Verilog Lab HDL

Assignment-1

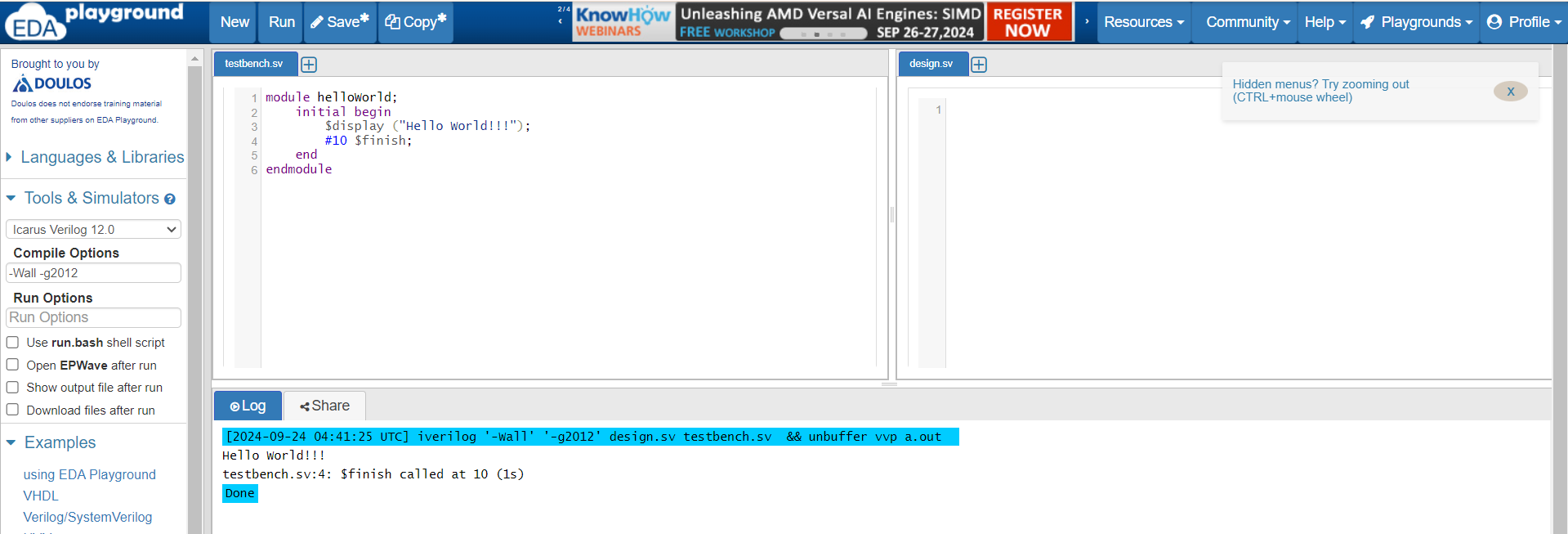
Name- Aye Myat Thiri Khin

ID-20293

Exercise-1

Q: Directly run the module *helloWorld* in the online compiler.

Ans:



module helloWorld;

initial begin

$display ("Hello World!!!");

#10 $finish;

end

endmodule

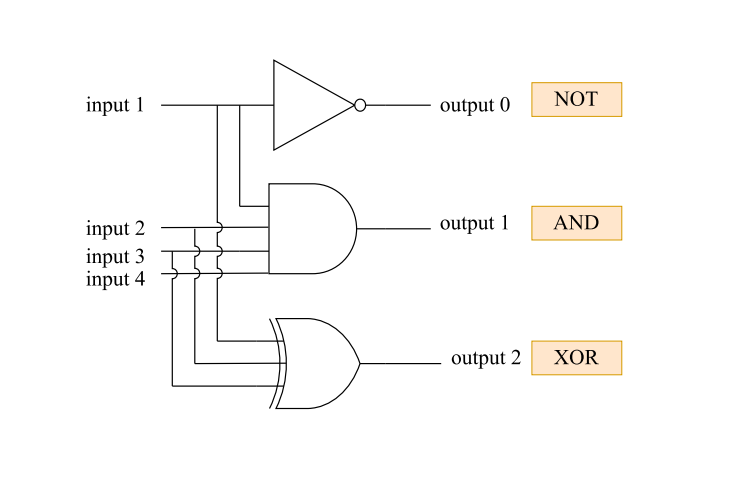
Result Output :

