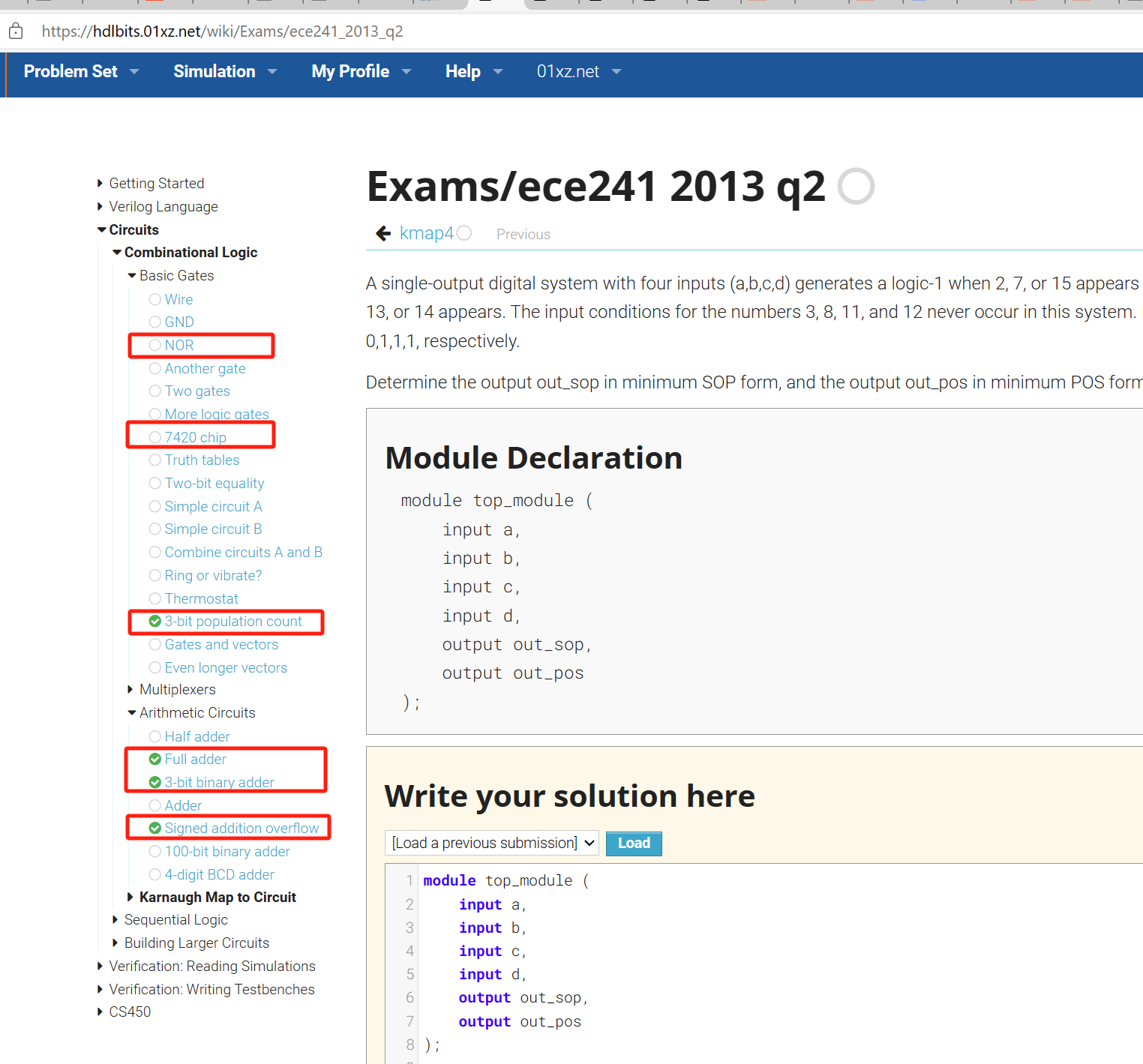
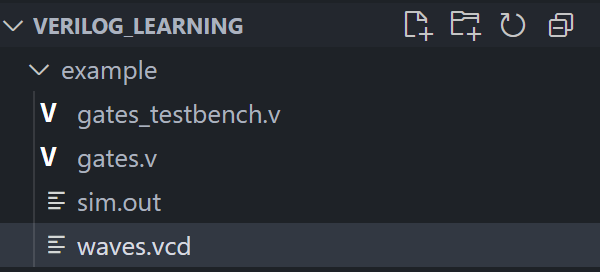
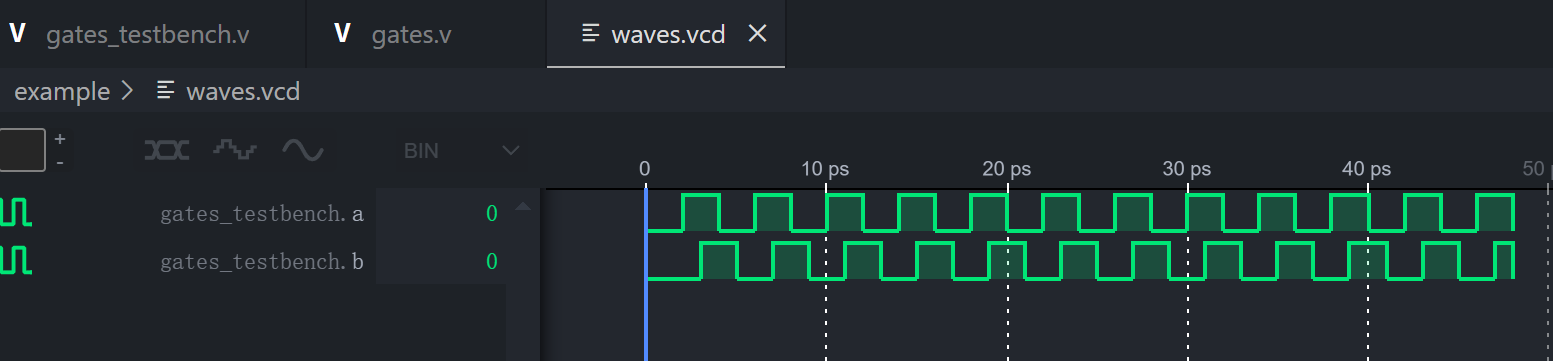
# Homework-Verilog-1

