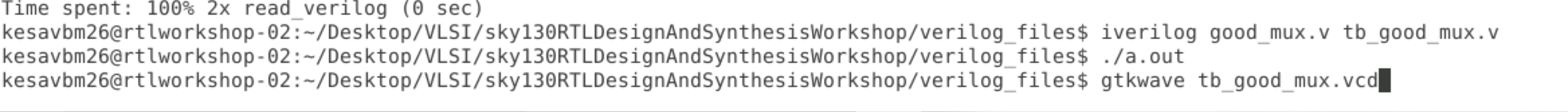
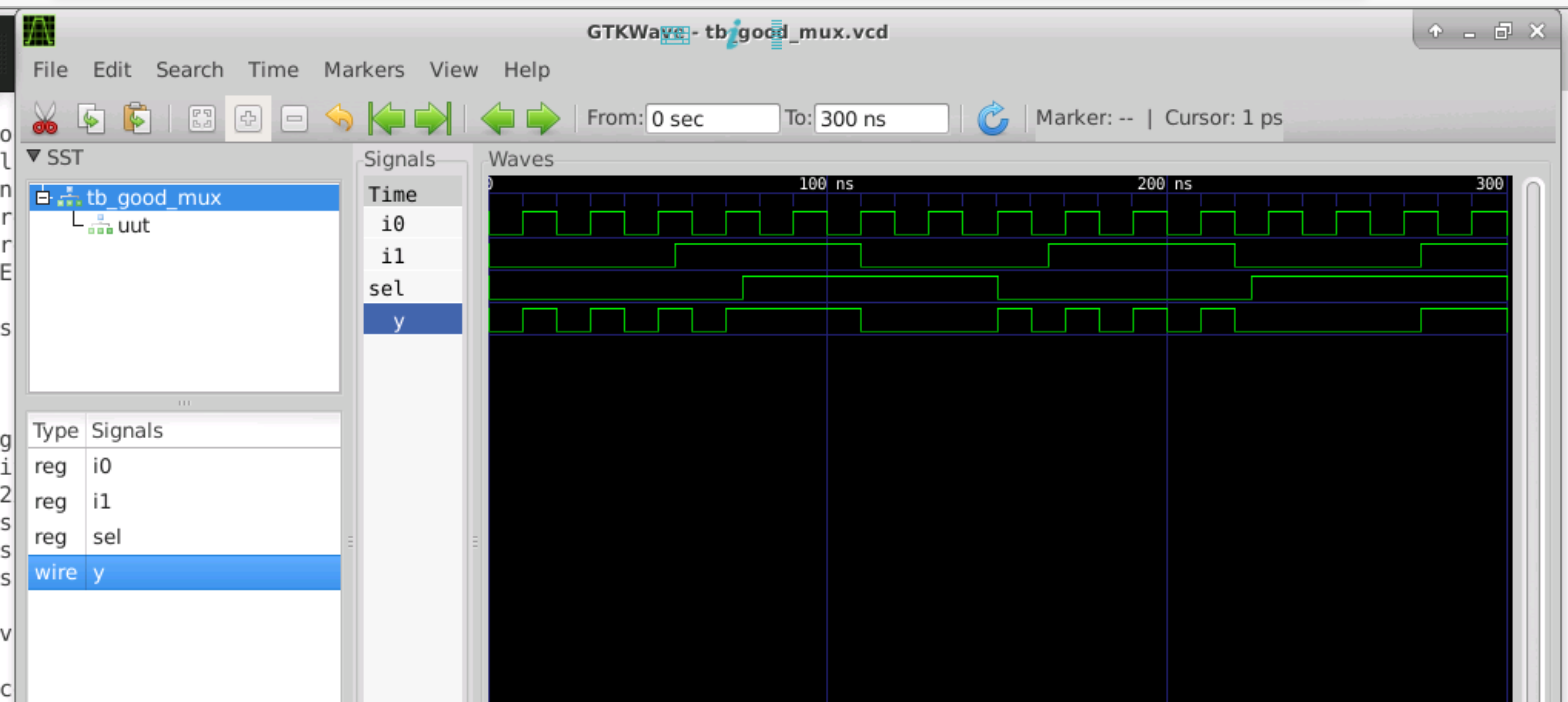
