

VLSI Physical Design
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 33
Time Closure (Part 2)

So, we continue with a discussion on Timing Closure. In this lecture we shall be discussing in some detail how this static timing analysis works. Now recall what we mentioned during our last lecture. Static timing analysis is a software tool using which you can estimate the worst case delays in a combination circuit netlist. So, this, the analysis is very useful to detect timing violations in a design in a circuit net list. So, that we can move to the next step and try to correct those violations. So, we proceed with this lecture.

(Refer Slide Time: 01:06)

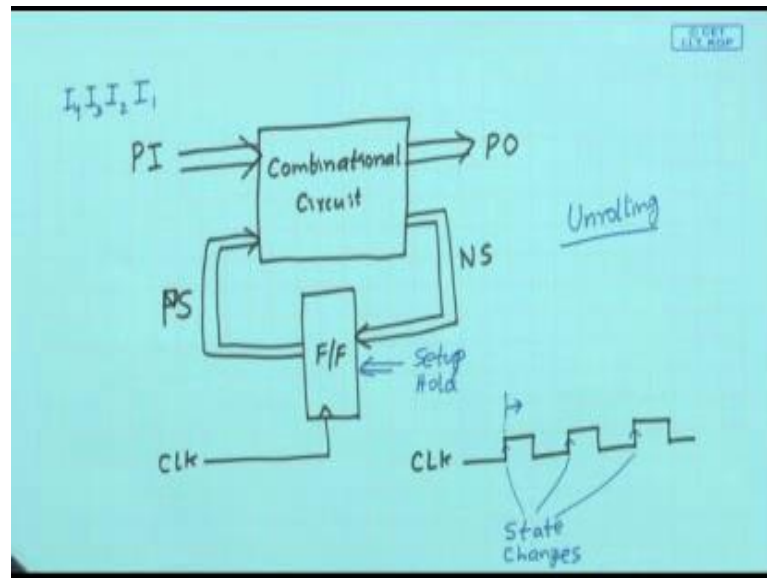
Timing Analysis and Performance Constraints

- Almost all digital ICs are synchronous Finite State Machines (FSM).
 - Transitions occur at a set clock frequency.
 - A sequential circuit, *unrolled in time*:

The diagram illustrates a synchronous sequential circuit unrolled in time. It consists of three stages of combinational logic, labeled 'Combinational Logic Copy 1', 'Combinational Logic Copy 2', and 'Combinational Logic Copy 3'. Each copy is connected to a flip-flop (FF) block. The output of one FF block is the input to the next combinational logic copy. A common 'Clock' signal is connected to the clock input of every FF block. Vertical dashed lines separate the combinational logic blocks, indicating the discrete time steps of the clock.

IIIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

(Refer Slide Time: 01:21)



Here what we talk about at the beginning is that we say that almost all digital ICs are synchronous finite set machines. Let us try to explain this. So, how does a finite set machine look like? Let us see that first. So, any finite set machine; that means, synchronous sequential circuit can be modeled as follows. So, you have a combinational circuit you have a set of flip flops. So, I am showing them separately. So, some primary inputs are coming these are the primary inputs, some primary outputs are coming out PO. Some outputs are going to the input of the flip flop. So, this we call it as the next state and this we call it as the present state PS. And there is a clock which will be feeding the flip flop.

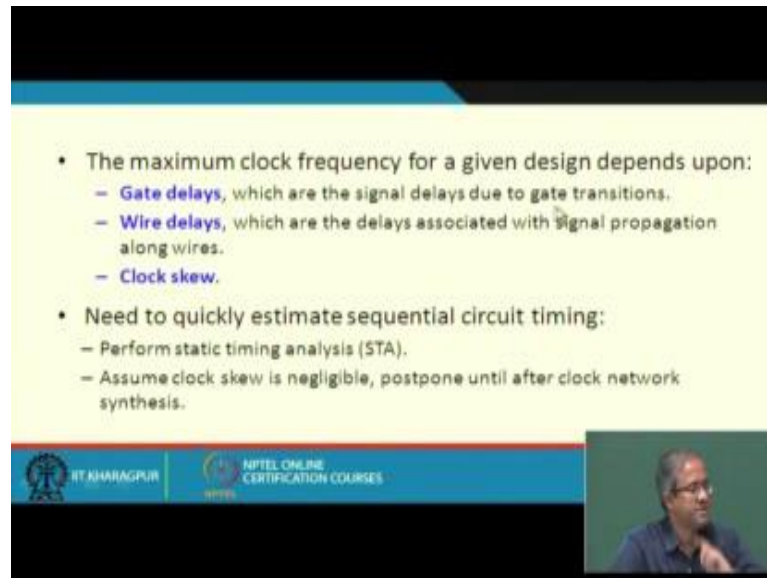
So, normally in this model what is assumed is that the clock is a periodic signal that goes like this. Let say it is triggered at the positive edge of the clock, so at every positive edge of the clock here, here and here, this state changes take place. So, what do we mean by state changes, you see for a particular specimen after this clock is present let say after this time, the inputs to the combination circuit are stable the PI is coming this PS is also stable. So, combination circuit will be having some delay. After that delay it will be computing PO and NS. And when the next h comes this NS is already at the input of the flip flop. So, when the next clock comes this NS will get stored and it will now become the PS for the next cycle right.

