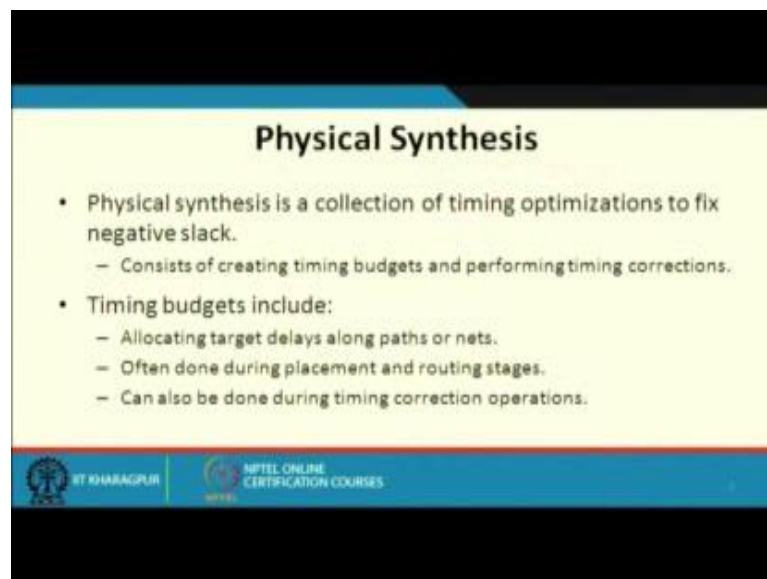


VLSI Physical Design
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 39
Physical Synthesis (Part 1)



So, we now talk about some of the techniques for so called physical synthesis. That when we say physical synthesis what we actually mean is we are trying to adjust a control some of the negative slacks. So when you do timing analysis, static timing analysis on a netlist we may find that some of the nets they are having negative slacks so some corrective actions need to be taken. So the required arrival time is less than the actual arrival time. So for those cases you need to carry out some kind of corrective steps this is what is meant by the so called physical synthesis steps. So we start our discussions.

(Refer Slide Time: 01:07)



Physical Synthesis

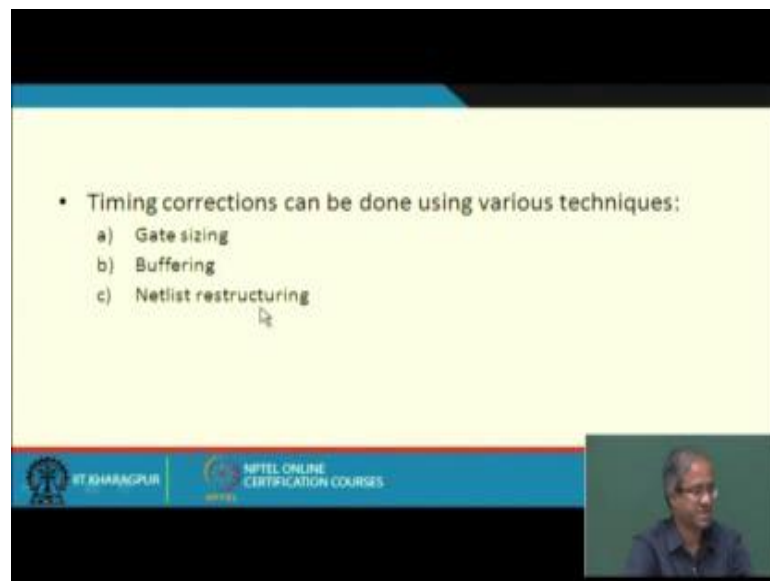
- Physical synthesis is a collection of timing optimizations to fix negative slack.
 - Consists of creating timing budgets and performing timing corrections.
- Timing budgets include:
 - Allocating target delays along paths or nets.
 - Often done during placement and routing stages.
 - Can also be done during timing correction operations.

 IIT KHARAGPUR  NPTEL ONLINE CERTIFICATION COURSES

So, physical synthesis as I have said is nothing, but collection of timing optimization techniques. The main objective for doing this is to try and control the negative slacks. Try and eliminate the negative slacks make them 0 or positive. So earlier we had seen what a timing budget is. So this can consist of creating some timing budgets. And based on these budgets you try to check whether anywhere this budget is getting you can say not satisfied, if you try to satisfy these budgets at every point in the circuit and whenever we see that there is some kind of a violation you try to take some corrective steps.

Now, here we shall be looking at what are the possible corrective steps we can take in order to control these kind of timing violations. So when we say timing budgets here they include target delays along I means either an entire path or along shorter segment of the paths which are the nets. This can be done either during placement or routing stages or also can be done later on during timing corrections; that means, you see after doing everything that some of the timings are not getting satisfied. So, you can carry out these corrective actions then also.

