**CS303 – Logic & Digital System Design**

**CS303 – Lab#3 Report**

**Name-Surname/SID : Barışcan Köse 25413**

**1.Project Description**

The aim of this project is to use Verilog to build 20 bit adder/subtractors. First design will be implemented with ripple carry adder concept in mind. Second design will be implemented with 4 of 5-bit Carry Look-Ahead Adders in hierarchical design.Verilog code, simulation and implementation results are down below.

**2.First Design**

**2.1. Verilog Code**

module RCA20(A,B,Sum,Carry,OVF,SUB

);

input SUB;

input [19:0]A,B;

wire [19:0]D;

wire [19:0]E;

output[19:0] Sum;

output Carry ;

output OVF;

assign D[0]=B[0]^SUB;

assign D[1]=B[1]^SUB;

assign D[2]=B[2]^SUB;

assign D[3]=B[3]^SUB;

assign D[4]=B[4]^SUB;

assign D[5]=B[5]^SUB;

assign D[6]=B[6]^SUB;

assign D[7]=B[7]^SUB;

assign D[8]=B[8]^SUB;

assign D[9]=B[9]^SUB;

assign D[10]=B[10]^SUB;

assign D[11]=B[11]^SUB;

assign D[12]=B[12]^SUB;

assign D[13]=B[13]^SUB;

assign D[14]=B[14]^SUB;

assign D[15]=B[15]^SUB;

assign D[16]=B[16]^SUB;

assign D[17]=B[17]^SUB;

assign D[18]=B[18]^SUB;

assign D[19]=B[19]^SUB;

FullAdderVerilog FA0(.A(A[0]),.B(D[0]),.Cin(SUB),.Sum(Sum[0]),.Carry(E[0]));

FullAdderVerilog FA1(.A(A[1]),.B(D[1]),.Cin(E[0]),.Sum(Sum[1]),.Carry(E[1]));

FullAdderVerilog FA2(.A(A[2]),.B(D[2]),.Cin(E[1]),.Sum(Sum[2]),.Carry(E[2]));

FullAdderVerilog FA3(.A(A[3]),.B(D[3]),.Cin(E[2]),.Sum(Sum[3]),.Carry(E[3]));

FullAdderVerilog FA4(.A(A[4]),.B(D[4]),.Cin(E[3]),.Sum(Sum[4]),.Carry(E[4]));

FullAdderVerilog FA5(.A(A[5]),.B(D[5]),.Cin(E[4]),.Sum(Sum[5]),.Carry(E[5]));

FullAdderVerilog FA6(.A(A[6]),.B(D[6]),.Cin(E[5]),.Sum(Sum[6]),.Carry(E[6]));

FullAdderVerilog FA7(.A(A[7]),.B(D[7]),.Cin(E[6]),.Sum(Sum[7]),.Carry(E[7]));

FullAdderVerilog FA8(.A(A[8]),.B(D[8]),.Cin(E[7]),.Sum(Sum[8]),.Carry(E[8]));

FullAdderVerilog FA9(.A(A[9]),.B(D[9]),.Cin(E[8]),.Sum(Sum[9]),.Carry(E[9]));

FullAdderVerilog FA10(.A(A[10]),.B(D[10]),.Cin(E[9]),.Sum(Sum[10]),.Carry(E[10]));

FullAdderVerilog FA11(.A(A[11]),.B(D[11]),.Cin(E[10]),.Sum(Sum[11]),.Carry(E[11]));

FullAdderVerilog FA12(.A(A[12]),.B(D[12]),.Cin(E[11]),.Sum(Sum[12]),.Carry(E[12]));

FullAdderVerilog FA13(.A(A[13]),.B(D[13]),.Cin(E[12]),.Sum(Sum[13]),.Carry(E[13]));

FullAdderVerilog FA14(.A(A[14]),.B(D[14]),.Cin(E[13]),.Sum(Sum[14]),.Carry(E[14]));

