|  |  |
| --- | --- |
| **Student**: | Michael Escue |
| **Assignment**: | Assignment 7 |
| **Class**: | ECE 351 |
| **Professor**: | Dr. Garrison Greenwood |
| **Term**: | Spring 2019 |

**Questions:**

1. **3rd party IP provides three files: a netlist; a behavioral wrapper; and a file showing the instantiation syntax. What is the purpose of the behavioral wrapper?**

As the top level of the design, wrappers allow maintain design hierarchy and maintainability between platforms with the option to swap instantiations at a lower level. They essentially connect multiple modules and are at the top of that hierarchy with each module connected within it.

1. **Why are black boxes synthesized?**

Black boxes are synthesized to use IP which has obscured functionality but provides an interface for instantiation. Black boxes are usually 3rd party or Vendor IP which has predictable performance and guaranteed functionality but is only provided as an EDIF file and must be instantiated using black-box instantiation methods/directives.

1. **Xilinx recommends control signals in modules be coded so they are active high. Why?**

Active low signals use more LUTS because they require inversion before driving a control port of a register. Only clock signals are automatically inverted when required. This inversion results in a LUT input being used for each control signal that is not inverted. Because the inversion cannot be combined into a single inverter block, a LUT is also required for each hierarchical design block. This also causes timing problems and consumes more power.

1. **In a Xilinx FPGA what is a control set? What does it contain?**

A control set is a group of signals such as enable, set, reset, and clock signals used for controlling a single slice. Other examples of control signals in a set are clock enable, gate enable, write enable, set, preset, reset, clear, clock, gate, and slice related write enables, chip select, synchronous reset, and clock signals.

1. **Xilinx says designs with asynchronous resets may use excess LUTs and registers. Briefly explain why.**

Certain dedicated hardware resources don’t have and don’t support an asynchronous reset. When an asynchronous reset is used, inferring resources such as multipliers, adders, subtractors, MACC, counters, comparators, shifters, multiplexer, pattern match, and others must be implemented using design fabric LUTs and flip flops. Also an asynchronous reset cannot be moved to the data path to optimize control set size and merge similar registers.