So, in this way it goes on, and again this clock period will be decided not only by the combination circuit delay, but also some characteristics of this flip flop this set up and hold times, set up and hold right. Now this model of a combination circuit can also be expressed as if you are unrolling it in time. Let say I have this inputs which I have coming in sequence, let say first I 1 then I 2 then I 3 then I 4 like this. Now in this model there are coming sequentially in every iteration I am applying one input and the next state is coming is becoming the present state. Then I apply I 2 then I 3 then I 4 and so on. Now sometimes just for modeling an analysis not for actual circuit realization we may have to do a process called unrolling.

So, we unroll this iterative structure. Like for example, I unroll it 3 times. So, that here what I am saying is that this I 1, I 2, I 3, these are as if applied at the same time, 3 3 3 we are dividing and in this diagram that I have shown. So, here actually we have unrolled the circuit in time 3 times. So, here if you see that same circuit we have unrolled it 3 times. So, the first copy the output of it goes to the flip flop, the output of it comes to the second copy the output goes to the flip flop these output of it comes to the third copy. Now what I have not shown is that here the input I 1 is coming here the input I 2 is coming here the input I 3 is coming. So, here in one clock we are doing the task of 3 clocks in the previous circuit.

Now, this kind of unrolling we often do in order to capture the delay rated things in the combinational circuits and how it can effect across clock citrus. So, just for that analysis sometimes we may have to unroll a sequential circuit like this.

(Refer Slide Time: 05:59)



- The maximum clock frequency for a given design depends upon:
 - Gate delays, which are the signal delays due to gate transitions.
 - Wire delays, which are the delays associated with signal propagation along wires.
 - Clock skew.
- Need to quickly estimate sequential circuit timing:
 - Perform static timing analysis (STA).
 - Assume clock skew is negligible, postpone until after clock network synthesis.

So, when we talk about the maximum clock frequency that you can have in a given design there are 3 things that we can possibly control. Now this also depends on the set up and hold times that I have mentioned that set and hold times are something that we assume that we cannot control, because we are already picking up this storage cell from some from some technology library, those are already pre designed. And pre designed means the set up and hold times are already specified. So, as a designer I cannot make this set up time longer or shorter or the hold times longer or shorter those are fixed.

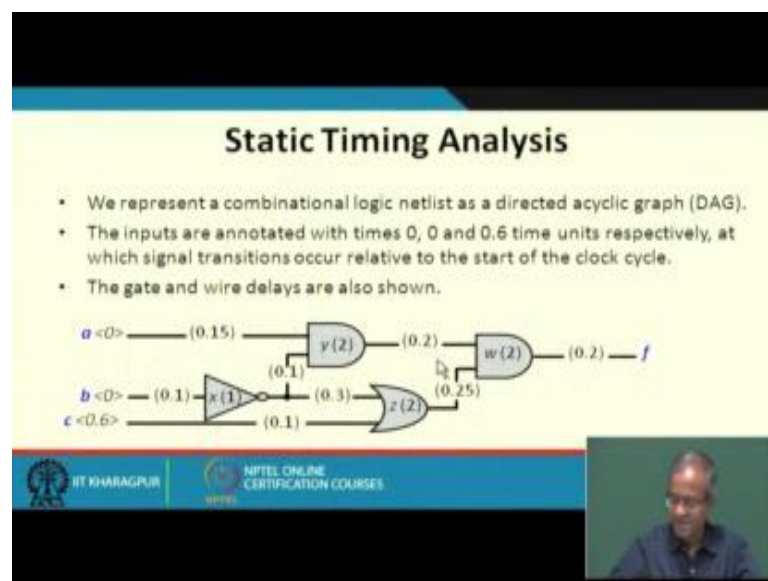
But the parameters with which we can play around is we can control the gate delays. So, we can make a gate smaller we can make a gate larger. This corresponds to the signal delays due to gate transitions. Gate transition means given a gate whenever one of the input changes from 0 to 1 or 1 to 0, after how much time the output of the gate will show that transition taking place that is the gate delay in terms of transitions. Similarly wire delays, which correspond to the signal propagation along the interconnecting wires. So, again we saw earlier we can control the wire delays, like we can make a wire delay shorter by inserting buffers, try and make it shorter, but sometimes also we may want to make wire longer increase the delay, like you recall in clock net routing to adjust this skew we may have to slow down one of the wires.

