

**CMOS Digital VLSI Design**  
**Prof. Sudeb Dasgupta**  
**Department of Electronics and Communication Engineering**  
**Indian Institute of Technology – Roorkee**

**Module No # 05**  
**Lecture No # 24**  
**Logical Efforts – III**

Hello everybody and welcome to the NPTEL online certification course of the CMOS digital VLSI design we will be having the third module now on logical effort in the first two modules in the module of logical effort we had seen what is logical effort based on basic principles of logical effort we have also seen how logical effort is calculated for simple inverter and then for a various gates of various inputs and we also primarily came to know that logical efforts method can be used to calculate the total delay between input and output.

The second thing which we did was taking an primary example we had looked into the fact that more complicated the gate is or larger number of input the gate as ah you will have the larger value of your logical effort what will do in this module is look into some design examples and then see how logical effort comes into play for this design examples so let me start the third module for you is basically a logical efforts and the outline of this module will be that I will be introduced to you the basic idea of logical effort as applied to a basic combinational logic cell.

**(Refer Slide Time: 01:43)**

## Outline

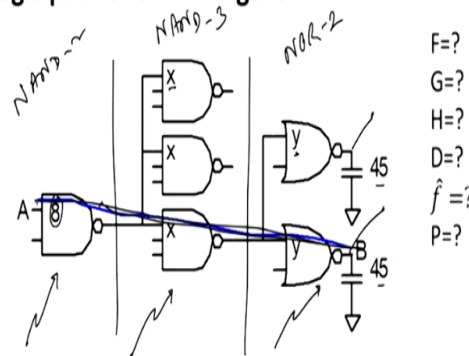
- Introduction
- Design Example for Multi-Stage Network ←
- Choosing the length of the path > 'critical path'
- FORK Design
- Logical Efforts for Asymmetric Gates
- Recapitulation

We will take up the multi stage network which means that we will be using the multi stage network to explain to you how logical effort will be useful in determining the value of the gate once we know that can we therefore determine what is the critical path available to us once we have actually received the logical effort delay. If we will also look into what is known as the FORK design and see how does a logical effort tries to find out the delay for that.

And then for Asymmetric gates in the sense that when you have when your delay paths are not equal and you have Asymmetric gates available to you whose parasitic delays at different then how to do you use to do that finally we will be concluding the lecture by certain examples and show you the overall course. Now as we have discussed now we will be looking forward into having maybe a delay mechanism or calculating delay.

**(Refer Slide Time: 02:44)**

**Example 1 Calculate the gate sizes  $x$  and  $y$  for least delay for the logic path shown in figure?**



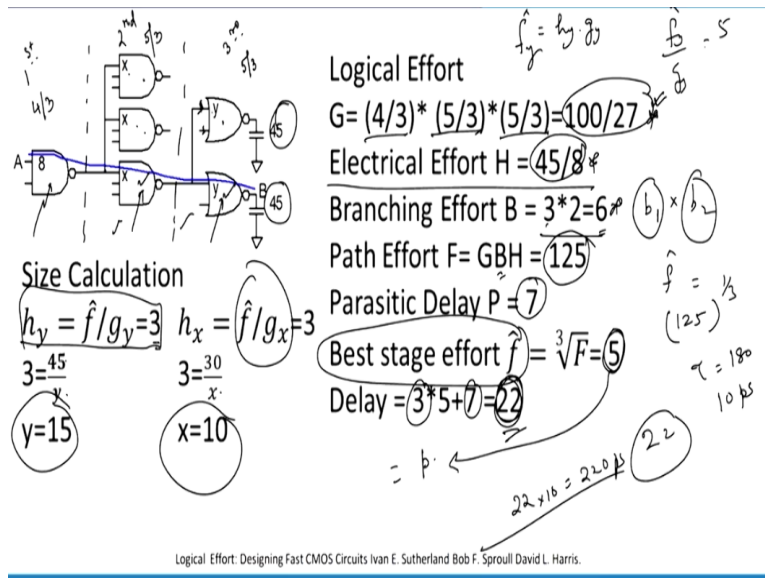
Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris.  
&  
Internet Sources

When you have blocks of various types of gates for example this is basically a 3 level logic so this is 1 level 1 we have level 2 and level 3 in level 1 we have a two input so this is NAND 2 in level 2 you have got all the gates are basically NAND 3 and in third you have actually NOR 2 right and they are terminating at 45 CL value which you see or the input is basically 8 input CL is basically 8 and output CL for whole logic is basically 45 so with these inputs we need to find out the values of these gates X and Y so that you have the lease delay available to you right and.

