CMOS Digital VLSI Design

Prof. Sudeb Dasgupta
Department of Electronics & Communication Engineering
Indian Institute of Technology Roorkee

Module No # 06
Lecture No # 26
Sequential Logic Design – II

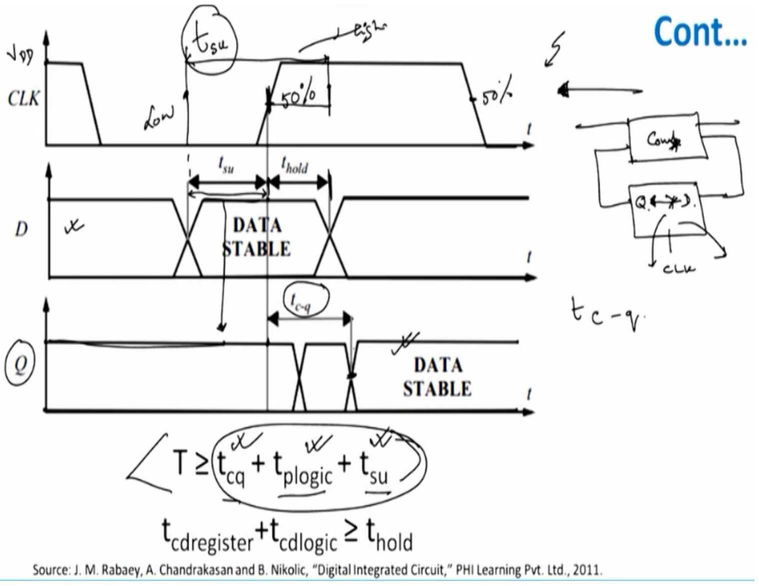
Hell everybody and welcome to the next session of the module of the NPTEL online certification course on CMOS digital VLSI design and this is the sequential design module number 2 if you remember in the previous module we had discussed the various aspects of sequential logic design and how is it different from the combinational logic and we also saw that sequential logic can be analog of memory design.

Because if you remember in this sequential logic the current state of output depends not only on the current state of input as was in the combinational logic but it also depends on the previous state of input. So you need to know the previous state and therefore in time domain you are actually trying to store a particular kind of input for a at least one clock period of time and that is where the concept of memory comes into picture and therefore there is a comeback as I discussed with you in the previous turn that you will have a combinational logic and then there will be a feedback loop connected to the combination logic through a resistor this register will have certain amount of delay.

And so the input at t = 0 at t = 1 let us suppose will be there + input at t -1 will be also there with you then the sequential logic will be able to take a call about the output. So that was what we have learnt yesterday in the previous turn. We have also learnt about certain things for example certain definitions of sequential logic for examples set up time volt time contamination delay tcq delay and so on and hence so forth we will start from where we have left in the previous turn and try to explain to you in the aspect of sequential logic right.

(Refer Slide Time: 02:17)

Cont...



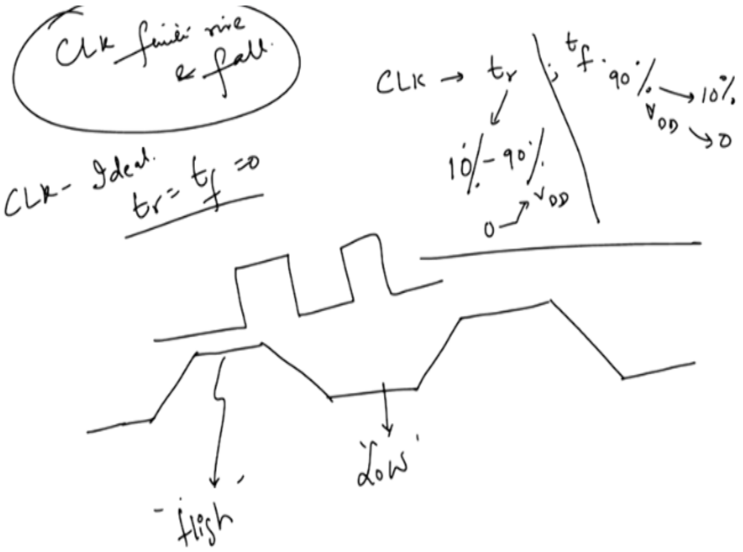
$$T \geq t_{cq} + t_{plogic} + t_{su}$$

$$t_{cdregister} + t_{cdlogic} \geq t_{hold}$$

Source: J. M. Rabaey, A. Chandrakasan and B. Nikolic, "Digital Integrated Circuit," PHI Learning Pvt. Ltd., 2011.

