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

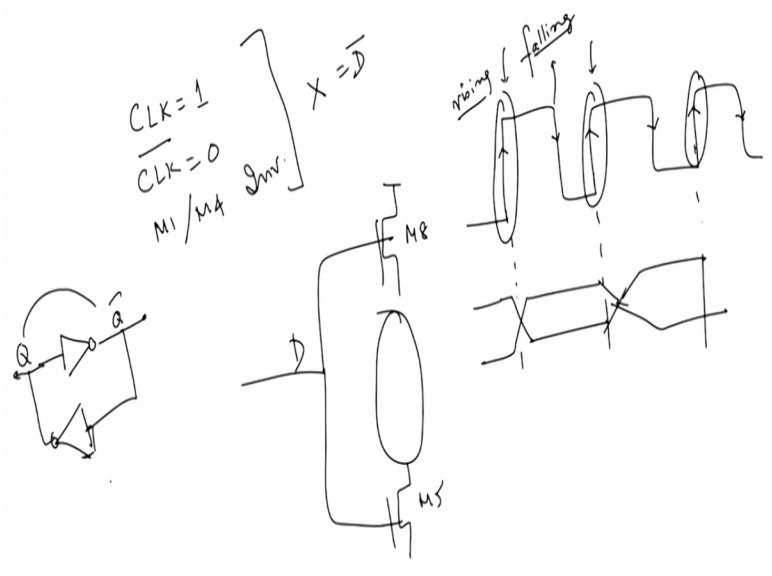
Module No # 07
Lecture No # 31
SEQUENTIAL LOGIC DESIGN-VII

Hello everybody and welcome to the NPTEL online certification course on CMOS digital VLSI design. And we will be doing this sequential logic design module number 7. In our previous lecture, we had seen that there was a problem of clock overlap, right. And we saw that if this a clock overlap there is a chance that whatever data is inserted on to the master part of the sequential logic will actually be also present at the output part which means that there is a race condition available with you which primarily means that any data which you are inserting is at the same instant of time available at the output sides. So you are not able to process the data internally.

This was possible because your clock and clock bar where both having the same values which resulted in NMOS and PMOS of 2 opposite transmission gate switching on simultaneously. And therefore the data was going from one into another and since we are using CMOS inverters as logic outputs. So the swing was actually maximize to high value. To remove that we had seen that C2MOS logic was used.

Now the C2MOS logic was such that even if there was a clock overlap, there was no way in which there is a short circuit path between the output and the ground. And therefore the logic was actually stored at the output even if there is a clock overlap available, right. So, this we have already learned through our extensive lectures in the previous turn. Now what we will do is basically look into the fact of dual edge trigger. So, till now you are actually looking at edge triggering happening at only one side.

(Refer Slide Time: 02:16)

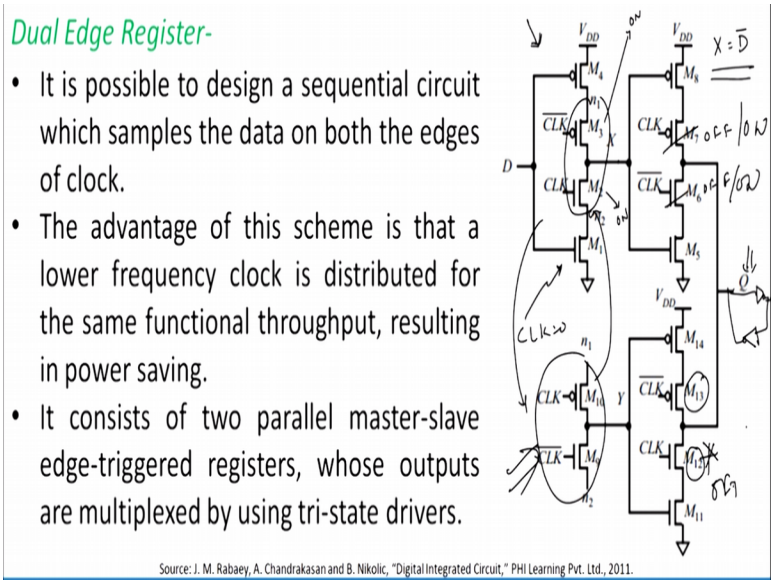


Which means that if you have a clock and the clock is this like this which is giving me a fixed value, then if it is a positive edge triggered clock then during the rising edge of the clock which is this or this you are able to sample the data. So the data was getting sample exactly at the rising edge of the clock, right. And the sampling was there provided you do not valid the set of time and whole time evaluations.