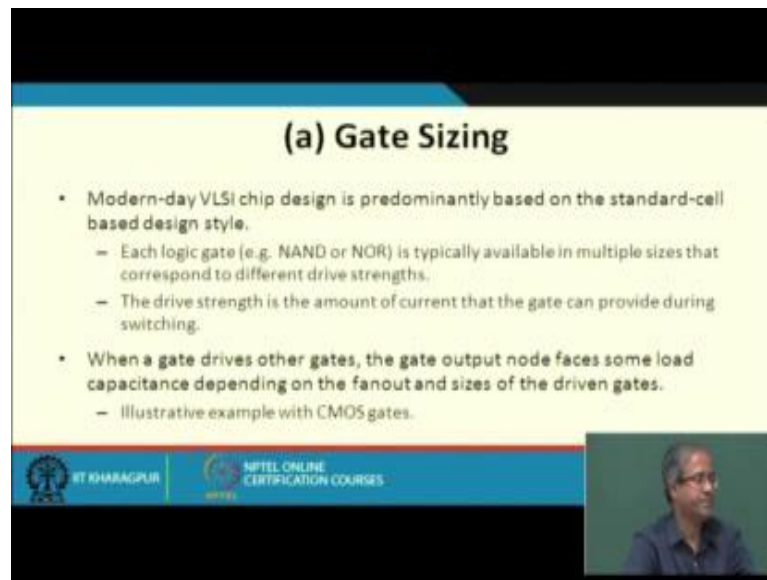
(Refer Slide Time: 02:51)



- Timing corrections can be done using various techniques:
 - a) Gate sizing
 - b) Buffering
 - c) Netlist restructuring

Broadly speaking timing corrections can be done using 3 approaches. First one is referred to as gate sizing, second one called buffering in which involves introducing buffers in the net list, and third one is by modifying your net list in some way this is called net list restructuring.

(Refer Slide Time: 03:19)



(a) Gate Sizing

- Modern-day VLSI chip design is predominantly based on the standard-cell based design style.
 - Each logic gate (e.g. NAND or NOR) is typically available in multiple sizes that correspond to different drive strengths.
 - The drive strength is the amount of current that the gate can provide during switching.
- When a gate drives other gates, the gate output node faces some load capacitance depending on the fanout and sizes of the driven gates.
 - Illustrative example with CMOS gates.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

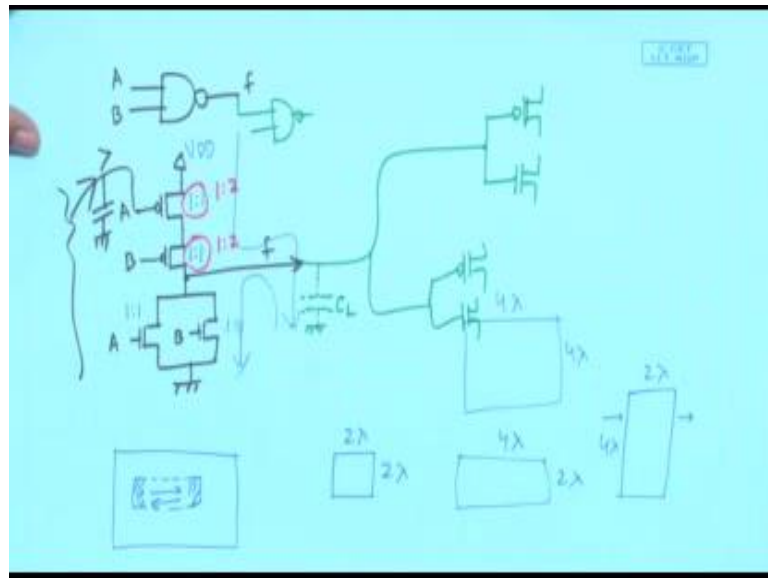
So, let us start with the first one. Gate sizing this is the very simple and a very commonly used technique for adjusting the delays in a circuit. Now one thing you observe is that today as I had said earlier most of the well assigned circuits that are that are manufactured they are based on the semi-custom or the standard cell designed style. So what is the basic idea there? We have a technology library where there is a set of cells which are pre designed and are kept in the library they are all of the same height their widths can vary. So, whatever is required I pick up a cell from the library, I put it in my layout. And the layout is nicely arranged in terms of rows.

Now what we talk about here is that let us take an example, let us say I need a 2 input NAND gate. So I expect that in the library there would be a 2 input NAND cell. There is a cell indeed, but not just one there can be multiple such 2 input NAND gates were the sizes of the gates vary. So I shall explain what this varying of the size means.

So, what I have just now mentioned is that each logic gate or the simple functional blocks which are supposed to be there in a standard cell library are available in multiple sizes. See roughly speaking a larger gate will have a larger current driving capability. That is why we say here is that they can have different drive strengths. Larger gate will have larger current driving capacity; smaller gates will be having less current driving capacity. So this drive strength actually determines the current, which the gate can provide to the load during switching. Now this is an important point I shall be explaining

this. So when a gate is driving some other gate the gate output actually faces load capacitances. So, on what factor does this capacitance depend? It depends on the fanout how many gates it is driving and also the size of the driven gates. Let us try to just explain this fact. Let us look at a simple example to input NAND.

(Refer Slide Time: 06:03)



So, I am showing you a CMOS level transistor level realization of a 2 input NAND. So in the pull down you have the 2 n type transistors nMOS, and here you have the pMOS transistors, a and b and this is the output let us say f. So, here this is a this is b and this is f.

