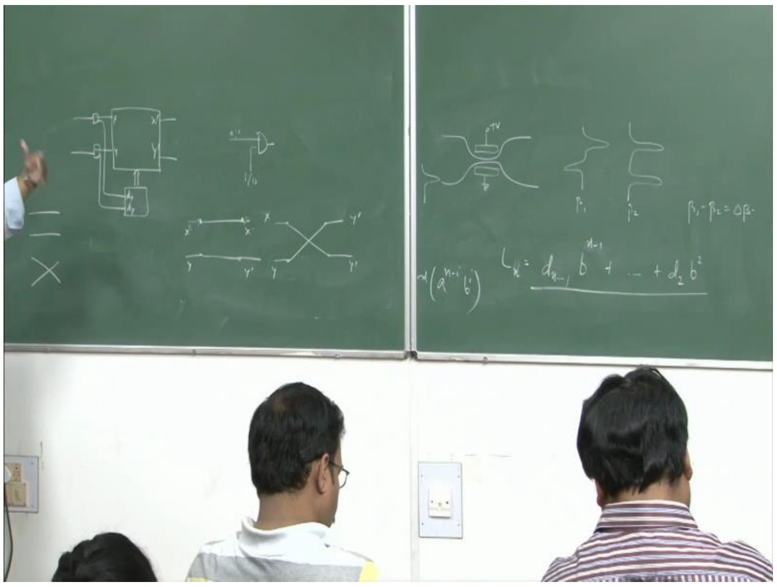
Now, I can go and estimate for i plus L n, after n th stage, sorry, I am using, I think, small n, everywhere. So, what will be L n? I can just use this expression. So, when i reaches to n, all these terms will actually, go away. This is for i plus 1, remember; n minus i plus 1. So, when I am looking at n, s minus n minus n, that will be s 0 a raise to power something, at that is for how, it should actually, come. In fact, that term will not be there.

Student: Excuse me, sir. b in b last term?

No, that cannot be b. This is actually, from i th stage; after that, there is a interconnect and after that, I am doing a switching operation, which is a digit, which is used there, for routing purpose. That is that one, n minus i. In fact, you should always, write this thing this way for clarity. This priority digit is used for routing there, actually. So, all these terms will be actually, almost gone. By the time you come to end, and you will end up in getting d n minus 1, b raise to power n minus 1, because it is i plus first stage; it was b raise to power i. So, nth stage; it will be n minus 1 and so on, and this is nothing but the output port address.

So, shuffle net interconnection actually, is a delta routing network; delta interconnected network. Of course, if there is any confusion, you always try to find out what will happen for n minus 1 stage. Find out what is going to be L n minus; not L n, but L n minus 1. From here, you should be able to get. There will be only, one term left, which will be done away within next step. You will end up always, in the routing at the destination output port address. So, that was the example; that shuffle net s; it has worked like a delta router network, but so far, you follow the principle, you can always create anything. Interconnection is your choice. How this kind of switch will be implemented?

(Refer Slide Time: 37:55)



If it is a 2 by 2 switch, it is very simple. You look at the inputs, which are coming in. There are only two inputs, and they will have a bit, which is the deciding bit, actually. There has to be always, a controller, sitting in. In fact, I can make a parallel interconnector also; keep on sending the control bits; one by one, separately, as a parallel line, or it can be attached as a tag in front of the packet. If it is attached as something in front of packet, you require interface processor. It receives the whole packet, gets the tags out. These tags are being sent to the controller; so d x d y. Interface board has been designed to take specific bit out of the tag, depending on which stage, you are putting in. This can only keep an action of whether; it is in cross or bar state. If it is in cross state, the mapping is this; cross is this, and bar is this. So, that control has to be exerted. That is it; nothing else has to be done. So, if there is a conflict; both of them want to go to the upper port, or both of them want to go to the lower port; you just randomly, pick one, and forget about the other. This is inefficient in that sense, actually.

Student: Sir, how it is tag delta cross once again, sir, delta cross parallel?

If this is x and x prime connection, that is the bar. A cross actually, means your this input is now connected to this; this input is connected to this; that is the cross state. You can actually, in electronics, it is pretty simple. You can just build up an AND gate and put it there; that is acting as a switch, but it is a non relational switch, actually. Because this is an AND gate switching; how fast AND gate can change state from 1 to 0; it basically, will decide. Because what is happening is logically, in AND gate, only one input you are using as control; other one is this, but this is actually, changing every time. It goes from 0 to 1. This transition bandwidth is decided by this gate itself; gate design. It is not a relational switch. We call it a non relational switch; there is no relation. These are all governed by the hardware; how fast you can? What is a bit rate, which can be put in? There is another kind, which is, one example is optical coupler.