[2024-09-24 04:41:25 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv && unbuffer vvp a.out   
Hello World!!!  
testbench.sv:4: $finish called at 10 (1s)  
Done

--------------------------------------------------------------------------------------------------Exercise-2

Q: Based on the module *gates*, draw the digital circuit schematic.

Ans:



module gates();

wire out0, out1, out2;

reg in1, in2, in3, in4;

not U1(out0, in1);

and U2(out1, in1, in2, in3, in4);

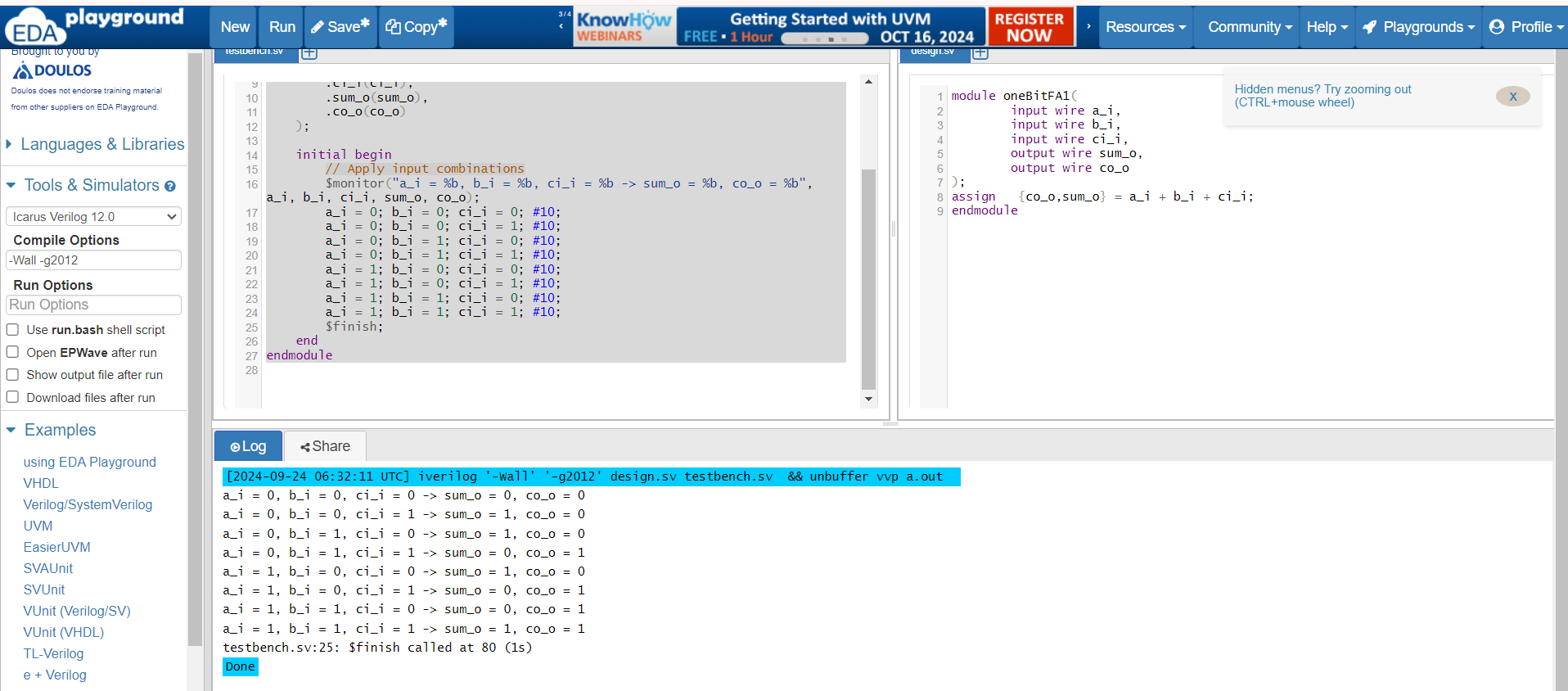
xor U3(out2, in1, in2, in3);