Now the question that we are trying to ask is that well how small can a transistors be. Now if you look at a transistor say from a top view a MOS transistor, there is a region which is the channel. This is the channel on one side you have this source on the other side you have the drain. So current will be flowing in this channel. Now the question arises this is source this is drain or the reverse and this is the channel. Now the question arises that what is the size of this channel we need to use. Now you see from natural optimization point of view why should we unnecessarily make a gate larger the channel larger. We will try to use it as small as it serves my purpose. You see the sizes of these channels they are all based on some kind of featured size. This is a very standard technology or standard procedure a channel can be minimum sized 2 lambda by 2 lambda. This is the minimum featured size it is called. And there is another thing that the

pMOS transistors their mobility; that means, the whole mobility is slightly less; that means, the holes move slightly slower as compared to the electrons in the nMOS transistors.

So, to make this switching speed comparable the pMOS transistors are made slightly bigger. You see you can make a transistor like this; you can make a transistor like this, 2λ by 4λ . I mean you can make it like this 2λ by 4λ or you can make a square larger type, 4λ by 4λ . There are so many ways you can change the size of a transistor means the channel. So when I say the size of a transistor means I refer to the size of a channel.

Now, what I am saying is that, well I am assuming that this p type and n type are all identical, but actually p type will be slightly bigger because of the mobility issues, so the basic transistor size is 2λ by 2λ we refer to as 1 is to 1, 1 is to 1, 1 is to 1, 1 is to 1, so this 1 is to 1 refers to the case that the length and the width are the same, but suppose I make the channel wider. So I make the channel wider like this I mean source is on one side drain is on other side. So because it is wider the resistance will become less. So if resistances become less it can drive more current. So what I am saying is that I can make a gate larger so called larger by changing the dimension to say instead of 1 is to 1, is to 1, I can make it 1 is to 2 something like this.

So, I make the gates wider. So if I make it wider it can drive more current. So the basic idea is something like this. And just another thing you just see that suppose this gate is driving similar to other gates. Similar gates mean what the output let us say this is f this f is going to the input of another NAND gate. Let us say so this point will be connected to one p type transistor and it will also be connected to one n type transistor like this. So for every transistor the gate to substrate capacitance is quite substantial. So an equivalent circuit we may recognize we may treat that it is driving a capacity of load C_L . So this capacitive load is dependent on the fanout if there are 2 such gates the value of C_L will be doubled because there is another such pairs of transistors that are being driven. So this fanout determines the load capacitance and when we say that the output of a gate is changing from 0 to 1 or 1 to 0, so when I say it is changing from 0 to 1 which means the capacitor is initially discharged the output was 0 now it is charging this is V_{DD} it is charging via this path.

Now, when I say that the output is going from 1 to 0 we say that it is discharging via this path. So this charging and discharging time will depend on 2 things. One is the value of the capacitance, larger the value slower will be this process and the resistance along this paths, lower the resistance faster will be the charging discharge. So if I make a gate larger this discharging will be faster so my gate delay will be less, so this is roughly what we mean by sizing of a gate how we making a gate larger can reduce it is delay. So we are not going into the further detail inside this MOS level designs, but I think this much is sufficient to know and appreciate, but that as I make a gate larger which means I make my transistors wider, the channels wider. So I can drive more currents so my RC delay will become less, so the gate delay 0 to 1, 1 to 0 rising and fall time delay will become less. And more the number of fanouts higher will be the capacitive load more will be my delay because I have to charge larger capacitors.

(Refer Slide Time: 13:15)

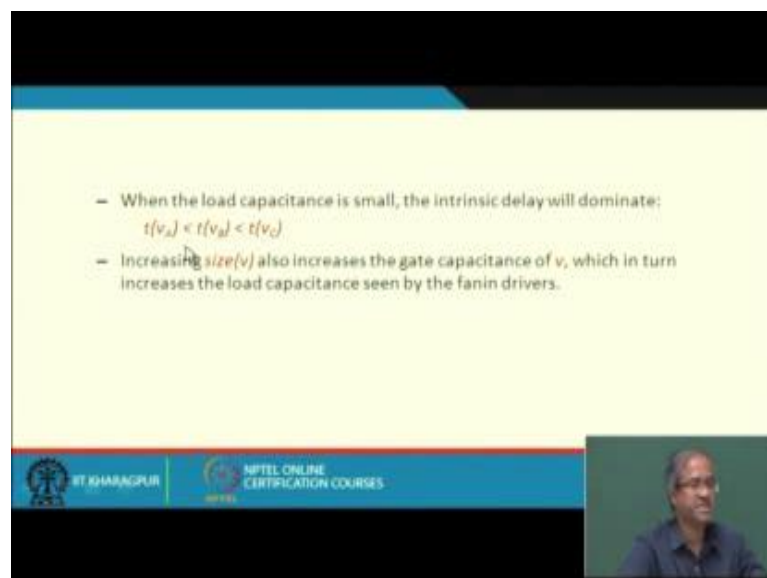
- Suppose that a gate v has three versions A , B and C of differing sizes:
 $size(v_A) < size(v_B) < size(v_C)$
- A gate with larger size will have lower output resistance and can drive a larger load capacitance with smaller load-dependent delay.
- However, a larger gate also has larger intrinsic delay due to the parasitic output capacitance of the gate itself.
 - When the load capacitance is large, the load-dependent delay will dominate:
 $t(v_C) < t(v_B) < t(v_A)$