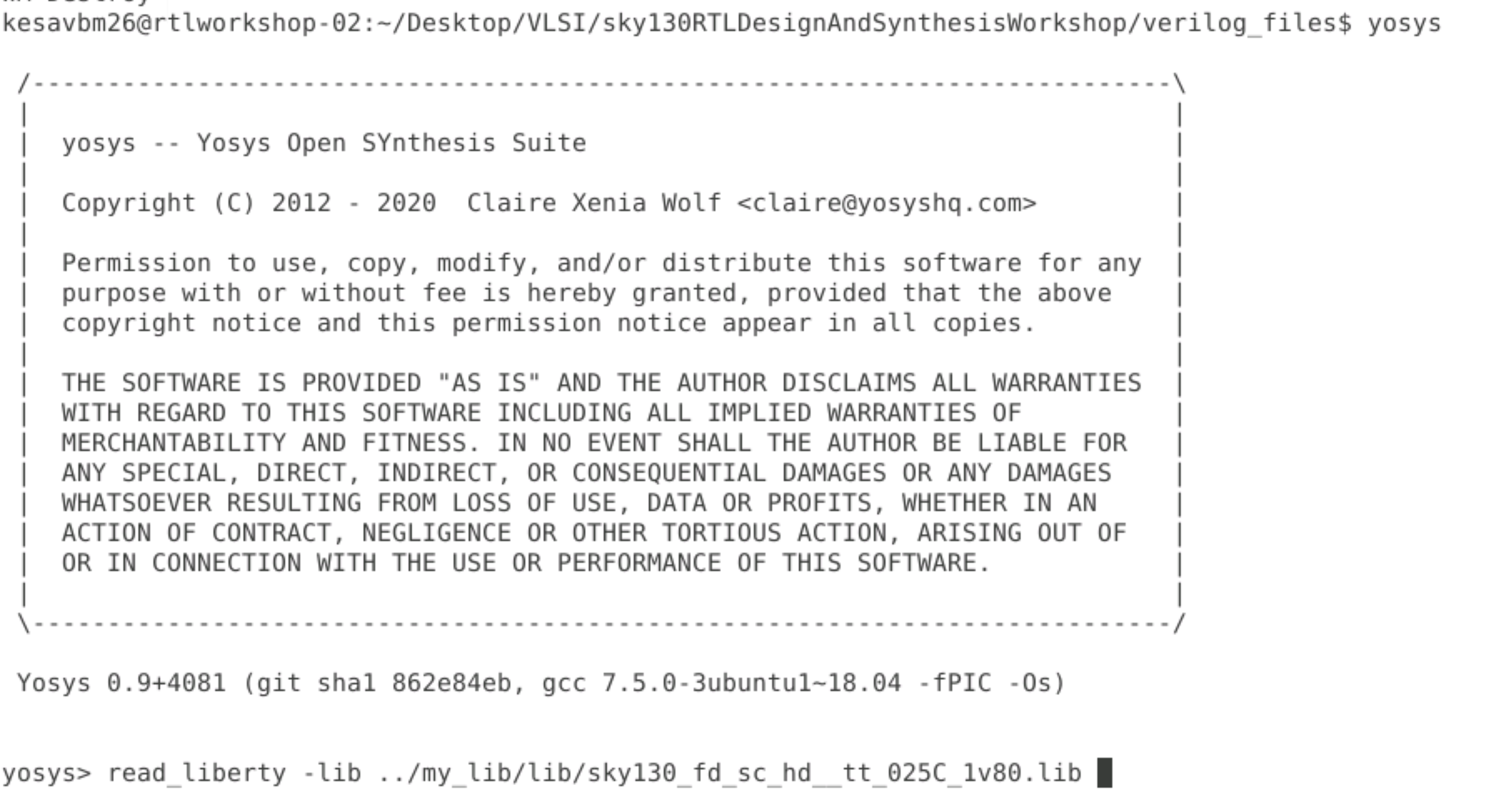
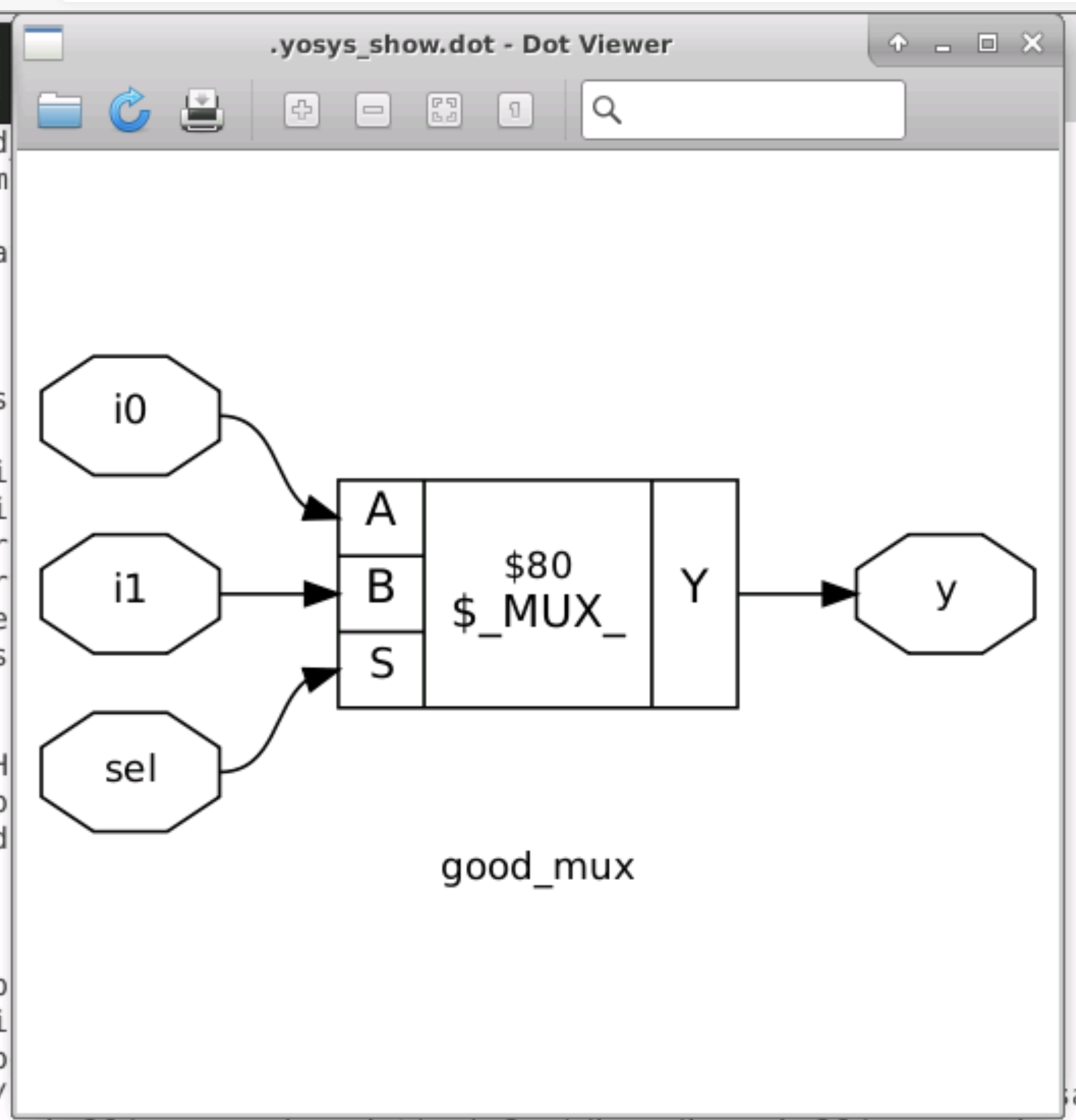
