

Synplicity Synplify -- FPGA Synthesis

Bug List

5/18/01

Bill Lenihan
Raytheon Systems Co. (formerly Hughes Aircraft Company)
2000 E. Imperial Hwy., R2-V514
El Segundo, CA 90245
310-334-8324
wlenihan@west.raytheon.com

Contact the author for testcases.

date	case #	bug #	Problem	Solution / Workaround
9-20-00	22529	10379	<p>I want to pass initial values into a Xilinx Virtex Block RAM, essentially making it a ROM instead of a RAM.</p> <p>The code only seems to pass the INIT_00 value, and will not pass the other 15 INIT values (01 to 0F), even if I use backslash, comma, semicolon, etc., as a line extender.</p> <p>If I put the entire synthesis directive on one line it will pass all the values to the edif netlist, but then the code becomes unreadable. What is the correct way to extend a synthesis directive over more than 1 line?</p>	<p>This is a known bug in the Synplify software.</p> <p>In order to use the xc_props attribute or the parameters, you need to have the arguments all on one line.</p>
9-25-00	22726	16434	<p>I'm synthesizing a design into virtex-E. One of my outputs is an LVDS signal (I assign in scope).</p> <p>I notice in the EDIF netlist produced that the output is assigned w/ an OBUF_LVDS, but it is still a single-ended signal, not the differential pair (P & N sides) that LVDS should have. Will this be fixed in a future</p>	<p>This is a known issue with synplicity. We currently have a bug open on it.</p> <p>Attached is the example to instantiate LVDS.</p>

date	case #	bug #	Problem	Solution / Workaround
			release? If so, is there a Bug number assigned to it?	
9-19-00	22438	22522	<p>QUESTION:</p> <p>I know how to set the default clock frequency for a specific project: after I establish the project I go to</p> <p style="margin-left: 40px;">Project Implementation Options Global Constraints Frequency (Mhz) = ____</p> <p>But how do I establish a global default frequency for every project, so that it's there automatically. (i.e., right now, every new project has default frequency = 0 Mhz (but uses 1 Mhz as default in timing analysis!), but I want them all to start w/ a default frequency of 100 Mhz). How can I set that?</p>	<p>As of now the default frequency for Synplify is 0 MHz. It can not be changed to 100 MHz as a default. We have filed an enhancement bug on this issue.</p>
9-19-00	22438	N/A	<p>I thought the top-level design to compile was the one at the bottom of the project's verilog folder. But sometimes Synplify compiles another design in the hdl file list -- one that isn't at the bottom. How can I force Synplify to synthesize the design I want?</p>	<p>For verilog, top level module does not necessarily have to be at the bottom of the project's verilog folder. If you want to have a particular module as a top level module, please do the following:</p> <p>In synplify project window, click on implementation options and you will see a verilog tab. If you click on the verilog tab, you will see a Top Level Module window. Please type the module name in it and run synplify. Synplify will use that module as a top level and will select verilog files from your project accordingly.</p>
10-06-00	23323	23481	<p>The synthesis pragma: /* synthesis xc_map="lut" */ does not always work. Sometimes Synplify maps the logic into other resources like MUXF5/6. Since one leg of the MUXF6 must be</p>	<p>put the pragma /* synthesis syn_keep = 1 */ on output wires from xc_map="lut" components.</p>

date	case #	bug #	Problem	Solution / Workaround
			<p>fed by the MUXF5 in the same slice, RLOCs will sometimes fail with this mapping. [RLOCs are often used in conjunction to direct the mapping.]</p> <p>Synplify must map to LUTs when encountering /* synthesis xc_map="lut" */</p>	
04-04-01	31615	31637	<p>See code below (uncomment each assign statement, one at a time, and try synthesizing). Why does the Exclusive-NOR (a ~^ b) pass, but expressions involving the NAND (a ~& b) and NOR (a ~ b) operations fail in Synplify? Is this a temporary oversight in Synplify, or a permanent part of the un-synthesizable subset of Verilog?</p> <pre> module gates (a, b, gatel); input a, b; output gatel; //assign gatel = a & b; //assign gatel = a b; //assign gatel = a ^ b; //assign gatel = a ~& b; // FAILS //assign gatel = a ~ b; // FAILS //assign gatel = a ~^ b; //assign gatel = ~(a & b); //assign gatel = ~(a b); //assign gatel = ~(a ^ b); endmodule </pre>	<p>You have found a bug in the Verilog compiler. I have already submitted bug#31637 to address this issue. To answer your question, it is a temporary oversight.</p> <p>Regards, David Synplicity CAE</p>