endmodule

-------------------------Exercise-3

Q: Create the testbench first, and then simulate the design modules *oneBitFA1*, *oneBitFA2*.

Ans:



module testbench\_oneBitFA1;

reg a\_i, b\_i, ci\_i;

wire sum\_o, co\_o;

// Instantiate the oneBitFA1 module

oneBitFA1 uut (

.a\_i(a\_i),

.b\_i(b\_i),

.ci\_i(ci\_i),

.sum\_o(sum\_o),

.co\_o(co\_o)

);

initial begin

// Apply input combinations

$monitor("a\_i = %b, b\_i = %b, ci\_i = %b -> sum\_o = %b, co\_o = %b", a\_i, b\_i, ci\_i, sum\_o, co\_o);

a\_i = 0; b\_i = 0; ci\_i = 0; #10;

a\_i = 0; b\_i = 0; ci\_i = 1; #10;

a\_i = 0; b\_i = 1; ci\_i = 0; #10;

a\_i = 0; b\_i = 1; ci\_i = 1; #10;

a\_i = 1; b\_i = 0; ci\_i = 0; #10;

a\_i = 1; b\_i = 0; ci\_i = 1; #10;

a\_i = 1; b\_i = 1; ci\_i = 0; #10;

a\_i = 1; b\_i = 1; ci\_i = 1; #10;

$finish;

end

endmodule

Design:

