

Do's and Don'ts for Writing Verilog  
14:332:437 Concepts in Digital Systems Design  
Fall 2004  
Electrical and Computer Engineering Department  
College of Engineering  
Rutgers University

Prof. Michael L. Bushnell

November 4, 2004

Please follow these practices in writing your Verilog code:

1. NEVER set the same variable from two different *always* blocks – each *always* block must never drive a signal that another *always* block is driving. This is equivalent to short-circuiting two logic gate outputs.
2. The blocking and non-blocking assignment operators are drastically different. The blocking = operator changes the signal value immediately. The non-blocking <= operator changes the signal value at some time in the future (like after the *always* block stops executing). So, this code malfunctions:

$$\mathbf{c} <= \mathbf{a} + \mathbf{b}; \mathbf{d} <= \mathbf{c} + \mathbf{e}; \tag{1}$$

because the new value of **c** (**a + b**) is not what is used in the statement **d <= c + e** – instead, the old **c** value is used. Corrected code:

$$\mathbf{c} = \mathbf{a} + \mathbf{b}; \mathbf{d} <= \mathbf{c} + \mathbf{e}; \tag{2}$$

3. Wire names are a problem. Declaring signals as *wires* means that the electrical wire has a name. When you connect two modules to the wire, one should drive the signal and the other should read it. This is the mechanism for connecting the hardware that you designed to the testbench. The connection is done in a higher-level system module that makes an instance of the testbench and an instance of the design, declares the signals connecting them as wires, and indicates which port of which instance is connected to which wire.
4. Do not use **always @(\*)** for your *always* block sensitivity list unless you mean the *always* block to be combinational.
5. The Verilog translator often turns a sequential *always* block into a combinational one, even when you intended it to be sequential. The rules are: An *always* block is combinational if every execution path through it sets the same outputs and if all inputs to the *always* block are in the sensitivity list without edge specifiers. However, some sequential *always* blocks with edge specifiers in the sensitivity list are also treated as combinational. You can prevent

this by adding another input to the *always* block that is never referenced in the block with an edge specifier in the sensitivity list. This will force the block to be sequential, because now there is a path through the block for the other input that does not set all of the output variables.