So as I discussed with you in the previous turn and this is the diagram which is in front of you that you see here the first one is basically the clock here the first one is the clock this one is the clock and clocks are assumed here to be having a finite it has been shown to you have finite rise in fall time right as per our earlier definition in a clock.

(Refer Slide Time: 02:38)



If you look at the clock that the earlier discussion clock will have tr and tf which is given by tf so tr is basically from 0 to 90 percent or generally 10 to 90% of our rise. So 0 to VDD right and fall is again 90 to 10 % of your so this is one of the definition of rise and fall so this is how we define for the clock and in most of the cases we will assume that you will have finite rise. So all my clocks will have finite rise and fall finite rise and fall. However If you consider the clock to be as ideal clock right then you will have almost tr = tf = 0.

So you will automatically get what you will get almost like a straight line rectangular pulses for a clock fine. Otherwise what is shown here is basically like this and it is (()) (03:34) to give you an idea about how the clock will behave in reality right. So that is the reason we have shown the clock in this manner and this is your high phase of the clock this is high phase of the clock and this is your low phase of your clock.

So I will have high and low phase of the clock available to me and depending upon the state of the clock the data will be sampled or will not be sampled. Now if you look at this diagram here the first part is basically the clock here which you see this is the clock which you see in the clock you have obviously this goes up to VDD need not to say that this is your low state and this is as your high state which is there with you right. So you have high and low state available with you.

I discuss with you the setup time is defined as that time before the rising edge of the clock when the data must be held stable right. You cannot have a data which is fluctuating near the rising edge of the clock if it is an edge triggered which is basically a register edge trigger design. Edge trigger I discussed with you that whenever the clock is in rising edge or a falling edge if the data is excepted from an input and it is evaluated we define that is a edge trigger design right or a edge trigger clock.

So when you have edge triggering which means that the edge you are excepting data then exactly few units of time before the rising edge of the clock or even the falling edge in case of the negative edge triggered your data must be held stable right that minimum amount of time is basically defined as setup time right as you can see therefore see if this is my data here. So data can be 1 or 0 whatever it is it does not matter so you see this is my approximately my clock half of the clock so this is basically 50% of rising edge of the clock this is 50% of the falling edge of the clock.

So around 50% of the clock before that this is my point before that your output should be of the data should be held stable right that is known as tsu or t setup when the clock so this will help you to except the data and keep you outside right. And therefore you see this is the Q is the output so if you remember the previous days slide if you remember I had a feedback so it is basically d to q right and I had a clock given here right and I was feeding here and then this is going into a combinational logic here and this combinational logic block was feeding like this.

So this was the combinational logic block now you see whatever the data you are feeding at d will be transferred to q with the certain delay that delay is primarily because of the input because there will be gate inside which will be actually having some parasitic delay available to it. So what I am trying to tell you therefore is that your data should be held stable at least tsu minutes or seconds nano seconds before the rising edge of the clock.

If it is held stable then only data is made available to you in the q you see so it si a rising age of the clock and you are q value is having a particular value which you see here right and your q value is here. Now I also define a new term here which is known as t hold right t hold is basically that once the rising edge of the clock has passed right or you were at the rising edge of the clock when you are doing an edge triggering hen the minimum time after the clock has based till which the data must be held sable for actual output to be evaluated is defined as must be hold.

So if you see this from here to here is basically my t hold which means my assuming that of after this point I am assuming that till this much time of the clock I have to actually have this stability condition available to me which means that this is the total time from this edge to this edge when you are data has to be held stable and you cannot do anything you cannot evaluate it if it is fluctuating very much right and this is know the prime limitation of sequential logic

That therefore the prime limitation therefore is in the speed actually and therefore you cannot allow the data to come into a sequential system at any speed you desire right you have to have a data input speed such that the minimum time for the data to be stable between tsu and t hold between the starting of tsu and the ending of s hold which is basically the holding time with this knowledge or with this idea let me explain to you what is basically known as tcq or tcq is basically the delay type.

