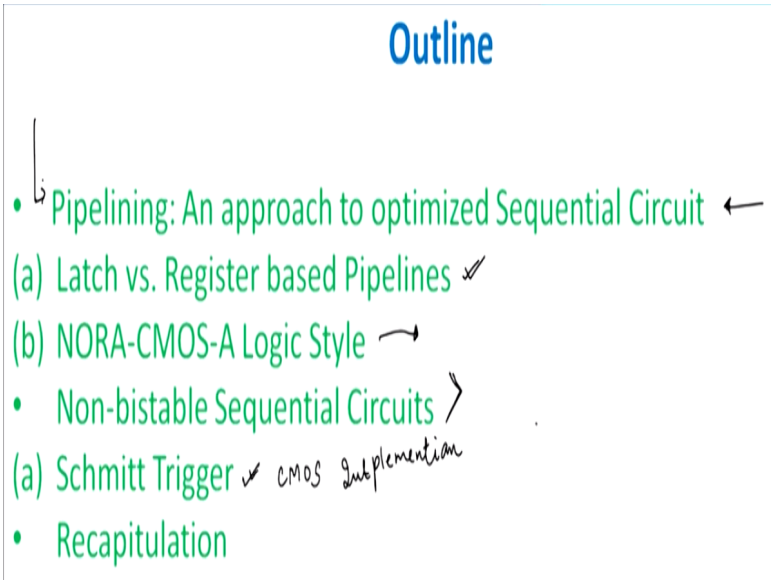
CMOS Digital VLSI Design
Prof. Sudeb Dasgupta
Department of Electronics and Communication Engineering
Indian Institute of Technology – Roorkee

Module No # 08
Lecture No # 37
Sequential Logic Design –IX

Hello and welcome to NPTEL online certification course of CMOS VLSO design today we will take up sequential logical design module number 9 and here we will be discussing certain things which happens to the pre curser whose pre cursor you have already learnt in the module of sequential design we have already learnt C2MOS we have also learnt two single phase clock design and we have seen how our through put can change with C2MOS what are the problem areas which 0, 0 and 1, 1 hold up so on and hence so forth. So we will be taking up today an the CMOS digital VLSI design we will be taking up these ideas.

(Refer Slide Time: 01:12)



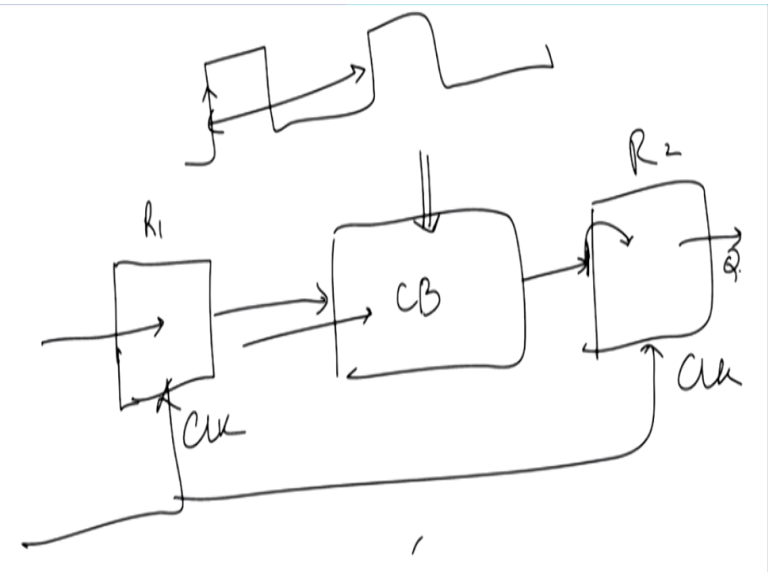
And the two ideas which will be taking up today basically the pipelining which is a very important path of design which is pipelining here and this is quite interesting in the sense that for all your heavy data centric design where you have high volumes of data to be evaluated generally people preferred to the pipeline structures and this is one of the way of optimizing sequential circuits.

Even in that will be looking into various types of lathes available as well as registers available for pipeline then we will go for your CMOS logic style and NORA CMOS logic style is a new styles developed for the pipelining purposes then we have bi-stable sequential circuits which means is that currently if you remember all your sequential elements where single is stable now we will be looking into bi-stable elements means can I two basic element point where my system is stable and I can.

I can bias the system for I can shift my system those two stable point by external triggering of pulse so in that we will be actually looking into shift trigger and a CMOS implementation of Schmitt trigger. So we look at Schmitt trigger as such and then we will be looking at CMOS implementation of Schmitt trigger right so these things will be carry forward in this module as far this module is concerned till now.

If you look very carefully what we were doing was as log as a data was available to you in input side we were able to evaluate it provided the data was able once set up time before the rising age of the clock right that and was also stable one whole time after the falling age of the clock. Now while doing so and if you remember the previous discussion of the previous modules if you have so general scheme of things as something like this.