module oneBitFA1(

input wire a\_i,

input wire b\_i,

input wire ci\_i,

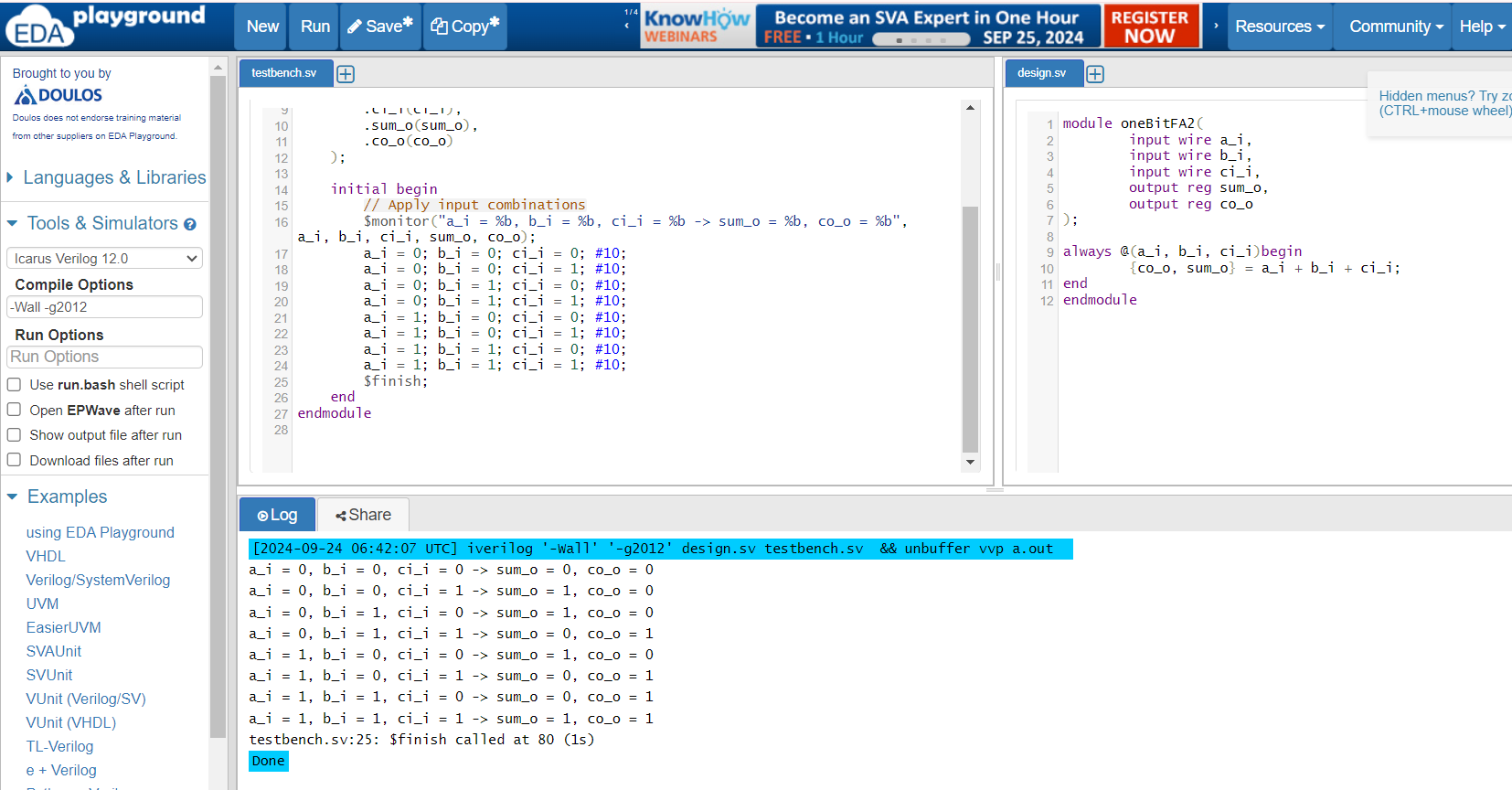
output wire sum\_o,

output wire co\_o

);

assign {co\_o,sum\_o} = a\_i + b\_i + ci\_i;

endmodule



module testbench\_oneBitFA2;

reg a\_i, b\_i, ci\_i;

wire sum\_o, co\_o;

// Instantiate the oneBitFA2 module

oneBitFA2 uut (

.a\_i(a\_i),

.b\_i(b\_i),

.ci\_i(ci\_i),

.sum\_o(sum\_o),

.co\_o(co\_o)

);

initial begin

// Apply input combinations

$monitor("a\_i = %b, b\_i = %b, ci\_i = %b -> sum\_o = %b, co\_o = %b", a\_i, b\_i, ci\_i, sum\_o, co\_o);

a\_i = 0; b\_i = 0; ci\_i = 0; #10;

a\_i = 0; b\_i = 0; ci\_i = 1; #10;

a\_i = 0; b\_i = 1; ci\_i = 0; #10;

a\_i = 0; b\_i = 1; ci\_i = 1; #10;

a\_i = 1; b\_i = 0; ci\_i = 0; #10;

a\_i = 1; b\_i = 0; ci\_i = 1; #10;

a\_i = 1; b\_i = 1; ci\_i = 0; #10;

a\_i = 1; b\_i = 1; ci\_i = 1; #10;

$finish;

end

endmodule

Design:

module oneBitFA2(

input wire a\_i,

input wire b\_i,

input wire ci\_i,

output reg sum\_o,

output reg co\_o

);

always @(a\_i, b\_i, ci\_i)begin  
{co\_o, sum\_o} = a\_i + b\_i + ci\_i;

end

endmodule