FullAdderVerilog FA15(.A(A[15]),.B(D[15]),.Cin(E[14]),.Sum(Sum[15]),.Carry(E[15]));

FullAdderVerilog FA16(.A(A[16]),.B(D[16]),.Cin(E[15]),.Sum(Sum[16]),.Carry(E[16]));

FullAdderVerilog FA17(.A(A[17]),.B(D[17]),.Cin(E[16]),.Sum(Sum[17]),.Carry(E[17]));

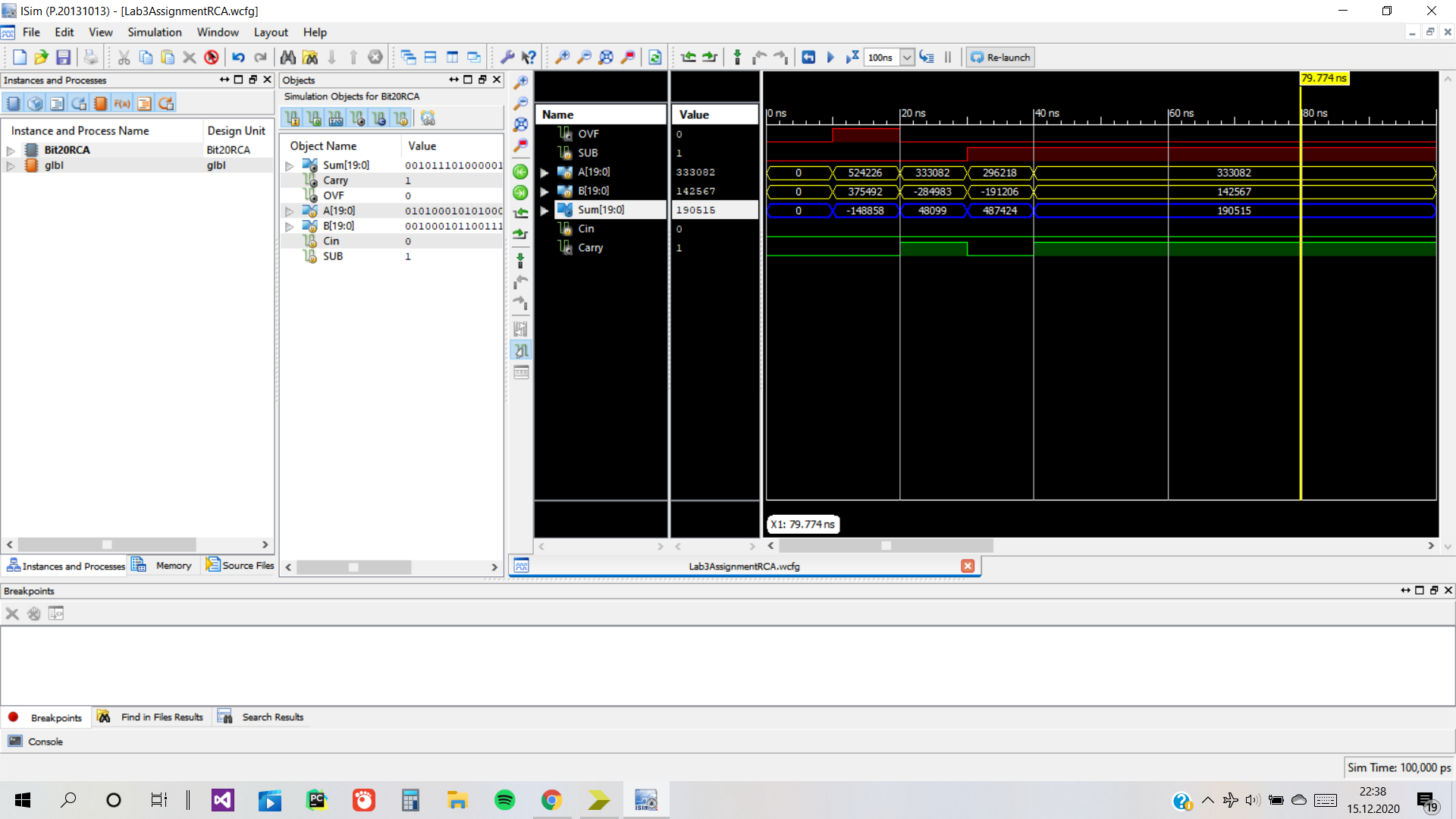
FullAdderVerilog FA18(.A(A[18]),.B(D[18]),.Cin(E[17]),.Sum(Sum[18]),.Carry(E[18]));