(Refer Slide Time: 03:12)



That if you have if you have resistive one R1 here right and you have combinational block here right CB and then you have an R2 this is basically this thing and then I am feeding a clock here

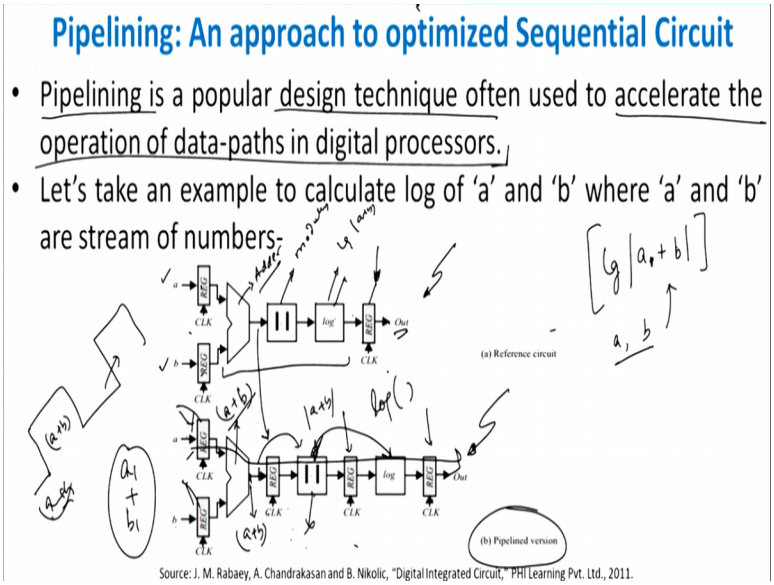
this is feeding a clock here right this is the clock also. Assuming the clock is perfect then at the first rising age of the clock right first rising age of the clock data is in certain here goes to CB because of some contamination delay of the CB this appears here if that is exactly equals to next rising age of the clock here this data is evaluated by R2 and able to get the value of Q here.

But please understand while this was happening your R1 was basically sitting ideal which means of the R2 was evaluating a data R1 was sitting ideal right and that effects the input of the system so what we are trying to do in a pipeline approaches but with varying clock cycles can I go on inserting data at a regular phase and my combinational block take care of evaluation of the resulting data for the combinational block which basically is the block available here that takes care of the evaluation of the data.

So after each clock we so what happens is that the latency is better latency is improved in this case at the cost of certain silicon area because you have to introduce certain things within the system in order to achieve such a profile. So with this knowledge or with this idea of the motivation that when you require a high true puts and your data rights input data rates are quiet high you need to convert yourself from ordinary sequential design from pipelining approach right.

So let us look at first of all the first bullet which is basically pipeline let us see how it works out.

(Refer Slide Time: 05:05)



So let us suppose I need to find out let me say log of say absolute value of A1 A + B this is the mathematical sort of approach which I want to do which means that given inputs A and B I need to evaluate log mode of + B right so this is the requirement which we require now as I discussed with you pipeline is a popular approach design approach often used to exhilarate the operation of the data paths in digital processors which means that in order to exhilarate the operations means the number of data input should increase at using pipeline.

So using pipelining architecture you can improve it let us take an example so if you see the first one this is the graph which is without your any pipelining approach whereas B if you look which is basically a pipelining version now if you look at this point inputs are a and b so with rising age of the clock this goes to the register it evaluates the sense and adder so this is basically is an adder it adds it right adds a and b then it goes to a modules which rights to find out the modular right and then it goes to the with convert this into logs the log of modular come to be a + b and then it goes to the next clock and come to register and here output available to you right.

What i need to ensure therefore is that this total delay + this delay + this delay + t set up of individual registers right must be equal to the clock period between this point and this point right. So since I have deriving from the same clock I need to ensure that the capital T of my clock because this is H trigger design capital T of the clock will ensure that will ensure that.

So what I am try to tell you is let us suppose I have this let us suppose I have a clock available with me right then if this is the first rising age of the clock where you are evaluating it then this length which is capital T should be at least equal to the total delay path then only what you will happen is that your this clock the second clock which is kept in the register here we will be able to evaluate the after the log scale as been done.

So this is the standard method of in a sequential logic how will you evaluate log mode of a + b now let me do one small thing and the small thing is let me introduce between the adder and this modulus a small another shift register and the resistor. Similarly between modulus and the locks scale another resistor is certain and finally the third resistor will be available here in spite of 1, 2, 3 resistors in a typical scenario I will have five resistor in pipelining version right and let us see how it works in that sense.

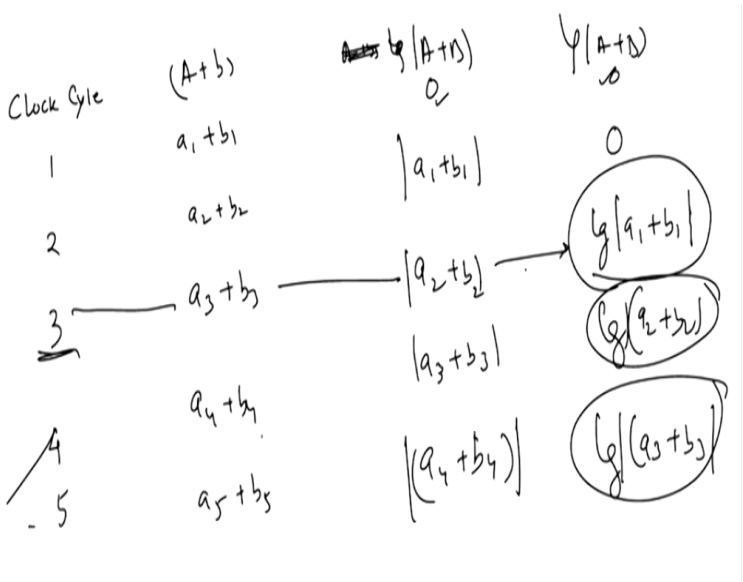
So what I am trying to tell you here is that in the first clock cycle may be A and B will get added up with the added up to this added option right and will come here in this next clock cycle. Please understand this will go to here and this become mode of a + b but then but then your this register will now be ready to accept a fresh set of a + b. Please see you do not have race condition because at no point of time this is path which is through this and this path is transparent base these resistors will not allow you for the part to be transparent.

