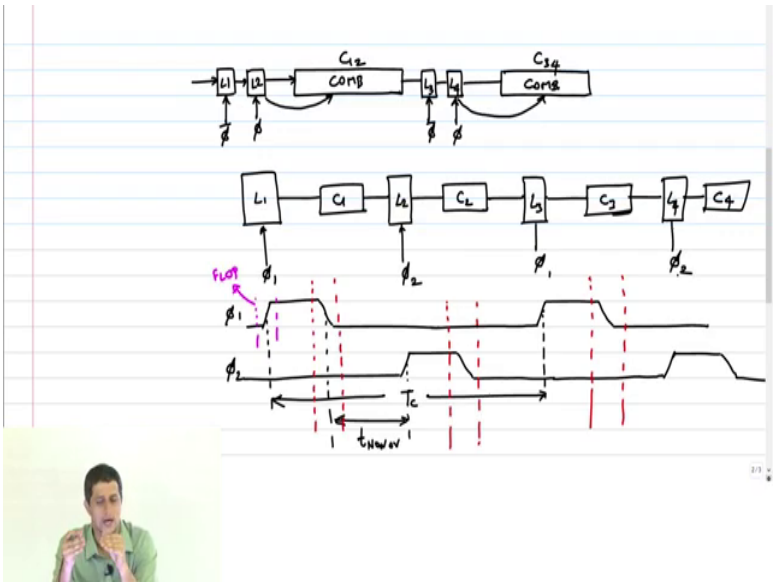
Digital IC Design
Prof. Janakiraman Viraraghavan
Department of Electrical Engineering
Indian Institute of Technology, Madras

Lecture - 76
Latch - Max and Min Delay Constraints

(Refer Slide Time: 00:17)



So, let us now proceed to see how we can make this even faster right by exploiting certain design considerations of the flop. So, if you look at the flop, it is nothing but two latches back to back, right. So, if you look at that. So, this was phi and this was phi bar right or if you want a positive thing, then this would be like this. So, latch L 1, L 2, L 1, L 2 ok.

The issue with the flop is the edge is very hard; if you miss setting up the data before that rising edge, then you have lost that cycle, you dropped a break and it is gone, right. On the other hand all I have to do is to move this latch somewhere in between here; move this latch

there is another combinational circuit after this, right. I have to just split the combination circuit and move the latch around like this.

The advantage with the latch is, I do not have to set up the data before the rising edge. It is enough if I set up the data or not rising edge I use a sampling edge, right. I have to set up the data only before the sampling edge which is a full half cycle away almost; if there is no you know, if there is basically zero overlap right and if the duty cycle is 50 percent then it is a full half cycle away.

So, I get t c by 2 time approximately to you know perform more computation ok; but of course, because of the problem of. So, this is phi bar and phi again right; this is a combinational block, right. So, the idea here is now to take this and split it as follows; L 1 running let us say on phi ok. Now do not worry about whether it is a positive latch, negative latch; I am going to split this combinational block at c 1, then insert my L 2 here ok.

And maybe I should call this as L 3 and L 4, L 2, c 2, right. So, this is combinational block c 1, 2 this is c 3 4. L 3 this is phi bar phi, then you get c 3, L 4 and c 4, right this again phi bar. So, what you have done is you split the combination block c 1 2 into two blocks c 1 and c 2 such that the delays add up. T p d of combinational block c 1 2 is t p d of c 1 plus t p d of c 2; similarly for c 3 4 ok.

Now we will analyze what the timing constraints are for this system and we will see how much of a gain we are able to get ok. Now as I told you earlier in the when we discussed the whole time violation right, this overlapping clocks is a big problem. So, people generally try not to use just clock and clock bar; because inevitably the two edges will overlap at some point and you have a violation.

So, the best thing to do is to generate what is known as non overlapping clocks; two clocks phi 1 and phi 2 remember, see there is nothing sacrosanct about having the 50 percent duty cycle. I can change my clock, so that it has a very small; it has to use, basically I have to be able to sample that thing in that set up that hold window, it should be greater than that, that is

all right. So, you can make the duty cycle very small and have two phases such that there is no overlap at all; which means that, there can be no raise condition that happens there, right.