FullAdderVerilog FA19(.A(A[19]),.B(D[19]),.Cin(E[18]),.Sum(Sum[19]),.Carry(E[19]));

assign Carry=E[19];

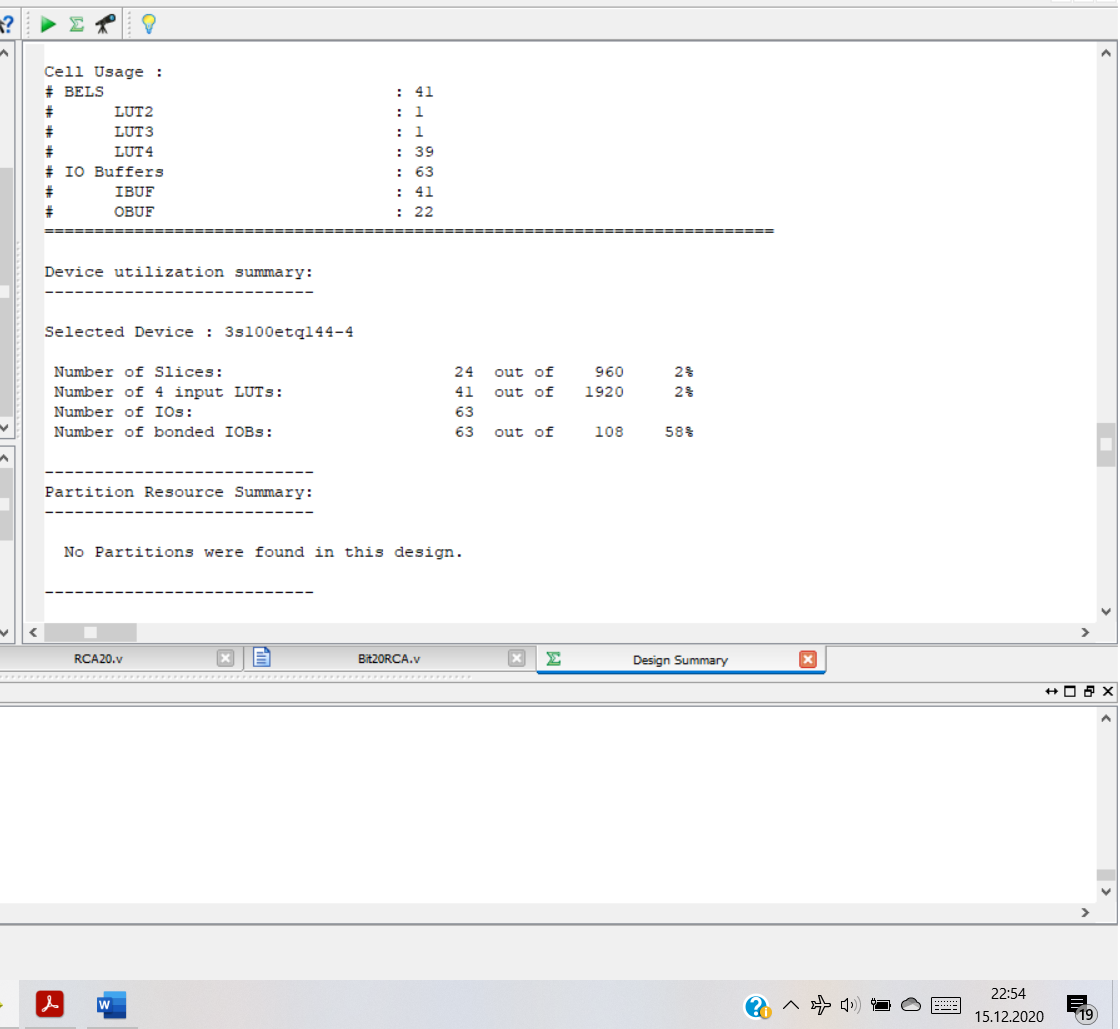
assign OVF=E[18]^E[19];