So the delay is basically propagating from A which is basically this gate right and then it goes on to the second level and then finally the third level which you see here and then comes out in an

output side. So the output is basically from the here which you this is basically a B so blue path or blue line which you see here right the blue line which you see here this blue line is primarily let us suppose the path across which you find want to find out the delay. So let us see how we can find the delay or explain to how we can find the delay for such a path.

**(Refer Slide Time: 04:07)**



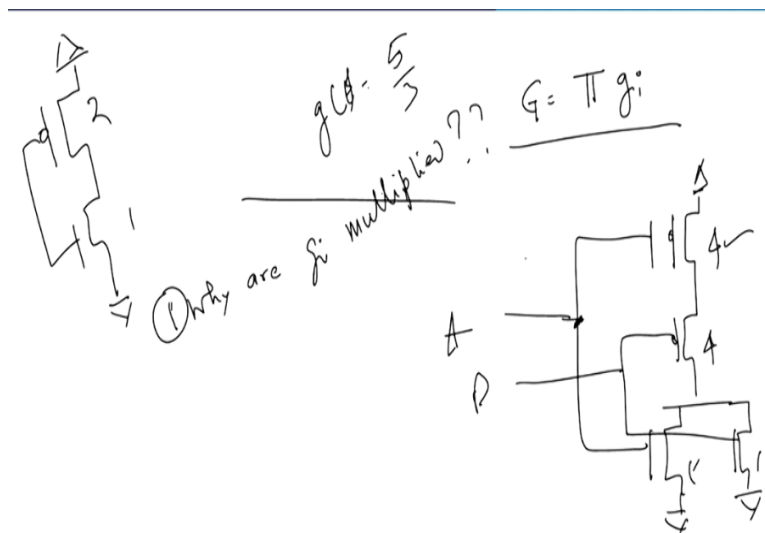
Logical Effort  
 $G = (4/3) * (5/3) * (5/3) = 100/27$   
 Electrical Effort  $H = 45/8$   
 Branching Effort  $B = 3 * 2 = 6$   
 Path Effort  $F = GBH = 125$   
 Parasitic Delay  $P = 7$   
 Best stage effort  $f = \sqrt[3]{F} = 5$   
 Delay  $= 3 * 5 + 7 = 22$   
 $22 * 16 = 220 \text{ ps}$

Size Calculation  
 $h_y = f/g_y = 3$      $h_x = f/g_x = 3$   
 $3 = 45/x$      $3 = 30/x$   
 $y = 15$      $x = 10$

Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris.

As I discussed with you the logical effort for a 2 input NAND gate is therefore.

**(Refer Slide Time: 04:17)**



Why are  $g_i$  multiplied??  
 $G = \Pi g_i$   
 $G = 11/3$

So what I told you was that for you in the path when you discussion the path or then the overall was basically the multiplication of  $g_i$  right. So we need to look at the three stages and each stage the value of  $g_i$  which you get will be multiplied with that of the second stage right and therefore

third stage. So if you look here the first stage which is this one the first stage this is the stage number 1 first if this is my stage number 2 second stage right and this is my stage number 3 this is my third stage.

So the first stage we have a two input NAND whose  $g$  is basically  $4/3$  I have then a 3 input NAND which is  $5/3$  and then followed by a two input NOR which is again  $5/3$  right. Two input NOR is primarily  $5/3$  I suppose you can understand this  $5$  from where I got this result as I discussed with you in the NOR gate if we take the NOR gate then then then the diagram is something like that and you have something like this right and then you have got two NAND gates 2 NMOS transistors.

So your input A and then this is your input B which you see this is your input B so if you compare that with your with your minimum sized inverter then  $2$  is to  $1$  is the value which you get then if you compare this with this one then this has to be  $4$  this has to be  $4$  and these one has to be one so the value of  $C$  in which once sees from this is basically  $4 + 1 = 5$  so I get  $g$  of  $e$  goes to  $5/3$  right.