So I define here tcq which is also defined as tc to q delay which is from here which is basically the rising edge of the clock right when your data has been held stable to a time when a time when your data actually falling down which means that from the data is again stable here right but it is falling down at this stage so you initially had your tcq is delay is this is the time when your system is evaluate your data from d to q right and this is basically your tcq.

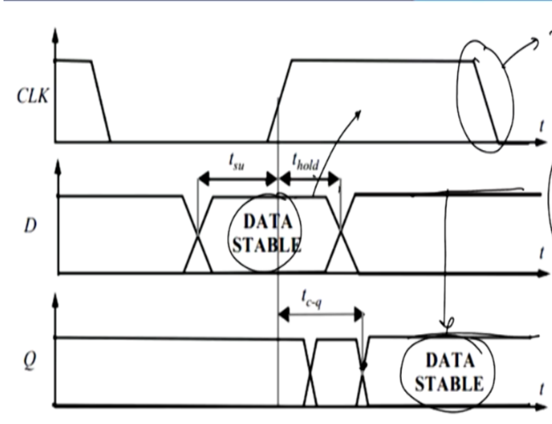
So if you look very carefully the capital T which is the time of the clock should be at least greater than tcq + tp logic + t setup I will explain to you what is tcq I mean to say. So tp logic is basically the logic of the combinational itself right because there will be some delay here also right that is some delay this delay + this delay + which is tcq this delay is combination logic and this delay is setup.

So you have to wait minimum this much amount of time before any output appears in front of you. So your clock period capital T should be at least greater than equal to at least this value right for proper evaluation. If it is less than this value by virtue of the lower tc setup delay or tcq delay or tp logic delay then you do not allow the output to be stable in the output side you do not know what will be the value of the output.

For example I will give you the example let us suppose your capital T is greater than is equal to tcq + tp logic + t setup fine but I have load my t capital T just below the value of tp logic then what will happen is the combinational see there will be a minimum time which the combinational logic cell will take for giving you a output based on its Boolean expression they will be some finite delay of the gate choose the critical path will be defined. You have to give that much amount of at least time for the combinational logic said to react right and that is reason why t should be greater than equal to this value which you see in front of you.

(Refer Slide Time: 10:07)

Cont...



$$T \geq t_{cq} + t_{plogic} + t_{su}$$

$$t_{cdregister} + t_{cdlogic} \geq t_{hold}$$

Source: J. M. Rabaey, A. Chandrakasan and B. Nikolic, "Digital Integrated Circuit," PHI Learning Pvt. Ltd., 2011.

So this is the reason why you require this one condition to be holding good. Let us look at the second condition right the second condition is after the clock accepted the data right your tcd register cd register explain to you and tcd logic which is basically the logic the Boolean expression which is kept there your basically solving it and you are getting the delay because of the logic gates this + the register in which you are storing the data.

So you have logic is upper one combinational logic delay and tcd register is basically the delay at the shift register or the register kept in the feedback loop. If you add these two delay that should be greater than or equal to old type why because if it is less than whole time then the data will fluctuate even before the whole will actually appear are you getting my point. So if you look very closely this is my whole time right this is the time when the data is to at held stable but then if this condition is not met right then you are in a problem that your data might change even before the whole time comes and therefore this equality will not be valid and your data will be spread.

So there are two basic conditions which one needs to be very careful about that t must be greater than equal to tcq + tp logic right + t setup right and tcd register right + tcd your logic must be greater than equal to t hold. If you sustain this two conditions one and two you automatically get the perfect picture of a sequential logic this the first condition which you see condition number 1is actually responsible for fixing the frequency of a sequential logic.

So people are trying to do it and they are trying to actually lower the setup time tp logic as well as tcq. Tcq is the delay between c and q in the combinational logic block and then sequential logical block and tp logic is the logic cell design which you see in front of you after this part when you have actually got this thing your data is again held stable. So your data should be stable here this is the part this is the part where you are stabling because in any case you see this is positive edge trigger so nothing is happening in negative part.