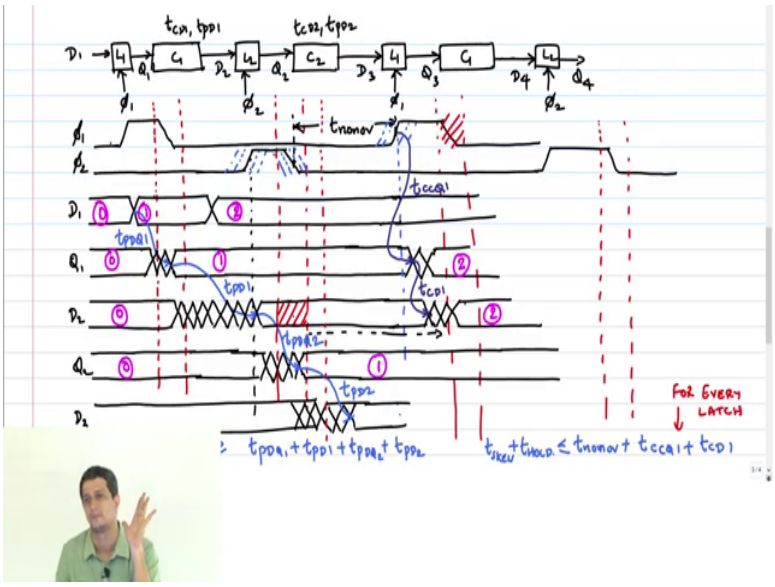
So, therefore, what people do is they convert this instead of phi and phi bar; they call it phi 1, phi 2, phi 1, phi 2. How do the timing look? You just get something like this, phi 1; phi 2 is phi 2. What is the clock period? It is basically from the rising edge of phi 1 to the next rising edge of phi 1, this is T c right. And this period is basically the t non-overlap; obviously, because the clocks are not overlapping there ok.

So, just to remind you that set up hold window in the red that we have to draw is actually here, for both phi 1 and phi 2; because by default this a latch L 1 L that I draw is a positive latch. If I want to make it a negative latch, I put a circle here; then the clock will get inverted to the because of the negative latch, ok.

So, anyway if I do not put the circle, it means it is a positive latch right. And these again this will be my set up hold window where data should not change, right. So, the point I was trying to make is; earlier we were the our constraint was the set up hold window was here for a flop, right. For flop this is the window; that means, data has to be set up before this rising edge itself. Now I have time till the clock falls, because that is the sampling edge for a latch, technically.

So, I have that much extra time to do something ok. And you can do many interesting things. So, we will first look at what the timing constraints are for such a system. Just like the previous case, we will try to draw it for both max and min delay in the same waveform, one after the other; then hopefully we will get a clearer picture of what is going to happen.

(Refer Slide Time: 08:02)



So, I will also draw this a little small first, this is my latch L 1 with clock phi 1, right. So, this is combinational block t c d 1, t p d, t c d 1 comma t p d 1 with the combinational block c 1; then I have L 2 which is controlled by clock phi 2 ok, then I have combinational block c 2, then going to back to latch L 3 controlled by phi 1 again, ok. And then you have the following block and all that; this is c 3 and so on ok.

So, first let us draw the two clocks right; this is going they are going to be 0 all the way until next phi 1. Similarly phi 2 can go high here and this is L1, L2 and somewhere here you will have the next L 4 controlled by phi 2 again ok. So, now, let us look at D 1, Q 1, D 2, Q 2, D 3 and Q 3 and D 4 and Q 4, then I do not think we need D 4, Q 4 and all that, but let us just look at this ok.

So, D 1. So, first thing, I am now trying to analyze my max delay constraint right; what is the worst case thing. So, what is the worst case; is it that data changes before this edge itself here, somewhere here or can the data change inside the stable state of the clock. It can change inside and that is really the worst case; because I eat up some time and I still have to achieve this state, right. If you make the data change before the rising edge, then you have lot more time to finish your thing, ok.

So, let us also before that draw our violation window, this is my violation window for L for phi 1 right for L 2 again, this is my violation window, ok. So, what happens is, D 1 is now changing somewhere inside here right and this is my event 1; that has to be sampled and then held across right. This is not event 1 time sorry, this is event 1; you add some other data, this is event 0. If you do not know what it is, so let us not worry about it; we change this to event 1.