So the next sampling would take place here, right, of the next sampling will take place here, the third sampling will take place let us suppose here and so and so forth right. So what is the tell me therefore is that apart from set up type and whole time constraints. The frequency of the output is actually being governed by only these clock edges which are basically the left, the rising clock edges.

Now if I have a shift register base design allows me to sample the data in both the rising as well as of the falling edges, right. So this is the rising edge, right and this is basically your falling edge, right. So if I am able to sample my data in both the edges, the output frequency will double itself for sampling of the system. So that was the reason why people have used dual edge trigger design which you see in front of you as I discussed with you in the previous turn.

(Refer Slide Time: 03:43)



Let us suppose your clock is basically a high then M2 will be on and clock bar is low which primarily means M3 will be on. And therefore this set will be transparent and a mod X will be therefore equals to D bar which will be stored here. But if you see, if your clock is this is so what is happening in the first case is let me give an idea but if clock = 0, right. Let me again come back if clock equals to 1, then clock bar will be equals to 0.

This will imply that this first register which you see in front of you, right, will be transparent because this will be on. So M3 and M2 will be on so this will be on, right and this will be also on. As a result, whatever the value of D will be available at X with a inverted form because M1 and M4 form an inverted pair, right. So M1 and M4 form an inverter this will result in what X becoming equals to D bar fine.

With X becoming equals to D bar, but you see I have just not taken clock to be equals to 0 and clock to be equals to 1 and clock bar equals to 0. So when clock equals to 1, then M7 and M8 are off state. M7 and M6 right are in off state which means that the effective circuitry if you look at the output side you will have this something like this. So this will be M8 and this will be actually your M5 this will be M5 and this will be connected like this. So you will have be connected like this in this manner. But since your middle data path is not there you will have the output your X will always have a fix. So basically your Q will hold the previous data whatever the data is right. And the best form is that if you can put in the Q1 2 inverters connected back to back.

This will also ensure to me that there is always a you will always be storing the data over the period of time. Assuming that there are no leakage paths available to you, right, so if there are

no leakage path in the system, then I could safely assume that any data stored at node Q, will store the data as long you have a feedback path available to you. So generally what people do to do that at output Q this is let us suppose output Q they put 2 cross coupled inverters in this manner, right, with the properly W/L ratios.

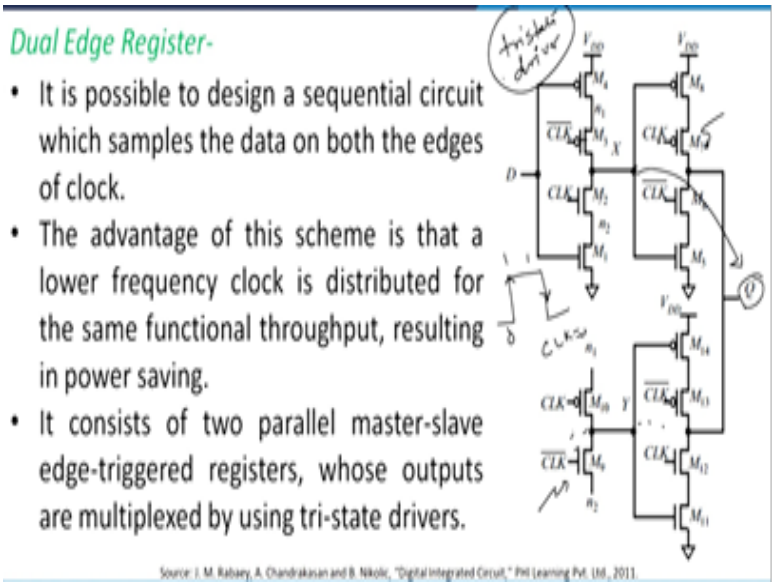
This ensures to me that whatever the value of Q is actually available to me that value, same value is stored as so this will becomes Q bar this comes out in Q so the same value is stored. It is recharged time and again and its statics CMOS implementation. It will either go to VDD or ground depending upon whether the input value is above the switching threshold or below switching threshold, respectively.

Now, so what is it tell me therefore that now what is happened with the condition that clock equals to 1 and clock bar equals to 0, our this transistor M5 to M8 transistors there all a not coming into picture and therefore Q actually stores the original value of data. Similarly what will happen is when your clock and clock bar so what I am trying to tell you is that I can just replace this right into this fine I can just remove this part and replace it by this.

So what will happen is in that case, your D will be transparent why because when clock goes to so what I was using was when clock was 1, let me see clock is equals to 0. So when clock equals to 0 if I use this a profile, then M10 becomes on clock bar is equal to 1, M9 becomes on clock bar equals clock equals to 0 means M7 is on now, right and clock bar is this means this also on and therefore D is transparent to Q and you can go there.