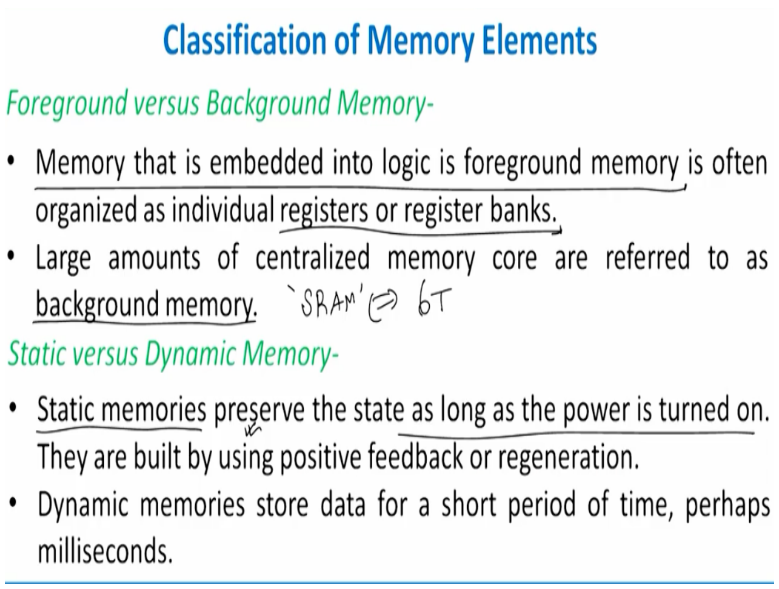
So when you see a data so in a negative edge of the clock nothing happens and if therefore data is held stable and your output is exactly falling in the input here right and when this will again happen this will again happen at least one clock delay. So when the next rising edge of the clock comes you will actually see a change available to you fine. So I wanted to stress this point as far as stability is concerned or for that matter a clock is concerned.

Now the idea here is that and that is the reason you always need to have a stable data base beyond the particular limit right and that is the limit of the frequency so the frequency of the operation long output is basically 1 / t which you get therefore that sets the operational limit of design. So once your part so I have you have understood the basic concept of tcq tc register tcd logic.

We have also understood the meaning of t setup and t holder right how they are related to the maximum frequency operation of a sequential logic right we will be giving you a numerical examples during the assignments which will be also helping you to clarify this issues in the much more manner. Let me therefore come to the classification of the memory we are already done this part that if you have if you have therefore if you look very carefully the register which you are discussing in the previous turn previous part the register which was holding this data this capital D was basically behaving as the simple register simple memory right.

So single register will be acting behaving like a simple memory design but this is basically a fore ground memory right this are all fore ground memory. So if you look at the fore ground difference between the fore ground and back ground memory.

(Refer Slide Time: 14:11)



Classification of Memory Elements

Foreground versus Background Memory-

- Memory that is embedded into logic is foreground memory is often organized as individual registers or register banks.
- Large amounts of centralized memory core are referred to as background memory. `SRAM' ⇔ 6T

Static versus Dynamic Memory-

- Static memories preserve the state as long as the power is turned on. They are built by using positive feedback or regeneration.
- Dynamic memories store data for a short period of time, perhaps milliseconds.

So you will have memory that is embedded into the logic is the fore ground memory right which means that this memory though is not primarily utilized this memory is actually utilizes logic but it also works as memory right so it is often for example individual registers or register banks right we will discuss this as we move along. For example there are 4 types of shift registers and you actually store a data till certain clock period of time for example if you doing an 8 bit shift register right then you will storing a 8 bits of data over a clock period 8 clock period of time so on hence so forth.

So you can actually do that whereas large centralized memory referred to as back ground memory for example static RAM, SRAM static random access memory these are actually based on most of them are based on 6T design we will not go into details at this stage we will do a separate module for that. But we have got fore ground memory which is primarily virtue of the registers which you are having these are embedded in the logic itself.

So you have big logic and embedded within that you have these you have these memories right and therefore they are known as fore ground memories and we have the back ground memories which are very large in dimension and they can store large bit of data and they are basically your back ground memory right okay. Now static memory preserves so let us look at this static and dynamic memory difference.

Static memory actually preserve the state as long as the power is on and so when you for example we take a shift registers you are all have the register which we are discussing till now.

you switch on the power data is just gone right that not sure in case of a memories right and in other memories for example your back ground memory right for example your SRAM cell or even that matter your hard disk available in your computer which are basically a back ground memory or even your flash memory which is available in your pen drive.