So, then we may have to do something called snaking some kind of zig zag kind of a path we may take. So, that the total length increases and the delay gets balanced. And

third is of course, clock skew clock skew something that also determines the maximum clock frequency and clock skew is something this also depends on the design. So, you can have a design with a very good clock skew very small for some designs where it can be significantly higher. Now to estimate the timing to estimate this gate delays and wire delays clock skew you are ignoring for time being. So, we perform a process called static timing analysis. So, in this analysis as I said we assume that clock skew is negligible as compared to the gate and wire delays.

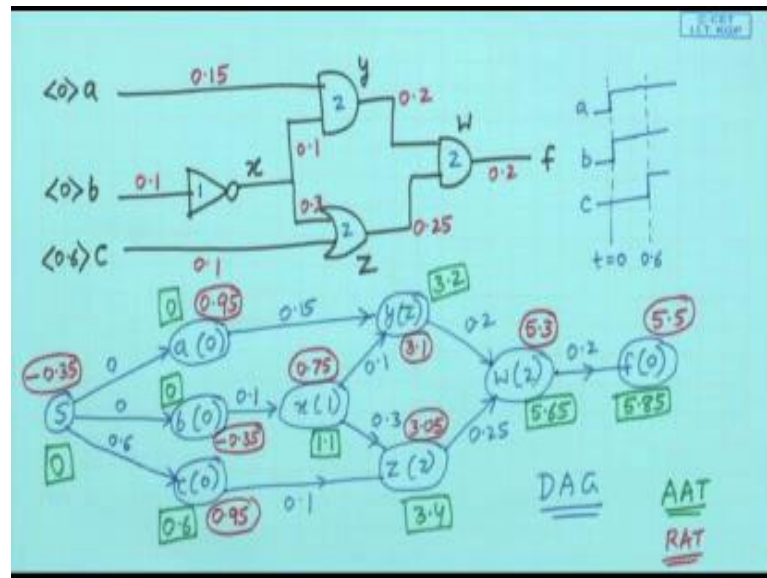
Clock skew we shall be considering separately. So, once the clock network is being designed synthesized, then only we shall look at the clock skews and adjusted it as required, but while we are doing static timing analysis we look at only the gate delays and the wire delays.

(Refer Slide Time: 09:10)



Let see what static timing analysis this process actually does. Static timing analysis the first point you notice that it works only for a combination circuit not for a sequential circuit. So, it takes segment of a given net list between 2 storage or flip flop stages it is a pure combinational circuit it estimates the worst case delays roughly it is like that. So, the illustration that I shall be giving for this process, here the combination logic will be expressed as a direct cyclic graph. Direct acyclic graph means graph where every edges has an arrow it shows the direction of signal transition and there is no feedback, there is no directed cycle - directed acyclic means no cycles.

(Refer Slide Time: 10:21)



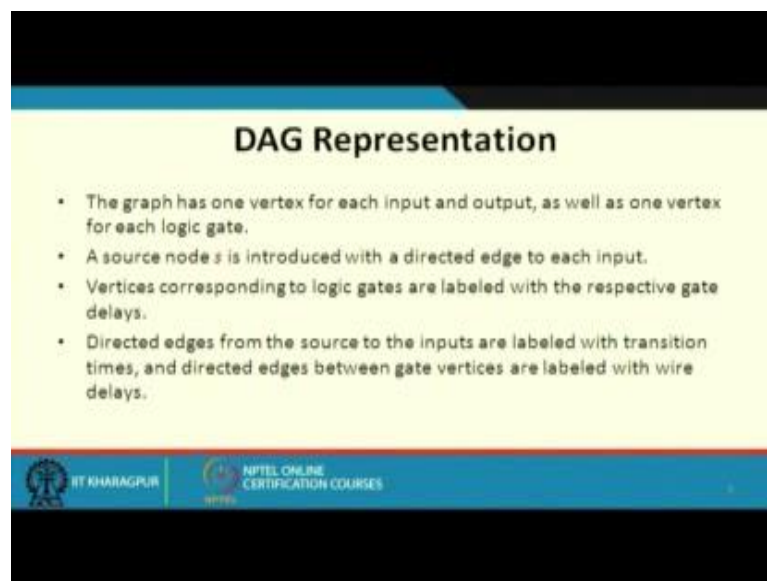
So, we take an example. So, we consider an example like let us just draw this circuit separately. So, we have taken an example like this. This is a simple combinational circuit we have taken. So, the input let us call it a b and c, this is x this gate is y this is z this is w and the output is f. So, what we are assuming here is that that let say we will just illustrate the static timing analysis with respect to the example of this circuit, let say I look at the access of time this represents my time t equal to 0. So, what we assume that there are some signal transitions appearing at a b and c. So, we are assuming that at the signal transition is taking place exactly at time t equal to 0, at b this can be 1 to 0 or 0 to 1 it does not matter. I am just showing 1 0 to 1. At b it is or also at 0 and let say another time scale this is 0.6 little later.