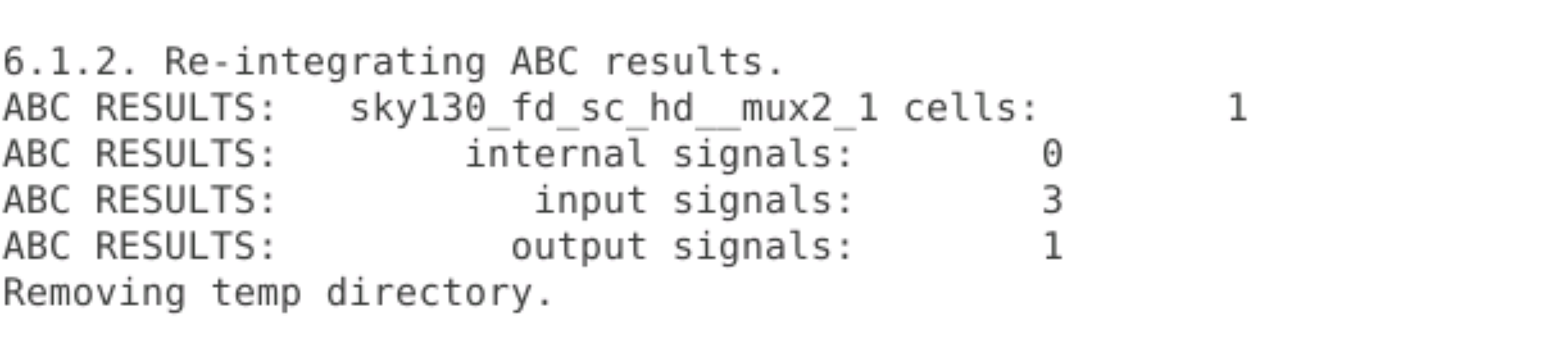
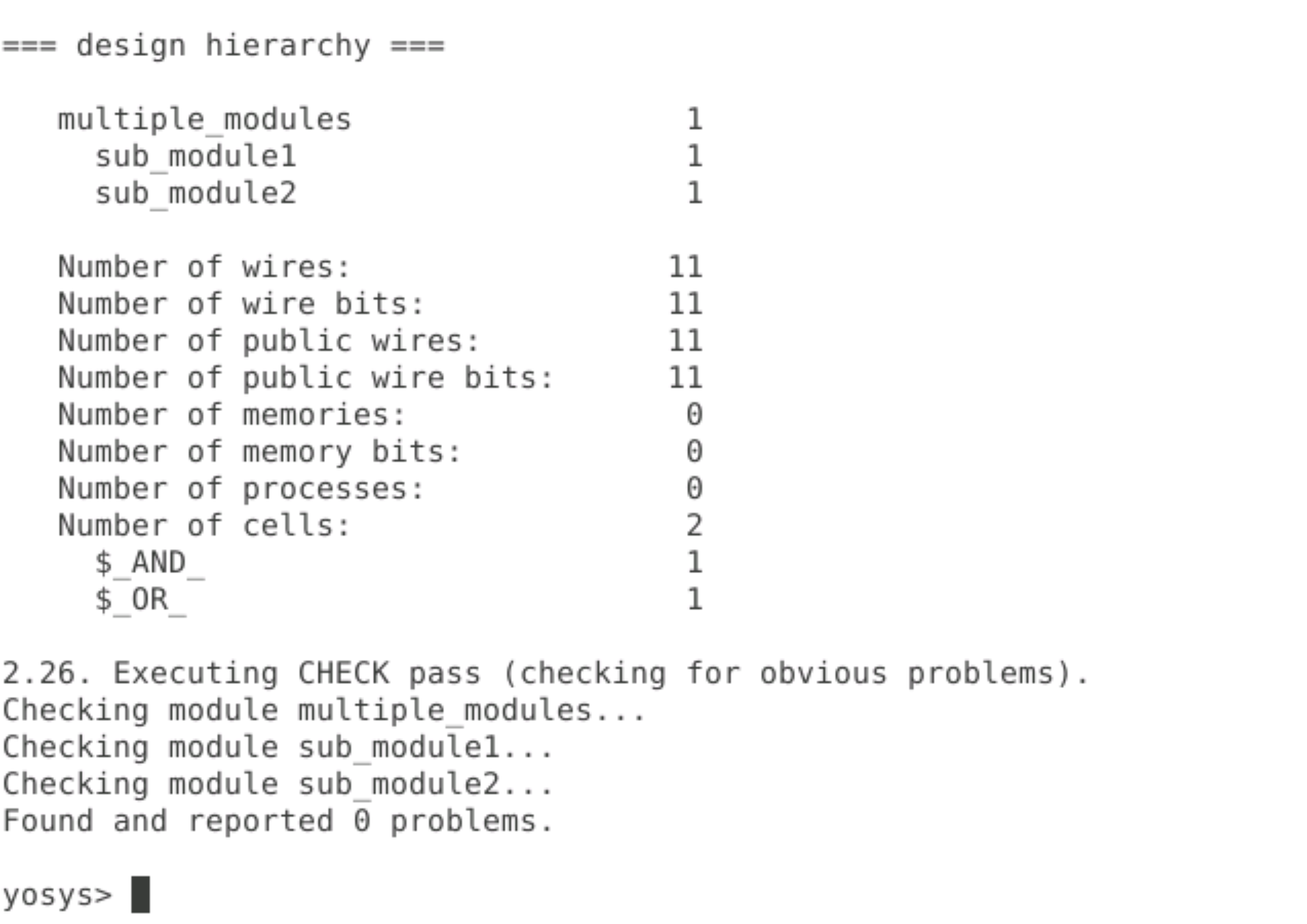
Understanding of following points took place at the end of five day workshop:

