

**Architectural Design of Digital Integrated Circuits**  
**Prof. Indranil Hatai**  
**School of VLSI Technology**  
**Indian Institute of Engineering Science and Technology, Shibpur, Howrah**

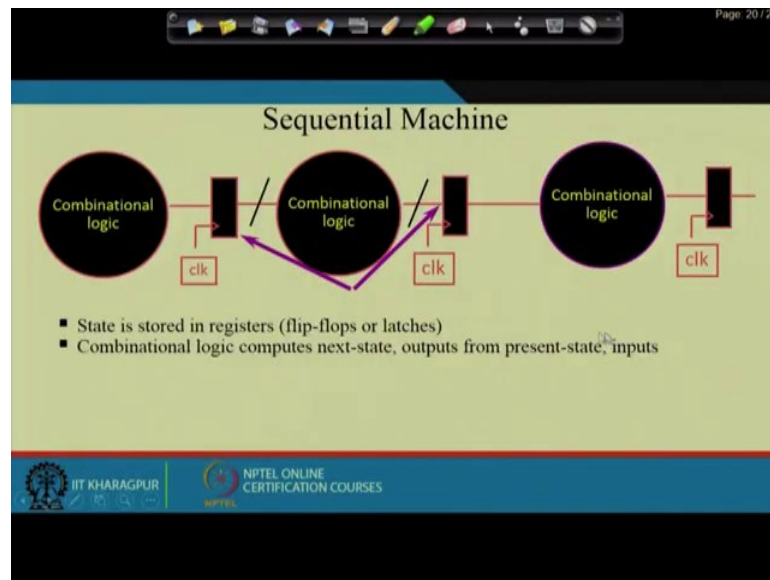
**Lecture – 51**  
**Timing Analysis Basics**

Hello everyone, welcome to the course on Architectural Design of ICs. So, today we will see one new topic which is this Timing Analysis Basics, not only basics we will also see the advance features or advance techniques what we use in timing analysis while we are basically doing or while we are basically designing one digital IC design. Why this timing analysis is very much important is that; whenever we are designing complex circuit complex digital design circuits, so, at that time there must be millions of nets, millions of gates, millions of interconnecting paths.

So, unless and until we are basically guarantees that every path or every of this; that means, this interconnections or these gates or the connection from input to the outputs are perfectly working fine, unless and until we are guarantees that it is not perfect that at that time of; that means, testing this process of this IC design they may fail.

So, that is why this timing analysis is one of the important topic or important aspects which we follow in every digital design, present day digital system design. So, that is why we will see today, we will see the this basics how it is actually with what it starts and then we will see some of the advance features like clock domain crossing, what is that and when one actually a circuit can circuit may fail to work perfectly fine. So, those advanced things also we will see in the later on of this particular lecture series ok.

(Refer Slide Time: 02:15)



So, let us start with the very first thing. In any sequential machine, we know that there are two type of circuit maybe there which is one of the logic is, in every digital circuits there are two type of circuits; one is this combinational logic another one is the sequential logic. So, whenever in digital IC whenever we are having both of them together; that means, we have some combinational logic or we can name that as combo logic and then it has passes through the series of registers.

Now, every registers may be actually connected with the clock. If every clock is connected with the same clock then we can say that this is synchronous design, if actually the clocks are different for this different registers or different this sequential elements, so, at that time we can say that, that it is design is asynchronous circuit. So, we will see this; that means techniques which will be applied for synchronous circuit as well as the techniques which will be applied for asynchronous circuit design.

So, actually in sequential circuit, there are basically states; the input state or the output state or the next state from which state to or what state it will go, the states are basically stored in the registers. The registers may be flip flop or they may be latches. Whereas, the combinational logic basically computes the operations; that means, whether it will go to which state whether that means, whether it will go to suppose, if there are three states whether it will go to S 0 to S 1 or S 0 to S 2 at what particular input if we give, so, at that time it will go to S 0 or it will go to S 1 or it will go to S 3. So, something like that logic

basically depends upon the combinational logic which computes the next state as well as the output from the present state depending on the what will be the value of the present inputs.