Every element has to now operate on event 1, ok. So, what will happen to Q 1? First of all what is the delay that will be involved; for latch L 1 data has changed after the clock went high t D Q by t c D Q or t p D Q, right correct. So, this line will start changing somewhere here right Q 1 and this delay is what t p D Q 1 for latch 1, right. Q 1 has changed; what happens to c 1 now. So, usual thing, it is just a combinational block t c d 1, t c d 1 and t p d 1 is the delay that is going to be incurred now.

So, the moment this line starts changing, you can have a very quick change ok. Let us, again what is the worst case? Should D 2 settle before the phi 2 arises or can it go inside? It can go inside right; same logic that you apply for L 1 you also applied to L 2 right; therefore, this change is going to take it right in, ok. And this delay is what t propagation delay for block 1, t c d 1, t p d 1 correct; and this block is what t c d 2, t p d 2 right.

Now what happens to Q 2 after that? Sorry Q 2; at what point will Q 2 start changing, can it start changing here? It cannot, because the clock has not it then L 2 is opaque; it is holding the previous state, right. It is holding event, which event here; holding event 0 here right, even this was holding event 0, correct.

So, therefore, this line will wait all the way till the clock rises, right. Clock will rise and then you will see some change and what is that delay, what is the contamination delay there? I think that is very interesting, what is the contamination delay there? T c c q, because only after the clock t raises will that thing go through, right. So, you will start seeing some change here ok, we will come to that thing and it will settle after sorry this one.

What is this delay? T, what p D Q 2, right. Now we will assume just one thing here ok; in the next cycle right I want to just finish before this edge, I will tell you why. There are two cases ok; some cases you can go into the next cycle also that is called time borrowing fine; but there are certain places. So, essentially what I am saying is, you can borrow within a half cycle right; you can borrow time from phi 2 under all conditions, but to borrow time from the neighboring from the next cycle across cycles you have some constraints, you cannot do it in all systems ok. We will look at examples for that.

So, let us assume that we got to finish by this particular time. So, what will happen to D 3, what will happen to D 3? So, it is going to basically go like this right and some contamination delay and then it will settle to its proper state, right. Now since I am assuming that this data change is happening before the rising edge of phi 1 itself, there is no set up time problem; it can happen just at that clock edge and it can still go through proper, because the setup time really is for the next gap right. So, there is no issue there. So, do not worry about that ok.

So, now, what is the timing constraint that I can write, ok? What is this delay t c or t p D 2 correct. So, what is the timing constraint I can right now on the clock? This entire thing should finish in one clock cycle. How far can this edge move, if I do not want to borrow time from the next cycle? It should finish exactly at t c, right. So, therefore, I am just go now I am going to write the sum of all of this, t p D Q 1 plus t p D 1 plus t p D Q 2 plus t D Q plus t p D 2; what is the inequality? T c should be greater than or equal to this, right; this is my max delay ok.

Now let us try to analyze the min delay constraint, ok. What is the, when can D 1 change first of all? After the correct, the time is not t c. So, again the assumption is this thing changes just

at the clock edge. If you do not want to borrow time, but it is t p D Q 1; that is all the delay is need to D Q 2 not c 2 Q that is the only thing.

So, it is just at the boundary for long ok, which one no, there is no negative, this is not a flip flop, it is the latch. Latches are by default I told you positive latches; when the clock is 1 all these latches conduct, clock should be 1 for the latch to be transparent that is all. It is not a negative latch; I told you it if I want a negative latch I have to put a bubble at the clock only then it becomes a negative latch.

Rising t p D 2 can change, it can actually change you know later. But now I am just sorry I am just imposing a constraint that I wanted to finish in that cycle. Because there are cases later where I cannot do this time borrowing from the neighboring cycle; technically there is nothing that stops me from making t p D 2 go all the way till this boundary. I can take it all the way till here; as long as that happens, it will still function correctly, because that latch can sample it and hold it at perfectly, no issue at all.

But I am just imposing the constraint saying, right now I do not want time borrowing from neighboring cycles. What we are doing here by allowing D 2 to go into the you know level 1 of phi 2 is we are time borrowing from the half cycle, which is ok; which can be done under all conditions. We will see that example where it is not possible, where it is possible; then you will really understand, ok.