So I get  $5/3$  here and therefore I get  $5/3$  so you see  $4/3$  multiplied by  $5/3$  multiplied by  $5/3$  and that is  $100 / 27$  is the total  $g$ . So the we multiply the logical effort as we move from this point to this point stage 1 to stage 2, stage 2 to stage 3. I will leave as an exercise to you that why do you acutely multiply why do not we add the logical effort why do we multiply logical efforts to get the overall logical effort why do not we add the logical efforts here.

I just think about yourself look why is it like that right so the simple question is therefore why are  $g_i$  multiply right and they are not why they are not added right. So I need to find out the question to this answer the answer to this question sorry then we come to the electrical effort which is  $h$  is if you remember  $C_{out}$  by  $C_{in}$  so  $C_{out}$  was actually given as  $45$  your  $C_{in}$  is already given as  $8$  so I get  $45 / 8$  as my  $h$  value right.

Let us look at the branching effort now branching effort is primarily if you see at this point you have got you have got 1, 2, 3 branches and then you have got two branches in the second half so  $3$  into  $2$  is basically  $6$  of branch is  $b_1$  into  $b_2$  which you get branching at the first second stage and  $b_2$  is branching at the second stage. So I get  $3$  into  $2 = 6$  so therefore the total path effort

from point A to point B is nothing but the multiplication of logical effort across the path multiplied by the electrical effort across the path multiplied by the branching effort across the path.

So if you multiply this this and this across the path I get 125 as the total path effort right as since I am using a two input NAND gate two input NAND gate here 3 input NAND here and 2 input NAND here if you add all the parasitic across the delay it comes out to be 7 right I will come out to be 7 just add this + this and this from your previous table in the previous module and I get 7 right.

What we do is that you can see the best stage effort now which is best stage effort basically means that you need to find out FF cap which is basically capital F which is the path effort it we look at the path effort which is 125 and then if try to find out the cube root of that because there are three stages available to us and therefore stage of it means the idea is that why do find it basically to tell you an idea is that each stage contributing the overall delay of the system.

So if we if I assume equally divided among three stages I just need to find out the cube of root of 125 to find out each stage so I get that equals to 5 right we get 5. So if you come back to your original discussion therefore we saw this to be is equal to  $p$  into  $h^5$  I get the best stage delay right and then I multiplied with 3 + the parasitic delay which is 7 I get 22 is the intrinsic available to me which is the available delay intrinsic available delay to me.

Now so this so we have found out 22 dyna meter 22 is the intrinsic delay between point A and point B if we know what is the tau d value of let us suppose this is 180 nano meter say tau d value is 10 nano second or 10 Pico second so 22 into 10 happens to be equals to 220 Pico seconds I can find out the delay from here. Now how can you find out the sizing which means that how can you size your second stage and third stage so that you get the minimum delay condition.

Now idea here is that if you find out  $h_y = f \text{ cap upon } g_y$  right I will tell you the reason  $f \text{ cap}$  is nothing but  $f \text{ gap}$  is nothing but  $h_y$  into  $g_y$ .  $G_y$  is the logical effort multiplied by  $h_y$  is the path of  $y$ th stage right. So for the  $y$ th stage  $f_y$  happens to be  $f_y g_y$  right so if you want to find out  $f_y$  it will be nothing but  $f_y \text{ cap my } g_y$ . You already know  $f_y$  is 5 so 5 sorry we already know  $f \text{ star}$

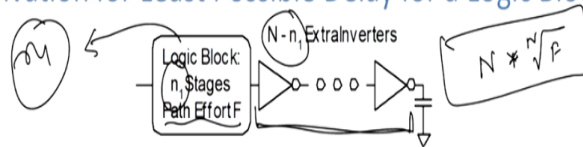
in this case is basically so you want to find out for the value of this thing this is already known as 3  $t_{hy}$  is already known as 3 because it is there so I get 45 by  $y$  into 45 by  $y = C$  so I get  $y = 50$  right.

Now so that is the reason why I get 15 in this case why which is this we have 45 loading which is available to you so you had a 45 loading  $45/y =$  this thing similarly and the  $x$  value we get is that  $f \text{ bar} / gh$  which is basically again equal to 3 stages here 3 input pass here again so you have 3 into  $30 / x = 10$ . So now I know that in order to ensure a minimum delay between point A and point B I required to have my second stage devices which is 3 input NAND gate to be 10 times larger as compared to the minimum sized inverter and third stage should be fifteen times larger as compared to the minimum sized inverter right and that will ensure that you will automatically get the same value.