So what happens is that when you are adding so this will be added and then you go to register here register will do a model when you transform. So in a next in the first rising age a and b are evaluated in the registers right and it goes via this through this adder and becomes a + b right. So at this age you have already a + b available now right in the next clock cycle when the rising age of the clock takes place this a + B is evaluated and going to the module counter modulus system.

And this first set of registers are now accepting new data bath so I have a1, b1 coming into picture and then a1 + b1 in mode of that now after second clock pulse. So second clock pulse what is issue that you have already done the modulus here and you have already got a + b second a + b available path is it okay. In the third clock pulse this will shift forward for the log domain associated with it in the meantime the second a + b will appear as the in the modulus part here.

So you are able to get the picture that that your combinational logical blocks in the previous case had to wait before the clock cycle was available to it but now here by inserting few register in the middle I do not have to wait for whole clock it starts to evaluate does the combinational logical block and by the time it does that does that numerically it the system is ready to accept fresh data from the outside one right. So this is the typical concept which you see if you typically look at a typical profile of this at least where you have log addition.

(Refer Slide Time: 10:52)



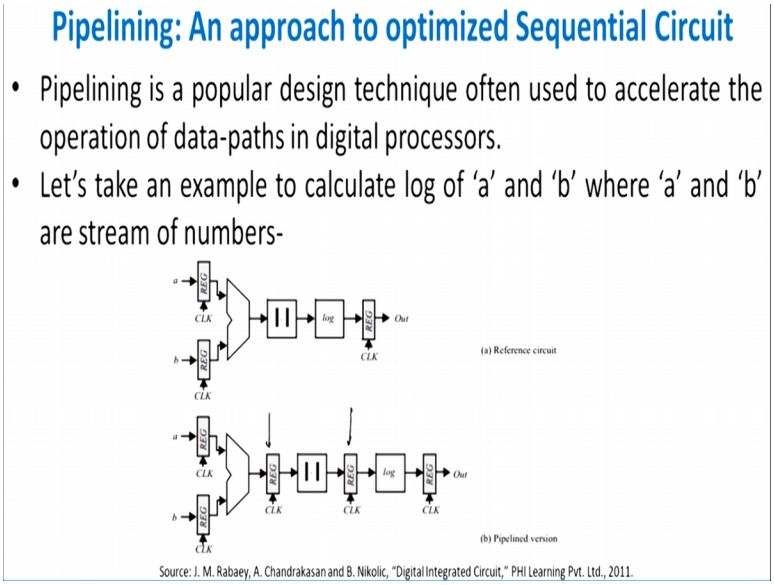
Then if I write clock clock cycle let us suppose and we refer to as say 1, 2, 3, 4, 5 and then we say this to be as adder. So a and b will be adding right so you will get a + b right then you will get log mode of a+b and then log of a + b right. So in the first clock cycle right I can get a1 + a1 right but nothing is happening at this and this in the next clock cycle this becomes a2 + b2 right is it okay but then by this time I get log of a1 + b1 right. I am sorry I get modulus of so I get I get just a minute what is get is basically modulus a and b I get a1 + b1.

But nothing is happening with third column this column is still 0 nothing is there in fact so these two are 0 actually. In the third part a3+b3 what has happened now is the a1 + b1 will shift to the log so this will be log of a1 + b1 and here I will get a2 + b2 log. So this will be a3, a4 + b4 this is a5 + b5 right. So now what will happen this at a4 + b4 you will have what you will have a3 + b3 what and then log of a2 + b2 right.

Then we will have a4 + b4 right sorry this is a1 + b1 a2 + b2 so this mode of this and then this we log of a3 + b3 right. So which means that even after three clock pulses you were able to evaluate log of a1 + b1 after fourth clock pulse log of a2 + b2 after fifth clock pulse a3 + b3 log of a3 + b3 which means that within 5 clock pulses you are able to evaluate all the three at least the three input data which you see.

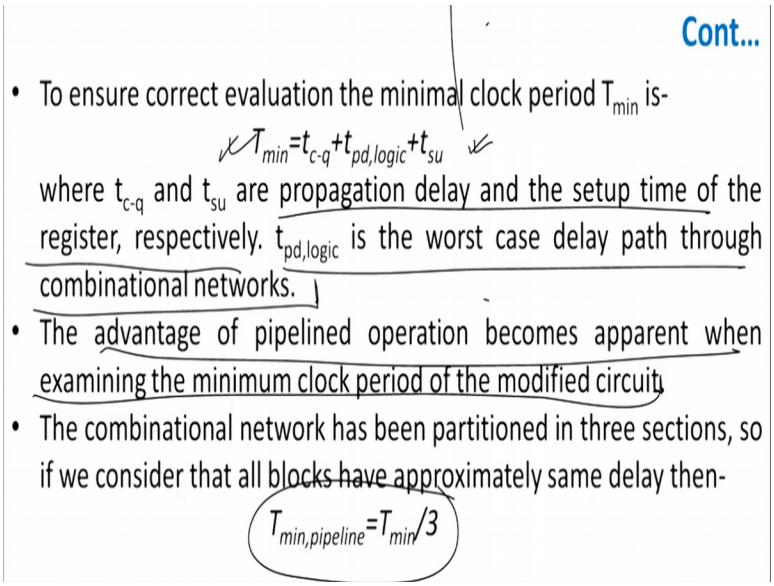
Input data which is a1b1, a2b2, a3b3 and so and hence so forth so after three clock cycles I am able to evaluate log a1 + a2 so the final evaluation available after that.