(Refer Slide Time: 04:22)

The slide is titled "Why Clocks?" and contains the following text:

- Clocks provide the means to synchronize
  - By allowing events to happen at known timing boundaries, we can sequence these events
- Greatly simplifies building of state machines
- No need to worry about variable delay through combinational logic (CL)
  - All signals delayed until clock edge (clock imposes the worst case delay)

Below the text are two diagrams. The first, labeled "FSM", shows a cloud labeled "Comb Logic" with a clock symbol inside, connected to a box labeled "register". A feedback arrow goes from the register back to the combinational logic. The second, labeled "Dataflow", shows a box labeled "register" connected to a cloud labeled "Comb Logic" with a clock symbol inside, which is then connected to another box labeled "register".

The slide footer includes the IIT KHARAGPUR logo and the text "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a man is visible in the bottom right corner.

So, after that in every actually the sequential circuit; the main important factor is that the clock. Then the thing is that, why clock? Basically this clock provides the means to synchronize the circuit. So, synchronize means in digital circuit synchronization means; the operations should be performed at what particular instant I need. That means, every operation if I say that within this particular clock period or within this particular event every circuit will start say operation or it will start performing the corresponding delivering the outputs or it will capturing the inputs. So, it depends upon the operation or the event of the clock.

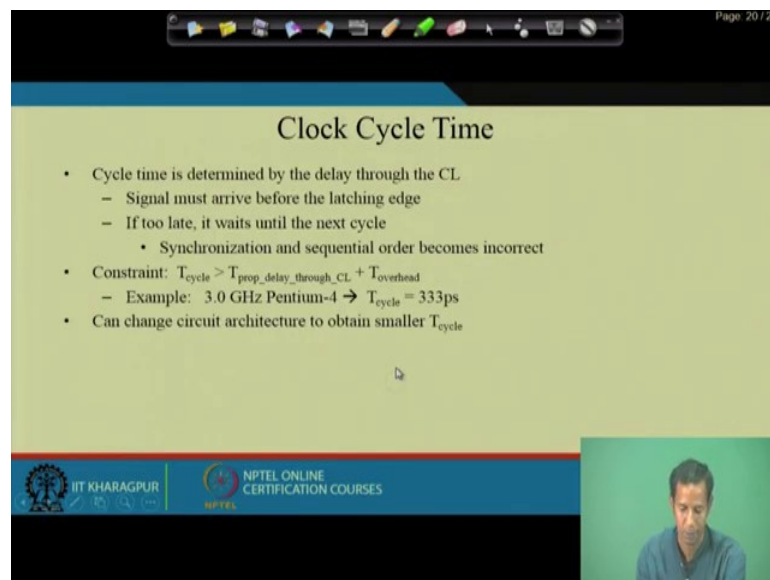
So, that is why this clock is basically present in digital circuit to synchronize by allowing the events to happen at known timing boundaries which we can sequence this event particular events and we can sequence these particular events depending on the corresponding combinational logic what we have for the particular design. And it basically greatly simplifies building of the state machine. So, as we know that there are two type of state machine mealy machines and as well as the that means this moore machines and no need to worry about the variable delay through the combinational logic where all the signals are basically delayed until the clock edge.

So, what does it mean is that; suppose, as in the previous slide we have seen that we have three different combinational logic and the registers are same where they have been driven by the same clock. So, at that time three different combinational logic they may have three different delays.

So, at that time it is not that though they have this combinational logic have variable delay or different delay from each other so, but the operation will be synchronized only based on the clock edge. That means, the data will be delivered to the next state depending on the event of clock edge only whatever is the delay of the circuit or delay of this combinational logic that does not affect the corresponding performance of that particular circuit.

So, this if we see this actually here you see this is the example of one finite state machine and this is just one simple; that means, the sequential circuit example. So, where you see that you have this registers and in between you have this combinational logic which is the data flow part of this particular circuit.

(Refer Slide Time: 07:15)



The slide is titled "Clock Cycle Time" and contains the following text:

- Cycle time is determined by the delay through the CL
  - Signal must arrive before the latching edge
  - If too late, it waits until the next cycle
    - Synchronization and sequential order becomes incorrect
- Constraint:  $T_{\text{cycle}} > T_{\text{prop\_delay\_through\_CL}} + T_{\text{overhead}}$ 
  - Example: 3.0 GHz Pentium-4  $\rightarrow T_{\text{cycle}} = 333\text{ps}$
- Can change circuit architecture to obtain smaller  $T_{\text{cycle}}$

The slide also features a video inset of a man speaking in the bottom right corner, and logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom.

