CAN 'OUT' EVER BE A 1?

**Timing Specifications**

- \( T_{\text{setup} \& \text{thold}} = 0.1 \text{ ns} \)
- \( \text{clock} \rightarrow Q = 1.0 \text{ ns} \)
- All gate delays = 1.0 ns
- Clock period = 100 ns
Reduce the following Boolean expression to a minimal sum of products form:

\[ F = \overline{A} + \overline{B} + \overline{A} \overline{B} \]
Design a 4-bit complex binary counter with the following function, implement it.

<table>
<thead>
<tr>
<th>Load</th>
<th>Count</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>C = old C</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Load Parallel</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Up Count</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Down Count</td>
</tr>
</tbody>
</table>
Which of the following is the right Syntax for blocking / non-blocking assignments?

Circle all correct answers.

a) assign ps = A;
b) assign ps <= A;
c) always_comb begin
   if (reset)
      ps = A;
d) always_comb begin
   if (reset)
      ps <= A;
e) always_ff @(posedge) begin
   case (in)
      'b0 : ps = A;
f) always_ff @(posedge) begin
   case (in)
      'b0 : ps <= A;
Implement with flip flops and gates a string recognizer that outputs true whenever it sees 1111 or 1001.
Perform the operations and identify if there's overflow.

Assume sign magnitude:

\[ 01111 - 10101 \quad 10111 - 11001 \]

Assume 2s complement:

\[ 01111 - 10101 \quad 10111 - 11001 \]
Design Make a XOR gate using a 2:1 mux.
Determine the functionality of this circuit, and simplify it to as few basic gates (AND, NAND, OR, XOR, etc.) as possible.
Design a circuit that takes in a 3-bit number in S-M and outputs the equivalent 2's comp. number.
Draw a state diagram for a 2-bit input where the output is true when in the last 4 bits of input including the current input, only 2 bits are '1'. At startup, goes to '00'.
Implement $F = \overline{AB} + AC + \overline{AC}B$ using a 3:8 decoder. Use extra gates where needed.
What are the legal ranges for third and the clock period for this circuit?
How many 2:1 muxes would you need to construct a 4 input lookup table? (using only 2:1 muxes)

What about an 8 input LUT?
Draw an XNOR circuit using only NOR gates
Here is the block diagram of a 1 input finite state machine. Implementing it by Verilog.
module aModule #(parameter WIDTH=3) (clk, reset, iniVal, incr, mode, out);
    input logic clk, reset, incr, mode;
    input logic [WIDTH-1:0] iniVal;
    output logic [WIDTH-1:0] out;
    logic [1:0] value;

    always_comb begin
        case (mode)
            1'b0: value = 2'b01;
            default: value = 2'b10;
        endcase
    end

    always_ff @(posedge clk) begin
        if (reset) out <= iniVal;
        else if (incr) out <= out + value;
    end
endmodule

For the verilog given above, implement the circuit. The above module is instantiated in another module as shown below, and it is optimized for minimum resource utilization:
module test (clk, reset, in, out);
    input logic clk, reset, in;
    output logic [1:0] out;
    aModule #(WIDTH(2)) a
    (.clk, .reset, .iniVal(2'b10), .incr(1'b1), .mode(in), .out);
endmodule
Given the following timing diagram, draw a combinational circuit diagram that conforms to the timing diagram.

- Inputs are A, B, and C. Intermediate signals are X and Y. Output is Out.
- All signals in the timing diagram must have a correspondingly labeled wire in the circuit diagram.
- Assume all gates have a gate delay of one time unit. Each grid line in the diagram is one time unit wide.
Implement a 3-bit counter that starts counting up to 7. When it reaches 7, it should start counting down to 0. When it reaches 0, it should start counting up to 7 again. This cycle should repeat indefinitely.

Whenever the counter is at a prime number (2, 3, 5, 7), the system should output 1. Otherwise, the system should output 0.
a) Find the function to tell whether a 4-bit sign-magnitude number \((A_3A_2A_1A_0)\) is divisible by 4. Then implement the function with a 4:16 decoder.

b) Repeat part (a) using a 4-bit 2's complement number \((A_3A_2A_1A_0)\).
a) Design a half-adder using 4:1 MUX

b) Using half-adder, design a full adder
A local merchant wants a sticker machine for her store. Stickers cost 50 ¢. This machine has one coin slot, which will only accept nickels, dimes, and quarters. This machine has one output, Give_Sticker. Note, this machine does not give change, but will accept a sum greater than the cost of a sticker, in which case the machine should retain the balance. Draw the corresponding state diagram for this machine.