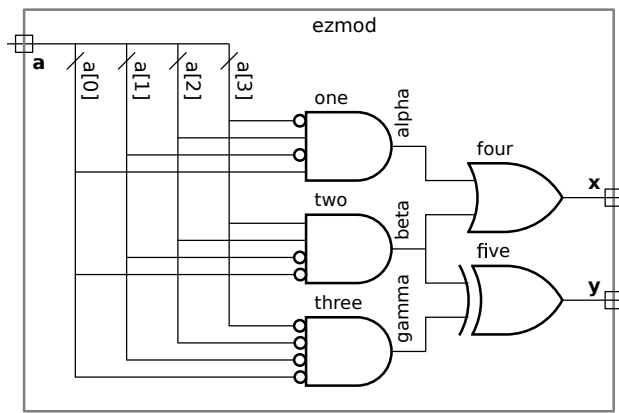


The questions below can be answered without using EDA software, paper and pencil will suffice. Please turn in the solution on paper. Homework 2 will require the use of Verilog implementations. Nevertheless, runnable SystemVerilog code for this assignment can be found at <http://www.ece.lsu.edu/koppel/v/2016/hw01.v> (plain Verilog) and <http://www.ece.lsu.edu/koppel/v/2016/hw01.v.html> (syntax-highlighted HTML).

Those who are rusty about the correspondence between Verilog code and hardware might want to look at the solution to EE 3755 Fall 2013 Homework 1, at http://www.ece.lsu.edu/ee3755/2013f/hw01_sol.pdf.

Problem 1: Show a Verilog explicit structural description of the module illustrated below.



- Base the names of ports, wires, and instances on labels in the illustration.
- Of course, use only primitives and wires. See Table 28-1 of IEEE Std 1800-2012 for a list of gates.

Problem 2: Answer the following questions about Verilog primitives as defined in IEEE Std 1800-2012. (See Chapter 28.) Indicate the exact section number where the answer is found.

(a) The standard provides a `not` primitive and a `nor` primitive, among others. One can easily argue that a 1-input `nor` gate is the same as a `not` gate. Does the standard actually allow Verilog code to instantiate a 1-input `nor` gate?

(b) Based on the standard, is there anything that can be done with a `not` primitive that can't be done with a 1-input `nor` primitive? (Don't try to answer this too deeply, just show an instantiation.)

Problem 3: Output match of module `is_1133`, shown below, is 1 iff its input `d` (digits) is 1133 in BCD (which has the same representation as 1133₁₆). The module instantiates BCD digit detection modules `is_1` and `is_3`.

```
module is_1( output wire match, input wire [3:0] d );
    wire z321;
    nor o0(z321,d[3],d[2],d[1]);
    and a1(match,z321,d[0]);
endmodule
```

```
module is_3( output wire match, input wire [3:0] d );
    wire z32;
    nor o0(z32,d[3],d[2]);
    and a1(match,z32,d[1],d[0]);
endmodule
```

```
module is_1133( output wire match, input wire [15:0] d );
    uwire          m1, m2, m3, m4;

    is_1 i0(m1, d[15:12]);
    is_1 i1(m2, d[11:8]);
    is_3 i2(m3, d[7:4]);
    is_3 i3(m4, d[3:0]);

    and a1(match, m1, m2, m3, m4);
endmodule
```

(a) Draw a diagram of `is_1133` based on the explicit structural description above. Show the insides of the `is_1` and `is_3` modules. Label the diagram using the same wire and instance names used in the Verilog descriptions.

(b) Design a module `is_1133_is` that does the same thing as `is_1133`, but that uses implicit structural code. The correct solution requires adding only one line to the shell shown below. Don't forget that the value in `d` is in BCD.

```
module is_1133_is( output wire match, input wire [15:0] d );
```

```
endmodule
```

Problem 4: When completed the output of module `is_1235` is 1 iff the input is 1235 in BCD.

```
module is_1235( output wire match, input wire [15:0] d );
```

```
endmodule
```

(a) Complete the module. The module must be explicitly structural except for the use of the concatenation operator (see Section 11.4.12). The module **must** use `is_1` and `is_3` to detect the digits. Do not assume or design an `is_2` or `is_5` and don't put in logic to detect those digits.

(b) Draw a diagram of the completed module, which should be very similar to the diagram from the previous problem.