Now interestingly you see the third stage you have got lower delay lower number of inputs which effectiveness means that its input capacitance is relatively low and therefore you require its sizes to be high why the sizes of these NOR gate to be high to drop the same amount of current as you expected to be sort of minimum size of inverter. So we have therefore understood that given a combinational logic circuit where you have a primary input and an output you would be able to find out the path effort best path effort and therefore the  $W/l$  ratio of individual transistors to a certain or to clarify those papers.

**(Refer Slide Time: 12:16)**

### Derivation for Least Possible Delay for a Logic Block



Consider the circuit containing  $N$  no of stages,  $n_1$  no of stages are in the Logic Block and  $N - n_1$  Are the extra inverters. Least delay for the logic path will be

$$D = N * F^{\frac{1}{N}} + \sum_{i=1}^{n_1} p_i + (N - n_1) p_{inv}$$

$$\frac{\partial D}{\partial N} = 0 = -\frac{F^{\frac{1}{N}} * \ln(F^{\frac{1}{N}}) + F^{\frac{1}{N}} + p_{inv}}{N^2}$$

Let  $F^{\frac{1}{N}} = \rho$

$$0 = \rho(1 - \ln \rho) + p_{inv}$$

Handwritten notes include:  $\sqrt[N]{F} = 1 - \ln \frac{\sqrt[N]{F}}{1 + \frac{p_{inv}}{\sqrt[N]{F}}}$  and  $\sqrt[N]{F} = \rho$ .

So let us say you have logic block right logic block which is basically having  $n_1$  stages with path effort of  $f$  and then followed by inverters which are aligned in series with each other and there are  $n - n_1$  inverters. So there are capital  $n - n_1$  inverters why because the total  $n$  total they were  $n$  number of stages out of which  $n_1$  has already been utilized for the logic block so the left overs are basically extra inverters available to us.

So if you look at the circuit here consider containing  $N$  number of stages with  $n_1$  number of stages in the logic blocks so out of total  $n$  number of stages there are  $n_1$  number of inverters within the logic block and capital  $n - n_1$  number of basic inverters extra inverters available to you. So for this logic path I get can define delay to be equals to will be obviously equals to  $n$  multiplied by root of  $f$  as discussed with you the previous turn that if there are  $n$  number of stages we can find out square root of  $fn$  as the best logical path that if you multiply by  $n$ .

$N$  is the number of stages I get the minimum delay available to you plus sum of your parasitic which is available to you  $+ n$  because  $n$  are inverters. So for each inverter I know the value of parasitic this for the block right this is for the both inside the block we are saying and this is outside the block inverter. So it is capital  $N - \text{small } n$  into  $p$  inverter.

$P$  inverter is the parasitic delay for the each inverter now if you want to minimize it the you need to just you need to solve this you need to find out differential of  $d/n \text{ del } n \text{ del } d \text{ del } l$  and if you solve it we get finally that  $0 = \log - 1 - \log \ln \text{ of } \text{row} + p \text{ of } n$  where  $\text{row}$  is basically root of  $n$  so  $n$ th root of  $f$  right. So where this is root of  $\text{row}$  right so what is get is basically  $f$   $n$ th root of  $f$  right multiplies  $1 - n \ln \text{ of } \ln n \text{th root of } f \text{ right } p \text{ inverse equals to } 0 \text{ inverter} = 0 \text{ right}$ . And this is simple derivation which is basically based on simple calculation of derivation which means that we have just derived it here.

**(Refer Slide Time: 14:36)**

## Best Stage Effort

- $\rho(1 - \ln \rho) + P_{inv} = 0$ 
 $n^D$ 
 $\rho(1 - \ln \rho) = -1$
- Equation has no closed form solution
 $\sqrt[n]{F} = 4$
- For  $P_{inv} = 1$  the value of  $\rho = 3.59$
- Therefore the best stage effort will be nearly 4

Now this implies that this equation need to be solved in order to obtain the value of row and from if you know the number of stages you can find the value of f or if we know the value of f we can get the number of stages. Now the only problem is that it has got no closed form solutions right. So this has got no closed form solutions you do not have a closed form solutions available to me.