So, let us take an example. Suppose we have a gate v and there are 3 versions of the gates which are stored in the library. Size of A is the smallest size of v A is less than size of v B less than size v C . So this something which we already said, so if a gate has larger size which means it will have lower output resistance, and which can drive a larger load capacitance with possibly the same or less delay. Now there are 2 things. So if you make a gate larger is another point which is also coming into the picture. So if you are given making a gate larger you are making the channel larger in size. So inside the gate there will also be an intrinsic capacitance which will be coming into play here. So earlier we

were talking about the load capacitance some other gate is driving another gate, but now I am thinking of the same gate, inside the gate there can be some intrinsic capacitance between the channel and the grounds various other things layers.

So, when the load capacitance is large, so this intrinsic capacitance will not matter much. So as you make the gates larger your delays will become less. So this relationship will hold. This is for the case when the intrinsic capacitance are small and the load capacitance is larger. So charging discharging time will be the predominant delay here. Charging the load capacitance and discharging the load capacitance. So since v_A is the smallest this delay of v_A will be the largest v_C is the largest delay will be the smallest, but if the reverse is true.

(Refer Slide Time: 15:17)



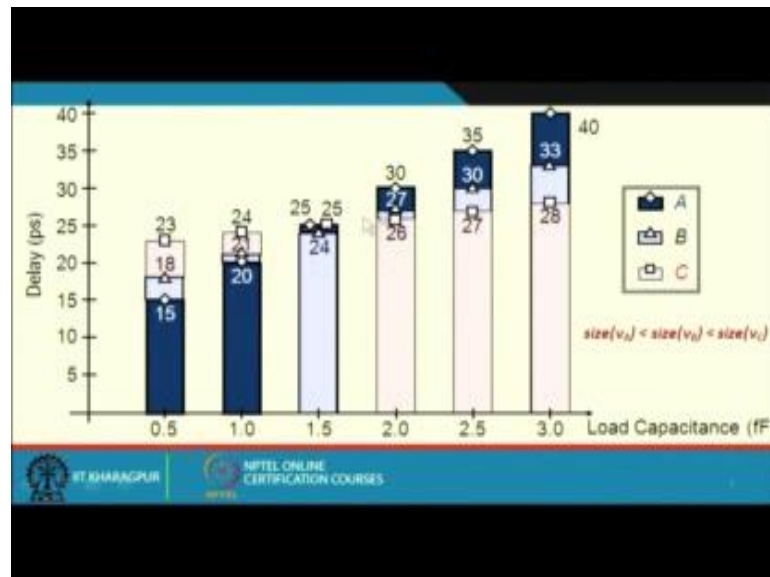
- When the load capacitance is small, the intrinsic delay will dominate:
 $t(v_A) < t(v_B) < t(v_C)$
- Increasing *size(v)* also increases the gate capacitance of *v*, which in turn increases the load capacitance seen by the fanin drivers.

Suppose assuming that we do not have an appreciable load capacitance it is small so now, the intrinsic delay will dominate and so if you make a capas if you make a gate larger it is delay will also go up. So by making a gate larger whether or not the delay will go up or go down it depends on number of things. You must also look into that. Whether it is driving a high capacitive load or it is not driving that higher load, depending on that the delay might either go up or may go down.

And another point is that if you increase the size of the gate, it may also affect the delay of the preceding stage. Because you see let us come back to this diagram again. Suppose if I make these gates larger and if there is some other gates from the previous stage

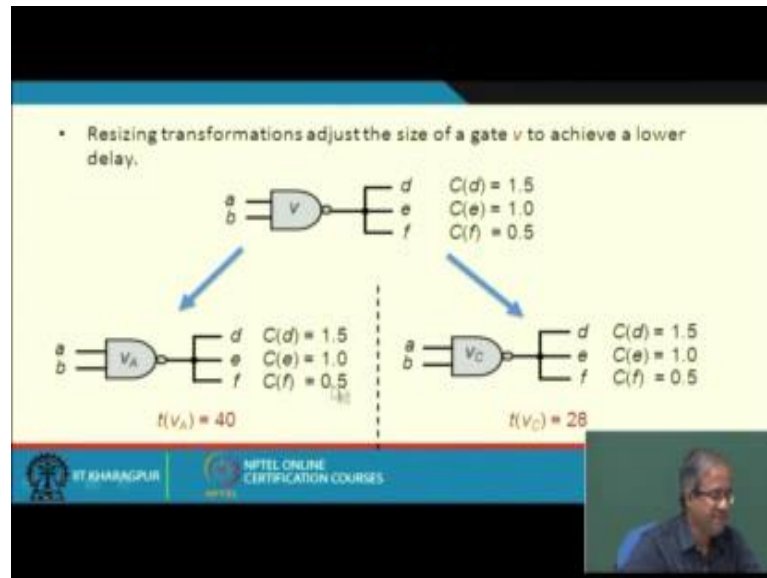
which is driving this. So this will also be facing a capacitance. So if I make this gate larger the value of the capacitance is proportional to the area. So if I make the area larger this capacitance value will also become larger. So this delay will also increase. So this is something you have to keep in mind, all right.