(Refer Slide Time: 13:25)



So if this is okay to you let us understand what is the idea here is then let us look what is happening here. In such a scenario or in such a case when you do have a problem with the this thing then you see at these two registers here this one and this one allows you to do allows you to do a set of blocking and permanence so it allows you to block the system and then allows you to be transparent whenever a new block is coming here and it is allows you to go to the next clock cycle right. So that is what the pipeline version is all about now to ensure correct evaluation the minimum clock period earlier as well.

(Refer Slide Time: 14:05)



Where tcq + tpd logic + tsetup. Now tcq and t setup are proportional are propagation delays and set up time of register respectively and tpd logic is the worst case delay path to the combinational logical networks. Which means that if you look very carefully sorry if you look back very carefully here and let us see what happens that a in the first case which is this one right.

In the first case and the let us compare the first with second case with pipeline and non-pipeline case if you look back then I said the tpd logic is the first case delay path with the combinational logical path now you see in the in this case the combinational logical block is some of the adder + modulus making system and a log making system right whereas so this is typically a large value right whereas in this case just adder or modulus or log scale.

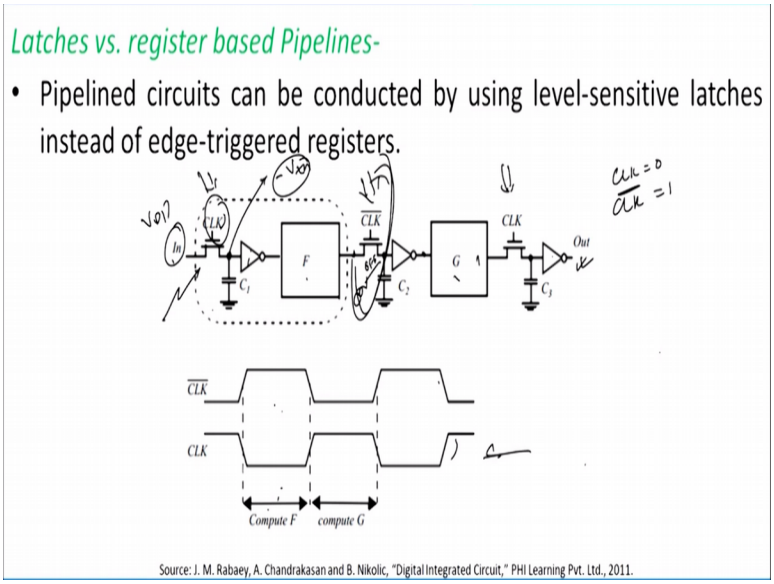
So here the total line will be larger here the total time will be shorter which means that you will be able to operate the pipeline at a much higher frequency is compared to the first case where through puts also will be high right and that is the reason we generally prefer these two be as the which we define this to be as the pipeline advantage. So therefore the advantage of pipeline or operation becomes apparent when exactly the minimum clock period of the modified circuit right.

Let us look at the minimum clock period now since your whole circuit has been divided into three parts right then my minimum pipeline period should be goes to t minimum by 3 right. t min is what t min is the minimum time period in the non-pipelined architecture for the this to be true which is this is to be true right for a non-pipelining architecture when you are pipelining you architecture since the same time is being divided into the three equals part assuming that all are equal delay then I will t min pipeline equals to t min / 3 right.

So you have reduced your minimum pipeline period which means that frequency of input will be larger and I will be able to operate at much higher frequency through puts will also larger in that case right. So that is the reason why we prefer to have a pipeline approach for most practical purposes. Now let us look at the latches versus resistors we have already discussed this point earlier but if you want to look at the pipeline issue and we want to compare latches and register let us see how it works out.

As I discussed with you registers are head sensitive design and latches are level sensitive right so whenever the goes so registers will be registering systems or transferring systems when you have 0 to 1 or 1 to 0 for positive and negative feedback respectively whereas when you are working with pipeline architecture for your latches then you are actually looking at the 1 value or 0 value. If you see here if you look at all these diagram in front of you I have a input here and this is NMOS past on register logic which is given here.

(Refer Slide Time: 17:13)



So I have an NMOS past on register logic here right and I am giving an input here so what is happening is input will come appear here with a minus VT and drop right. So if you giving VDD then this will appear VDD by Vtn it does not matter because since you are combining this with F which is the combinational block through an inverter I would expect to see the full swing up to VDD available to me right.