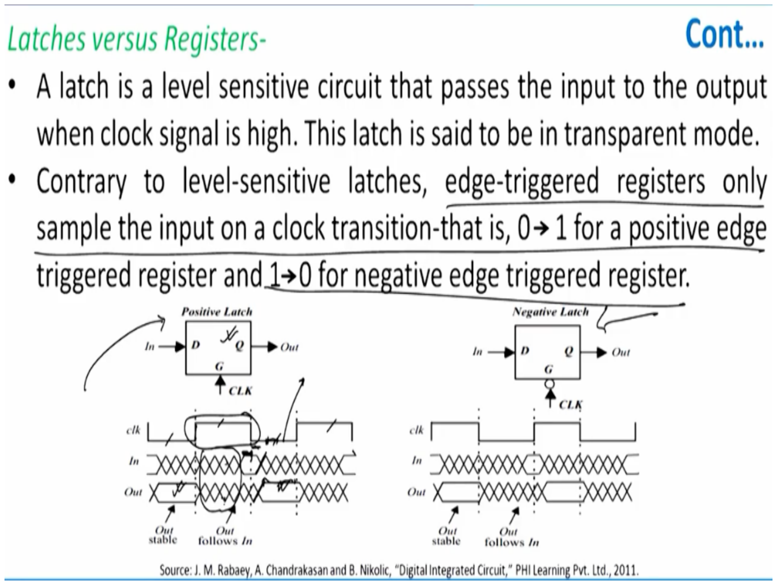
They are all memories which are basically they preserve the state even the power is off right so you must have heard a volatile memory and non-volatile memory right. So is a volatile memory means when you switch on the power everything goes off. For example you static memories are all volatile memories when they go off whereas when you talk about non-volatile memories you are talking basically of all your memory cards the whole range of hard disk portable hard disk so on and hence so forth.

So these are all static memories and hey work very fine now what is eh dynamic memory is basically a memory which stores for a small period of time and then it forgets. So irrespective of the value of the voltage available to it it does not depend on that prima-face it so dynamic memories stores data for short period of time typically the (()) (17:03) milli seconds and then they move off you will ask me where it is used they are generally using in those place where you can you need to you need to extract the data at you need to store the data for very short period of time right.

And you just have to do computation with that right and that will be quite useful when we do a dynamic memory design fine. So we have therefore understood two important points one is the fore ground and back ground memory. For ground memory is basically embedded in logic for example a shift register and back ground memory is primarily 60 SRAM cell which is primarily in the background of the system and lets stored large bits of data.

We have also understood static memory and dynamic memory right and these two are different types of memory is available to us for a practical use okay. Let me therefore before we move further give a differentiation between what is known as latches and registers right I will just important point latch is the level sensitive and register is head sensitive in terms of clock.

(Refer Slide Time: 18:08)



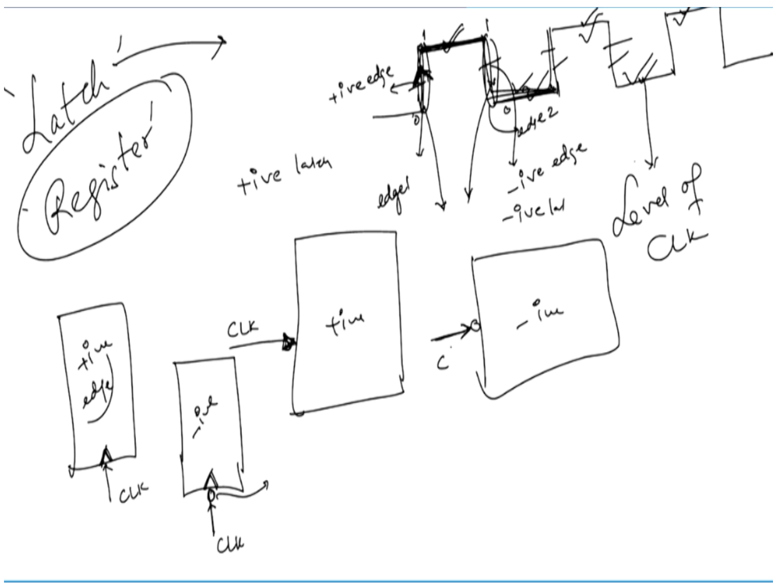
Latches versus Registers- Cont...