So what we do you do need to (( )) (15:00) way of doing it that means that if I take p inversion = 1 for an inverter for an inverter p in = 1 then I get row  $1 - \ln$  of row = -1 from here I get the value of row and then comes out to be 3.59 right. So therefore the so row is basically nothing but the f of root n and this is nothing but the stage effort best stage effort. So I therefore say that the best stage effort nearly equals to 4 3.59 is nearly equal to 4 so if you are able to sustain a stage effort of approximately = 4 your job is almost done in terms of finding out the value of the p inversion or the total delay across these two bars.

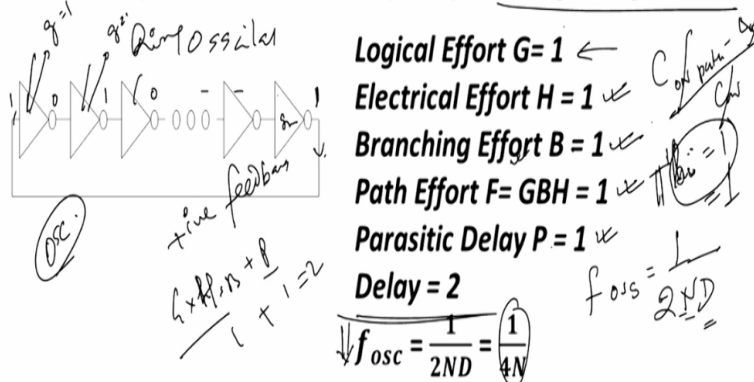
So you have ensure that you have to first of all ensure the value of  $P_{inv}$  which is basically the parasitic delay of the inverters I am assuming that to be equals to 1 and therefore the row value comes out to be 3.5 and which is approximately equals to 4 right. So from here if you know the value of f then I can calculate indirectly the value of n as to be the integer of course similarly if you know the value of n I can get the optimal value of the stage ratio in order to have this best efforts available to you.

**(Refer Slide Time: 16:16)**



## Design Example

Example 2 Estimate the frequency of N stages ring oscillator



Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris.

Internet Sources

Now let us suppose so this is for a combinational logical and it worth finding out for combinational logical block but let me show to you for n stage ring oscillator which is basically means that you have an inverter so inverter is 1, 2, 3 for I hope you understand this point it is suppose this is 1 0 1 0 and so on and hence so forth and finally you will get 1 here which means that there will be a positive feedback here positive feedback right whenever there is a positive feedback there will be oscillation available to you and as result this will result in what is stable oscillation and this is also the configuration is also known as ring oscillator right it is the ring oscillator design.

This is quite interesting in the sense that the total logical path effort because all are one remember so all your g's are 1 this g this g is because it is 1 input so because all are inverter right so all are equal size inverters if this is true that g's are one if  $g_1$  into  $g_2$  into  $g_3$  into  $g$  to the power n will always be equals to H similarly since all as our inverter its output load to input load will always be equals to 1 right. So you get path effort 1 since there are no branching I will get this to be equals to 1 and therefore the total path effort equals to  $g$  into  $h$  into  $p$  is also equals to 1.

Since it is a inverter I get parasitic  $P = 1$  and therefore delay is nothing but  $G$  into  $H$  into  $B +$  parasitic delay  $P$  so this is  $1 + 1 = 2$  I get delay equals to 2 fine now therefore I can get a frequency of oscillation  $f$  oscillation to be equals to  $1$  upon  $2ND$  right where  $D$  is the delay  $N$  is the number of delay available to you and therefore if delay is 2 2 into 2 is 4  $1 / 4$  I get so

oscillation is  $1/4$  which means that if the number of stages is increased my  $f$  oscillation will start to fall down of course and that is the reason you can optimize the frequency of oscillation by looking at the value of  $n$  here right.

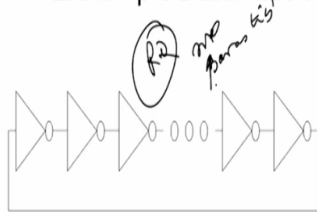
But the idea here is or the logic behind all this thing is that we assuming all the inverters here are equally sized right and they are equally loaded in input and output then only we can assume that the logical effort for each stage is 1 not only the that the product of the logical stage will therefore equals to 1 between input and output electrical effort 1 make sense because capacitance seeing at the input and output also equal and therefore  $h = 1$  since there are no branching available here  $b=1$  and so if you remember it is  $C$  on path /  $C$  off path /  $C$  on.