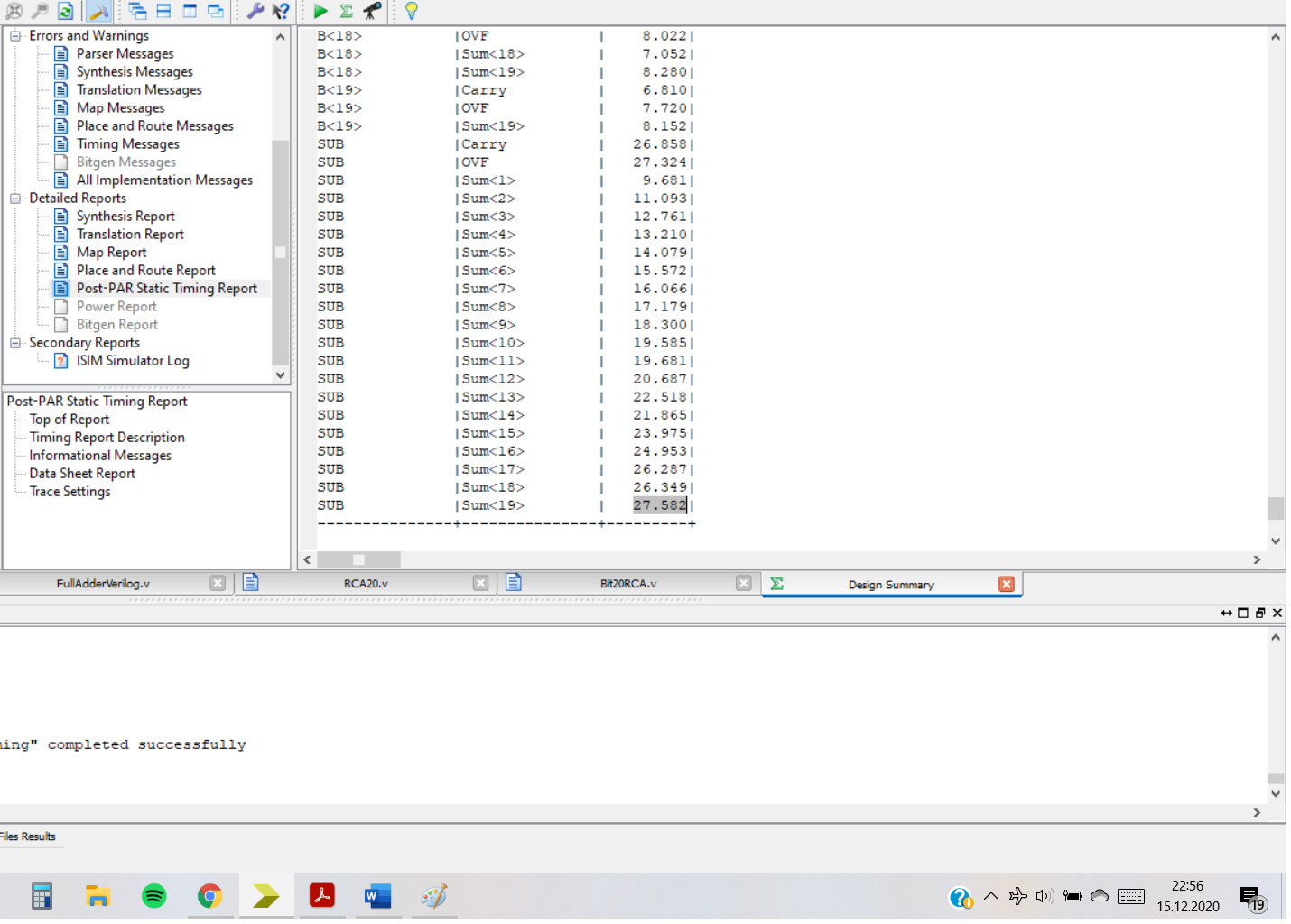
endmodule

**2.2. Simulation**

**2.3.Implementation Results**

The implementation uses 41 of 4-input LTUs.The critical path delay is 27.582 ns which is the path from SUB to Sum[19].

****

****

**3.Second Design**

**3.1.Verilog Code**

module Bit20CLAAdder(A,B,Sum,Carry,SUB,OVF

);

input [19:0] A,B;

input SUB;

output [19:0]Sum;

output Carry,OVF;

wire [3:0] W;

wire [19:0]D;

assign D[0]=B[0]^SUB;

assign D[1]=B[1]^SUB;

assign D[2]=B[2]^SUB;

assign D[3]=B[3]^SUB;

assign D[4]=B[4]^SUB;

assign D[5]=B[5]^SUB;

assign D[6]=B[6]^SUB;

assign D[7]=B[7]^SUB;

assign D[8]=B[8]^SUB;

assign D[9]=B[9]^SUB;

assign D[10]=B[10]^SUB;

assign D[11]=B[11]^SUB;

assign D[12]=B[12]^SUB;

assign D[13]=B[13]^SUB;

assign D[14]=B[14]^SUB;

assign D[15]=B[15]^SUB;

assign D[16]=B[16]^SUB;

assign D[17]=B[17]^SUB;

assign D[18]=B[18]^SUB;

assign D[19]=B[19]^SUB;

Bit5CLAAdder CLA1 (.A(A[4:0]),.B(D[4:0]),.Cin(SUB),.Sum(Sum[4:0]),.Carry(W[0]));