Also you can see here, in that case, provided the same thing is replicated here. Even if I so in that case what will happen is this will go switch off. This will go to the off state because in that case M13 and M12 will be off, right. And therefore there will be no path available to you.

(Refer Slide Time: 07:57)

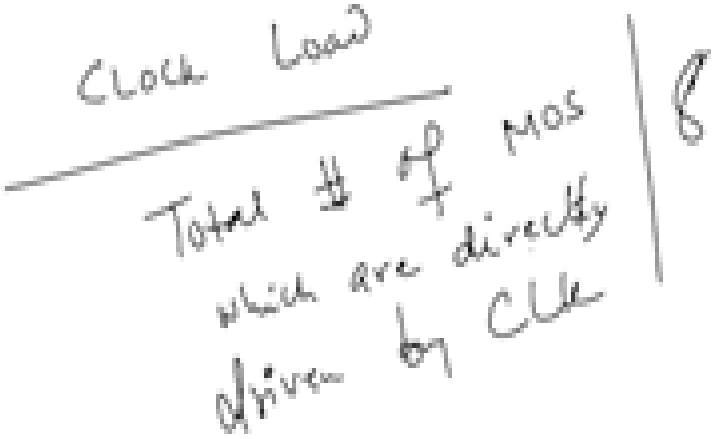


So what I am trying to tell you therefore is that if your design is such that you do have a clock here and a clock inverted register here, then I will be able to store the data at point Q in a much easier fashion and I can also make it transparent. So when the next clock cycle whatever the value of X is stored here will be transported to Q, right and that is the reason we get this type of what is known as at so this thing.

So you see both in the positive edge triggered when the clock is going from 0 to 1 and as well as from 1 to 0. In the negative edge triggered get in both the cases, the data D is actually being a going to clock Q. If when the first case, when the clock is equals to 0 and suppose it is going from 0. It was initially 0, then clock 0 implies that this will be on, right and clock 0 will M 10 and M 9 will be on and therefore this D will have this path available to it, right.

So if this is clock, see clock equals to 0 then clock bar equals to 1 what we will do is that, it will find this path into its itself. When clock goes 1 and 1 to 0 then when clock equals to 1, then M2 is on M 3 is on and therefore data D will be transparent and going to capital Q, fine. So what we have learned from this is that I will be doing therefore this is basically 2 master slave a parallel. So this is 1 master slave, this is 1 master slave latches, right. And there using tri states, these are all known as tri-state drivers. So I am using tri-state drivers and they are basically all master slave edge trigger flip-flop.

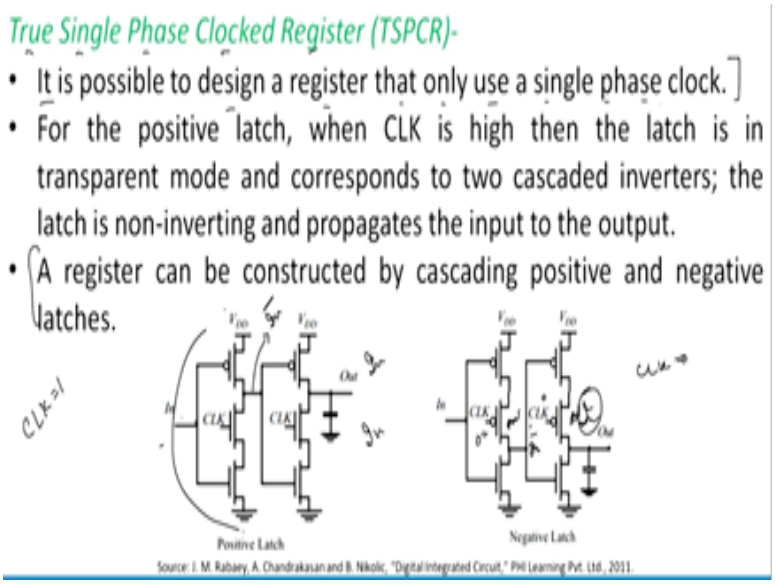
(Refer Slide Time: 09:47)



We define a new term here which is known as the clock load. Clock load is defined as total number of a MOS devices which are directly driven by clock. So if clock is driving, say for example, 8 transistor, the clock load is basically 8, right higher the clock load, more is the power dissipation because the clock has to do much more amount of work in order to drive the transistors, right. And that's the reason, the clock load varies typically 8 if you see because 1, 2, 3, 4, 5, 6, 7, 8 so M2, M3, M6, M7, M12, M13, M9 and M10 are all driven by clock, right.