So, in c the transition is taking place little later, you see this can always happen because a b c is coming from some other circuit previously right. So, it is not necessarily true that all the inputs will be changing state at the same time. So, the first thing is that in static timing analysis we have to annotate the primary inputs by saying that when the inputs are transiting or changing state. So, in this example if you see, here you are assuming this a b c these inputs I am showing them like this 0 0 and 0.6. This indicates the time at which the signal transitions take place in the inputs right. There are a few other things we are assuming we are assuming some delays of these gates which I have shown inside. See x

delays 1 2 2 and 2. So, let us note this down and this diagram also the delay of this is 1 this is 2, this is 2, and this is 2.

Not only that we are also just annotating on this diagram, the delays of the interconnection wires, in some units it is showing as 0.15 0.1 like this. So, I am also showing the delays like this. So, this delay is 0.1 this delay is 0.15, this is 0.1 this is 0.2, this is $\times 2$ this is 0.1, $\times 2$ this is 0.3, this is 0.25 and the output line delay is 0.2. So, this will be the input of your static timing analysis tool. So, I specify not only this also I specify the input arrival times 0, 0 and 0.6. So, I specify the input arrival times, I specify the gate delays I specify the wire delays. Now you can understand you can provide realistic estimate of the wire delays provided you already had made some placement. So, it is assumed that already placement is done, you have some very good measures of the interconnection wire delays with that you are doing this static timing analysis.

(Refer Slide Time: 14:31)



DAG Representation

- The graph has one vertex for each input and output, as well as one vertex for each logic gate.
- A source node s is introduced with a directed edge to each input.
- Vertices corresponding to logic gates are labeled with the respective gate delays.
- Directed edges from the source to the inputs are labeled with transition times, and directed edges between gate vertices are labeled with wire delays.

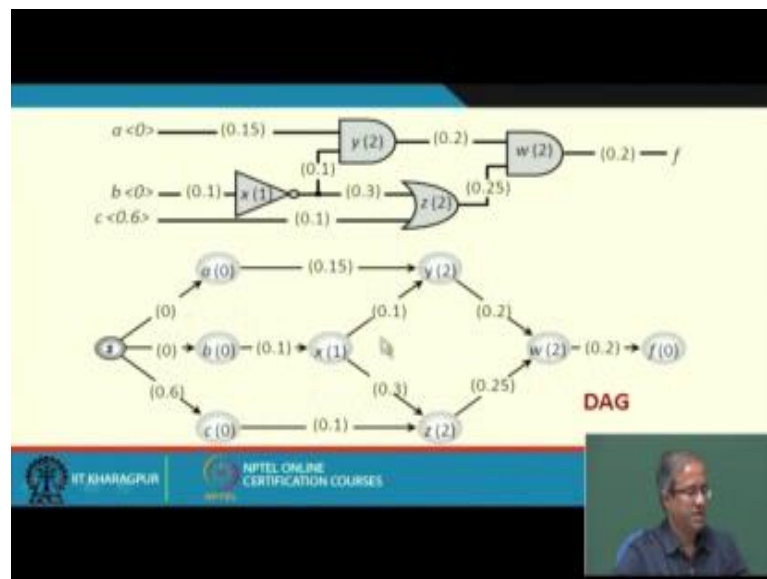
IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, the next step is to model this as a direct acyclic graph. So, this graph will be having one vertex for each input and output as well as one vertex for each logic gate right. So, you introduce a dummy source node s from where there will be an edge to each of the inputs. And the logic gate vertices will be labeled with the delays and the directed edges will indicate wire delays. So, let see for this example what will happen.

For this case let us try to just work out. So, here the inputs are a b c. So, I am also just indicating the delay in bracket. So, in the primary input there are no delays these are the primary input and I have introduced a dummy vertex s from s there will be an h to each of the primary inputs. So, I am assuming the delays are 0 0 and 0.6 because 0.6 is arrival time at c. And from b there will be an h 2 x, which has a delay of 1. And this weight is 0.1 and then we have y with a delay of 2 there is an edge like this with a weight 0.1 there is an h from here also with a weight 0.15.

Then similarly we have z. Again with 2 from x we have an edge 2.3 and from c you have an edge to 0.1 and here you have the gate w with a delay of 2 this is 0.25, this is 0.2 and finally, output on the f that does not have a delay, but this interconnection is 0.2. So, this is your direct acyclic graph representation of your given gate level net list right. So, from the given gate level net list and the arrival times, you create a graph like this. Now your static timing analysis your tool will work on this graph representation right.

(Refer Slide Time: 17:21)



(Refer Slide Time: 17:36)

Actual Arrival Time (AAT)

- The AAT of a given node $v \in V$, denoted as $AAT(v)$, is defined as the latest transition time at v measured from the beginning of the clock cycle.
 - By convention, $AAT(v)$ records the arrival time at the **output side** of node v .
 - In the previous example, $AAT(x) = 0.1 + 1 = 1.1$, $AAT(y) = 1.1 + 0.1 + 2 = 3.2$.
- Formal definition:
$$AAT(v) = \max_{u \in F(v)} (AAT(u) + t(u, v))$$
where $F(v)$ is the set of all nodes from which there exists a directed edge to v , and $t(u, v)$ is the delay corresponding to the (u, v) edge.

