|  |  |
| --- | --- |
| **Student**: | Michael Escue |
| **Assignment**: | Assignment 6 |
| **Class**: | ECE 351 |
| **Professor**: | Dr. Garrison Greenwood |
| **Term**: | Spring 2019 |

**Questions:**

1. **Consider the following code:**

always @(A)

if(A < 10)

Y = 0;

else if (A > 10 & A <= 15)

Y = 1;

* **Assume A can assume a value between -2 and +22.Why would a synthesizer believe a latch is necessary? Simply saying there is no “else” clause is not the answer. That causes a latch to be inferred but doesn’t explain why the synthesizer puts one in the design. I want to know why a latch is necessary.**

1. **What are three ways of preventing latch inference in case statements?**
2. **Why should module outputs be registered?**
3. **Consider the following module:**

module U1(in1, in2, y ,clk);

input in1, in2, clk;

output y;

reg s;

always @(posedge clk)

s = in1 ^ in2;

assign y = ~s;

endmodule

* **Edit the code so the module has a registered output.**

1. **Consider the following Verilog code:**

if(A)

y = 1;

else if(B)

y = 2;

else if(C)

y = 3;

else if(D)

y = 4;

else

y = 0;

* **Show what would most likely be synthesized.**

1. **Rewrite the code from the previous problem as a case statement. Show what would be synthesized if the parallel\_case synthesis directive is used.**