So, then how to calculate this period or the clock cycle time? So, what is this clock cycle time? Clock cycle time is determined by the delay through the combinational logic, why? Because, all the signals has to be arrived before the latching edge and if the signals arrives too late it waits until the next cycle; that means, at that particular cycles or at that

particular clock edge it will miss and then what will happen? Because of that the synchronization and the sequential order will become incorrect.

So, that means, if I, that means; delayed or if my input or from which is coming from the; that means, combinational logic to the registers if they are being delayed. So, at that time the data will be missed at that particular event which I basically need intend to do, but at that time it will totally mess up the synchronization and the sequential order of my following design.

So, at that time how we can actually how we calculate the cycle time is that that depends upon the that propagation delay through the combinational logic plus the T overhead, what is the T overhead? T overhead is the; that means, the delay for the registers. In registers also there will be one delay which will be associated it is not that the registers are delay free.

So, the registers itself contains some of the delay. So, that delay plus the combinational delay so, both the delay basically defines the cycle time. So, if I actually as an example if we consider the 3 gigahertz Pentium 4 at that time the cycle time for this is that 333 picoseconds. So, and if we want to obtain smaller t cycles, so, smaller T cycle means what basically I can optimize I can maximize the operating frequency. So, to actually obtain this T cycle minimum we have to change the circuit's architecture.

(Refer Slide Time: 09:41)

**Pipelining**

- For dataflow:
  - Instead of a long critical path, split the critical path into chunks
  - Insert registers to store intermediate results
  - This allows 2 waves of data to coexist within the CL
- Can we extend this ad infinitum?
  - Overhead eventually limits the pipelining
    - E.g., 1.5 to 2 gate delays for latch or FF
  - Granularity limits as well
    - Minimum time quantum: delay of a gate

$T_{\text{cycle}} > T_{\text{pd}} + T_{\text{overhead}}$

$T_{\text{cycle}} > \max(t_{\text{pd1}}, t_{\text{pd2}}) + T_{\text{overhead}}$

The slide includes two circuit diagrams. The first diagram shows a single combinational logic block (CL) with inputs A and B, and output A+B, flanked by two registers. The propagation delay is labeled  $t_{\text{pd}}$ . The second diagram shows a two-stage pipeline. The first stage has a combinational logic block (CL) with input A and output A, flanked by two registers. The propagation delay for this stage is  $t_{\text{pd1}}$ . The second stage has a combinational logic block (CL) with input B and output B, flanked by two registers. The propagation delay for this stage is  $t_{\text{pd2}}$ . The overall cycle time is determined by the maximum of  $t_{\text{pd1}}$  and  $t_{\text{pd2}}$  plus the overhead.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Then, how we can do? So, one of the technique is pipelining. So, what is this pipelining? Pipelining means instead of for this dataflow part instead of; that means, continuing one long critical path if we can split the critical path into several chunks ok. So, if we can say; that means, define that this the combinational part if we divide into several chunks; that means, we are subdividing the main operation into several sub tasks and then we can put the; that means, registers if to store the intermediate results which basically allows two waves of data to co-exist within the combinational logic ok.

So, actually if we see the example here suppose, this is my registers and registers in between I have this combinational logic which is basically combination of two tasks which is task A and task B. The total path delay for this combinational circuit which is this propagation delay of this combinational logic is then the; that means, the T cycle time will be  $t_{pd}$  which is the delay for this critic; that means, combinational logic plus this T overhead which is the register's overhead or registers delay.

So, why it will be register delay? Because, the registers are basically deliver actually it associated with clock to q delay; that means, from the clock input to the output pin. So, here it will start from this output of the register which is the q pin which will be traversing the data will be traversing from this output from this registers to the combinational logic then the clock to q delay. So, that is why it is only  $T_{pd}$  plus T overhead of; that means, the only delay of one registers not the delay of two registers.

So, after that what I did; that means, depending on the techniques of pipelining if we divide the corresponding work or task of A and task of B into two different combinational circuit and in between them if I place one registers. So, at that time this will be the  $t_{pd1}$ ; that means, the propagation delay which will be associated with only clock A and the propagation delay which will be associated with clock; that means, task B.

So, at that time what will be the; that means, minimum cycle time which will be the maximum of  $t_{pd1}$  or  $t_{pd2}$  plus T overhead. So, at that time I can minimize the cycle time because now it is already this total  $t_{pd}$  is now divided by 2. So, it is if suppose this in this case if this is 4 so, at that time I can divide it into that means same by 2 and 2 at that time it will be 2 plus if the T overhead is 1. So, 2 plus 1, it will be 3 which initially it was like 4 plus 1, 5 ok.