I am again assuming that is all clocks and clock bars are perfectly non-overlapping in nature. So that is very important condition here. Even if there overlapping you can see yourself that this is not make your life difficult in terms of dual edge triggering the flip-flops, right. So this you keep in mind as far as tri-state, so this one is the known as the tri-state driver which you see, right. This words very fine with are almost all the system design as far as this concern.

(Refer Slide Time: 11:02)



We come to the next case that is basically known as true single phase clocked register. See the problem of the previous all discussion was that you do have a clock which is basically the clock by itself is you have to generate clock and clock bar, right. And we already know that if you want to generate clock and clock bar then you have to be very much certain that there are as minimum number of 0 0 and 1 1 overlaps. Because that we have already discussed that will that is the problem for us and to do to remove that we went C 2 MOS logic and then we went for a dual edge trigger design in this case.

So, many people have done that and quite a lot researches and come out with an idea, what is known as the true single phase clock register? So this is also known as TSP CR so that is true. So this is the word is true single phase clocked register, right. So what I am trying to say is the first one if we look very carefully, it is possible to design a register that only uses the single phase clock. So that is the only term which you should be careful about you should be able to understand.

For a positive latch this is (())(12:10) suppose the clock is high. Then this if clock is so if the clock is say the clock is equals to 1. It means that this inverter acting as an inverter, right. I will get the inverted form of inverter here. Since clock equals to 1 this will be also acting as an inverter and therefore out will be equals to in and therefore in input will be coming from this point to this point.

Similarly, if you have PMOS transistors here and if you have clock equals to 0, you can automatically give therefore this to be equals to 0 meaning this will be on state. Similarly, this is equal to this will be on. And therefore you can directly feed from this point to this point,

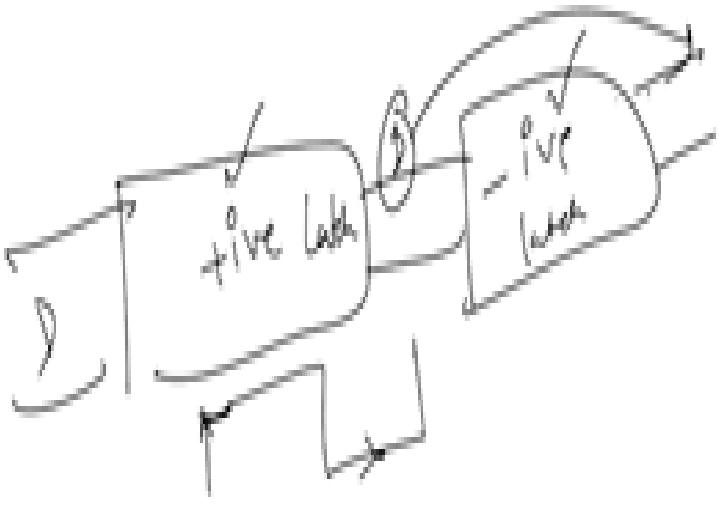
the whole value of the voltages which you see here, right. And therefore whatever input you give, you will be appearing as in bar here whatever input you give will appear as in bar here.

And this in bar will be again inverted through an static inverter and the output will be equals to in, right. So this we have already seen and this so what we do therefore is that this is a basically a latch so this is not a register, right. So this is not an edge trigger design is basically a latch where I am assuming that at high clock frequencies or at high values of clock or input clock my data will be able to the system will be able to sample may input data and store it in the output for in here at any point of time why is it important?

Because then if you want to store a data and the problem is discussed with you already are also that there are chances the data might get corrupted by virtue of its leaking. So there will be a leakage current because of sub-threshold leakage and so and so far. So you need to periodically refresh the data, the output note. And therefore this clocks helps you to do that and gives you to cascaded inverters and they give you a very proper output available to you. As I discussed with you therefore that the clock route in such a case just 2 and I (()) (14:11) with this 2 clock load of 2 I am able to therefore have this type of profile.

Now, if I want to therefore construct want to construct the register I just need 2 cascade one positive and one negative latch. So that is what is written here there are so when you see if you cascade this one so what I am trying to tell you is that.

