EE577B Homework #2

Verilog

Due: 9/27/2001

1. Verilog Concepts

1-1 Write a behavioral description that swaps the values in two registers without using a temporary register. The new values should appear #2 after the positive edge. Complete the following module.

```
module swapIt(clk)
      reg [15:0] A, B;
      input clk;

      always @(posedge clk)
            // here is your part..

end module
```

Submission: swapit.v, test_swapit.v, signalscan output waveform

1-2 Design 5 different behavioral Verilog models for2-input OR gate and simulate them. Find properly working model(s) for transport delay and properly working model(s) for inertial delay. Explain why?

```
Model 1:  LHS blocking
      #4 O = (A | B);
Model 2:  LHS non-blocking
      #4 O <= (A | B);
Model 3:  RHS blocking
      O = #4 (A | B);
Model 4:  RHS non-blocking
      O <= #4 (A | B);
Model 5: Continuous assignment
      assign #4 O = (A | B);
```

Submission: or2.v, test_or2.v, signalscan output waveform, answer for the questions

2. Verilog Coding

2.1 Modeling a 5-bit counter

```
A. Use the following module header and the port interface for the counter:
      `timescale 1ns / 1ns
      module counter(cnt,clk,data,rst,load);
            output [4:0] cnt;
            input [4:0] data;
            input clk, rst, load;

B. Use the following specifications to create the 5-bit counter.
      - The counter has an asynchronous low-asserted rst pin. When the counter resets,
      the output is 0 after a delay of 3 ns.
      - When load is high and rst is high, the counter is loaded with the value on
      the data pin, at the positive edge of the clock with a delay of 3 ns.
      - When load is low and rst is high, the counter advances, at every positive
      edge of the clock, with a delay of 4 ns.

C. Testing with a stimulus file
      - Provide a clock with a period of 20 ns.
      - Assert rst for 20 ns from the start to verify that counter resets.
      - Allow the counter to count until 5 and then load it with the hex 1D.
      - Increment the counter 5 times.
      - Finish the simulation 10 ns after this.
```

Submission: counter.v, test_counter.v, signalscan waveform

2.2 Modeling a RAM with bi-directional data bus

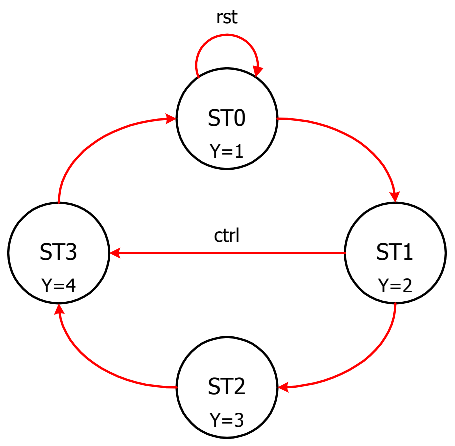
```
A. the following module header and the port interface for the RAM:
      `timescale 1ns / 1ns
      module mem(data, addr, read, write);
            inout [7:0] data;
            input [4:0] addr;
            input read, write;

            reg [...] memory [...];

B. Use the following specifications.
      - The MSB of each word should be bit 7.
      - The memory data bus is bi-directional.
      - At the positive edge of the write signal, the value on the data bus should be
      written into the memory location addressed by the addr bus.
      - When the read line is asserted, the contents of the memory location addressed
      by the addr bus should be continually driven to the data bus. (Use a continuous
      assignment.)
      - The memory should be capable of block reads; if the addr bus changes while
      read is asserted, the contents of the new memory location addressed should be
      immediately transferred to the data bus.
      - When the read control line is not asserted, the memory should place the data
      bus in a high impedance state.

C. Test your memory design using the test_mem.v stimulus file located at
      ~ee577/ee577bb/hw/test_mem.v
      - Observe the errors that are reported and answer the following question.
            Why is the bi-directional bus data declared as data type wire?

            What should happen if data were declared as data type reg, so that the
            stimulus could use procedural assignments to place values on the data bus?
      - Correct these errors by modifying the test_mem.v file.
```

```
	- Run the simulation again to verify the correct functionality of the memory.
```

Submission: mem.v, modified test_mem.v, answer to the questions, signalscan waveform

2.3 Simple FSM design

Using the following state diagram, implement in verilog using SM-NSL-OFL coding style. (See page 8 of Structural and RTL Verilog Examples)



```
A. Use the following module header.
      module FSM(clk,rst,ctrl,Y)
            input clk, rst, ctrl;
            output [2:0] Y;
            reg [2:0] Y;

            parameter [1:0] ST0 = 0, ST1 = 1, ST2 = 2, ST3 = 3;

B. rst is asynchronous reset. ctrl signal is sampled synchronously.
C. Use ~ee577/ee577bb/test_fsm.v for your test bench.
```

Submission: fsm.v, signalscan waveform

3. Verilog-XL with schematic composer

Complete the handout #7, #8 about Verilog-XL integration with schematic composer.

Submission: signalscan waveforms for each of handout.