- A latch is a level sensitive circuit that passes the input to the output when clock signal is high. This latch is said to be in transparent mode.
- Contrary to level-sensitive latches, edge-triggered registers only sample the input on a clock transition-that is, 0→1 for a positive edge triggered register and 1→0 for negative edge triggered register.

Source: J. M. Rabaey, A. Chandrakasan and B. Nikolic, "Digital Integrated Circuit," PHI Learning Pvt. Ltd., 2011.

I explain to what do I mean by level sensitive and edge sensitive.

(Refer Slide time: 18:14)



See if you have a clock here these are synchronize design these are all asynchronize at this stage synchronize means that your data is synchronize with the clock. So whenever the clock is available that the data the system except this the data during certain periods of clock and in certain periods of clock it does not except the data then we define that to be as synchronize design with respect to the clock so the data is synchronize with respect to clock you can also have a sequential logic whether is no clock where this only might be clock the data is coming independent of the clock right and your data is just behaving like a asynchronize then the that time is defined as asynchronize design or asynchronize sequential circuit.

Let me discuss with you the concept of latch right so i have a latch here and register right so will just discuss the difference between the two and explain to you what do I mean by that you see this clock as got two parts this clock has got edge so this is known as positive edge as I discuss with you and this is actually known as right a negative edge so I have go this to be as edge 1et us this to give you an idea edge to so I have got edge 1 and edge 2 this is edge 1 and I have edge 2 here this is edge 2 right.

And you have a level here level means the clock is high this is 0 to 1 transition this is 1 to 0 transition but this is the place where the clock is high and this is place where the clock is low right it is 0 this and this are referred to as level of the clock and this and this are all referred to as edge of the clock right. So whenever your sampling the data at the edge of the clock then we define that to be as the register right.

So if you sampling the data at this point or at this point or we defined that to be as the register sampling means you are excepting the data for it is this thing for computation right. Whereas that is known as registers whereas and therefore we referred to us those as your edge trigger as I explain to you edge. Edge trigger register means are the only sample inputs of the clock transition that is 0 to 1 for the positive edge trigger and 1 to 0 for the negative am I clear?

That therefore edge triggering can have at two within the same clock period can have two can have to is basically 0 to 1 then it is positive if it is 1 to 0 it is negative. So if do like this and this I can have actually 1 edge triggering done her 1, 2, 3 right so it is single clock period I can actually do a once I can do a edge triggering whereas what is level triggering and they are in register level triggering is when your output levels are either fixed to high or low at that point of time the sample the data is sampled inside then we define that to be as latch right.

So let me give you difference let me give you what I mean to say latch is a level sensitive circuit that passes the input to the output when the clock signal is high then the latch is said to the transparent mode which means that when the clock signal is high I the latch is able to transmit my data from input to output we define that to be as the level sensitive design primarily a latch right you get also pass it at low values of 0.

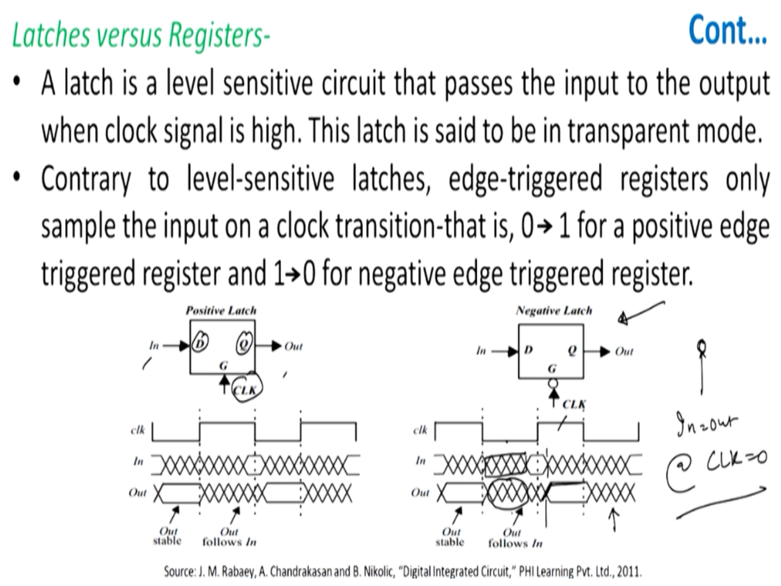
So do not worry even it 1 you can pass if it is 0 if it passes do not worry about that but that is a latch what is a edge sensitive this is what you get as edge sensitive therefore it is a register. I will give you difference between a positive latch and negative latch so if you look at the diagram this is my positive and this is my negative latch positive latch is latch which allows the data to be transparent or allows the latch to be transparent when the clock is high. So you see you have a clock cycle like and then it goes like this then and then this is it okay.