(Refer Slide Time: 14:35)



If you have a positive latch right and it cascade that with a negative then when your input here D again the same concept exactly the previous case is that in the positive latch when the

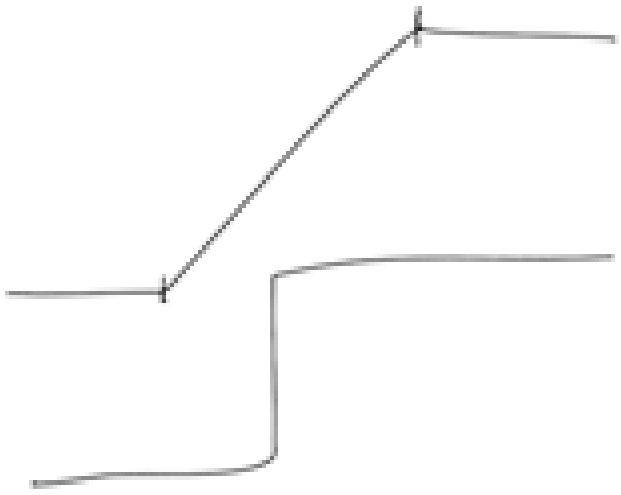
rising edge of the clock or when it is clock equals to 1. D will be appearing here but that is point this will not work because this is a negative latch right and therefore this will stop here. Now when it is fallen in the clock is fallen down.

This will become transparent, this will become off and therefore the D will be available into a in the output side, right. So I can actually cascade one positive and one negative latch and using TSPC and then make it what is known as a register, right a basically a clock register. And the problem is that whenever you look at these 2 figures where and you have 1 clock 1 positive latch and 1 negative latch driving an NMOS logic is relatively easier as compared to driving a PMOS logic.

And the reason being PMOS logics are basically volts of the carriers and therefore the mobility is typically much smaller, right. And hence driving a PMOS logic is other more difficult as compare to driving an NMOS logic which what is the therefore tell me that if you have an NMOS logic which is there in the positive latch. I can easily drive it from on to off state and vice-versa.

And therefore your switching time will be very small, right, and you have to very causes about it. This is the first very important point which you should know. The second important point is that the clock rise time and fall time should be approximately should be very low, right. Your rise time and fall time should be very low. I will give it example. Let us suppose your clock rise time is not very low. It is typically very high like that the rise time is something like this I do not have a I can show it to you.

(Refer Slide Time: 16:33)

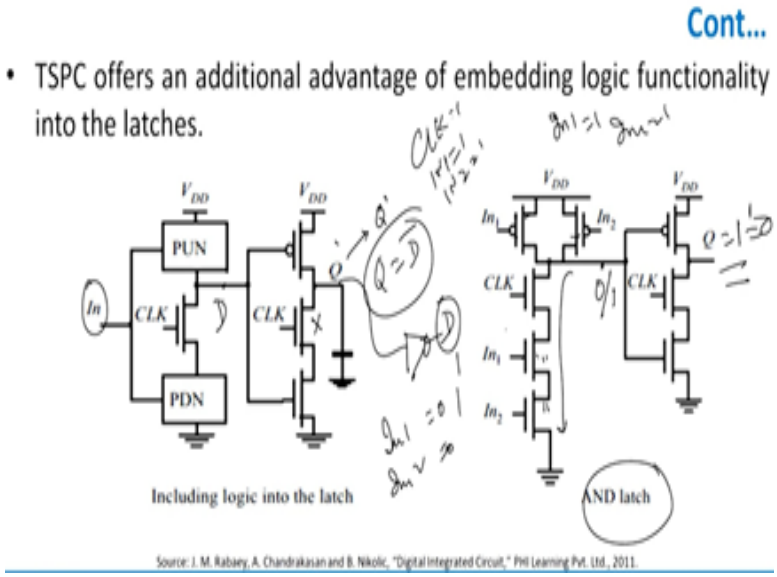


Say, the rise time is, one is rise time like this another is rise time like this. So I have typically large rise time here, right. If the rise time is large then please understand what will happen in a large case. If you allow therefore the clock to rise a for a larger duration of time. Then you are allowing the data to be sampled. Even when the rising edge of the clock is available to you, right. And therefore you will be violating the set of time a here if you have a very if you do not have a very sharp input edge, right. So your input edge should be as sharp as possible.

Typical industry regulation is that your clock should be approximately 5 times more speed as compare to the input data. That is the industry standardized people try to follow in this case. So we have discussed that point and we have also seen that we can cascade a positive and negative latch to obtain TSPCR logic available to me. So what people did was that quite interesting is that they found out that I can actually see, we remember when we were doing static CMOS logic.

We told you that I can I insert NAND gate, NOR gate, XOR gate and I can embed logic within the within the system itself. How did I do that? I made NMOS logic depending upon the Boolean expression available to you and once you have done that, this is the complementary of that will be a PMOS logic in the pull up network. So all your parallel networks in the pull down will become series networks in the pull up and all series networks in pull down become parallel network in pull up, right. With this, we were able to formulate a Boolean expression. Exactly the same thing you can do it in the latches well. So you can embed a logic functionality in the latch itself. And how do you do that is something like this like I can show it to you.