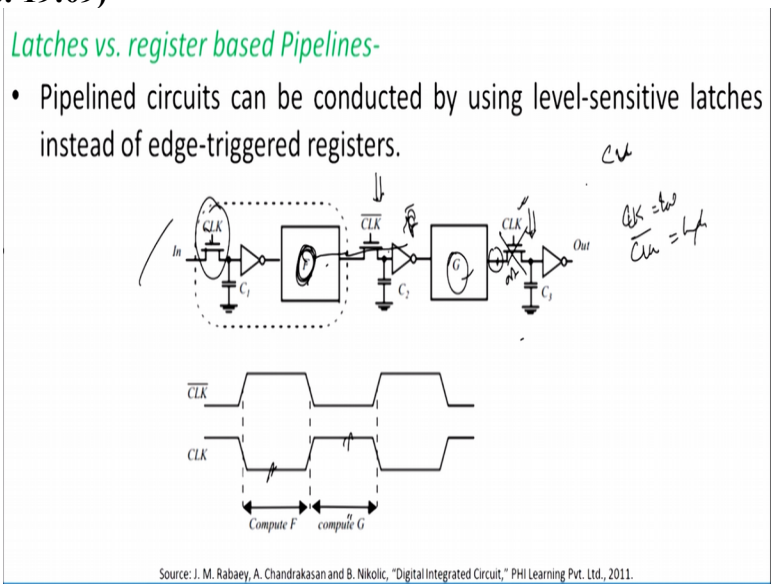
So when clock = high this is on the clock bar low means this is off right and therefore by the time it evaluates at particular point the value through the combinational logical block let us suppose clock goes to 0 and clock bar goes to 1. So clock bar goes to once means that you have a high logic which means this is open now which means this is on right and then you are able to evaluate at C2 and then reverse of that will come here and so on and hence so forth.

Similarly now but you see quite interesting that if using three dimensional clock so this is positive clock and this is negative clock right or negative clock means a complementary right

negative clock. So when the clock the is high this is also transparent which means that the previous values stored at C2 is now being able to fed into G block and you are able to get the output here.

Where as in the previous case in the case when clock = low or high my F starts to evaluate at a particular point. But my this is clock bar it does not allow you to evaluate. So there is a there is a blockage of data or there is no transparency between block F, G and output right and that is the good reason why we should use latch based design. So this is basically a latch based design and this is basically a latch based design and as you can see if you look at the second diagram here which is this one when clock bar is high and clock and therefore let me this is rub the whole thing.

(Refer Slide Time: 19:09)



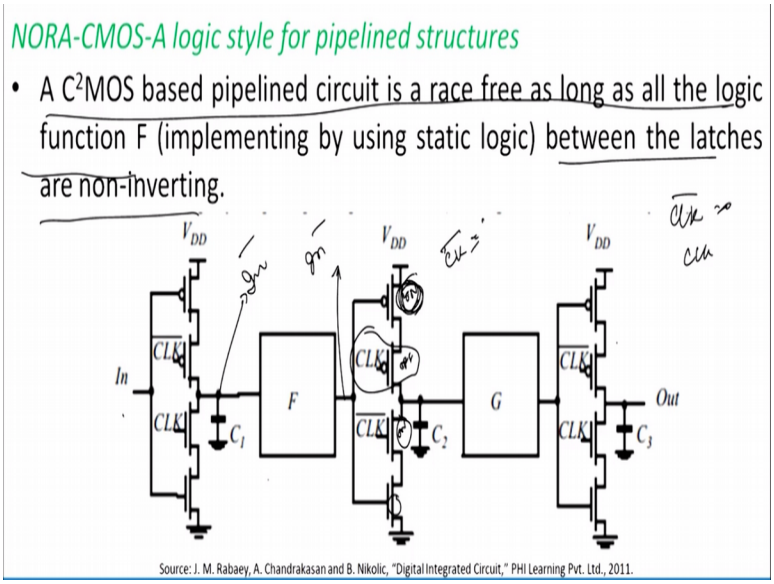
So when your clock is low and clock bar is high right clock bar is high when you compute f whatever the initial value of F is being computed here this becomes F bar let us suppose and then it does some manipulation in this combinatorial logic block or sequential logical block and it is pasted here but for what case clock = low right which means that when clock = low this means that this is high clock bar as a result sorry this will be also low which means that this is in the off state and this is hold the data for larger duration of time.

But in the second case when clock bar goes high then sorry when clock goes high let us suppose which is this one clock goes high then this becomes on this becomes on right and therefore G

starts to evaluate. So that is the reason we are able to compute G right when your so understand this logic here so when your clock is low right you are making this input OPEC to inputs.

So inputs even if input various this combinational logical block of F is not able to see it and you are also allowing since clock since clock equals to low and clock bar = high right so clock bar = high means this is high now which means that F is able to evaluate but when F is able to evaluate F bar appears here but by the time F is basically since clock bar is high clock is low which means that this is off state and your output retains the early of your (()) (20:46). So this is the basic idea of pipelining shift register or pipelining versus register a based pipelining.

(Refer Slide Time: 20:54)