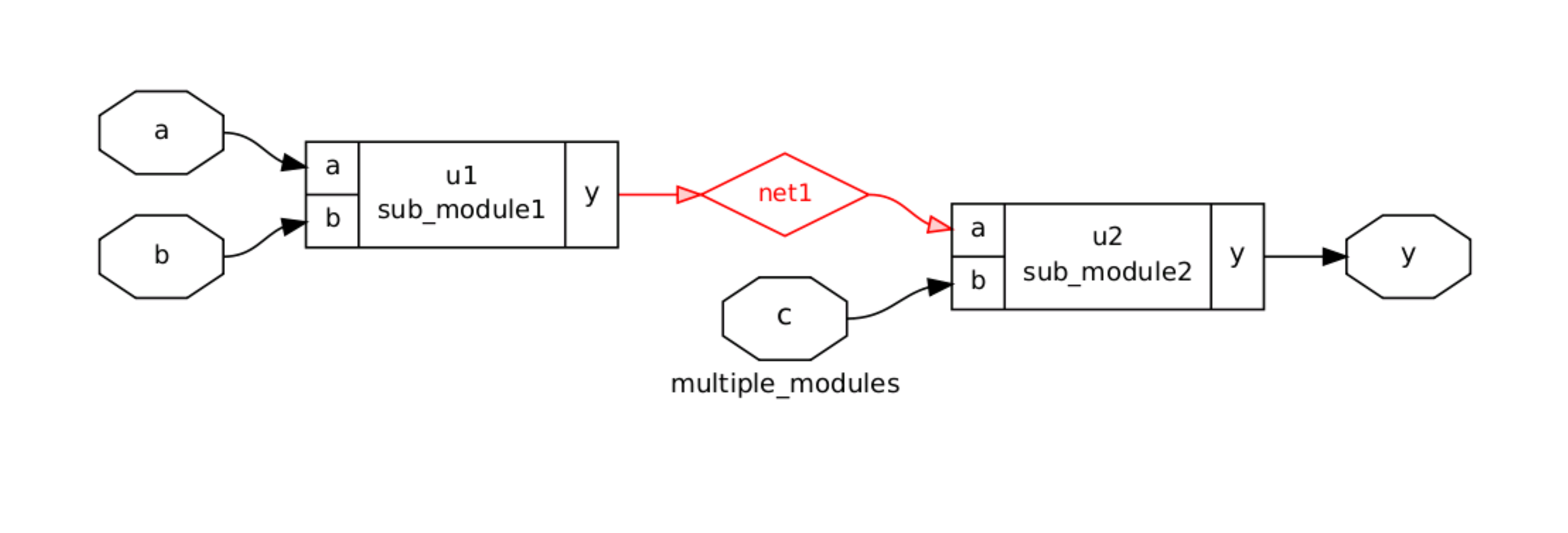
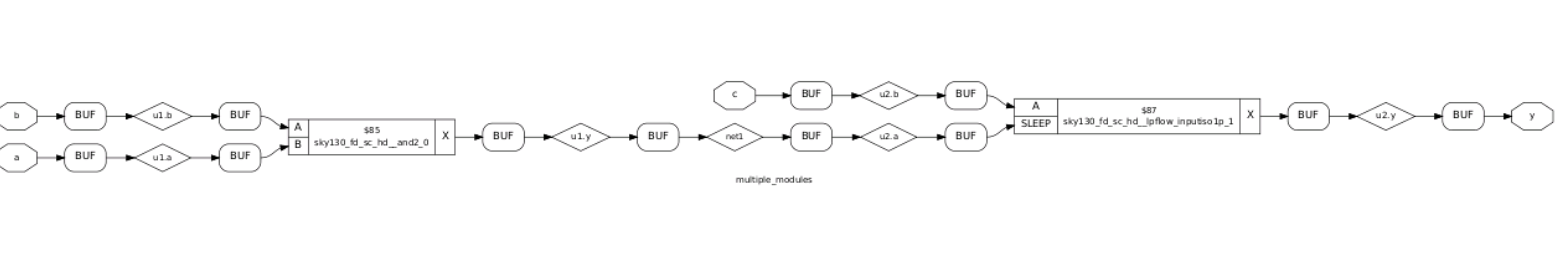
1. The fundamental definition of Simulator, Design and Test-bench was learnt.  
   How stimulator really works, based on changes in the input, corresponding changes in the output is computed. What is the role of testbench, was clearly understood.
2. Understanding of simulation flow using iverilog. Used and practised iverilog, in the linux terminal, to generate the value change dump format.
3. Once a.out is generated, developed an understanding of how it is executed, hence it would dump the vcd file.
4. The VCD file dumped was opned using the gtkwave viewer, where each signal was viewed, and the whole waveform was fit, to understand the functioning of the circuit.
5. Understanding of what Synthesis really means, why we really need a synthesis, how a mapping is done to a particular technology node (our case 130 nm), and its corresponding gates. So synthesis tool, which consumes the design and the liberty file gives out the netlist.
6. Understood that simulation at netlist level and at design level, functional requirements would be same and hence the same tb can be used.
7. Visually understood how the mapping was done and how the circuit was built using, show command after synthesis.
8. Understood the importance of fast, medium and slow gates (same functionality) and how it affects the performance, area and power consumption, their necessities because of setup and hold time constraints



1. Understanding of the cells are selected based on the requirements. Liberty file and the contents inside the file, the naming convention, and the definition of PVT was understood.
2. Hierarchical design, when it needs to be preserved and when flattending must be done. Understanding of their requirements individually was learnt. Logical effort required for stacked pmos is more and hence for optimization, NAND is preserved and used.
3. Understanding of the necessities of flip flops as storage element was learnt and the way they are coded ar RTL level. Along with the different resets and when the reset applied comes into effect was learnt.
4. Multiplication of 2 to the power n, would be shifting the number of binary bits to the right and appending 0’s at the end would result in the correct value, it doesn't require mathematical computation or lot of gates in terms of hardware was shown.
