CSE370: Introduction to Digital Design
Winter 1999

Homework Set 7

DUE: Feb 26, 1999, 12:30 pm

Please show *all* of your work. Solutions not involving DesignWorks do not have to be typeset, but may be if desired. In any case, your solutions must be legible.

1) In DesignWorks or by hand, show how to implement a T flip-flop starting with a D flip-flop.

2) Design a circuit that asserts its output (sets OUT=1) whenever it detects the pattern 010 in a continuous serial input data string, and de-asserts its output (sets OUT=0) whenever the pattern is not 010. The inputs to your counter are data (the serial input data) and clock. The output is the single bit OUT. Use D flip-flops and some combinational logic. Assume that all your flip-flops have true and complementary outputs (i.e. Q and Q'). Turn in your state diagram, K-maps, circuit, etc. DesignWorks is optional.

3) Design a 4-bit binary coded decimal (BCD) counter that counts in the following sequence: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001 and then back to 0000, 0001, etc.

   a) Draw the state diagram.
   b) Draw a state-transition table showing the present state, next state, and flip-flop inputs assuming both D-type and T-type flip-flops.
   c) Draw K-maps for the flip-flop input functions, and minimize the combinational logic.
   d) Implement your counter using D flip-flops. Draw a schematic in DesignWorks, and verify your design.
   e) Implement the counter using T flip-flops. Draw a schematic in DesignWorks, and verify your design.
   f) Modify your counter from (d) to make it self-starting. Draw a schematic in DesignWorks, and verify your design. If it is already self-starting, simply explain why this is so.
   g) Implement your counter in Verilog using a "case" statement. Turn in your Verilog code.

4) Using D flip-flops and combinational logic, design a 4-bit register that has two 4-bit data inputs, a load input (*load*), and a select input (*sel*). *load* tells the register to load new data, and *sel* chooses which of two possible inputs (A or B) the register loads. Design your register as a DesignWorks schematic. Turn in your schematic, and a clearly labeled waveform that showing your circuit operating correctly.

DesignWorks hints:

1. You can use $display statements to show what is going on inside your Verilog modules. You have to put them inside an "always" block.
2. In your Verilog modules, you can display vectors that correspond to your input and output ports using the %h specifier (%b denotes binary, %d denotes decimal, and %h denotes hex). Put the $display statement at the bottom of your always block.
3. Turn in printouts of your circuits, subcircuits, and Verilog models as Grant suggested on the mailing list.