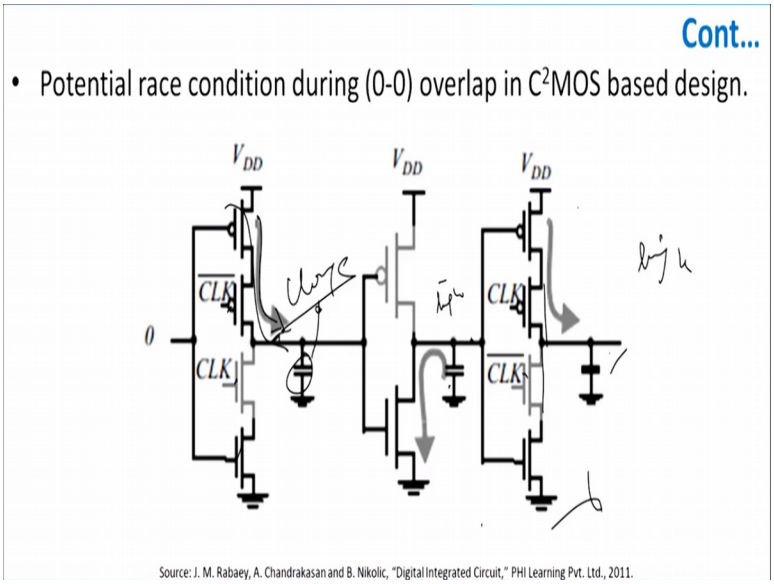
So let me come to NORA another logic which is basically NORA or CMOS this is the logic style for pipeline architecture. So whatever do here is that as long as the whole statement is that C2MOS based pipeline circuit is race free as long as all the logic functional F between latch or non-inverting this is very important but as long as the latches or the output of the latches are non-inverting we have to ensure that it is non-inverting then I could safely assume that this will be race free your circuit right.

So it will be race free circuit right and you will actually get no variation in the output as long as this race free circuit condition is removed right and I will explain to you how it works out? So let us suppose I have given a input in my clock bar = low so my clock bar = low right is low clock = high and therefore input get evaluated at C1 as input bar it is fed into F. F have does the

evaluation let us say it is input bar then input bar primarily means that this will be on state right this will be on this will be again on but we are not sure about what will happen is that your clock since clock bar is there so clock will be high which means that this will be switched off.

When this is switched off you do have the voltage so this is on state and this is off state if this is on it will try to evaluate this transistor by removing it from off state and some output will be available here which finally goes to G and then you have got this basic design for you right. Now the idea is here is that exactly like CMOS NORA style you are removing all the concepts of period overlap so on and hence so forth right. So if in overlap does not matter as long as I am able to make it race free logic function right this is what you have learnt till now in this case.

(Refer Slide Time: 23:05)



Now as I discussed with you it is 0, 0, 1 lap as I discussed with you then this is the path through which you will have a decay of your current right so current will be decaying from this path right this is on this is on obviously this and this are off and therefore this will be the path. So this will be the path and it will do what it will charge your capacitors C to the value you want to see it right.

After that it goes to a simple static invertor which converts it from 0 to 1 and vice versa to remove noise as well as to improve the peak to peak variation for us AC voltages as carried out. Then it is again fed back to the logic where it is complementary of nature so I have got clock

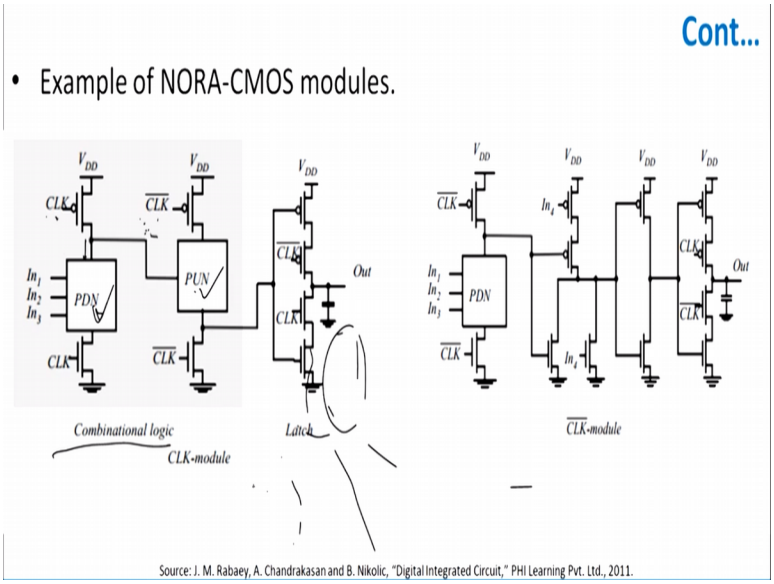
here and clock bar here so if the clock is let us suppose say it is high let us suppose and for high value of input V in let us suppose this is my clock this is V in coming from here.

Then when your clock is high and clock bar is low this is becomes on right and you will have direct connectivity to the output and also the input will have direct connectivity with the output right. So you convert into a basic inverter and then you get the output as per the basic inverter. Now if there is a 0, 0 overlap suppose let us quite interesting it will if you understood 0, 0 overlap in previous case is exactly the same thing here in this case also.

If this is 0, 0 overlap then my clock bar and clock is on state and therefore there will be finite chance that they will be a circuit path available to you between VDD and ground and you have to avoid the circuit path to practical purposes of your future purposes. So this is the disadvantage of the so if you see even if you have 0, 0 overlap I am still able to sustain a 1 in the output side by this diagram.

So what happens is the clock and clock bar are opposite so if the clock is basically 0 clock bar = 1 which means that both the transistors clock this and this are on and therefore this is the direct path of VDD to ground available to you which increases the static power dissipation of a circuit right. So this we have learnt just now and this is quite interesting as far as CMOS logic is concerned so examples of NORA CMOS logic is that you do have a clock here.