(Refer Slide Time: 18:27)



For example, you do have a this is suppose your latch, right. So this is one of your latch which is basically a negative latch and then you put input here and then depending upon the value of PUN suppose clock is equals to is high which means that this will be shorted. And therefore this will act as an inverter and in value will have an in minus or whatever by depending upon the value of the upon the logic gate a some value will be a stored here, right.

By the time, when the storing is taking place, let us suppose clock goes low. When the clock goes low, then if you look at this diagram clock goes low means this becomes off. And therefore whatever the new value, initial value of Q is being stored here, right. So you do not have any transparency available here and you are able to transfer you are not going to transfer data from point D to point Q. So Q is holding its original data. In the next phase of clock, when clock becomes again equals to 1 what happens it takes finite amount of time for this input to sample here.

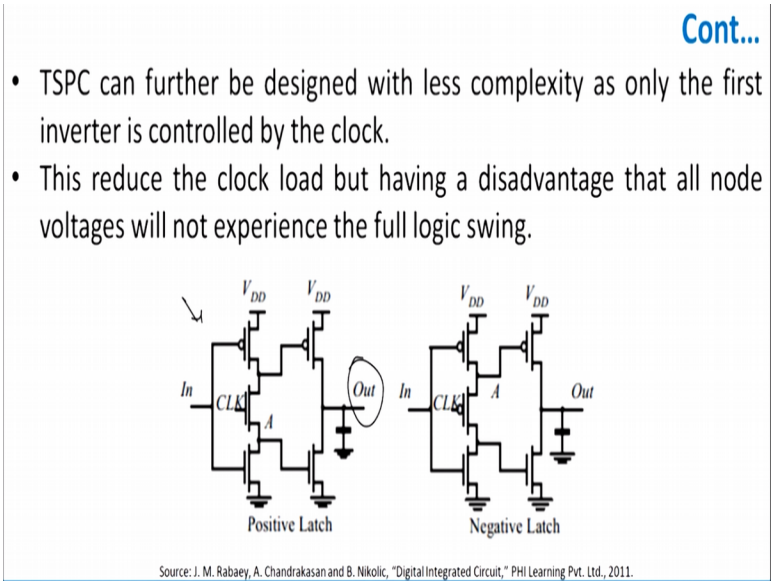
By that time, this is D will be able to make and therefore Q will be equals to D bar available to me cross it through a static inverter and you automatically get a D here. But then you to ensure that during that phase of time you do not let this network. The first network go to a like making it on. If it then make it on chances are that new value of output can appear at point D which might destroy the whole value, right.

And therefore that is the reason we should be very careful as far as designing this logics are concerned. If you look at the right hand side which is basically an AND latch then if you look very carefully in 1 in 2 are in the series and in 2 are in parallel here. So a let us suppose our I

am got in 1 = 1 in 2 = 2 1. Then you will have at = 1 at clock equals to 1 if my in = 1 and in 1 = 1, in 2 = 1. Then this network is activated and this will fall to 0. If this is 0, if clock is equals to high then Q will be always equals to 1 available to you, fine.

So I will have 1 1, I will have 1 available here, fine. Similarly if clock is low and if either of the inputs input 1 and input 2 is either of them is 0. Then this will be going high and therefore this will clock is high and therefore this will be equals to 0 and therefore I will get a pure inverter available to me. I think I am I made it clear what is happening in this case that you do have a logic functionality embedded here. I can have an NAND gate, NOR gate, I can have an exclusive OR gate all embedded in the system, right. So this is one advantage of using a DSP seen.

(Refer Slide Time: 21:27)



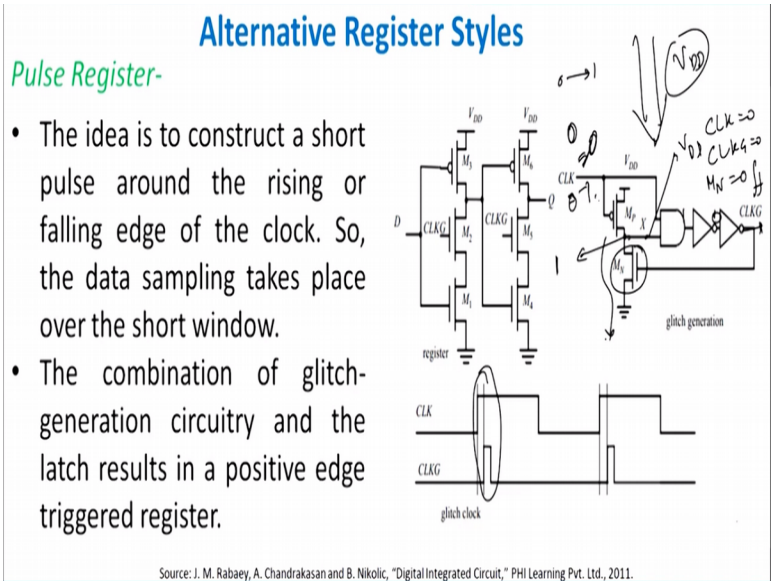
If you look very carefully, a another methodology which people have been adopting is that if you look at this is the positive latch. The first one is the positive latch, right, and the second one is pure inverter where I am feeding. For example, if clock equals to high, right, the clock is high then these 2 will be shorted and therefore I will have out equals to in bar, right. If this is not shorted in its open then the original value of out will be stored here, available here, right.

