Name: Junzhe Wu Net id: Junzhew3

1. What is the difference between local parameter and parameter in Verilog? When would you use either one of these declarations?
   (1) The default value of normal parameter can be changed when the module is instantiated. But the value of local parameter cannot be changed.
   (2) We use normal parameter to define a set of attributes for the module which can characterize its behavior as well as its physical representation.
   We use local parameter to protect some important value from accidental or incorrect redefinition by an end-user.
2. What is the difference between wires and registers in Verilog? List several syntax rules for both wires and registers?
   (1) Registers represent data storage elements. Registers retain value until another value is placed onto them. Values of registers can be changed anytime in a simulation by assigning a new value to the register.
   A Wire is an asynchronous connection between the PC and an HDL endpoint and no values get stored in wire.
   (2) They can both be connected to input port of a module instantiation.
   They can both be used as outputs within an actual module declaration.
   They can both be used to model combinational logic.
   They can both appear on the right-hand side of assign statements and **always@** block **=** or **<=** signs.
3. When designing a finite state machine module, when would you use a wire and when would you use register?
   (1) Using wire:
   Wire elements can be used to connect output port of a module instantiation, but register cannot.
   Wire elements can be used as inputs within an actual module declaration, but register cannot.
   Wire elements are the only legal type on the left-hand side of an assign statement.
   (2) Using register:
   Register is the only legal type on the left-hand side of an **always@** block **=** or **<=** sign.
   Register is the only legal type on the left-hand side of an initial block **=** sign (used in Test Benches).
   Register can be used to create sequential logic, but wire cannot.
4. What is the difference between blocking and non-blocking assignments in Verilog?
   (1) Blocking assignments happen sequentially. In other word, if an **always@** block contains multiple **=** assignments, you should think of the assignments being set one after another.
   (2) Non-blocking assignments happen in parallel. In other word, if an **always@** block contains multiple **<=** assignments, which are literally written in Verilog sequentially, you should think of all the assignments being set at exactly the same time.