(Refer Slide Time: 25:23)



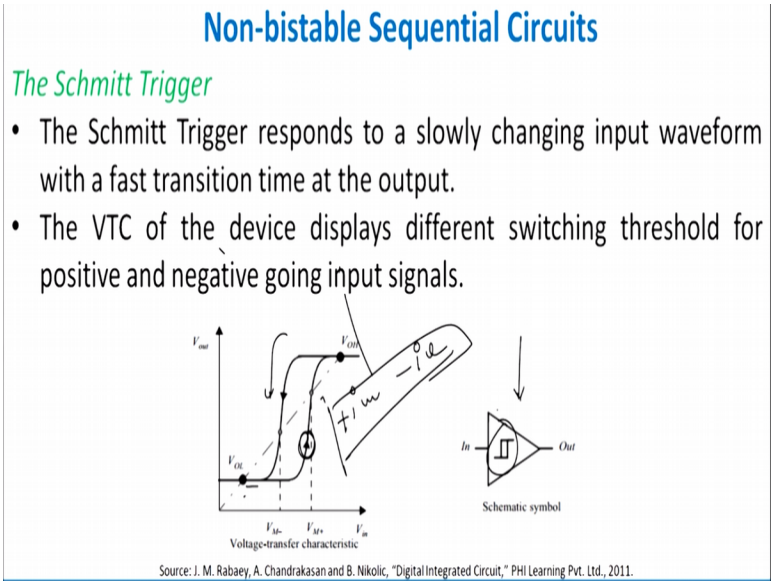
So you will look at a combinational logical block if you look so this is PDN and this is PUN which is basically your upper network then I have a clock another clock bar which is acting as a current source here and depending on the values of In1, In2, In3 PDN with either evaluating or not evaluating right. Let us suppose it evaluate the voltage here starts to fall down and it starts to fall down this PUN gets activated either In on off whatever and because of that output will vary this output goes to a third invertor wherein if you feed it to this.

So if you clock bar is low and clock is high this will be transfer end and whatever is happening here will be visible to you in the output side. So people used this quite a lot for understanding the NORA CMOS logic so that is what this is the clock module which you see this is a you can also have a clock bar module also what do you do you interchange clock to clock bar so what you do you have a clock and you have a clock you had a clock and it need a clock this clock was cascaded or added with CLK which is the clock bar.

Now by some problem of other CLK bar is not working which means this circuit is not working then at least with the previous technique available to you I will be able to achieve voltage almost equals to VDD and again use that as a exit point from circuitry right and that is what happening in this example of CMOS NORA logic and reality. Now let me discuss with you the brief idea of non by stables sequential circuits.

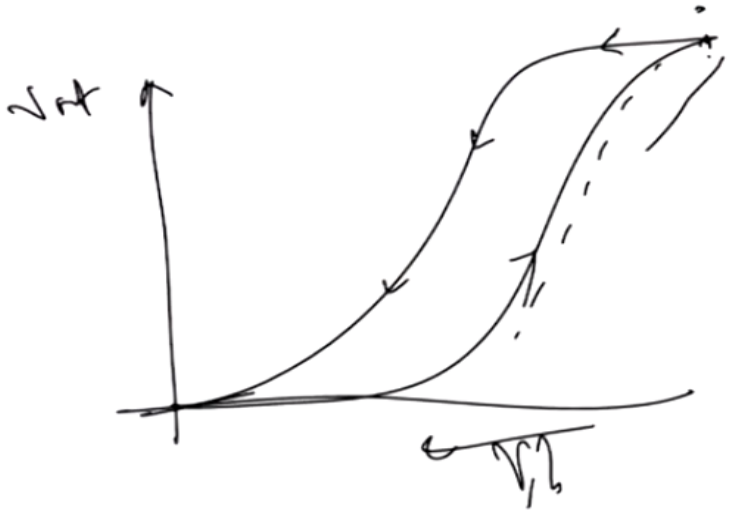
Now the smith trigger is basically a trigger which gives you an output by virtue of small change in input by the by the user.

(Refer Slide Time: 27:10)



So if you are by the user gives you a small change in the input wave form then will be corresponding to a very fast transition in the output side provided your smith trigger is there with you right and that lag of timing between input and output is quiet critically understanding the various design aspects of smith trigger. Now the advantage or to at extent disadvantage of a smith trigger is that that its V versus V out versus V in is V out.

(Refer Slide Time: 27:47)



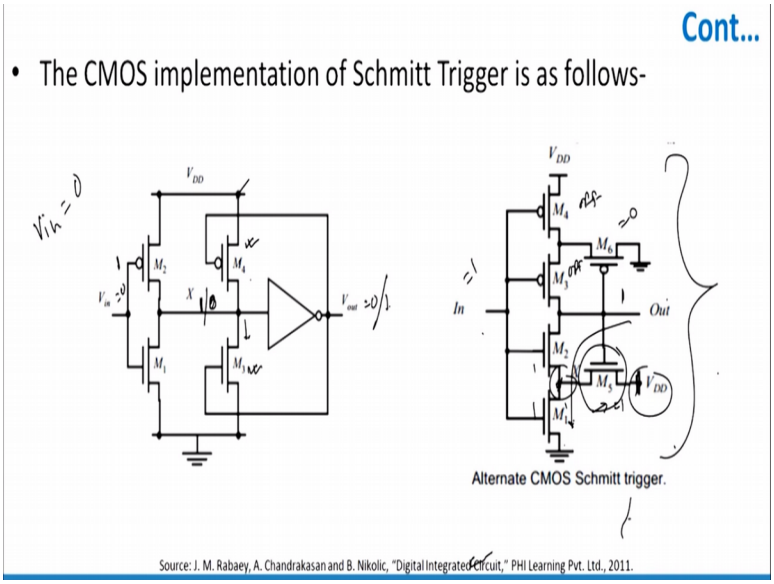
If out plot V out versus V in so this is V out and this is V in so if you plot V out versus V in what you get is something like this you get a something like this so if you go like this V in V out increase reaches to maximum value and then swing it backward it follows like this which means that it has not follow the original path through which has not come right then the reason being for

a design which is let us say when it reaches to the highest point in V in you already have all your data's parameters confirming to the point such that output is lash to the value right and that is the reason you cannot change it very often.