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

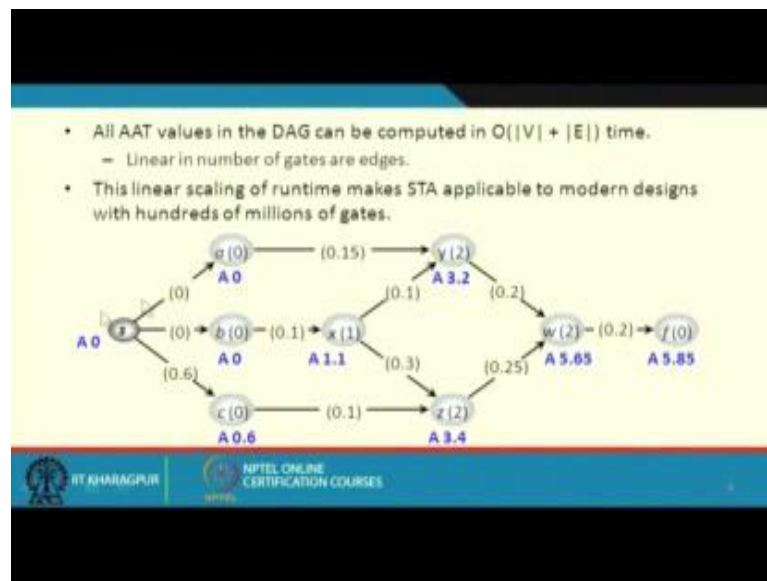
So, let see. Here we show this graph that same graph I had drawn. We first construct this graph. Now with respect to this graph we then calculate sometimes. First thing is called the actual arrival time. The actual arrival time of a node this is actually belongs to this symbol is not coming correctly v belongs to capital V .

So, the arrival time of a given node small v that belongs to a set V , which is denoted as actual arrival time of v , is defined as the latest transition time at that point measuring from the beginning of the clock cycle. So, as a matter of convention this AAT, will record the arrival time at the output side of node v . Like you look at this example again look at this example again. So, let say let us look at this s , well s this nothing like arrival this is a dummy node. So, I assume that the arrival time here is 0 I am showing like this. A , a transition takes places at times 0, b takes place at time 0, and c takes place at time 0.6 this is already there in the now let us look at the others.

For x the delay of x is 1 and the delay of the input b which is at time 0 is 0.1. So, it will be 1 plus 0.1 only after 1.1 the output of x will be available. So, I label this with 1.1. Now once I have leveled this with 1.1 you look at this y . So, there are 2 paths we have to consider both the paths. In the first path it will be 2 plus 0.15, 2.15 and along this path it will be 1.1 plus 0.1 plus 2; that means, this is larger you record the larger number 3.2. Similarly, in z there are 2 paths one is 0.6 plus 0.1 plus 2, 2.7 other side 1 plus 1, 0.3, 2 that is 3.4. 3.4 is larger, record that.

Come here there are again 2 paths 3.2 plus 0.2 plus 2, or 3.4 plus 0.25 plus 2. This is larger it will be 5.65. And in the output side finally, plus 0.2 it is 5.85. So, in green whatever I have shown you are actually getting actual arrival times of the signal transitions at every node assuming the gate delays and the wire delays. This can be done by a single pass through the circuit right. So, the complexity of this process will be number of vertices plus number of edges not more than that linear in the size of the circuit. It can be done for large circuits also.

(Refer Slide Time: 20:58)



So, this is shown in this diagram exactly what you have worked out. So, as I said this all actual arrival time values shown in blue here in this diagram, can be computed in order number of vertices plus number of edges because we have to visit every vertex exactly once we have to traverse every edge also exactly once in this process.

You have to do something called a topological sorting first; then from left to right you have to traverse. Because of this linear time complexity, the static timing analysis can be very easily used for very large circuits comprising of millions of gates as well. So, this becomes an attractive tool for very quick estimate of the timing delays and hence some I can say timing violations.

(Refer Slide Time: 21:52)

Required Arrival Time (RAT)