So there is no  $C$  off so this goes to 01 this by this cancels I get this = 1 so you remember  $C$  on path + -  $C$  off path /  $C$  on path right. This was the formula for my branch since there is only one path so  $C$  off branch goes away and this cancels out and therefore I get  $h I =$  sorry  $b_i = 0$ . Now you do a multiplication of that there will be  $V_{u1}$  so I get that is the reason we get a such a delay such a delay profile which you see right. So now we get oscillation which you get  $1/4$  right when is the number of stages.

**(Refer Slide Time: 19:49)**

### Design Example

Example 2 Estimate the frequency of  $N$  stages ring oscillator



**Logical Effort  $G = 1$**   
**Electrical Effort  $H = 1$**   
**Branching Effort  $B = 1$**   
**Path Effort  $F = GBH = 1$**   
**Parasitic Delay  $P = 1$**   
**Delay = 2**

$$f_{osc} = \frac{1}{2ND} = \frac{1}{4N}$$

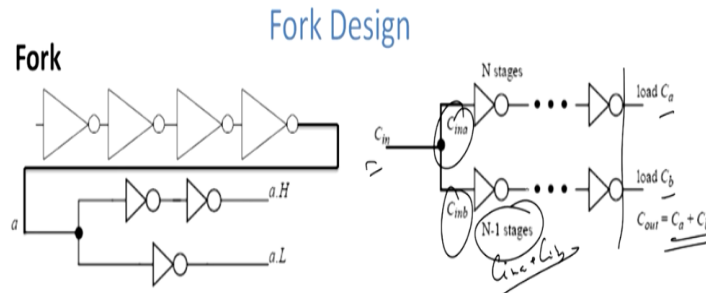
Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris.

&  
Internet Sources

Now if you look at this all the stages have got so now again assumption here has been that all the stages will give you exactly the same delay reality not true when you actually fabricate it on you will see the they will be large amount of parasitic which will be coming into this whole RO so for this RO to be operative I assume that there are no parasitic no parasitic's available to you and

it works fine even for any stage ratio of that matter any sizing of the PMOS and NMOS transistor within right. We next move to Fork design now Fork design is something like this that so you have let us suppose you have 2 signals and these are A the idea is that.

**(Refer Slide Time: 20:41)**



- *Signal a and a' should appear at both the outputs at the same time*
- *Size both the paths to have equal delays*
- *Need to consider 'p' if the number of stages in the fork's limbs is not large*

Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris.

Whenever you do a design you ensure that if there is a forking of the signal which means that the signals are deviating from original path and going to some low impedance path then you should ensure that at the primary output both the inputs which you have given appears simultaneous for proper evaluation right and that is the reason we generally say that if this is my  $C_{in}$  right if this is my  $C_{in}$  as a  $C_{inA}$   $C_{inB}$  there are  $n$  number of stages  $C_a$  and  $C_b$  where  $C_{out} = C_a + C_b$  right input = in A + in B at this stage and then (()) (21:20) stages available to you right.

**(Refer Slide Time: 21:24)**

## Fork Design

$$d_{fork} = (N-1) \left( \frac{C_b}{C_{inb}} \right)^{1/(N-1)} + (N-1)p = (N) \left( \frac{C_a}{C_{ina}} \right)^{1/N} + (N)p$$

- **Minimum delay desirable within the fork: One of the limbs have ideal  $p$** 
  - **For a given  $N$  and fixed  $C_a/C_b$ :  $C_a/C_{ina}$ ,  $C_b/C_{inb}$  and thereby  $C_{ina}/C_{inb}$  are unique**
  - **For a given  $N$  and fixed  $C_a/C_b$ :  $H_{fork} = (C_a + C_b)/C_{in-fork}$  can be used to design the 'preceding' buffer**

When this is the Fork design which you see now in the Fork design therefore delay in the Fork  $d$  equals to  $n - 1$  right because there are  $n-1$  stages available  $C_b$  is the output bulk capacitance divided by  $C$  in  $B$ .  $C$  in  $B$  is the input bulk capacitance hold to the power  $1 / N - 1$  and if you solve this you get  $= n$  of  $C$  upon  $C / n + np$  right. So if one of the Fork which is basically if you look at the Fork if one of the Fork got a ideal delay right as got a one of the limbs as got a ideal row then I can simply say that  $C_{ina} / C_{inb}$  is basically my input at  $a$  to input at  $b$  will have a unique value of course will have a unique value available to you.

