Alexandria University

Faculty of Engineering

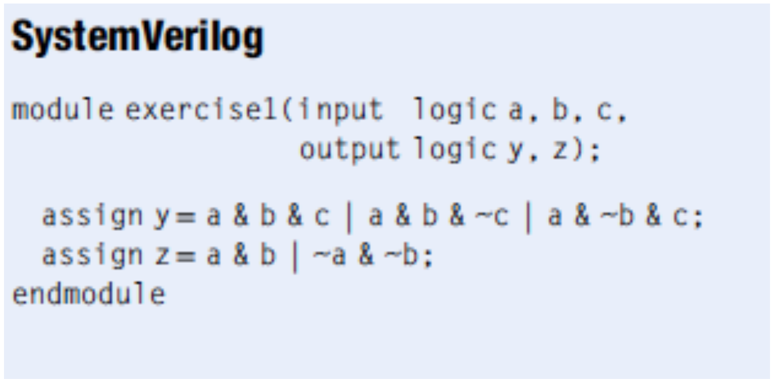
Computer and Communications Department

CC322: Computer Architecture

Sheet 0: Hardware Description Languages

Exercises 1-13 from the reference book: "Digital design and computer Architecture by David Harris, 2nd Edition"

1. Sketch a schematic of the circuit described by the following HDL code. Simplify the schematic so that it shows a minimum number of gates.

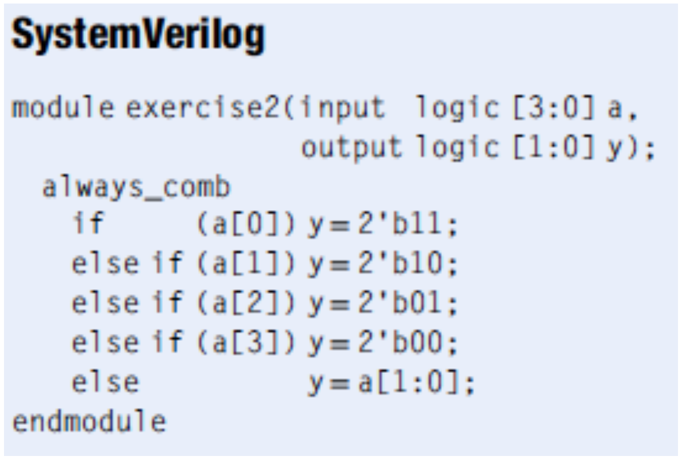


SystemVerilog

```
module exercise1(input  logic a, b, c,
                 output logic y, z);

  assign y = a & b & c | a & b & ~c | a & ~b & c;
  assign z = a & b | ~a & ~b;
endmodule
```

2. Sketch a schematic of the circuit described by the following HDL code. Simplify the schematic so that it shows a minimum number of gates.



SystemVerilog

```
module exercise2(input  logic [3:0] a,
                 output logic [1:0] y);
  always_comb
    if      (a[0]) y = 2'b11;
    else if (a[1]) y = 2'b10;
    else if (a[2]) y = 2'b01;
    else if (a[3]) y = 2'b00;
    else           y = a[1:0];
endmodule
```

3. Write an HDL module that computes a four-input XOR function. The input is $\mathbf{a}_{3:0}$, and the output is **y**.

4. Write a self-checking testbench for Exercise 3 Create a test vector file containing all 16 test cases. Simulate the circuit and show that it works. Introduce an error in the test vector file and show that the testbench reports a mismatch.

5. Write an HDL module for a hexadecimal seven-segment display decoder. The decoder should handle the digits A, B, C, D, E, and F as well as 0–9.

6. Write a self-checking testbench for Exercise 5 Create a test vector file containing all 16 test cases. Simulate the circuit and show that it works. Introduce an error in the test vector file and show that the testbench reports a mismatch.

7. Write an 8:1 multiplexer module called mux8 with inputs $S_{2:0}$, d0, d1, d2, d3, d4, d5, d6, d7, and output y.

8. Write a structural module to compute the logic function, y = ab + b c+abc, using multiplexer logic. Use the 8:1 multiplexer from Exercise 7.

9. Repeat Exercise 8 using a 4:1 multiplexer and as many NOT gates as you need.

10. Write an HDL module for an eight-input priority circuit.

11. Write an HDL module for a 2:4 decoder.

12. Write an HDL module for an SR latch.

13. Write an HDL module for a JK flip-flop. The flip-flop has inputs, clk, J, and K, and output Q. On the rising edge of the clock, Q keeps its old value if J = K = 0. It sets Q to 1 if J = 1, resets Q to 0 if K = 1, and inverts Q if J = K = 1.