VLSI Physical Design with Timing Analysis

Dr. Bishnu Prasad Das

Department of Electronics and Communication Engineering

Indian Institute of Technology, Roorkee

Week - 12

Lecture 58

OpenSTA Static Timing Analyzer

Welcome to the course on VLSI Physical Design with Timing Analysis. In this lecture, we will discuss about OpenSTA which is a static timing analysis tool. So, the content of this lecture we will discuss about the background of the OpenSTA tool, then we will have a four lab example we will discuss in this lecture. So, we are discussing about what is the input and output of a static timing analysis tool. In this case, we have used OpenSTA, but however, these things can be applicable for any kind of static timing analysis tool. So, first of all, what are the inputs to the static timing analysis tool? We have basically dot lib which contains the timing information of the standard cell library. So, the dot lib file should be input to your static timing analysis tool because it contains all the timing information of the logic gates of the standard cell library. The second thing which is going to the static timing analysis tool is the netlist. So, here this netlist is maybe it is coming from the logic synthesis tool that is the gate level netlist or it can come from the physical synthesis tool which is having the information about the parasitics also. So, this netlist can be coming out of the logic synthesis tool or it can come from the physical synthesis tool.

So, that is the second input to your static timing analysis. Then we have some kind of timing constraints. So, the timing constraint we need to give it to the static timing analysis tool such as create clock, set input delay, set output delay, all these should be alSo, given as input to the static timing analysis tool. Then the fourth one is the script actually. Script means you have some kind of input conditions, all these alSo, go to the static timing analysis tool.

Then we have this file, this file is the post route parasitic extraction file that is the standard parasitic exchange format. So, that is alSo, going as input to your static timing analysis tool. So, all these files are going as input to the static timing analysis tool and what are the outputs coming out of it. So, we have timing reports and quality checks all are coming out whether my setup is meeting or not, my whole constraints are meeting or not, all these are coming as report. Then we have one more file SDF.

SDF stands for standard delay format which is coming out of the static timing analysis tool. This SDF file is used by verification tools for doing the dynamic timing analysis and it is improve the speed of dynamic timing analysis by using this SDF file or the standard delay format file. So, this is the place where you can get help related to STA. STA minus help will give all these. Then you can install the open STA which is coming from the open road installation and this is an example which is given in this location you can download.

And we can copy these files. These two files are needed one is the library and the example one. So, this is the lab one actually what you can say. So, then we can run this one in two mode one is with a SPEF file and So, here one thing here we will run this tool in two mode without SPEF file and this is with SPEF file. So, then we can see what is the difference in the report. If you can see this is without SPEF you can see the slack is 9.43 and with SPEF it is drastically reduced to 1.52 and if you can see the delay here is changed to large number here. So, this is the first lab and this is in the same lab we can dump the capacitance, slew, delay, timing information all in tabular format. For detailed reporting in the lab 2 we are reporting timing without SDF and with SDF.

Basically we are checking whether what is the impact using SDF without using the SDF. You can see here without using the SDF this is zero workload model you have a slack of 9.43 and with SDF you have a slack of 6.20. So, SDF is a standard delay format that has delay numbers that can be used at any stedge to speed up the run time. The SDF is used to do dynamic timing analysis for timing verification. By using the SDF tool the run time of the dynamic timing analysis tool is improved. Then we have another lab, lab 3 where we are doing timing analysis with on-chip variation and here we are using Derate factor what we discussed in our OCB class or CRPR class. So, this Derate factor will be used to accurately calculate the delay with on-chip variation effects. And if you can see So, this is the slack without Derate and this is the slack with Derate. So, in this case we are doing a lab experiment to check that what is the delay of the in2reg path on the output coming from the AOCs and what is the delay for raised to raised path. So, this is alSo, found it can be found out using this timing analysis tool. In all the 4 labs whatever we have demonstrated we have taken a very small design without using the clock tree, but in real design we have clock tree inserted inside the design. So, whenever you are doing the timing analysis with the clock tree inserted we need to use a command called set underscore propagated underscore clock to know the clock network delays in the timing reports. The command set underscore propagate underscore clock is used to show the clock network delay in the timing calculation reports.

So, we will discuss some of the SDC, SDC stands for Synopsis Design Constraint. So, this synopsics design constraints are basically predominantly used in static timing analysis. So, whenever you do any kind of design we need to give some constraint to the design and those constraints are read by the tools to optimize your design. So, we have to give some constraint to the tools basically to optimize the design. So, we will start with the clock which is the one of the fundamental signal that is used.