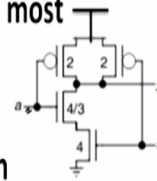
So for a given  $n$  and a fixed  $C_a / C_b$   $C_a$  and  $C_b$  is basically the input capacitance at this node and this node right  $C_a / C_b$  for a fixed value of  $C_a/C_b$  I can write  $H_{fork} = C_a + C_b$  upon  $C$  in fork right. So this is the formula which people use that  $C_a + C_b / C_{in} = fork$  so I have got  $C_a$  by  $C_b$  right divided by you have got  $C$  in fork and this is the used in design so this was used in the design of the buffer as well but get  $H_{fork}$  to be  $= C_a + C_b / C_{in}$  Fork and if the typical value of  $C_a = C_b$  and equals to  $C$  in fork for all three are equal then simple we can simply write this to be equals to  $C_a$  right and this is a well-defined formula.

**(Refer Slide Time: 23:12)**

## Asymmetric Gates

- Asymmetric logic gates favor one input over another.

- Example: suppose input A of a NAND gate is most critical.



- Select sizes so pull-up and pull-down still match inverter Place critical input closest to output

Asymmetric Gate

Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris

Now let us look at the asymmetric logic so what happens is that asymmetric logic is basically means that it tries to favor one input over another right for example let us suppose the input A of a NAND gate is the most critical gate because that is just giving you the larger delay then a you have to select the sizes so that the pull up and pull down network still match the inverter place critical input closest to the input.

Which means that you need to still match the input and output so that the output delays is exactly equals to that of the inverter right. So asymmetric gates is not used to often but just for information sake that if you have gates whose aspect ratios are quite different from each other the pull up and pull down are quite different from each other you should be very cautious and you have to put you have ensure that to get the minimum delay is to ensure that that the input which is critical is closes to the output right.

So that is very important to discuss this point earlier also that you have to keep your input closes to the output whenever this is working in a working condition right and it is sounds a proper also because this is what we are supposed to do for Asymmetric gate. Now what we will do is we will take up the logical effort of this Asymmetric gate right.

**(Refer Slide Time: 24:36)**

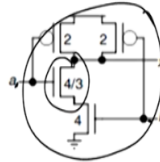
## Design Example

- ❑ Calculate logical effort for the following logic?
- ❑ Select sizes so pull-up and pull-down still match unit inverter
- ❑ Place critical input closest to output

Logical effort at input  $a = 10/9$

Logical effort at input  $b = 2$

Total logical effort =  $g(a) + g(b) = 28/9$



Logical Effort: Designing Fast CMOS Circuits Ivan E. Sutherland Bob F. Sproull David L. Harris  
&  
Internet Sources

We will select the sizes for pull up and pull down network such that it still matches the inverter unit inverter and we put the critical signal closest to the output. So logical effort here it is basically 2 input NAND so it will be 2 input NAND but no you have this into consideration so whatever you get  $10/9$  as a logical effort at input  $b$  which is this one in this case happens to be 2 because it sees to be 2 input NAND gate here.

So I get so total effort is nothing but some of these two and I get this to be equals to  $28/9$  so the  $28/9$  is the total logical effort to the system. So with this so we have therefore found out or we have actually understood using one design a example how to calculate the logical effort for the any logic given to you right how to select the value of your pull up and pull down transistors in order to match the unit inverter right and also we have seen that if you add  $g_a + g_b$  which is logical effort for A and B they might be unequal and show you different results all together.

**(Refer Slide Time: 25:47)**

## Recapitulation

- The least delay of logical circuits can be obtained by adding the Inverters.
- The best stage effort for the logic circuit will be nearly 4
- The FORK circuit can be designed using logical effort
- Delay of Asymmetric gates based logic circuits can be calculated.

So to recapitulate or to continue the whole talk for logical effort part what we seen is that least delay of logical circuits can be obtained by adding an inverters because inverters as got logical effort of 1 you see a very good result available to me the best stage effort for the logical block will be nearly 4 we have seen that it was 3.59 if you remember but it is a fabricated then closest you go is basically 4.

The Fork circuit which I have showed it you can be purely designed by logical circuits but we are not doing it currently and we have also seen that there is a delay asymmetric gate based logical circuits can be calculated so I have a big circuit available to me I can use logical effort to find the total logical effort across the path then your path delay across the path and so on and hence so forth so that will be quite useful in finding out the total delay of a block right. So I think I will stop here thank you very much.