The idea here is that your clock loads are reduced because if you look at the negative latch in the second phase of the latch does not have a clock here. So clock loads are reduced and but the disadvantage is that you will not experience the full swing because you removed the

inverter from the last edge. So that peak to peak swing known be available to you in this case, right that is the basic issue.

Negative latch you can understand here again when clock equals to 0 only you will have this to be on and then you will have this A will be equals to in bar and if these 2 are shorted in A. A bar will go through this and out you will get as A right. So this we have already seen through our previous discussion and explanation also, right and we have next explain what the basic idea is in this case.

(Refer Slide Time: 22:47)



There we will finish this whole topic by using 2 registers styles here. And the first is the pulse register style so I have been using till now a master slave configuration which means that you are at master which was sampling the data, storing it and the slave as responsible for transferring the data in the output world when the master is switched off. So at no point at time you had direct contact between input and output.

Now there is also an another method by which we say that which is known as the pulse register method by which we can do the same thing as we are doing in clock register but only thing is the configuration is slightly changed. I will like you to see this fundamental work here which is basically if you look at this particular point. So let us suppose that my clock was equals to let us say 0 right. So 0 means this is an AND gate, right.

So this will be obviously 0. This will be 1, this will be 0. So if your CLK = 0 CLKG will also be equals to 0 this will switch off M N. So M N will be off state, right. When M N is off the and your MP is on X will hold D =V DD, right. Is it okay? Because this is equals to 0 and

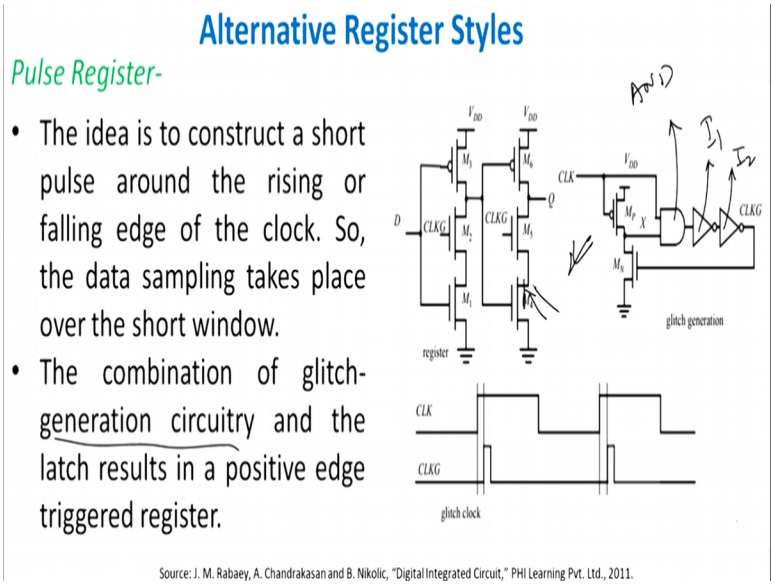
therefore clock = your M P will be on any V DD will appear at X and therefore X will be = VDD, right. It will be true.

But when clock goes from 0 to 1, right. This is the finite period of time when both the clocks might overlap and as a result what bit happen is quite interesting that when clock goes from 0 to 1, right. It goes to 1, let us suppose it goes to 1 here. This is 0 to 1 means this will cut off, right. This will mean that this X value which is at this particular point will go to 0 when the clock pulses passed.

But there will be a finite duration of time when both the clock and clock G will be high which is you see here, right. And therefore both clock and clock G will be high. This is also known as the glitch clock, right. This output is basically known as a glitch clock. Why because that if therefore what is initial equals to 1 in as so 0 o 1 it goes, right. But suppose, this M P is switched on and X was initially it takes some time to go to 0 and therefore this is actually equals to 1.