So, in this case we are using create underscore clock this is the command to set the clock constraint and here the clock period is 10. So, if you can take a clock here then the period of the clock is 10 nanosecond. So, nanosecond is defined in your dot lib files. So, what is the unit of time? So, if the time unit is something different then the corresponding number will be reflected. So, then if you can see here your on period is 5 this is 5 nanosecond. So, this 5 corresponds to this one and this 10 corresponds to this one. Then your gate port clock, So, gate port clock means that the clock what is there in your design. So, this clock pin what is there in your design which is going to all the flip flops it is going to this flip flop as well as this it is going to this flip flop. So, this is related to create clock constraint. Then we will go to the set driving cell. So, let us say I have a design which is driven by one cell. So, then I can decide which cell to be used as a driving cell. So, the constraint here is that set underscore driving underscore cell is the command and there are some switches here like hyphen lib underscore cell is the switch and what type of cell I am using this buff x2. This is the buff x2, this is the cell name of this buffer driving cell. Then it is taken from a library VLSI 18.

So, VLSI 18 is the standard cell library from where we are taking that buff x2. Then we have using the pin 3, So, this pin is basically your pin 3. So, this will be used to evaluate the design. So, this constraint is given to the design So, that the driving cell is used buff x2 and it is connected to the pin 3. So, next we will go to the set output load constraint. So, the command for setting the output load is set underscore load. This is the command. Then there are some switches are there. What is the value of the load? So, the pin underscore load is the one of the switch which is defining 0.2.

So, 0.2 what is the unit that is defined in your dot lib file usually it is pico forward, but it always you look into that file to check that what is the unit in the dot lib file. So, then your output pin, this pin is basically out underscore pin bracket 3. So, the corresponding buffer inside the design will be designed such a way that it can drive 0.2 pico forward load. So, this load is very important parameter whenever you are designing any kind of things using the logic synthesis tool. If you are not adding any load, it will not add any buffer. Otherwise, if you add the load, then it will use the logical effort method to calculate the effective driver which can drive this 0.2 pico forward load. Now we will go to the another constraint set input delay constraint. So, this is the command for the set input delay. And if you can see here, you have one thing is written clock. So, we have a clock signal is there. And with respect to the clock signal, this is a clock my IN1 is set.

How much delay it will be with respect to the rising edge of the clock because here it is written rising. So, this delay is let us say 2 nanosecond and this is 2 nanosecond.

So, the total period is 4 nanosecond and your input IN1 is rising here. So, from here to here, my delay is 2.3 nanosecond. So, this 2.3 nanosecond is defined here. So, this is with respect to the clock edge. What is the delay of the input pin IN1? Now I will explain the another constraint set output delay constraint. So, the command for set output delay constraint is this one. Then we have basically the clock with respect to clock. How much my IN2? IN2 is a pin between the blocks. Now I have the clock is going to all the gates, all the gates. So, if you can have a clock here. So, this is your clock signal. Let us say this is 2 nanosecond. I am assuming this it is not given here, but I am assuming 2 nanosecond. My IN2, this is the pin, this IN2 is basically how much it is delay with respect to the clock edge. So, this delay with respect to the clock edge, rising edge of a clock is basically 3 nanosecond. So, it will come in the middle of the positive half cycle. So, this delay with respect to the rising edge of the clock is 3 nanosecond. So, this is basically set output delay command which is coming output from a one block and going as input to the other block. All these five constraints are frequently used in logic synthesis tool and timing analysis tool. Similarly you have many constraints available in the Synopsys design constraint file, you can go through it. So, here we are going to explain one open source tool called OpenSTA. So, here we have different labs are there, we will explain one at a time. So, how to open this OpenSTA tool, you have to type STA.

So, you have to type STA. So, then it will go to the OpenSTA tool prompt. Then we will do one experiment, what is the impact of doing the static timing analysis with parasitic and without parasitics. So, here we are the first line is the read liberty, basically it reads the dot lib file from the library. The first line is reading the dot lib file and it is a slow corner dot lib file. So, here we will run that command. Then it returns once means it is able to read that file. Then the second command is we are reading the Verilog file dot v file generated from the logic synthesis tool YOSYS. So, this will be alSo, provided. So, it was reading the file was proper that is why it is returning one here. Then the third step is basically the link design to the top Verilog code. So, it will do the linking of the design. Now this step is alSo, successful. Now after doing this, we will add some kind of SDC command, Synopsis Design Constraint here. There are three clocks are there. So, all the clocks, what it does is that it creates a clock whose period is 10 time unit.

The time unit is defined in the dot lib file. If it is nanosecond, it will be 10 nanosecond. If it is picosecond, it will be 10 picosecond. So, three clocks will be created using this command. Now after you do this, then you have a set input delay. Basically how much delay it will introduce between the clock and the input pins. So, that will be given here. So, after you set these initial parameters, then you can do the report check. Then you can do the report check. It is basically report check is equivalent to doing the timing analysis.