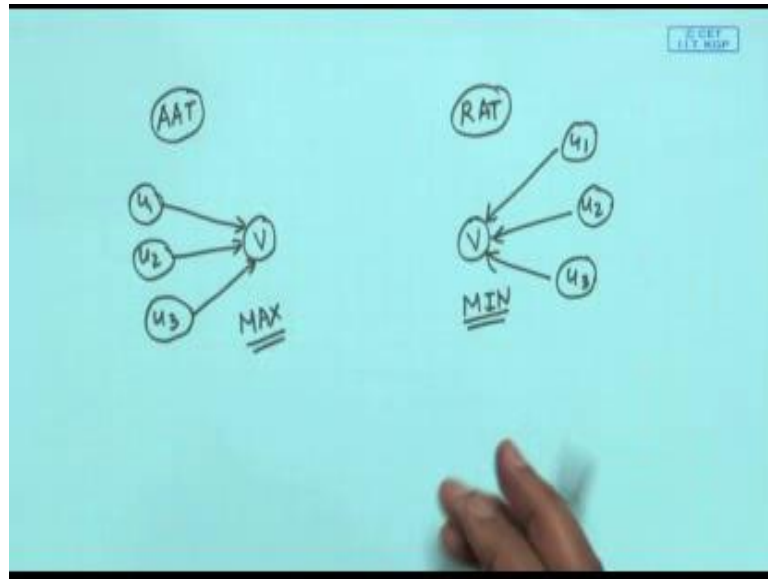
- The RAT of a given node $v \in V$, denoted as $RAT(v)$, is defined as the time by which the latest transition at a given node v must occur in order for the circuit to operate correctly within a given clock cycle.
 - Unlike AATs, which are determined from multiple paths from upstream inputs and flip-flop outputs, RATs are determined from multiple paths to downstream outputs and flip-flop inputs.
- Formal definition:
$$RAT(v) = \min_{u \in FO(v)} (RAT(u) - t(u, v))$$
where $FO(v)$ is the set of all vertices with a directed edge from v .

IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, once we have done this, next just going back once. So, this this this actual arrival time actually formally definition goes like this, so for a node v . So, if there are number of fan in nodes let say u belongs to fan in of v . So, for each of them you consider the actual arrival time of that node plus the wire delay take the maximum of that exactly what we have done right, that way you calculate AAT, now required arrival time is different. Well again this should be belonging to, required arrival time means you see the user have specified the required arrival time of my output. That is the maximum delay I want the minimum delay I want. So, from that value I can back trace and you can deduce what will be the required arrival time for the other nodes.

Like for calculating AAT, actual arrival time we assumed when the input transitions are taking place. And from there we are calculating and finding out the transitions in the other node, but for required arrival time we are doing in the reverse way. From the output side we are saying that well at time into of 5 I want the output to be stable. So, therefore, in the previous stage when it should be stable. Therefore, in the earlier stage when it should be stable like that that is RAT. So, RAT we define as the time by which the latest transition at a given node must occur. This is a required dead line, in order for the circuit to operate within a given clock cycle.

(Refer Slide Time: 23:47)



Now, here one thing that when we are calculating the AAT. So, what we are doing. For every vertex V we are looking at the input fan ins. Let say u_1, u_2, u_3 like this. So, you look at the AAT values here plus the delay plus this you take the max, you take the maximum, but when you are calculating the required arrival time we do the reverse way, for every vertex V we are looking at the other direction, if there is a fan out connection. So, there is a fan out of 3. Let say just call it u_1, u_2 and u_3 . Now here similarly you see what is the required arrival time for these nodes, subtract the delay of these lines subtract the delay of this gate. So, what RAT value will be getting here will be the minimum of these. So, for calculating required arrival time you have to calculate minimum because whichever is earliest you have to get V ready by that, but for this you are estimating the maximum delay you need to take max, right.

So, looking at the definition again, you can see just what I mentioned just now, at AAT at determined from multiple paths from upstream input; that means, which are coming at the fan in, but RAT is the determined from multiple paths at the fanout side this called downstream. So, the expression for RAT is just this. The RAT value of the output nodes minus the delay of the node, and you take the minimum of that right. So, let us let us take the same example to work it. So, we assume that the RAT value that is given to us of the final node is 5.5. So, I am assuming in this diagram. Let us come to it I am assuming that the RAT value is 5.5. So, in red we are marking RAT required arrival time. So, let see

how the RAT value of the other nodes can be calculated. So, what I doing from here when you come here we see that the weight of this node is 0.2.

So, this signal at the output should be ready at time 5.3. Right now from here when you come here you see. So, so at the output it was 5.3 the delay of the gate is 2. So, it is 5 point it is a 3.3 and 0.2. So, here it should be come at 3.1. Similarly, here 5.3 minus 2 is 3.3 minus 0.25, 3.05. So, here it will be 3.05. So, in this way you go on calculating. So, from here you come here minus 1, 3, 3 and this 1 it will become 2, but there is another path. So, you calculate both the path which one is smaller. So, you calculate like this. So, it will be 3.1 minus 2, it will be 2 0.1 minus 0.1, 3-point minus 2 it will be 1.1 minus 1 to 1, and this side would be 3.05, minus 2 it will be 1.05 minus 0.3 it will be 0.75 this is smaller. So, the value here will be 0.75 this smaller minimum.

Similarly, you proceed 3.2 minus 2 minus 0.15 this will become 0.95. Similarly, this will become minus 0.35. You see one value is becoming negative here 0.75 minus 1 minus 0.1 and here it will become 0.95. And finally, at this dummy node it is minus 0.35. These are the required arrival times right. So, now, we are mentioned that we can estimate the slack by subtracting AAT from RAT, RAT minus AAT. So, I am showing this slack computation here.