If you use, for example, an optical coupler, typically, these are built over electro-optic materials. If you do not apply electric field, this may be in bar state. When you apply electric field, you change delta beta, where, beta 1 and beta 2 are basically, difference of propagation constant of two modes. Two Eigen modes in this coupled system, is actually, like this; one is this, and one is this. So, one is with beta 1; other one is beta 2. The difference between them is known as delta beta. That is why this is known as delta beta coupler, actually. This delta beta can be changed by applying electric field. Usually, when you put a profile here, this will excite; this is not the mode of the system; there are only two modes.

So, this has to be now, represented by uniquely, by a linear combination of these two modes. If these are both of equal strength and you add them, you will just make a 180 degree phase shift for this one. So, these and these, will become in-phase component, and correctly, what is there; out of this, this is going to there. If you make it 180 degree phase shift, these two will be canceling each other, and these two will be adding. So, these two modes will be excited with equal strength, actually. As you propagate, and if you ensure this length is sufficient enough, so that, you will have a pi phase shift. What do you find out? These two will come in phase, and these two will go out of phase at the output.

So, whatever power you are inserting here, will come out at this port; that is the cross state. If you change your voltage, your delta beta into length; it becomes 2 pi or integral multiple of 2 pi; power will remain in the same fiber, but one of the important things; when I am going to cross to bar state, that will take a few milliseconds, but this time taken to go from cross to bar state, does not decide what will be the bit rate, which can be injected to the system, because this is not changing the state of system. So, this is transparent to input to output. This is a relational system. In case of a gate, this is not a relational system. Input and output are not at all, related bit rates. The system actually, tells what is the maximum bit rate, which can be there. So, that is also, another kind of configuration, which can be there, but scheme usually, will be very simple to this. If you have a kind of, naught a two; because in this case, logic is very simple. If both are same, choose one of them, randomly; other one, you simply drop, but where you will drop; what that drop means?

If this is a gate based implementation, only one path, because what will happen is you will have, going from here, you will have a gate; going from here also, you will have a gate, with OR option here. Similarly, from here, there is going to be one gate, something like this. If you put a naught, then there is a problem, actually, because both of them cannot be switched off, simultaneously. Sometimes, both of them might have to be switched off, if there is a contention, because you are dropping that packet. So, that packet will be lost. So, it has to be received, analyzed, and then retransmitted again, by this interface board. You have to repeat it, multiple times in the switch. It is not that; it is not like a cross bar where, only interface board does the analysis. Then, internal, just sets the cross points; thing passes through now. You require basically, all clock recovery, putting it to memory, analyzing, everything at this point, and every 2 by 2 cross point requires this.

So, amount of hardware required is higher, actually, compared to a simple cross bar; that is a penalty, which you pay for text switching. So, every 2 by 2 cross point require a processor; however, small it is, but it requires a processor. Cross bar; one processor is good enough. Every interface board; there has to be a processor. So, technically here, number of interface boards, which you require are; one is the main interface board in the beginning where, the texts are inserted. If it is an IP packet variable length switching, it also does a splicing into constant fixed length frames; output side, it will be spliced back, joint together, the splited stuff to make again, variable length IP packet. These are two special kinds of interface boards, and every stage will require interface boards at the input side; that is a complication here. This is, I think, is a problem, but with the good, now, in VLSI, it is technically, possible to implement all these details. So, this has actually, taken off as such. Now, can we improve this further?

Student: Sir, we are implementing 2 by 2 by gates, using VLSI, sir; that is now, this (( ))?

I have the whole matrix itself, is built in VLSI now.

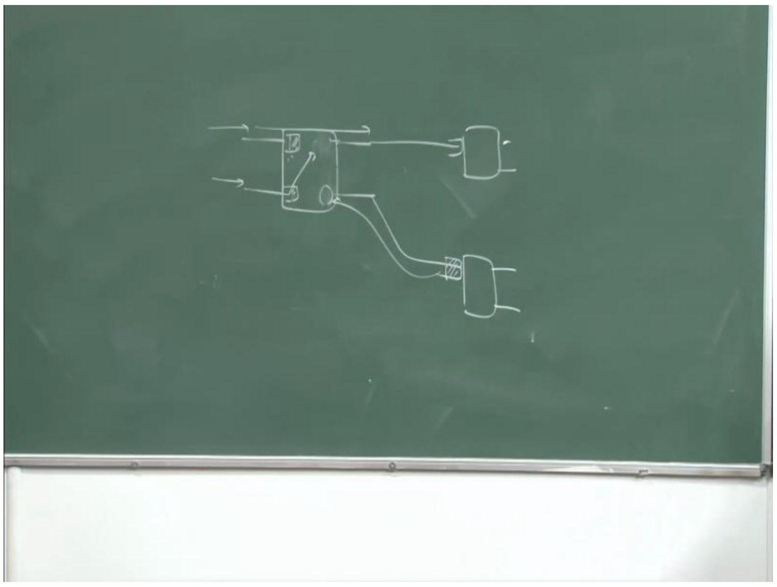
Student: But that will not have a processor in built, if we built in VLSI.

No, you can build a processor also; just repeat the same things, same design multiple times, and connect them with the verses.

Student: Sir, now, we have done away with cross bar. How that is a implemented in (( ))?

