**VLSI Design Flow: RTL to GDS**

**Dr. Sneh Saurabh**

**Department of Electronics and Communication Engineering**
**IIIT-Delhi**

**Lecture 27**
**Logic Optimization using Yosys**

Hello everybody. Welcome to the course VLSI Design Flow RTL-to-GDS. This is the tutorial for the sixth week. In this tutorial, we will be looking at logic optimization. Specifically, the objective of this tutorial is to gain hands-on experience on logic optimization using the open source tool, USES. And the requirement for this tutorial is that you should have USES installed on your system.

So, the installation and how to run USES was described in tutorial 5. So, if you have not yet installed USES on your system, please install by taking help of tutorial 5. And in this tutorial, we will also need a few files. The first one is the design file top.

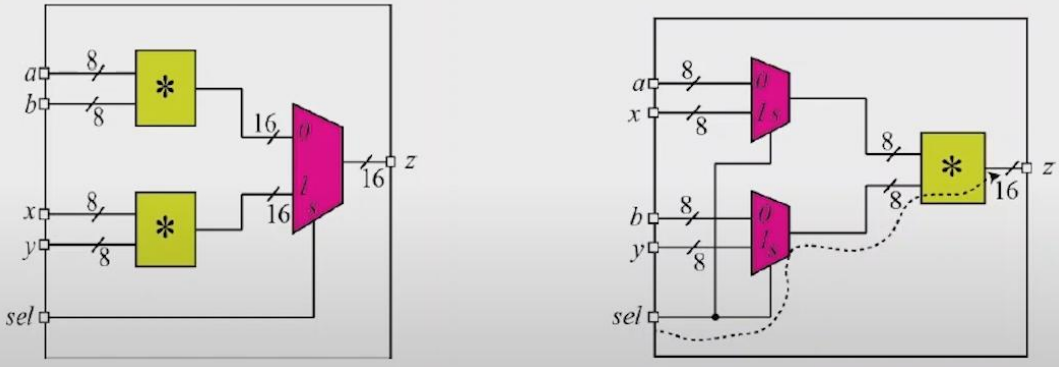
Then we need the script file opt.tcl and not underscore opt.tcl and technology library toy.library.

So, all these files are available on the NPTEL website as study material for week 6. So, you can download those files and then run the tutorial at your end by using these files. So, in this tutorial, we will be looking at an optimization strategy which is known as resource sharing. So, we have discussed resource sharing in lecture 13. So, resource sharing basically tries to save area in our design by using the same resource for multiple computation.

**Optimization:**

From Lecture 13 (Resource Sharing):

```
if (sel == 1'b0) z = a*b;
else z = x*y;
```




**Experiment:** Run Yosys and observe resource sharing and how it leads to decrease in area.

So, let me recap what we had discussed about resource sharing in lecture 13. So, suppose there is an RTL construct in which there is a statement like if select is equal to 0, then Z is assigned A multiplied by B else Z is assigned X multiplied by Y. So, if there is an RTL construct like this, then a direct translation of this RTL construct will lead to a circuit something like this. So, we have two multipliers, one multiplier corresponding to A star B or A multiplied by B and the other multiplier for the operation X multiplied by Y. And then using the select line either the output of one of the multipliers is selected and produced at Z.

So, this is the direct implementation, but we can actually save the resource that is the most costly resource in this circuit is the multiplier. So, the multiplier typically occupies a large area and therefore, we will want to use a smaller number of multipliers in our circuit. So, what the tool can do is it can transform the original circuit of this to the circuit which is shown on the right hand side. So, in this case we are using only one multiplier and the multiplier gets input through the multiplexer and either A and B or X and Y those are supplied as input to the multiplier and finally, Z produces either A multiplied by B or X multiplied by Y depending on the value of the select signal. Now, let us look into that or we carry out an experiment using open source tools to understand how resource sharing is done by the tool and what is its impact on the circuit area.

So, let us run the tool and see what happens. So, before running the tool let us see whether we have all the files that are required for carrying out these experiments. So, we have the script file and we have the design file and also the library. Now, let us first look

into the design file top dot v. So, I am using an editor gwim you can use any other editor also.

So, now if we look into this design in this Verilog code we have a top module whose name is top and it has only one module. So, this Verilog file has only one module and then within this always block we have the RTL construct that is what we talked about. So, we expect that the tool will actually optimize this and implement this circuit optimally by using only one multiplier. So, now let us first run the script that does not opt for TCL. So, not opting for TCL does not do the resource sharing optimization.