(Refer Slide Time: 16:46)



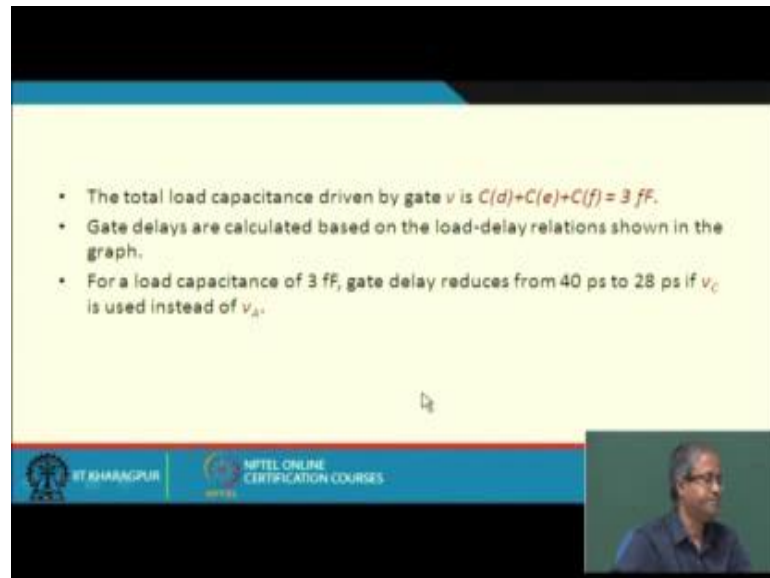
Now, this chart actually shows us some representative values for the 3 cases like this A B C whereas I said that a. Is smallest and the C is the largest, size of A is less than size of B less than size of C. So this graph shows the variation of gate delay with load capacitance. So load capacitances are plotted in femtofarads and delay in picoseconds. So what this graph actually means so means I am showing 1. Suppose if the load capacitance is 2.5 femtofarads then with your smallest gate A your delay will be 35 picoseconds with B it will be thirty with C it will be 27, right. So for the variance load capacitance is these are the typical variations that have been seen. So here we shall be taking some examples which we are using these values for illustrations. That is why I am showing this graph.

(Refer Slide Time: 18:02)



So, resizing transformation what it says. It says that I change the size of the gate to achieve a lower delay, like suppose I have a circuit netlist like this; I take one gate 2 input NAND gate which is driving 3 other points. Let us assume that the sizes of these destination gates are different and their total capacitances are like this, 1.5 femtofarads all 1.0 and 0 0.5. Now because this capacitance will lie in parallel as you know if the capacitances are in parallel the total load capacitances will get added up. So the total will become 3, 3 femtofarads. So I am showing 2 alternatives. If I use v_A ; that means, smallest gate with a load of 3 femtofarad, let us go back with a load of 3 femtofarads v_A the delay is 40. So for this case the delay of the gate will be 40 picoseconds, but if I use a larger gate v_C so you see v_C the delay is 28. So this gate will be 28. So you see just by changing the size of the gate I can change the delay from 40 to 28. So it is quite a wide variation that I can make by adjusting the size of the gate.

(Refer Slide Time: 19:44)



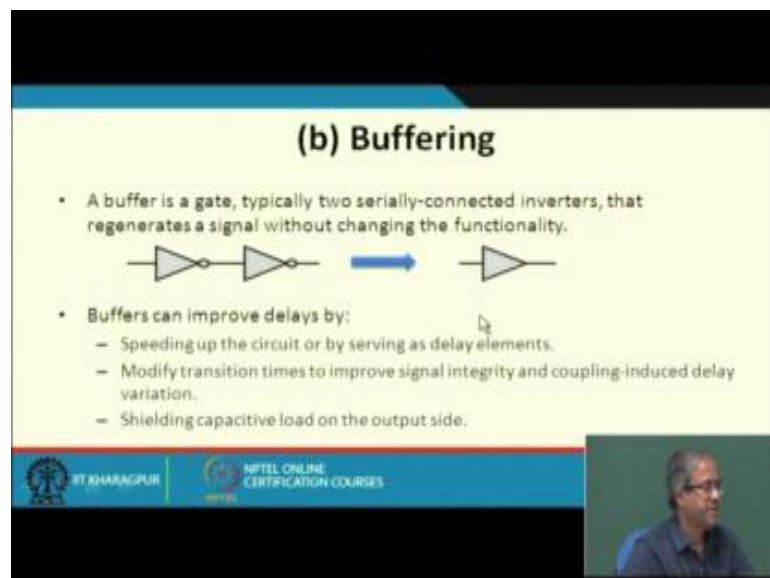
• The total load capacitance driven by gate v is $C(d)+C(e)+C(f) = 3 \text{ fF}$.

• Gate delays are calculated based on the load-delay relations shown in the graph.