But typically we use thermal thermometers to find out the values of resistance is there right or temperature is there at this particular point where your V out is typically very large. Now so what I am going to tell you is that for the negative for the positive half when the voltage is rising I will have one switching threshold and the voltage is falling I will have multiple negative threshold fusing threshold right.

Which means that the rate of change of the threshold with respect to the input we also depend upon the will also depend upon the value of your device swathing threshold of the device right and that is quite interesting or quite this thing or not to. To appreciate this point we have a schematic which is in out and you have a systematic symbol is used in layout.

(Refer Slide Time 29:29)



And this is quite an interesting symbol which can be used for Schmitt trigger now CMOS implementation of smith trigger is we will discuss but we will before you move forward therefore let me let me just give you a let me give you a idea what this Schmitt trigger is all about let me just give you a let me give you a idea that Schmitt trigger is all about. So if you look back here carefully in this case I have V in here so V in is let us suppose 1 this will come out to

be 0 if you feed it into a inverter right and you this with this become 0 this is 1 right and V out will be also inverter I will get output to be equals to 1.

If right output will be equals to 1 so for every input I get output = 1 we will ask for what is the input so if it is 1 M3 switched on M3 switch is on implies that this voltage will be pull down to ground right. M3 switch is on that it will pull down to ground but if this pull down to ground the 0 becomes even stronger than 0 right. Suppose this would have been 0 and this would have been 1 then this will come out to be sorry I did a mistake I will just get back to you right.

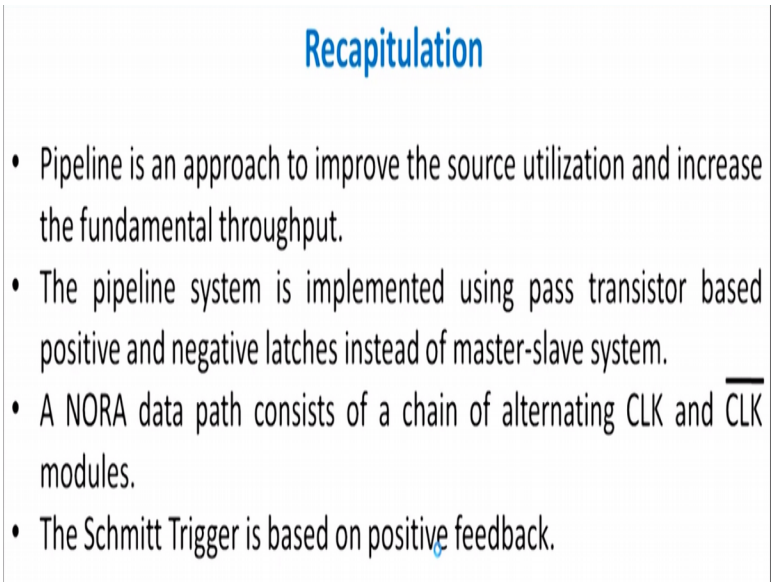
So if V in = 0 right then this X becomes 1 then this X becomes 1 V out is actually equals to 0 and then you do a feedback here. So M4 will be on right and then M3 will be off so when M4 is off there is a direct path between VDD and that particular point and therefore the voltage levels there will raise up there by charging voltage with in M4 and a charging point an whereas when you put input = 1 this becomes 0 and this becomes 1 then this 1 is fed back into M3 tries to pull it down but you already had 0 available so pulling it d has got no influence everything is that it will do it faster right.

As a result this will be must faster in this case and you are able to achieve a much higher one so this is a alternate CMOS architecture which you see in front of you and we use therefore when your input = high M1 and M2 switch is on as it switches on this input goes to 0 output goes to 0 means this value X goes to approximately = 0 and therefore VDD – 0 point is always = VDD with VDD and M5 into saturation I automatically get a definitely get the value.

So let us suppose M1 and M2 are on and if I give input = to the let us suppose this is 1 then these two are 1 so this voltages are connected to ground since this is 1 this is off this is off and this is 1 this out will have a path to discharge here and get discharge to M1. Now if let us suppose this is out = 1 primarily meaning M5 is also = 1 right then M6 is always = 0. Now when M5 = 1 I have connected to VDD it means that M5 tries to connect to VDD to this path fine.

So M5 is basically a past transistor which connects VDD to X right as it connects there will be a potential loss but that is a off shoot of this problem of Schmitt trigger right. So let me therefore give you a basic recapitulation what you have done let me give you a recapitulation.

(Refer Slide Time: 32:48)



We had learnt in this module about the pipelining approach we have used how to implement pipelining using pass transistor and a instead of master slave conventional slave approach what is the difference between negative and positive latches we have also seen a NORA data path consists of chain of alternative clock and clock bar and therefore routing clock and clock bar of course the distance is major concern. We also finally look into (()) (33:10) Schmitt trigger and saw that positive feedback I was able to achieve a much better profiling in case of a CMOS inverter design right with this I will finish of this is this module thank you very much.