So, let us first see that if we do not do optimization what will happen. So, we now invoke yosys by typing in yosys. So, yosys is already installed on my laptop and that is why I can just invoke yosys and then we read the design, read the design translation processes that are always blocked to netlist and then clean up and do some cleanup to remove unused cells. So, we copy these lines of code and paste it in yosys. So, we run these commands.

So, now our design is loaded and no optimization is done. So, we can run the command show and see what our design looks like. So, when we run the command show then this is displayed. So, we can see that there are two multipliers one multiplier and the second multiplier and the inputs are a b and x y and it is going to multiplexer and then it is going to the d flip flop and to the out. So, this is an unoptimized version of our design.

So, we close this window and return to yosys share. So, yosys throws out quite a few warnings and some messages which we can safely ignore are related to the GUI that we open after running the command show. So, these are not important warnings or messages and we can simply ignore them. Now, once we have got this design what we do take mapping to internal cell library then we do mapping of the of the flip flops to the cells in our library and then map the rest of the combinational circuit elements to the cells in our library and then do the cleanup and then tell the tool that give us the statistics about the design. So, it will report the area.

So, we copy these all commands and paste it in the yosys share and run it. So, the design was synthesized and finally, it reported the area of the chip as 13,944. So, this is the chip area in the library units. Now, we can actually get the verilog netlist out of it by running this command. We copy this command and paste it in the yosys window.

So, we get the final netlist as netlist underscore final underscore unop dot b because we have not carried out the optimization. So, we exit the yosys shell and we can see whether our netlist was created. So, this netlist is created and this netlist contains

instances of the library cells. So, we can open this netlist and see what is this content. So, we can see that there are inverters, NAND gates and so on.

There will be flip flops and other cells from the library. So, there are flip flops also and multiplexers, there are some multiplexers and we also have flip flops DFF, R and Q. So, these are flip flops. So, the R circuit got synthesized and the instances from the libraries were picked and put in the netlist. Now, the next thing is that now let us run the script which is doing the optimization.

So, we open this file op dot tcl and then again invoke the yosys tool. And first we do the same thing that we had done for an unoptimized thing script that loads the design. And then convert the processes or synthesize the processes and then ask it to show the netlist. So, the design currently contains two multipliers as earlier. Now, after this what we do is that we carry out some more commands or we run some more commands in yosys shell.

So, these commands are opt and share aggressive manners. So, this command is basically sharing the resources. So, let us run these two commands. We copy these commands and paste it in the yosys shell. So, now we expect that some optimization must be done.

So, there are messages here for example, it is saying analyzing resource sharing options for multipliers. And then it is saying that according to the sat solver these pairs of cells can be shared. So, the tool actually inferred or detected an opportunity of optimization and it transformed our design to a new design. So, what is this new design? Let us look. So, we run the command show then it will display the updated design.

So, we can see that now we have only one multiplier. Only one multiplier is here and there are multiplexers which are selecting either A to X or B to B and Y. So, this is the optimization that we have discussed and the tool has done it. Now, we do the same thing that we had done earlier, which is to do the technology mapping and ask it to report the area and other statistics about the design. So, we copy this command and paste it in the yosys shell. Now, the chip area is being reported as 7,546.

So, remember that earlier it was around 13,900 or so. So, the A chip area has now decreased. So, this is what we expect out of resource sharing. So, to get the netlist we can run the command to write verilog. We copy this and paste here and now we expect that the tool must have generated this netlist underscore final underscore op dot v.

So, we do ls minus lrt to see the most recent file. So, the most recent file generated is

this. So, this is the optimized netlist in which there will be less number of gates than in an optimized unoptimized version. So, we can also see that the size of this file is smaller. So, the optimized netlist has got a smaller size compared to unoptimized one.

So, the number of gates in the optimized netlist is less. Its size on the disk is also less and the tool has done resource sharing. So, to summarize in this tutorial what we have done is that we have looked into how a synthesis tool, open source synthesis tool yosys performs resource sharing. So, we carried out an experiment to understand how resource sharing is done by the tool. So, now this is just one of the examples of logic optimization.

Now, you have got a library, you have an open source tool and you can write the very log code and carry out many more such experiments at your end which will help you understand this logic synthesis tool and its optimization in more detail. So, that is the end. Thank you very much.