So, by this actually subdividing the data flow part or this combinational logic we can and then putting the; that means, the extra registers in between we can increase the corresponding maximum frequency or we can minimize or we what in other aspects or other context I can say that we can minimize the corresponding critical cycle time.

Now, at what particular extra cost now as we have already put this registers in between these two. So, we have put some extra area with this technique of pipelining to achieve the gain in terms of speed ok.

(Refer Slide Time: 13:42)

The slide is titled "Let's Revisit Cycle Time and Path Delay". It contains the following content:

- Cycle time ( $T$ ) cannot be smaller than longest path delay ( $T_{max}$ )
- Longest (critical) path delay is a function of:
  - Total gate, wire delays
  - logic levels

The diagram shows a circuit with two flip-flops, Q1 and Q2, connected by a combinational logic path. The clock signal is labeled "clock" and has two clock edges,  $T_{clock1}$  and  $T_{clock2}$ . The data signal is labeled "data" and shows a series of pulses. The cycle time is indicated by a horizontal line above the data signal. The critical path is highlighted in purple and labeled "critical path, ~5 logic levels". The relationship  $T_{max} \leq T$  is shown in red text.

Page 20/22

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, now, again let us revisit the cycle time and the path delay. So, in this particular case if we see that this is the clock and this is the data. So, data is basically now at this particular; that means, this cross that means, whether when there is the cross; that means, at that time the data are basically changing. When there is; that means, only white color; that means, at that time the data are not changing.

So, if we; that means, say that this basically this flip flop this is a two flip flops. So, when these two flip flops are connected with some of the combinational logic in between. So, at that time I actually to avoid the violations, always we have to consider that at the clock edge the data must be stable it will not change it is value.

So, Q here you see at every of this edge clock edge the data are basically fixed it is not changing it is value. So, that is why this cycle time cannot be smaller than the longest

path delay which is the  $T_{max}$  and then longest path delay is a function of this total gate and the wire delays, where this gate delays are basically depends on the number of logic levels. That means, as in this particular case you see there are two paths which is basically following. So, in this particular path I am having 1, 2, 3, 4 number of gate delays whereas, in this particular path 1, 2, 3, 4, 5. So, 5 number of gate delays.

So, at that time so, this will come and this will be considered as the critical path which consider this  $T_{max}$  and the; that means, the cycle time should not be less than this  $T_{max}$ . If this particular; that means, this cycle time that becomes lesser than that then at that time the data at that time it will. If this happens at this at that time this edge will actually, at that time if this edge will come here where the data are basically changing.

(Refer Slide Time: 15:57)

The slide is titled "Cycle Time - Setup Time". It contains the following elements:

- Text:**
  - For FFs to correctly capture data, must be stable for:
  - Setup time ( $T_{setup}$ ) before clock arrives
- Timing Diagram:** Shows a data signal (represented by 'X's) and a clock signal ( $T_{clock1}$ ). A vertical line indicates the setup time window before the clock edge. Below the diagram, the equation  $T_{max} + T_{setup} < T$  is written in red.
- Logic Circuit Diagram:** Shows a circuit with two flip-flops, Q1 and Q2. A path of logic gates (AND, OR, NOT) connects Q1 to Q2. This path is labeled "critical path, ~5 logic levels". Clock signals  $T_{clock1}$  and  $T_{clock2}$  are shown at the inputs of Q1 and Q2 respectively.
- Logos:** IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.
- Video:** A small video inset of a presenter in the bottom right corner.

So, when this edge occurs were the data are changing at that time it will not work, work perfectly fine sorry, it will not validate that what data it has been captured. So, that is why it will be in the ambiguous state or you can I can say that that is will be in the metastable state. So, that is why we are we just avoided to do that ok.