Visit <https://hdlbits.01xz.net/> and solve the following problems：NOR，7420 chip，3-bit population count，full adder，3-bit full adder，signed addition overflow. [Problem sets - HDLBits](https://hdlbits.01xz.net/wiki/Problem_sets#Combinational_Logic)

Upload your answers (with snapshot of the simulation result on <https://hdlbits.01xz.net>) to the course webpage on canvas system.



1. Configure a verilog simulator environment on your personal computer (e.g. Icarus Verilog/verilator + GTKwave) and get familiar with its basic usage.



1.NOR gate

module top\_module (

input wire in1,

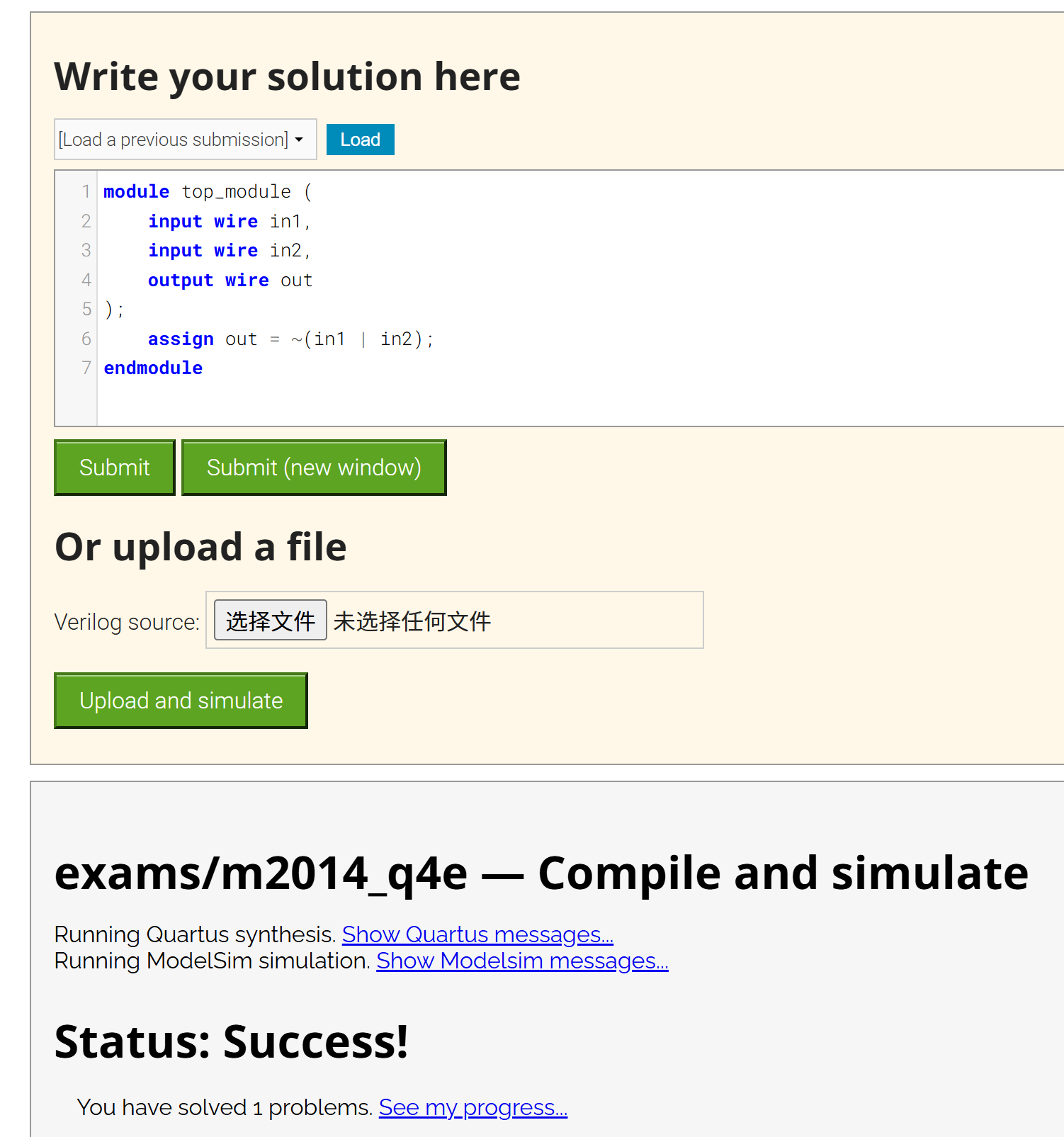
input wire in2,

output wire out

);

assign out = ~(in1 | in2);

endmodule



2.chip7420

module top\_module (

input wire p1a, p1b, p1c, p1d,

output wire p1y,

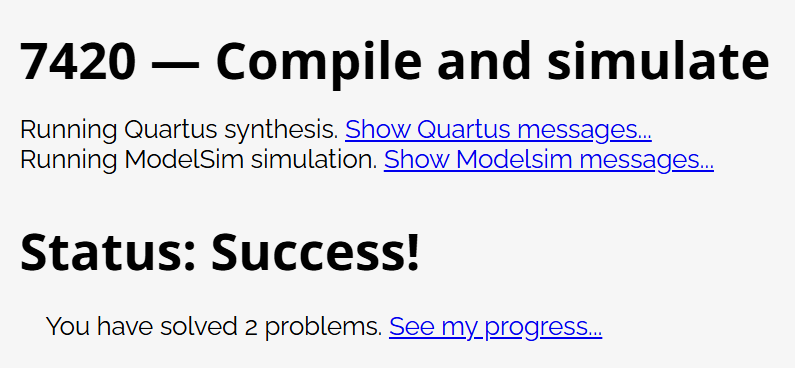
input wire p2a, p2b, p2c, p2d,

output wire p2y );

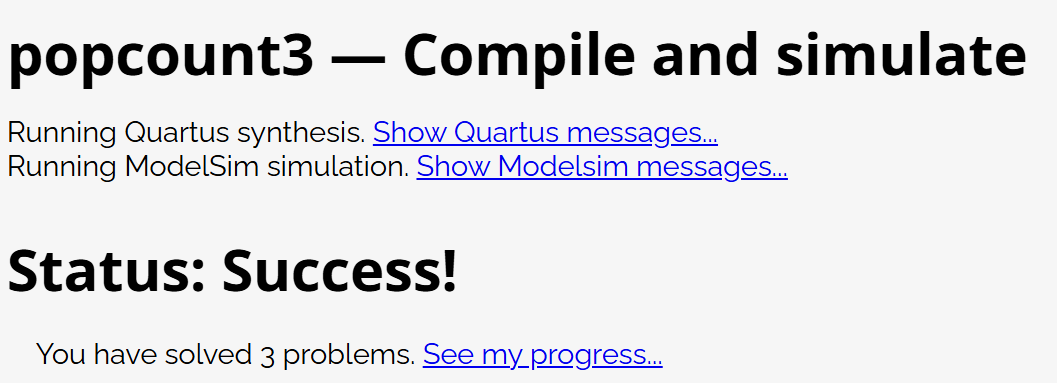
assign p1y = ~(p1a&p1b&p1c&p1d);