(Refer Slide Time: 28:56)

Slack Computation

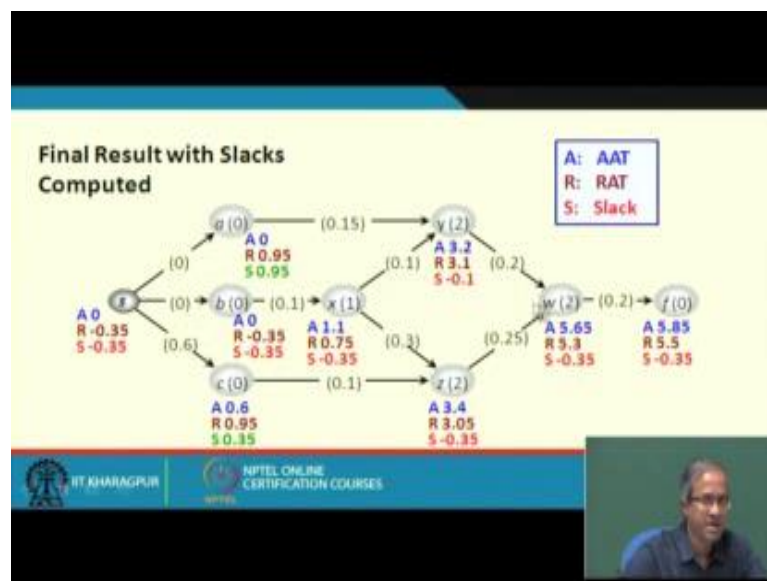
- The correct operation of the chip with respect to setup constraints (e.g. maximum path delay), requires that AAT at each node does not exceed RAT.
 - That is, for all vertices $v \in V$, we must have $AAT(v) \leq RAT(v)$.
- The slack of a node v is computed as:
$$slack(v) = RAT(v) - AAT(v)$$
 - Critical paths or critical nets are signals that have negative slack.
 - Non-critical paths or non-critical nets have positive slack.

IIIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, coming back to the slide, we for every vertex v belonging to capital set of vertices the requirement is the actual arrival time should not be greater than the required arrival time. So, for every node we calculate the slack the difference. So, slack can be positive slack can be negative. Slack positive means it is all right. We are not violating the constraint, but if it is negative which means that there is a problem. So, all paths with a negative slack they are refer to as critical paths. Because it is the critical paths we have to look at it very carefully, and try to adjust its so, that the slack again becomes 0 or positive. So, whichever path is having a negative slack this is our way of identifying which path is critical. Because you see there can be millions of paths the designer cannot tell that well these are my critical paths it is not possible.

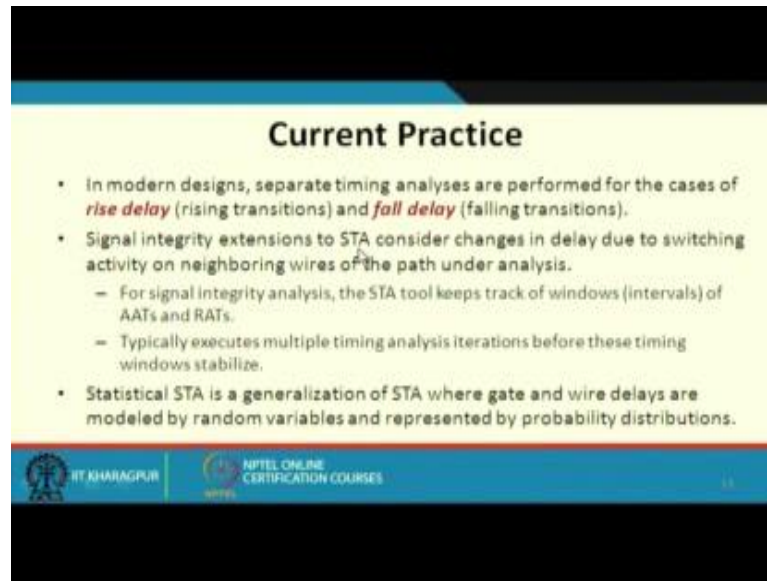
So, static timing analysis will give you a quick way of telling which of the paths are critical and which are not right.

(Refer Slide Time: 30:08)



So, this diagram actually summarizes. So, exactly what I worked out, in this diagram in blue I am showing the actual arrival times, in this brown I am showing the required arrival times, and in red I am showing the slacks, this a AAT minus this RAT minus AAT. So, you see that other than 2 nodes a and c all the other nodes are having negative slacks. So, I have to make some adjustments in my circuit. So, that these negative slacks become positive again this is what we want, but my static timing analysis gives me a result like this fine.

