VHDL SYNTHESIS TOOL

Tutorial

last updated October 2, 1997

The purpose of this tutorial is as a follow-on to the VHDL simulation tools tutorial. In that tutorial, we set up the directories and files necessary to simulate VHDL. Then, we take a simple design, count3.vhd, and demonstrate how to compile it and simulate it through the use of a testbench. In this tutorial, we will take that simple design and synthesize it through the use of the Synopsys synthesis tools. Again, we will first have to add and modify the necessary files. Then, we will synthesize count3.vhd and go over the use of the synthesis tool.

I. Setup

The files can be found in my home directory in the Riggs 10 lab: *fstiver*.

First, copy the following files from *~fstiver/TA426/turorial/counter* to your *ece426/tutorial/counter directory*:

compile.script
.synopsys_dc.setup

Create the following directory in your counter directory: *WORK*

.synopsys_dc.setup is used by the synthesis tool to set up the paths to the proper libraries and to the desired *WORK* directory. It needs to be in any directory that you start-up the synthesis tool in.

The *compile.script* file is not required by the synthesis tool. It controls the synthesis tool through all of the steps of the synthesis process. The synthesis tool can be controlled manually from either the pull-down menus or the command line. However, many of the steps of synthesis are difficult to understand and go beyond the scope of this tutorial. Therefore, we will use the *compile.script*. (The other reason for using the *compile.script* is that synthesis can take from 5 minutes for a simple design to over 1 hour for a complex design. It would be a tedious process to input the separate commands by hand when the *compile.script* does it automatically.)

First, edit the *.synopsys_dc.setup* (remember this is different from the *.synopsys_vss.setup* used for simulation). Go to line 29 and adjust the path in order to point to your local *WORK* directory. Typically, for every part that you synthesize you should synthesize it in its own directory with its own *WORK* directory. It is not necessary for the file containing the actual part to be in the directory that you are synthesizing in. If the file containing the part to be synthesized is in another directory, you just need to set up the proper search path (line 4 *search_path = {*). This can be helpful when you have a hierachical design with several levels of instantiation or a design with multiple parts instantiated into a single part. The remaining parts of this file point to the proper libraries for the synthesis tool and to the Xilinx libraries (Xilinx is a major manufacturer of fpga's and the synthesis tool needs to know what architecture it is synthesizing for).

Now, edit the *compile.script* file. At this level it is not necessary to understand what this script is doing. Some of the commands are easy to decipher, while others require a deep understanding of the synthesis tool. Go to line 4 and set *TOP* = ***filename*** of your top level design (minus the *.vhd*). Modify line 6 to be your name. If there is only one part being synthesized, this is all you need to do. However, if you have a design that includes several parts instantiated in it, you need to make some additional modifications. Between lines 7 and 9, include the following line for each part in order from lowest level of the design to top level:

analyze -format vhdl ***filename.vhd*** (this time we do include the *.vhd*). (You do not need to include this line for the top level part because it is already taken care of by line 9.)

Once you have made these modifications, you are ready to synthesize your part.

II. Synthesis

To invoke the Synopsys synthesis tool, use the following command:

fpga_analyzer &

The synthesis tool will eventually come up although it might take a little time depending on the speed of the machine you are working on. A window titled: *Synopsys Design Analyzer* will come up. At this point there are only 2 menus you have access to. Using the *Setup menu*, select the *Command Window…* option. This will bring up another window titled: *Command Window*. When you initiate the *compile.script*, this will show you the commands that are being executed and any warnings. (Note: The synthesis tool produces many, many warnings even with a perfect design. It is a good idea to look at them, but, if you part simulates properly, you can typically ignore them.) Next, using the *Setup menu*, select the *Execute script…* option. This will bring up a window titled *Execute file*. Select *compile.script* from the list and select **OK**. Once you select **OK**, the synthesis tool will start work. The command window will start displaying the commands being executed and their result. If you part compiles and simulates correctly, the only errors you will usually receive here are a result of an improper or not included search path in the *.synopsys_dc.setup* or an improper part name or not included part in the *compile.script*. If you receive an error here, the synthesis tool will typically display an error window which describes the problem. Select the cancel script option, quit the synthesis tool (File menu, quit option) and troubleshoot your error.

When the synthesis tool completes operation, it displays your part in the work area of the *Synopsys Design Analyzer* window. It starts with a top-level design and you can work your way down to the actual logic created by double clicking on the design. At the base level, the tool displays the flip flops, logic and hard macros used to create your design. The hard macros are colored red. For example if you used a + or - in your design the synthesis tool uses an adder/subtractor or a incrementer/decrementer that was already designed and optimized by Xilinx. For a large design, you may have to zoom in order to get a good look at your design. To zoom, click your right mouse button. Then select Zoom with your right mouse button. You can then zoom using your left mouse button. To return to a full view, click your right mouse button and select Full View with your right mouse button. You can move around the hierarchy of the design created using the buttons on the left of the window. (Note: practice moving around in your design using these buttons and Zoom/Full View.)

In order to look at the warnings from the synthesis tool, input the following on the command line of the *Synopsys Design Analyzer* window:

check_design

The warnings will be displayed in the command window. You can scroll up this window to look at the warnings. Once you are finished with this, go ahead and quit out of the synthesis tool (File menu, Quit option).

At this point, the synthesis tool has created all of the files required by the Place and Route tool. The options we selected in the *compile.script* are for Xilinx parts. These options created the netlist (.sxnf files) needed by the Place and Route tool. The Place and Route tool can now be used to create a bit file needed to program the desired fpga with your design. (Note: Of course there are some other considerations I have left out, i.e. how do you lock down which pins on the fpga are used, the interface with the host, etc. But, for the scope of this tutorial, once you have synthesized your design, you could then place and route it and create a design for a fpga.)

The synthesis tool also creates several other files that can be helpful. The *command.log* file is a copy of everything that was displayed by the Command Window (Instead of scrolling around the command window, you could look at the warnings displayed by the *check_design* command here). If you have an error, it is usually helpful to look at this file to help in troubleshooting. Another file created is the ***filename****.fpga* which gives a listing of the logic required to implement your design. For larger designs this can give you an indication of whether your design will fit into your selected fpga. Finally, we also created a file named *ygtest.rpt.* Two of the commands issued by the *compile.script* was *check_design* and *check_timing* and the output of the commands was redirected to this file. (Note: The only files required by the Place and Route tool are the ones containing the netlists (.sxnf). All of the other files just contain information about the operation of the synthesis tool and the part created. They may or may not be useful.)