So 1 1 they will give you 1. M N will be on. M N on means it will try to remove the potential at node X right. But when you try to remove the potential at node X already your job has been done. And therefore even if the value of voltage at node X falls down you are not really worried about it. You are already extracted the voltage at the output side and your clock G is already gone high. Already remain like there because you have inverted current at node available to you, fine. This is basically the positive edge triggered because and there I will tell you the reason why is it a positive edge triggered design and the reason can be seen from this figure which you seen in front of you.

(Refer Slide Time: 25:52)

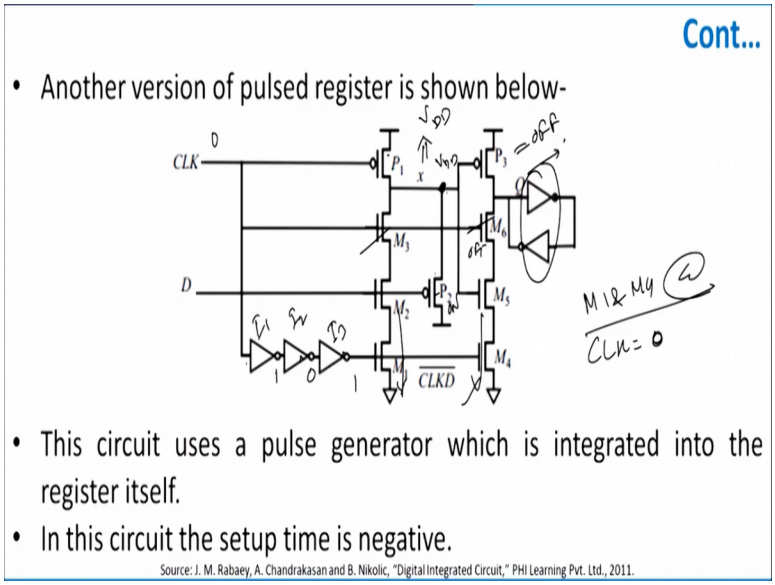


So if you look very carefully with the glitch clock is actually shifted from where from the clock and the reason being the clock has to travel through 1 AND gate, right, and 2 inverters I 1 and I 2. So the total delay for the gate 2 actually evaluate node X signal to M N will be actually sum of the delays of AND gate and 2 inverters here, right. And based on this M N will be either switched on after the certain period of time which is determined by this total sum of the delays of these 3 elements, right.

And then so what I am trying to tell you is the sampling is done at a very small window and this window opening is governed by the delays of these 3 basic elements. So the combination of glitch free or a glitch generation circuitry which is this much and the latch which is this much will result in a positive edge triggered latch, right. So think about it how is it possible? But that is what it is done that you generally I can replace a positive edge trigger design by this latch and its corresponding a glitch generation circuitry, right.

Again the cost you pay for it is basically high about dissipation because you are intentionally putting glitch into consideration. And therefore the power dissipation levels are relatively very high, right okay. Another version of pulse register is shown here. So when let us suppose when clock equals to 0 may be I do have a slide again.

(Refer Slide Time: 27:33)



When clock equals to 0 then M3 and M6 are off, right. This P 1 is on and therefore X is evaluated to VDD. X is evaluated to VDD means this point is basically equals to VDD, right. This point VDD primarily meaning that P3 is in the off state, fine and as a result, when P 3 is in the off state when clock equals to 0 I end up having nothing is happening here. So whatever the initial value of Q will be stored by this cross coupled inverter which is placed here and this store the data properly that is all.

Let us suppose clock so moreover when clock = 0, so this was 0, this will come here, this will be 1 0 1 which means that M1 will be switched on and M4 will be switched on, right, at a when clock goes from a goes is was initially goes 0. So M1 and M4 will be switched on as a result, M2 will switch on or switch off depending upon the value of D but surely these inverters will get switched on facilities.

Let me name is I 1, I 2 and I 3 and let us suppose transistor M1 and M4 all these 4 will get switched on very heavily because it sees a large voltage at the gate side of M3 and M4. Once this happens you might have a shorting of this voltage across this, I am right. No data will be lost but there will be a shorting of voltage available here. When D is equals to let us suppose low then this P2 becomes on, right. And therefore any voltage available at this point is going to the ground why are this P2 voltage, right. And that is the reason you lose voltages at a certain point of time to make it more robust.

Similarly D goes to 1 nothing will happen and therefore MN and M2 will be switched on but then previous case M2 was switched off, right. And therefore this is directly connected to VDD in this case, right. So this we have finished as far as understanding the basic pulse

register is concern, basic idea of pulse register what we will do in the next module is a look into the various aspects of this pulse register and then once you have done that we will actually discuss the last part of this talk and that is basically sense amplifier based shift register and we will recapitulate what we have done till now, fine. With this I will just take a break and thank you very much.