(Refer Slide Time: 31:02)



The slide is titled "Current Practice" and contains the following text:

- In modern designs, separate timing analyses are performed for the cases of *rise delay* (rising transitions) and *fall delay* (falling transitions).
- Signal integrity extensions to STA consider changes in delay due to switching activity on neighboring wires of the path under analysis.
 - For signal integrity analysis, the STA tool keeps track of windows (intervals) of AATs and RATs.
 - Typically executes multiple timing analysis iterations before these timing windows stabilize.
- Statistical STA is a generalization of STA where gate and wire delays are modeled by random variables and represented by probability distributions.

At the bottom of the slide, there are logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

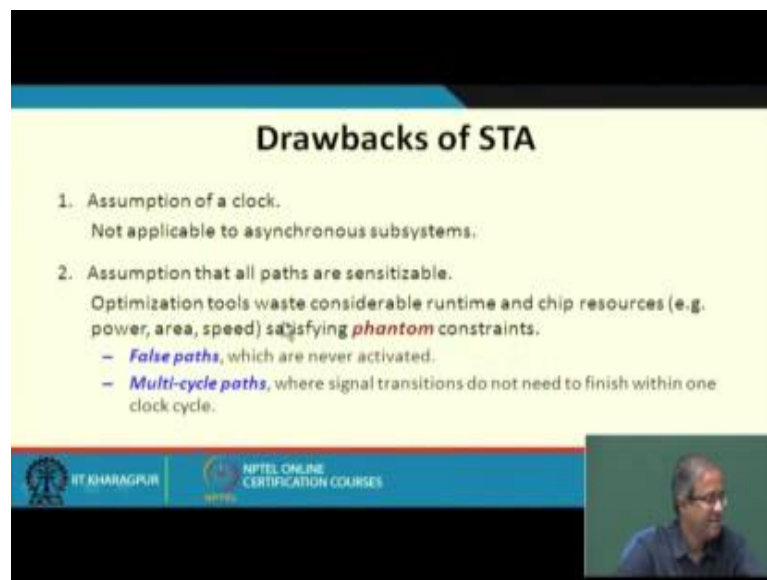
So, the current practice what you do nowadays is that not only the signal transitions and the delays, we separately look for rise time and fall time delays. Because you see I just mentioned during the last lecture, whenever there is a signal transition taking place in the output of a gate actually it is driving a load capacity.

So, it is either charging through the pull up network or discharging through the pull down network. Now the resistances of the pull up and pull down network may be different which means, different rise time and fall times. So, the delays when a signal is going from 0 to 1 and going from 1 to 0 may need not necessary be equal, they can be different also. So, a more realistic you can say means analysis of the delays should consider the different signal transition separately, because the way gates are designed gates are modeled those delays can also be different right. So, this is one thing you should separately consider rise delay and fall delay. And you can also you should incorporate signal integrate extensions to the simple analysis we just now consider, which means that if a neighboring wire changes state there will be a capacity of coupling on some wire and that wire can also incur some additional delays.

So, usually you need to keep some windows which means the neighborhood. You should try to keep track of which wires or which lines are near to each other. So, that transition in one line can affect the delays in the other line, that modeling is also required. And another method which is also used sometimes called statistical, static timing analysis

which means I am not fixing some values in the wire and gate delays, but I am giving a range. I am assuming it is a random variable and representing them by a probability distribution. So, probabilistically I am calculating the delays and these slacks. So, it gives a range actually not exact value it can give me a range.

(Refer Slide Time: 33:20)



Drawbacks of STA

1. Assumption of a clock.
Not applicable to asynchronous subsystems.
2. Assumption that all paths are sensitizable.
Optimization tools waste considerable runtime and chip resources (e.g. power, area, speed) satisfying *phantom* constraints.
 - *False paths*, which are never activated.
 - *Multi-cycle paths*, where signal transitions do not need to finish within one clock cycle.

IFT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, broadly speaking the static timing analysis is a very useful tool, but it suffers from a couple of drawbacks. First is that we assume that there is a clock, and we run timing analysis on the combination circuit block between pairs of registers or flip flops.

So, if there are asynchronous sub systems in a design which design require a clock. So, we cannot apply a state to those. Secondly, we are considering all paths to be equal; that means, there is a constant sensitization of a path we shall be looking at later. So, what you are saying is that we are trying to consider all paths and try to calculate RAT, AAT, satisfying some constraints some of them may be phantom constraints, phantom is virtual it can never happen. We consider that this path looks to be very long, let me try to optimize, but in practice what may happen signal will never flow to that path. So, if you can identify those path we can leave them out from our calculation.

So, maybe some of the efforts that we are putting in are not actually required right. So, there are 2 kinds of such thing, one is called false paths that are never activated. And some paths may be multi cycle paths. Which means the designer deliberately design in

such a way that between 2 flip flops signal transitions will require 2 periods, 2 clock periods not one. These are sometimes called multi cycle paths. So, in our next lecture we shall be looking at some of these draw backs and how to rectify them.

Thank you.