So, this is very important command. So, this command, if we have discussed in our static timing analysis, there are two things.

One is called a data arrival time and data required time. So, your data required time is basically 9.84, but data arrival time is 0.41. So, that difference is basically the slag, what we discuss in our timing analysis lectures. So, since the slag is positive, then design is meeting the timing. However, there was one point what we have not included in this timing analysis is the parasitic extracted from the actual design. So, what we are doing here it is that this is the analysis without using the parasitics. If you use the parasitics, how the timing will change, we will look into that. So, here the slag amount is 9.43. Now if I go back, first we will do report underscore parasitics underscore annotation. So, it will report how many unannotated nets are present. Then we will basically read the spec file which is coming from the physical synthesis. DSPF is the detailed parasitic extraction format consisting of RC information required to calculate interconnect delays. Now we have a report parasitic annotation that reports how many unannotated nets are present with their RC information. Now after doing this, what we have to do is that we have to check how my timing is changing. How my timing is changing after including the spec file, SPF spec file. So, if you can see here, now my timing basically if you can see is drastically changed. Earlier my slag was 9.43. The difference between your data required time minus data arrival time. However after you do this including the parasitics into account, my slag becomes 1.52 which is more realistic because in actual design our parasitic needs to be included because in the actual chip inside the chip we have interconnects. We need to include that while doing the timing analysis.

So, this is basically there is one more thing we can learn here. We can do maximum timing analysis and minimum timing analysis also. This command will do the maximum timing analysis. We will get the same report because we are doing the maximum timing analysis here. Because it is it you have basically data arrival time and data required time, we will get the same report. But we have if we can set the accuracy to 4 digits, then this is the basically command for that. So, if you can see here, now my accuracy is increased to 4 digits. Earlier case it was only 2 digits. So, this is 2 digits only. Now after I introduce this command max hyphen digit equal to 4, now I can express my delay in 4 digits after the decimal point. Now there is alSo, a command to show the capacitance value. So, here you can see we can see the capacitance value here using this one.

So, you can see this cap values. So, here the cap values is alSo, mentioned in the report. So, that is alSo, very useful to know how much capacitance loading is there at different nodes. So, now after this capacitance thing we can alSo, list out the capacitance input slew, what is the input pin, what is the nets and fan out all these things together in the report.

So, this is the command for that. This is the command for that. You can see here we have capacitance. The first one is a fan out. Each fan out is one, capacitance is this one, then the slew at the input of the logic gate, then the delay, then the time. And there are two parts. One is data arrival time calculation. This is the part responsible for data arrival time calculation. This is the part which is required for data required time calculation. Then we have to do the subtraction from the data required time minus data arrival time. So, that will give me my slag. And the slag is positive means your design is passing the constraint. If the slag is negative means that there is some kind of violation in the timing constraint. So, now this is all about the max timing constraint or the critical path analysis. Then we will alSo, look into the hold timing analysis of mean path analysis.

It is input to raise hold path. So, here you can do this one using report. If you can go here report underscore checks hyphen path underscore delay mean. So, in for hold analysis we have to write mean. So, here there is one interesting point here is that what I already taught in my timing courses. Here you need to find a data arrival time here, then the data required time here, then the subtraction will be basically data arrival time minus data required time in case of hold check. So, that is the reason your slag is negative data arrival time minus data required time in case of mean timing analysis. So, if you can see here you have a hold violation slag is violated we need to fix that one. So, this is the lab one. So, we will do the second lab here. First of all we will open the opensta tool using the command sta.

So, now this is the command prompt. Then this is the script for this one. Here we are basically looking for what is the importance of SDF file. So, basically if you can see here this is the thing what is same as your previous example. First one is the read liberty file. Dot lib file first one. So, reading is successful that is why it is returning one. Then the second one is reading the verilog file. This is the example second example or example one. So, basically this is reading the verilog file. Then we are linking the design with the top level using the link design command. So, now after doing this you have create clock. So, here we are doing the create clock and all the input file basically variable setting in this command. I am doing both together. Then you will do the report check.

So, here we are not used the SDF file. There are two cases. Case one we are doing the timing analysis report check without SDF in the first case. So, without SDF this is the slag 9.43 is the slag. Data arrival time this is data arrival time. This is the data required time. So, now we have data required time minus data arrival time is 9.43 your slag is met. So, design is met. But now what we are doing is that we are taking the SDF file here example dot SDF file is there where the actual delay number and interconnects. So, here we will do the timing analysis considering the SDF file which is generated from the timing analysis tool. So, this SDF file we are using here and checking how much slag will be there. So, first we will do the report annotated check. Then we will do that here read stf for this file which is generated from the SDF tool. We are reading this one. Now we are doing report annotated check. So, here basically if you can see is that number of things are annotated is 3. Number of setup arc and hold arc is annotated is 3. Now we will do the report checks. So, report checks will do. So, if you can see here now we are calculating the data arrival time first then the data required time second then the difference of that will give me my slag.