So, now let us work under these constraints; No, that is what I said look at where my violation window is. Violation window is on the falling edge. The latch samples when the clock falls, not when the clock rises. There is no set-up time constraint here is what I am trying to say, because the set up time is actually on the falling edge. You see this is my window, this is my violation window, set up hold time window; it is not here, so that is the key point about the latch.

You have the on cycle to actually finish some more computation ok. L 1 and L 3 are exactly the same assuming that there is no skew between the two clocks there.

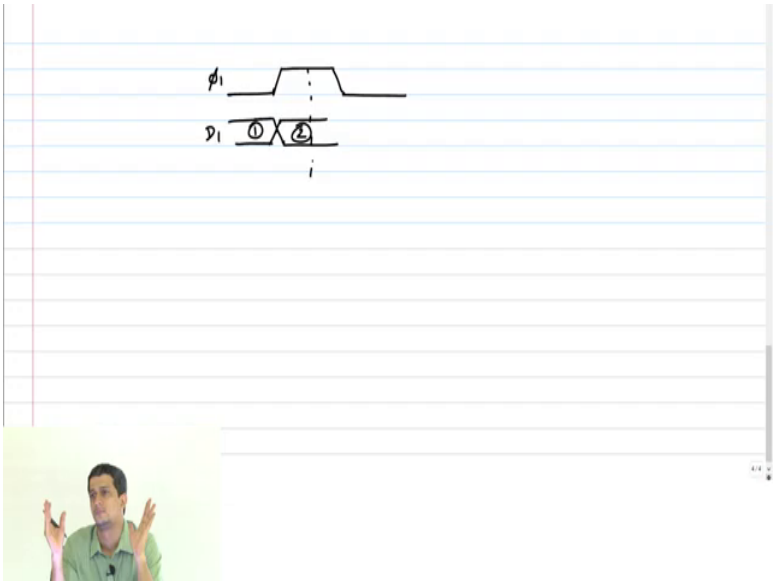
Student: (Refer Time: 25:08).

Well if you are under at least under ideal conditions they are the same, by design they are the same.

Student: Variation.

It can that is what is clock skew and other things, variations will be there. No, it can be due to other reasons also; but yeah by design they are expected to be the same. Yeah Balaji you had a question yes, after no that is what I am saying, because the latch is becoming transparent after that.

(Refer Slide Time: 26:05)



See let us assume right, let us assume that the change in a latch ok; my clock is here, let us assume that the data changed exactly here ok. This one is D 1, this is D 2, clock phi 1 ok. I will not say let us say D 1; event 1, event 2. What is the aim ultimately, when the latch falls, when the lock falls, it should hold which data; 2 that will happen right. Whether the data changes here or the data changes here, it does not matter; it has to hold, the latch has to hold the data for the remaining part.

And if that data is held correctly, sorry there is no problem; that is why in the latch you should not, that is why I said even before I started, mark your violation windows correctly, then you will not get confused. If there is no violation window, then there is no issue there, data can change at any point. Yes it you may ask whether it is t c Q; I mean c Q or now D Q, that may be the only difference. Right now we are saying, it should not go out of the fall oh no; it you should not go after the.

Student: (Refer Time: 27:34).

Yes.

So, it cannot go all the way till there, correct yes that is that depends on the splitting of the commutative law, correct ok. So, basically that depends on how much time you are borrowing, right. You are trying to borrow time; what you are doing is even before this phi 2 went high right, usually you would have wanted completed operation before that. But now I am saying I will borrow time in the on cycle of phi 2. So, question is how much time can I borrow? That is also a constraint that we have to worry about, you are right ok.

So, now let us come back to our min delay constraint. Where can D 1 change? D 1 is supposed to. So, if you look at Q 1; the aim of Q 1 is to simply hold this data for as long as possible, right. This is Q 1, right; so this has to hold event 1 that is the aim, then combinational block c 1, L 2, c 2 will operate correctly. At what point can D 1 change, so that Q 1 continues to hold L 1; I mean event 1 after the whole time window, right. So, now, let us assume that my D 1 change here, it does not matter, ok.

So, I am coming all the way till here and yeah maybe I have to extend this waveform, but we do it, right. That is my violation window for the next cycle. So, D 1 has changed and then phi 1 is going high in the next cycle. So, what will happen, there is now going to be a clock to Q contamination delay, which is going to cause what? Q 2 change immediately, right. So, you look at this, there is a I am sorry ok, this is D 1 right and this is by the way event 2, ok.