5. Understanding of combinational logic optimization and different techniques to achieve it was understood and illustrated.
6. Sequential logic optimization with the help of different techniques was analyzed and illustrated. And how these optimizations are done at the tool level, with the help of switches were discussed.
7. Different circuits involving D flip flop were illustrated indicating where the optimization can be done and where it won’t work, and how the tool is smart enough to perform it.   
     
     
   Hierarchical design:

  
  
Flatten:  


1. Understanding of gate level simulation, the necessity on why it must be done, due to synthesis and simulation mismatch. Understanding of how GLS is performed, with the help of gate level verilog models, and if they are timing annotated, how GLS can also help with timing validation. GLS fundamentally is to ensure the RTL - design aspect matches the post synthesis aspects of the design.
2. Understanding of how and when blocking and non-blocking statements needs to be used and how extra cautious one must be when using blocking statement. And whenever one deals with sequential circuits, non-blocking statements are advised. Various other issues with blocking and non-blocking statement was discussed.
3. Various illustrations and simulations were performed to show how synthesis and Simulation mismatch can take place.
4. Understanding on “IF” Condition and the way it implemented in RTL was illustrated, and how the first condition gets the priority was understood. And if incorrectly implemented, can lead to inferring a latch, which maybe acceptable in sequential circuits but needs to be avoided in combinational circuit at any cost.
5. Understanding of CASE statements, and how all the possibilities along with a default condition needs to be specified for correct implementation at hardware level was illustrated.
6. Two different kinds of for loop was illustrated, first for loop which comes under alwasys @, and is used for evaluation purpose. While the for generate loop is used for minimizing the manual effort to write a lot of RTL code, and hence for ease of hardware replication.