So, this one here the slag is 6.2 time unit where if you can see the our slag is reduced compared to the previous analysis. Previous analysis we can see the slag is 9.43 because here we have not taken the SDF file into account while doing the timing analysis. But in this case we are taken the SDF file into account doing the timing analysis that is why my slag is reduced. But if you look into the design this is more accurate this taking the SDF file into account is more accurate compared to without taking the SDF file. So, there was a small difference compared to SPEP versus SDF file. But whenever you are doing the timing analysis considering SPEP file it takes more time to do the timing analysis. But if you dump the SDF file from the timing analysis and include that in doing the timing analysis that can do the timing analysis in less time. So, that is the advantage of using SDF file while doing the timing analysis. So, this is the basically lab 2 where we are explaining the importance of SDF file. Now we will go into the basically third lab experiment where we are discussing the de-rating factor. So, importance of here is that how can we include a de-rating factors while doing the timing analysis. So, here if you can see here first of all we will open the tool which is open sta. So, now here we will do the third lab experiment. The third lab experiment is related to de-rating factor what we discuss in our timing analysis. So, here we are define the corners WC, type and BC. WC stands for host corner, TYP stands for typical corner, BC stands for best corner. So, we will define the corners then we will read the three files here the host case is whenever we are reading the host case our library we should use the slow library. The WC stands for host case and we are reading the slow library. Now we are basically reading the typical library or the nominal library then we will read the first library. So, now after you do this then we can read the verilog file, the small verilog file whatever we are using for this experiment.

Then we will link the design, link design top. So, this will be linked to that design that is done. Then we will like the previous examples will create the clock and set input delay variables. So, that we will do here after this then we will go to the report checks. So, there are two cases are there one case without de-rate factor. One case without de-rate factor. So, this is the slag actually. So, we are calculating the data arrival time. Then data required time. Then data required time minus data arrival time. This is my slag. So, there is no basically de-rating factor is included in this analysis. Now in the second case we are including the de-rating factor hyphen early as 0.9 hyphen late as 1.1. What we discussed in our lecture. So, this one is basically taken here on the early case 0.9 and late case 1.1. Now what we have to do is that report checks path delay min max we need to do. So, here after doing all this you can see here there is a term called clock reconvergence pessimism removal.

So, here those things are included and you can see that my slag is reduced little bit 9.39. Earlier it is 4.3 earlier it is 9.43. Now it is slag is reduced 9.39 because of this on chip variation effects. So, if you can see here the data arrival time. So, data arrival time if you can see here you have more number of So, here it is 0.26 in the previous case this is basically 0. 0.23, 0.31, 0.41. So, here if you can see here it is 0.26, 0.35, 0.45 like that. So, there is a change in the data arrival time. So, due to which my slag is alSo, modified. So, this is very important to consider this common clock reconvergence pessimism removal component while doing the timing analysis.

So, this is the third lab exercise which includes the D rating factor. Now we will start this timing analysis tool typing STA. Now we will do a experiment basically we have already done the logic synthesis using the open source tool YOSIS. What we are doing is that our analysis can be reported differently using that output netlist. Basically whatever the gate level netlist we are getting from the YOSIS can be used to do the timing analysis using the this is open STA tool. So, here we are reading that verilog file the same thing whatever we did and here till this point is same. So, we are reading the file. So, this dot v file whatever it is coming it is coming from the logic synthesis tool YOSIS. Now what we are doing here is that we are doing the report check hyphen path delay max. So, if you can see here our delay is between your input pin to the register. The delay is between the input pin to the register. However it is not the actual critical path but we need to find actual critical path using from register to register. How we can do that we can we have to do this kind of analysis using this command. So, if we do this. So, if you can see here we our basically delays from a register to the register.

So, this is from one register to other register. So, in that method we are actually reporting the timing from register to register. So, which is more accurate compared to input pin to the register which is not which is happening in the previous case. So, which is not correct. So, this is actually the second case is the actual register to register delay. So, this is the way wait how we can basically read the reports of the timing analysis tool such a way that we can actually look into the actual critical path in the design instead of doing into a some basically unnecessary path which is not the actual critical path in the design. So, this is the main point in this lecture. So, we discussed four labs related to static timing analysis.

Thank you for your attention. Thank you.