So, now Q 1 is coming here and then it starts changing this delay is what t c c Q of latch L 3 ok. Q 1 changes therefore, what no Q 3 changes right; no which one has changed, sorry Q 1 has changed I am sorry, Q 1 has changed. So, now, D 2 will change after the contamination delay, correct. So, now, you see what happens here; from this delay from the earliest change of Q of Q 1 to D 2 there is a delay of t c D, by the way this is not t c c Q 3 then 1, correct. So, this is t c D 1 right.

Now what is the constraint on D 2, what should happen at Q 2 tell me? Look at the previous cycle here what is the event that Q 2 should have sampled, one right. So, Q 2 is expected sorry, Q 2 is expected to hold event 1 all Q like this until the next thing. So, this has to be event 1 right; but this is event 2 already by the way and so is this. So, you agree that on D 2, event 2 has now changed while you are expecting it to hold event 1 correctly, right.

So, remember this is I am showing it on a timeline like this, right. And unfortunately maybe let me redraw it like this that will really help. I am going to call this again L 1 this c 1 and this back to L 2; because we are now talking of the event 2 on the same latches again. The event has changed, the data has changed to one to two; but we are still talking about the same latches and same combinational block. So, I will not call it L 3 L 4; I am just going to unroll that and call it L 1, L 2 again that makes sense ok.

So, in order for event 2 not to disturb event 1, what should happen? You remember that in order to sample event 1, data should not change in this window right; which means that, this change of t c c Q 1 plus t c D 1 should happen outside the hold window, correct. So, what is the constraint that I can write for that? So, by the way this delay is what, t non-overlap ok. So, what is the constraint for D 2's hold violation not to happen, ok?

Think what happens to this t non-overlap is 0, this clock edge has gone all the way till there; then you are very clearly this hold window will now move all the way here correct, it could move right depending on the delays. So, what is the constraint for the hold delay not to occur here, hold violation not to occur, change, ok. So, we have one thing. So, let us write it. So, going from this clock edge all the way to this delay, I have a delay of what t non-overlap plus t c c Q 1 plus t c D 1, right.

And on the other hand I have what t hold; what is the constraint now? T hold should be less than this, right. The change should happen outside that hold window ok. So, very clear, very quickly tell me what happens now if there is Q; these edges can start moving right, these edges can start moving.

So, first come to the max delay constraint, what happens if this edge moves a little bit? Is there a problem? If you do not want to borrow time there is no issue; I mean there is, it is basically insensitive to clocks skew. Given that skew itself is a very small number, if the you know deviation from the actual edge is very small; then it is true that max delay is not affected by q. So, therefore, this is what we call skew tolerant, max delays are is skew tolerant.

So, I will answer that in the next class we are out of time; the I will answer that in the next class. But in general it is skew tolerant; what happens to the min delay? Then it be affected by skew or not. So, what is it that I have to look at? I have to now look at this edge, right. Is does it affect it or not? It does; just exactly like how it would affect my earlier thing, I would simply add skew here, correct; because now my hold window has moved to the right.

If it is skew, this whole window would have moved here. So, the point is non-overlapping clocks is actually giving you a lot of window, that is how you are able to get over it. So, if your non-overlapping it is large enough, you will not have old violations at all; as a very evident on this waveform. It is only when this non-overlap becomes 0 and phi 2 is falling its almost coincides with phi 1 rising edge, then you will see this violation happening; using

clock and clock bar yes or if you are overlap period is very small, non-overlapping period is very small I am sorry ok.

Now remember that for max delay I have only one constraint; for min delay this constraint has to hold at every latch. I could do the same analysis sorry with D 2 changing and then analyzing what happens at D 1 again. So, this condition min delay has to hold for every latch. So, how many constraints do we have in a latch based system; one max delay and two min delay constraints, one for each latch. In a flop based system how many constraints should did we have; one max delay, one min delay.

I will leave you with a question come back on Monday thinking about it. Flop is made up of two latches, back to back latches; if I split the latches I am getting three constraints, if I combine the latches and make it back to back I am getting only two constraints, where did that third constraint go, ok? Think about it and come back.