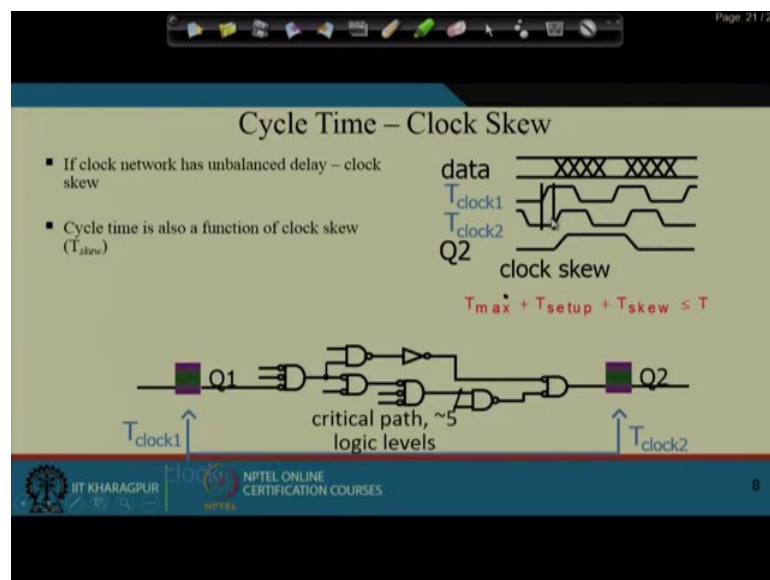
So, then while we are considering the cycle; that means, this clock at that time the two important actually things we have to consider. One thing is this set up time another one is this hold time. So, what is this set up time? The set up time is that we need; that means, before this clock edge appears, we need some times where the data should not be



changed; that means, to recognize the data properly we need some times before this clock edge occurs which is known as the set up time ok.

So, for every flip flops to correctly capture the data, the data must be stable as of the time period of  $T_{setup}$  which appears before the clock arrives or the clock edge arrives ok. So, before these clock edge arrivals of this, the data must be stable for some times which are known as the set up time. So, in this particular case; so, at that time the setup the cycle time should be  $T_{max}$  plus  $T_{setup}$  then only this particular this particular logic will work perfectly fine otherwise it may fail ok.

(Refer Slide Time: 17:59)



So, then if we say that what is this if we are having this clock skew. So, whenever when we will get clock skew; if these two particular registers are basically connected to two different clocks. It is not that the same clock is basically same clock is connected or this clock this will be like clock 1 and this will be clock 2, though they have connected with the same clock, why? Because in clock network in digital IC design they put some buffers in the clock tree path ok.

So, that means that; the particular delay where this clock appears to this Q 1, suppose this is a source pin. So, the paths or length from this particular pin to the Q 1 that is much lesser then the path delay which associated with the clock pin of Q 2. So, this difference between this delay of arrival of the clock basically creates the skew ok.

So, depending on that clock skew. Now you can see that this is the clock 1 this is the; that means, this clock 1 signal whereas, this clock 2 can be like they will be shifted by some of the time as they are coming a long path; that means, after a long path it is arriving to the Q 2. So, that is why this particular edge is not aligned to the clock 1, there basically there is some difference. So, this is basically nothing but the clock example of clock skew.

So, at that time; the cycle time if there is a clock skew so, at that time the minimum cycle time requirement will be  $T_{max}$  plus  $T_{setup}$  plus  $T_{clock\ skew}$ . So, unless and until we consider this skew part here which is present in the circuit, so, at that time this particular circuit will not work that means, at that time if this is only if I consider if there is skew, but still if we consider only max and setup  $T_{max}$  and  $T_{setup}$ . So, at that time that corresponding  $T$  value will sample the data at a very a faster rate, but depending on this  $T_{skew}$  it will not work perfectly fine ok.

(Refer Slide Time: 20:26)

**Cycle Time - Flip-Flop Delay (Clock to Q)**

- Cycle time is also a function of propagation delay of FF ( $T_{clk-to-Q}$  or  $T_{cq}$ )
- $T_{cq}$ : time from arrival of clock signal till change at FF output

$T_{max} + T_{setup} + T_{skew} + T_{clk-to-Q} \leq T$

The slide also features a circuit diagram with two flip-flops, Q1 and Q2, connected by a path of five logic levels. Timing diagrams show data, clock1, clock2, and Q2 signals. The slide is from IIT Kharagpur NPTEL Online Certification Courses.

So, then again if we consider this actually that we have already say actually consider the example of cycle time as  $T_{pd}$  or  $T_{max}$  plus this  $T$  overhead ok. But here that  $t$  overhead is nothing, but this clock to Q delay. Clock to Q delay means; this is the clock pin to the Q. If I consider actually at this particular edge, this is the output which is the input to this is the long path from input to the output. So, this is the combinational part

then the setup; that means, the set up time requirement then the skew requirement and then the delay of this particular register which is nothing but this clock to Q delay.