Packet switches are all implemented this way; no cross bars. Crossbars actually, are no more used. In fact, for even speech signaling, we do not require cross bars, anymore. Because speech is now, communicated over a packet switching system. So, underline switch is still, remains. This is transparent to your voice transport system. Earlier, voice transport system was knowing that there is a switch, beneath. You are setting a path. Now, for you, setting a path, actually, means a bus; simply putting, you are forming a packet, and putting something in the header. That is the switching functionality, as far as the voice transport is concerned. So, hardware takes care of that.

(Refer Slide Time: 47:50)



Now, how to improve the functionality? I have to introduce buffering. Buffering means if this is there, I am going to have one buffer. Now, if there are two packets coming, two packets coming to the system, then what will happen? There is a contention; one of them can go out, directly; other one can be buffered, actually. Now here, slightly a different thing, because I am now, introducing for the first time; the feedback system. There is a feedback loop, which is there. So, there is nothing actually, gets lost. You only transfer the packet to next stage, if you have got a buffer available; otherwise, you do not do this, actually; as simple as that.

Do we put an output buffer or input buffer; the question is this, if I want to build up a buffer. I am going to still build the same delta network, but now, every intermediate switch will have a buffer. Now, if I make this as an output buffer; I am connecting to some other switch. Now, you look at the situation, when the buffer is not here, but buffer is here. They are on the same chip, or same substrate, or same board. So, distances are very small; I can build up a backward connection, also. Whether you put the buffers here, or whether you put the buffers here; actually, are same; it does not matter. Now, question is how to analyze the performance. Now, there is one way, is to actually, build up a simulator with this, but doing simulation, you have to now build up a model of this system of every switch; how this is going to operate.

The model, which we are going to look is basically, based on a mechanism called feedback. So, if this buffer is there, if this buffer is free, I can send the information back, and this can maintain a state whether, buffer is available here, or not here. Based on that, whatever is occupied in this buffer, can move or cannot move. For example, there can be input; the packet for the same output. If this buffer is going to be free, then only, this flag say that the buffer is free, and both of them will be told; one of them will be picked; one will be pushed into this buffer; other one is still remain.

Now, packet from the previous stage can only, come when this buffer will get free. It is like; packets are hopping from one buffer to another buffer. If the packet in front of you actually, moves ahead, then only, you get the chance. Otherwise, you just stuck remain there. Under this situation, under given load, what is going to be throughput performance; that is basically, the issue. At some point of time, you will find out you cannot achieve higher throughput; throughput is limited; cannot be unity here, obviously; because there are contentions.

So, what I will do is actually, I am now ending here, because I think, I will cover up this thing. I am not taking two hour class today, honestly speaking. I am only taking one hour. I will take up another Saturday, some time, and then actually, take one hour extra there. The paper which, you have to follow for this one; I am just giving, so that, you actually, can go back and read it before you come to the class, because this is slightly, tricky part; especially, it says transition and how to change the matrixes, and how to compute. Simulator is not given code here, but once you understand this, then you can write a simulator.

So, doing simulation is not that trivial exercise. You have to build up a complete mathematical model. So, this paper actually, large part of it is actually, on how to build up a model of a switch. So, that can be simulated, and you will be using a patrinet model for this, actually. So, patrinets is what I taught in the previous semester; same thing will be used here, again.

(Refer Slide Time: 52:29)

So, this one is dias and jump. These are IEEE transactions on computers, actually. I write this one, because it actually, shows, tells for the first time, the complete details, before you start doing the simulation part, and theoretically, also an analysis can be built for this switch; that is one good thing.

Student: Title?

Title is analysis and simulation of buffered delta networks, and simulation of buffered delta networks.

Student: Sir, basically, we are doing input queuing with the output with the (( )).

But now in every node.

Student: And with the flag at the output, which corresponds to the input of the (( )).

So, you can move ahead, only if the queue in front of you is empty; otherwise, you remain in blocked state. So, input also, cannot come. Even, if a packet arrives at the input, but it cannot enter, because somebody else is already occupying your state. I am not dropping the packets in intermediate switches, remember; I am not dropping the packets, here. Now, that is a different case, again.

Student: But if you use output and input will drop if we use output buffering.

How I will do? There is nothing like output and input; what is the output of a switch, is the input for the next stage.

Student: So, this is buffer?

So, model is same. You take either, input or output. So, this is actually, input queue is being considered for every switch. A packet comes in, only one packet can be buffered here, and the packets from these buffers, will move to further stage. For example, this buffer and this buffer; both are free; both flags will show here, and this packet has to go here; this has to go here; they will go immediately. If both have to go here; only one go, and one will remain blocked. If both have to come here; one will go, and one will remain blocked. Based on that, this signal will go back. So, controversially, state signals are moving in backward direction; packets are moving in the forward direction. So, it requires exactly, n steps to move to n stages out of the switch. So, there are sub steps, which are being defined for movement, and after this, I think we will then, go to VOIP or SIP based systems, actually.