• For a load capacitance of 3 fF, gate delay reduces from 40 ps to 28 ps if v_c is used instead of v_d .

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

(Refer Slide Time: 19:50)



(b) Buffering

• A buffer is a gate, typically two serially-connected inverters, that regenerates a signal without changing the functionality.

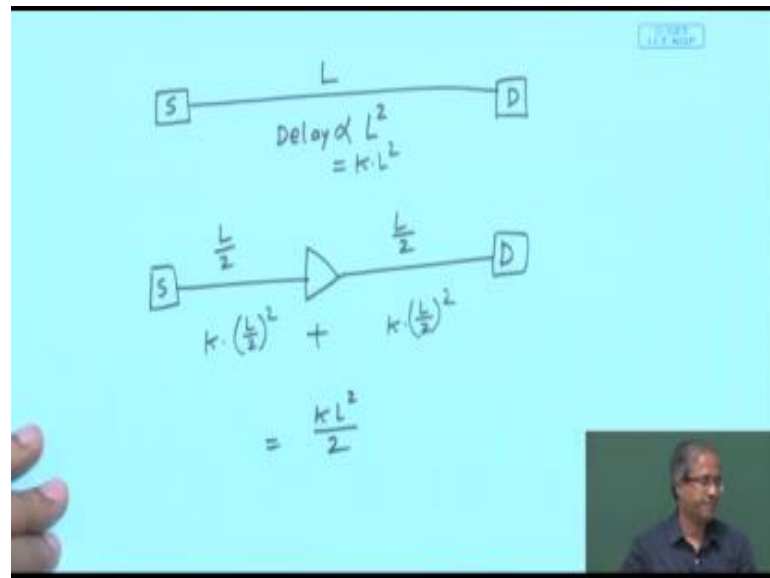
• Buffers can improve delays by:

- Speeding up the circuit or by serving as delay elements.
- Modify transition times to improve signal integrity and coupling-induced delay variation.
- Shielding capacitive load on the output side.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

This is what I have already said fine. So the next technique is called buffering. Buffering means we introduce buffers in a circuit in order to achieve something that something. Let us see what are the characteristics that get improved by introducing or inserting buffers in the circuit. So what is a buffer is typically constructed using a pair of inverters or if you want an inverting buffer then a single inverter can also suffice. Buffers can help you in doing a few things.

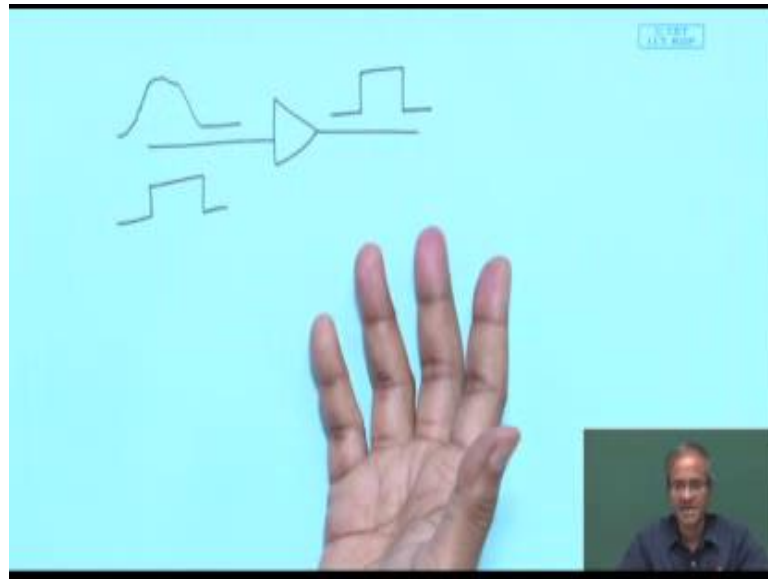
(Refer Slide Time: 20:51)



First, speeding up the circuit or by serving as delay elements - this is a point we shall be again talking about later the idea is as follows. Suppose I have a source I have a destination. There is a wire connecting that. So if this length of the wire is l , so the delay of this wire if you just compute the 1 mod delay model and compute the distributed RC model so the delay of this interconnection will be roughly proportional to the square of l .

So, now let us say I make a modification. Instead of this I insert a buffer in between. I break this total length into 2 parts l by 2 and l by 2. So let us say the delay is equal to some constant k into l square. So in this case for the first segment the delay will be k into l by 2 whole square. For the second part again the delay will be k into l by 2 whole square. So if you add this up what will be the total delay l square by 4 and l square by 4 it will be k l square by 2. So the delay has become half. So by introducing a buffer we can reduce the interconnection delay this is a very interesting and important technique for reducing delay. So this is the first thing - Speeding up the circuit.