So, this all this particular four terms summation of them defines the final cycle time of my circuit. So, in digital circuits, we have not actually in generic digital circuit we may not have considered all of this, but in digital IC design we have to consider each of these particular terms unless and until the circuit may not work after it is fabrication ok.

(Refer Slide Time: 21:46)

The slide is titled "Min Path Delay - Hold Time" and is page 21 of 23. It contains the following content:

- For FFs to correctly latch data, data must be stable during:
  - Hold time ( $T_{hold}$ ) after clock arrives
  - Determined by delay of shortest path in circuit ( $T_{min}$ ) and clock skew ( $T_{skew}$ )

The timing diagram shows a clock signal  $T_{clock1}$  and a data signal. The data signal is marked with 'X' during the hold time period. The equation  $T_{min} \geq T_{hold} + T_{skew}$  is shown in red.

The circuit diagram shows two flip-flops, Q1 and Q2, with clock inputs  $T_{clock1}$  and  $T_{clock2}$ . A "short path, ~3 logic levels" is highlighted between Q1 and Q2.

Logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES are visible at the bottom.

So, then again you actually there is another path which is hold time. So, now, what is hold time? Hold time is the data; that means, after the clock edge appears the data must be stable for some times to properly recognize that what the value the data contains, whether that data contains 0 or whether the data contains to 1 state.

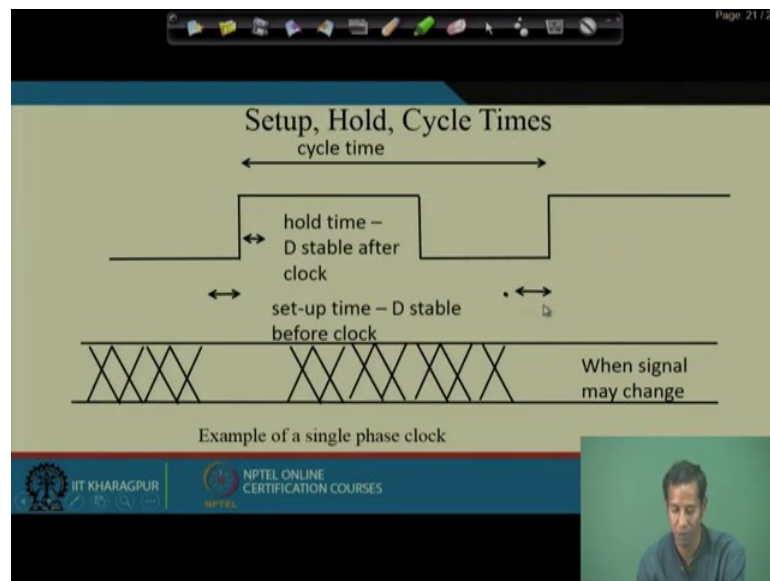
So, for that particular reason to properly recognize the state of the data the data should not change for it; that means, time period particular time period after the clock edge occurs. So, setup times is the time requirement before the clock edge arrives and that means, hold time is the time requirement after the clock edge arrives to properly recognize the data ok.

So, here you say hold time after the clock edge arrives. And it is basically determined by the delay of the shortest path in circuit and the clock skew. So, the T minimum is the T hold plus T skew ok. So, this is the minimum timing requirement that before this if the

data occurs or if the data appears. So, at that time this particular circuit will not work perfectly fine.

So, in this particular case the minimum path is this basically this one where it has to pass only three logic levels this is a shortest path. So, which comes with this  $T$  minimum computation ok.

(Refer Slide Time: 23:29)



So, if I actually if I; that means, take the example of one single phase clock. So, at that time here you see this is the time requirement for the setup and this is the time requirement for the hold and from this particular edge to the next stage is the cycle time.

So, within this summation of these two particular this hold time plus the set up time sorry, set up time plus the hold time the data should not be remained on the same stage it should not be change it is value within that particular time period. So, here you see here also the data should not be change. So, in between of this the data may change because they will not be sampled to the register.

Though they have change this value here for so many times, but that only the data will be sampled to the registers only what value it occurs at this particular edge and the data will be sampled which is occurs at this particular edge ok. So, intermediate values will be just ignored.