assign p2y = ~(p2a&p2b&p2c&p2d);

endmodule



3.popcount3



module top\_module(

    input [2:0] in,

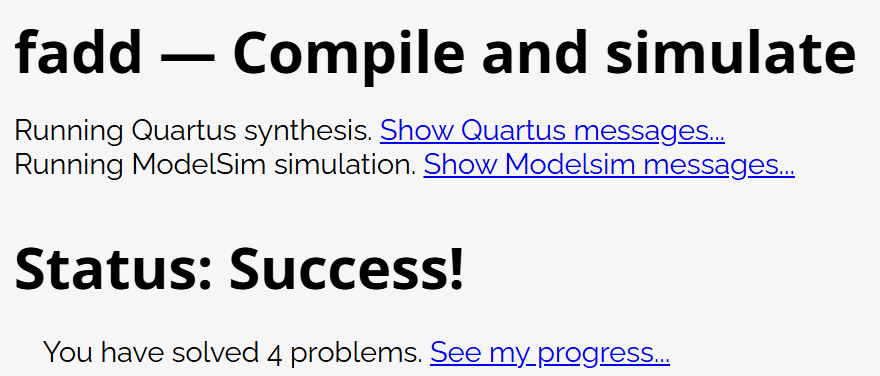
    output [1:0] out

);

    assign out = in[0] + in[1] + in[2];

endmodule

4.full-adder



module top\_module(

    input a, b, cin,

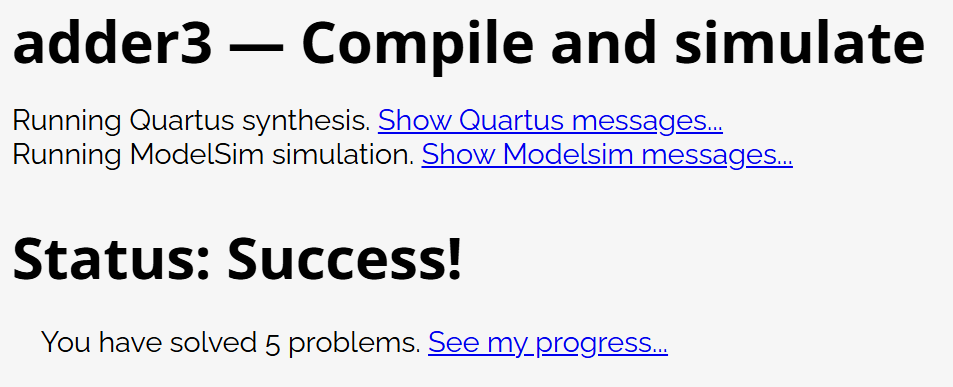
    output cout, sum );

    assign sum = a ^ b ^ cin;

    assign cout = (a & b) | (b & cin) | (a & cin);

endmodule

5.adder3



module top\_module(

    input [2:0] a, b,

    input cin,

    output [2:0] cout,

    output [2:0] sum

);

    assign sum[0] = a[0]^b[0]^cin;

    assign cout[0] = (a[0]&b[0]) | (b[0]&cin) | (a[0]&cin);

    assign sum[1] = a[1]^b[1]^cout[0];

    assign cout[1] = (a[1]&b[1]) | (b[1]&cout[0]) | (a[1]&cout[0]);

    assign sum[2] = a[2]^b[2]^cout[1];

    assign cout[2] = (a[2]&b[2]) | (b[2]&cout[1]) | (a[2]&cout[1]);

endmodule

module top\_module (

    input [7:0] a,

    input [7:0] b,

    output [7:0] s,

    output overflow

);

    assign s =a + b;

    assign overflow = (a[7] == b[7]) && (s[7] != a[7]);

endmodule

6.overflow