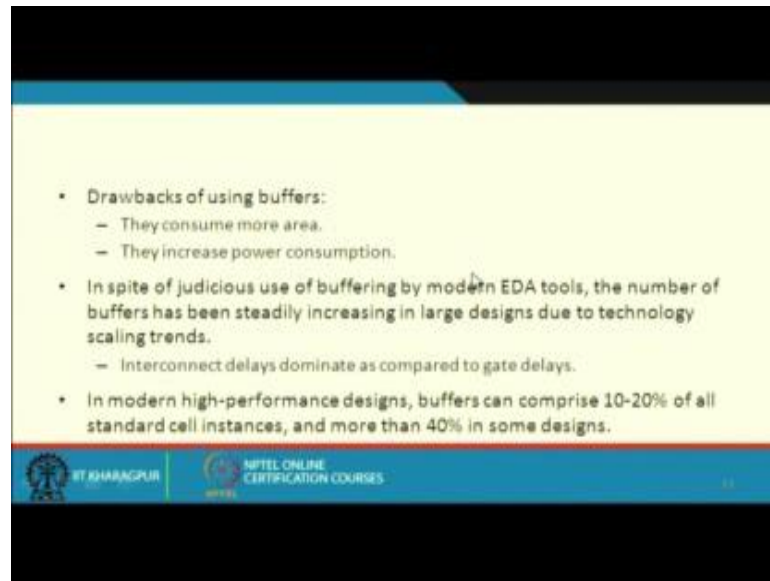
(Refer Slide Time: 22:30)



Second is modifying transition times to improve the signal integrity. So what this says is that suppose I have introduced a buffer in a circuit and the input signal it is not very good it has become like this, so instead of a neat pulse. So when I have a buffer the output of a buffer I will again regenerate this signal into a nice pulse. This is what I mean to say is that signal regeneration. So as signals propagate over a distance because of RC affects so contour of the signal gets degraded. The nature of the signal gets deformed. So if I introduce a buffer again regenerates the signal it makes it again very nice. And thirdly it can help shield capacitive load on the output side. Like for example, I have a buffer and at the output of the buffer there is a big capacitive load there are 2 capacitive load which it is driving.

Now suppose I introduce a buffer and from that buffer I am driving one of the loads and I use another buffer I am using driving another load that the other load, so I am sharing the load among the 2 buffers and the actual gate which was driving initially that is getting shielded. It is only driving the buffers not the capacitive loads. So these are some techniques. These are the improvements which can be done.

(Refer Slide Time: 24:00)



The slide contains the following text:

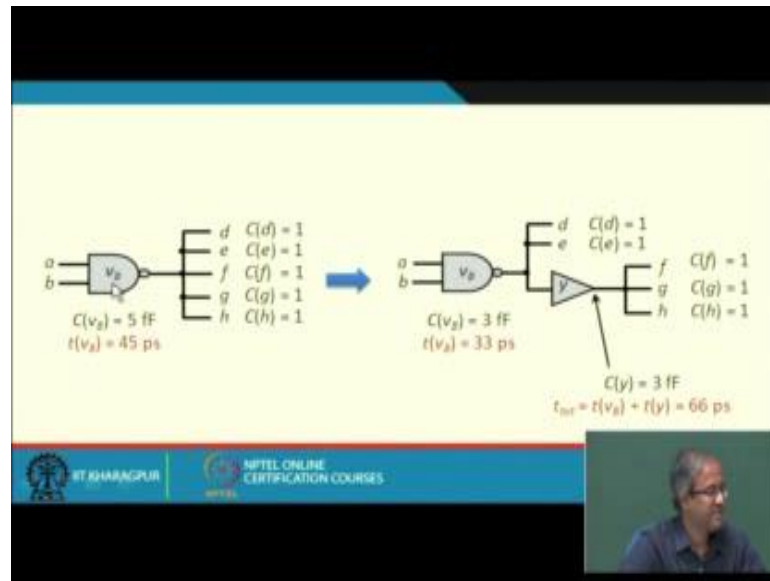
- Drawbacks of using buffers:
 - They consume more area.
 - They increase power consumption.
- In spite of judicious use of buffering by modern EDA tools, the number of buffers has been steadily increasing in large designs due to technology scaling trends.
 - Interconnect delays dominate as compared to gate delays.
- In modern high-performance designs, buffers can comprise 10-20% of all standard cell instances, and more than 40% in some designs.

At the bottom of the slide, there are logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

But of course, there are some obvious drawbacks. Buffer means some additional electronics as I said typically they will consist of 2 inverters back to back means 4 MOS transistors. They consume more area they will consume more power. So modern EDA tools they use buffers quite a lot because as you know the today's circuits are becoming very much performance aware. There is lot of performance driven optimization which are carried out. And buffer insertion is treated as a very popular technique.