(Refer Slide Time: 24:32)

Timing Constraints for Edge-Triggered FFs

$\text{Max}(T_{pd}) < T_{\text{cycle}} - T_{\text{setup}} - T_{c2q} - T_{\text{skew}}$   
 - Delay is too long for data to be captured

$\text{Min}(T_{pd}) > T_{\text{hold}} - T_{c2q} + T_{\text{skew}}$   
 - Delay is too short and data can race through, skipping a state

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, if we consider that depending on the equation; so, the maximum delay of propagation delay of the combinational logic that will be  $T_{\text{cycle}}$  minus  $T_{\text{setup}}$  minus  $T_{\text{clock 2 q}}$  minus  $T_{\text{skew}}$ . Actually, just the previous example where  $T_{\text{cycle}}$  is based basically the  $T_{\text{max}}$  plus  $T_{\text{setup}}$  plus  $T_{\text{clock 2 q}}$  plus  $T_{\text{skew}}$ . So, if the delay is too long, the data will be captured incorrectly. So, we will see one of the examples then. And if this minimum  $T_{pd}$  is basically the minimum propagation delay is that the; that means, the equation is that it should be greater than  $T_{\text{hold}}$  minus  $T_{\text{clock 2 q}}$  plus  $T_{\text{skew}}$ . So, if the delay is too short at that time the data cannot be recognized properly and it will skip one state ok.

(Refer Slide Time: 25:28)

Example of  $T_{pdmax}$  Violation

- Suppose there is skew between the registers in a dataflow (regA after regB)
- "i" gets its input values from regA at transition in Ck"
- CL output "o" arrives after Ck transition due to skew
- To correct this problem, can increase cycle time

Too late!

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, that is why this if we consider this example of this  $T_{pd\ max}$  violation. So, here you see we are having this one registers A here registers B here and in between you are having some combinational logic. So, this A is basically connected with inverted clock whereas, this register B is basically connected with a direct clock. So, this is not inverted clock sorry this is skewed clock bar.

So, now here actually this clock that this clock is basically like this and this clock bar as this that is skewed. So, it will be arrived after sometimes which is the amount of skew. So, now, at  $i$ ; this is the, that means, output from registers A which is basically changing at this particular edge ok. So, at this particular edge depending on this clock bar it is changing the data is changing so, but at this edge of the clock which is the requirement of this register B; it is holding the previous value of  $i$ .

So, it will hold that value up to the next edge. So, at the next edge only this updated  $i$  value they will be captured and they will be delivered to the next clock cycles so; that means, though this particular value should be contain the output here at this, but it is not holding that particular value it is missing the state and it is appearing at the next clock cycle edge ok. So, that is why this is the problem which can increase the cycle time ok.

(Refer Slide Time: 27:24)

Page 21/23

### Example of $T_{pd\ min}$ Violation: Race Through

- Suppose clock skew causes regA to be clocked before regB
- "i" passes through the CL with little delay ( $t_{pd\ min}$ )
- "o" arrives before the rising  $Ck'$  causes the data to be latched
- Cannot be fixed by changing frequency  $\rightarrow$  have rock instead of chip

The diagram shows a circuit with two registers, regA and regB, connected to a combinational logic block. regA is clocked by Ck (inverted) and regB by Ck' (direct). The timing diagram shows Ck and Ck' signals with a skew. A data signal 'i' is shown with a delay  $T_{pd\ min}$ . The output 'o' is shown to be latched too early by regB, labeled "Too early!".

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, then the example of  $T_{pd\ min}$ ; that means, if the propagation delay is minimum, at that time what violation or what is the violation we will get. So, the same thing same

circuit where this register B is now connected with the skewed clock whereas, this clock is connected with the register A.

So, at that time what will happen? This is the in which; that means, the original clock and this is the skewed clock. So, if I is basically changing the data over here depending on this output from this combinational logic they will come at this particular edge and edge this comes at the edge where this clock skewed clock is basically changing from 0 to 1. So, at that particular time it is that output from this combinational logic changing it is value. So, it will basically creates the problem or there will be the properly recognition of the data will be correct incorrect ok.

So, that is why this the O arrives before the clock the before the rising of the clock skewed clock causes the data to be latched ok and by changing actually this to solve the problem of set up time violation, if we increase the clock time period so, at that time we can solve the problem. But here by changing the actually with this frequency or changing the cycle time these particular problems may not be solved ok.

So, we have different technique to solve this setup time violation and the hold time violation. So, later on this particular actually series we will see what are the how to solve these problems in details; so.

Thank you for todays.