So when the clock is when the clock is high this is the point when the clock is high if you very carefully see then your output exactly follows the input you see the output exactly follows if you look from this point onwards and if you see look at this point the output exactly follows the input and your latches exactly transparent to the clock so when the clock is high I define this to be as the positive latch fine but then you have to also ensure by my previous definitions that there will be some setup time before which the data edge must be held good and then will be whole time before after the pulse which will allow to hold the clock.

So that is the reason your input as to stable before the falling edge of the clock or even before the if it is a basically if you doing a negative edge triggering you will have these being done right. Now if you if you when the clock when the latch is not transparent what it will do is the output will be held stable why it will be held stable? Because it is not depend upon the data which is coming in whatever the initial value of data was initially stored before which means that whatever the data it was storing here right sorry.

So when the clock is low in this case you have a positive latch therefore with the clock is low your system is not transparent and therefore the input is not equal to output but the output holds the last state of your transparency which means that suppose it went high right as it went high right because of that increase in input here after this you have already crossed to a place where your clock is low where your clock is low the data is held stable. So the last value of your the previous cycle will be actually stored as the fixed value of the next cycle when the clock latches low.

(Refer Slide Time: 24:47)



Latches versus Registers- Cont...

- A latch is a level sensitive circuit that passes the input to the output when clock signal is high. This latch is said to be in transparent mode.
- Contrary to level-sensitive latches, edge-triggered registers only sample the input on a clock transition-that is, 0→ 1 for a positive edge triggered register and 1→0 for negative edge triggered register.

Source: J. M. Rabaey, A. Chandrakasan and B. Nikolic, "Digital Integrated Circuit," PHI Learning Pvt. Ltd., 2011.

So this is the case when you have a positive latch when you have D I have a D right and I have a I have D here right I have Q this is and I have a clock here which I am giving here so when the input and output right then let us look at the negative latch right negative latch is in front of you in this diagram and the right hand side of your picture whenever we want to show a negative latch we generally give a circle in front of clock right.

So please understand whenever you want to do I will just give you certain things as important thing to at this stage will be helpful for you in the longer run but if you have a latch and you are driving latch using the gate by using a clock then you just simply write CLK so if you do not write anything here it is basically a positive latch right but if you want to show a negative latch then you need to give you a clock but you need ensure that your clock is ending in a ball.

So whenever you show like this this basically a negative latch so this is a negative latch and this is a positive latch so this is circle here. When you do an edge triggering and if you want to give a positive clock then you give a positive clock here and you just do not say things CLK but then you give a small this is the concept of edge triggering will give a small notch there and if you want to do a negative edge triggering this is the positive edge triggering if you do a negative edge triggering then you have to show that there will be this this but this will be terminating so this is a clock.

So if you have a circle like this you consider that to be as negative clock and if you have you do not have a circle then it is a positive but if you have a triangle like this it is edge trigger nothing it is latch so you just have to look at the diagram and see whether one is negative edge trigger or positive edge trigger and how does it work out right. So let us look at the negative edge trigger here in the negative latch as I discussed in the negative half of my clock when the clock is in the negative half your system is transparent.

So this is the negative half of the clock and therefore output here exactly follows the input here right so this is equal to this so in equals to out at clock = 0 and that is what is happening here so this part is exactly the same and then when the clock goes high your data is held stable to the initial value of the input and it is held stable right. So the initial value is held stable in negative latch similarly have in the positive latch.

So we have understood what is edge trigger positive edge trigger we have also understood what is negative latch then positive latch right we have understood the different between the two latches and how does it work as such in these two latching domain this we will take up in the next module at this stage I thank you for the patient hearing thank you.