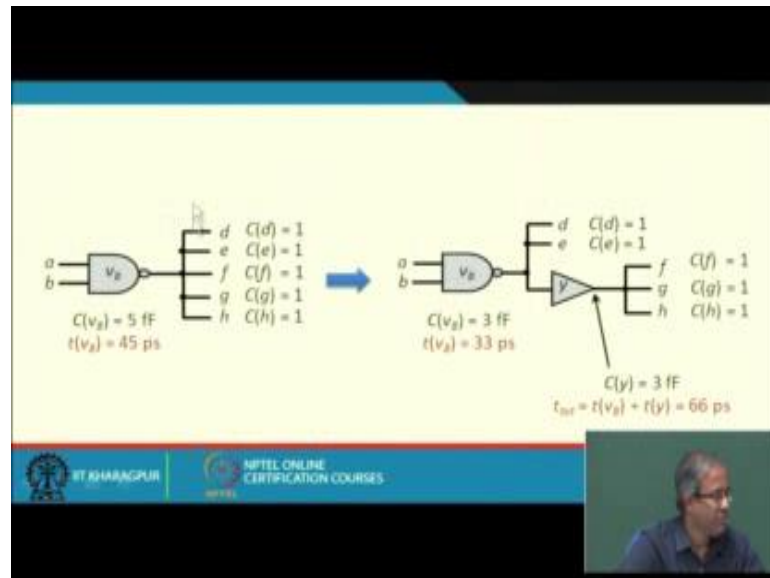
So in modern designs typically buffers can comprise 10 to 20 percent of all see buffers are also some standard cells. So means out of all standard cells 10 to 20 percent can be buffers and it can go up to 40 percent also in some designs this has been observed. So buffers although they are very good in providing some very desirable features, but there are some drawbacks as well you have to remember this. This is like a tradeoff.

(Refer Slide Time: 25:18)



So, let us see how buffering can improve. Let us take another example of a 2 input NAND gate which is driving 5 fanout lines, so each of them with the capacitance value of 1, so the total load capacitance will be 5 femtofarad and you are using v_2 this second gate. Now in this graph we have shown only up to 3. So we have not shown the whole thing just I have noted down here, so if you are using the intermediate v_2 then for 5 Pico farad 5ff femtofarad load capacitance, the delay comes to 45 picoseconds. So we have used this value. For this case the delay of the gate will be 45 picoseconds, but suppose we modify this fanout structure like this by inserting a buffer, first to d and e we keep as it is this y is a buffer we use it to drive the remaining 3.

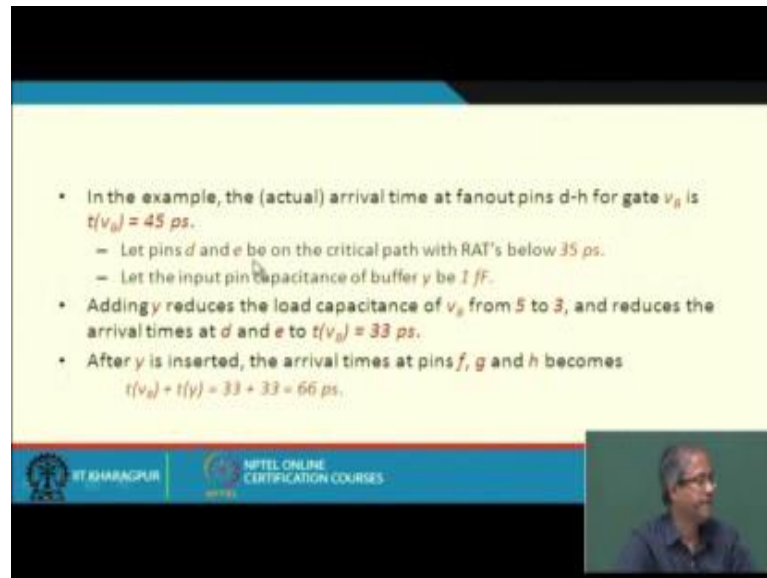
(Refer Slide Time: 26:13)



So, you see now v_B is also driving 1, 2, and 3. Y is also driving 1, 2, and 3. Let us assume that the input capacitance of y is also 1. So this will also be 3 femtofarad. This will also be femtofarad. So v_B with 3 femtofarad how much is it? 33. 33 is the delay. So this gate will have 33 this will have 33 again. So the total delay will be 33 plus 33, but this gate delay was 45 now this has decreased to 33. So if this d and e are some destinations where this signal has to be transmitted earlier. They have earlier required arrival times then you can use these strategies.

So, although the total delay is 66, but some of the outputs you can drive with the delay of 33, as if this is of course, much faster than this 45. So, this is what we have already said. This already we have explained.

(Refer Slide Time: 27:49)



- In the example, the (actual) arrival time at fanout pins d-h for gate v_g is $t(v_g) = 45 \text{ ps}$.
 - Let pins d and e be on the critical path with RAT's below 35 ps .
 - Let the input pin capacitance of buffer y be 1 ff .
- Adding y reduces the load capacitance of v_g from 5 to 3, and reduces the arrival times at d and e to $t(v_g) = 33 \text{ ps}$.
- After y is inserted, the arrival times at pins f , g and h becomes $t(v_g) + t(y) = 33 + 33 = 66 \text{ ps}$.

So with this we come to the end of this lecture. In the next lecture we shall be looking at some other physical design techniques where netlists restructuring various techniques we shall be looking at, so how we can adjust the delays for this. If you see this kind of delay adjustments are very commonly required for timing driven adjustments and constraints satisfaction. So whenever you have some kind of negative slacks somewhere you have to adjust the delays somehow to make this slack either 0 or positive. These are some of the very commonly used techniques.

Today we have seen gate sizing and buffering. Buffering again we shall see later in a different context, but for the time being.

Thank you.