Bit5CLAAdder CLA2 (.A(A[9:5]),.B(D[9:5]),.Cin(W[0]),.Sum(Sum[9:5]),.Carry(W[1]));

Bit5CLAAdder CLA3 (.A(A[14:10]),.B(D[14:10]),.Cin(W[1]),.Sum(Sum[14:10]),.Carry(W[2]));

Bit5CLAAdderL CLA4 (.A(A[19:15]),.B(D[19:15]),.Cin(W[2]),.Sum(Sum[19:15]),.Carry(W[3]),.Overflow(OVF));

assign Carry=W[3];

endmodule

module Bit5CLAAdder(A,B,Cin,Sum,Carry

);

input [4:0]A,B;

input Cin;

output [4:0]Sum;

output Carry;

wire [4:0]P,G,C;

assign P[0]=A[0]^B[0];

assign G[0]=A[0]&B[0];

assign P[1]=A[1]^B[1];

assign G[1]=A[1]&B[1];

assign P[2]=A[2]^B[2];

assign G[2]=A[2]&B[2];

assign P[3]=A[3]^B[3];

assign G[3]=A[3]&B[3];

assign P[4]=A[4]^B[4];

assign G[4]=A[4]&B[4];

assign Sum[0]=P[0]^Cin;

assign C[0]=G[0]|(P[0]&Cin);

assign Sum[1]=P[1]^C[0];

assign C[1]=G[1]|(P[1]&G[0])|(P[1]&P[0]&Cin);

assign Sum[2]=P[2]^C[1];

assign C[2]=G[2]|(P[2]&G[1])|(P[2]&P[1]&G[0])|(P[2]&P[1]&P[0]&Cin);

assign Sum[3]=P[3]^C[2];

assign C[3]=G[3]|(P[3]&G[2])|(P[3]&P[2]&G[1])|(P[3]&P[2]&P[1]&G[0])|(P[3]&P[2]&P[1]&P[0]&Cin);

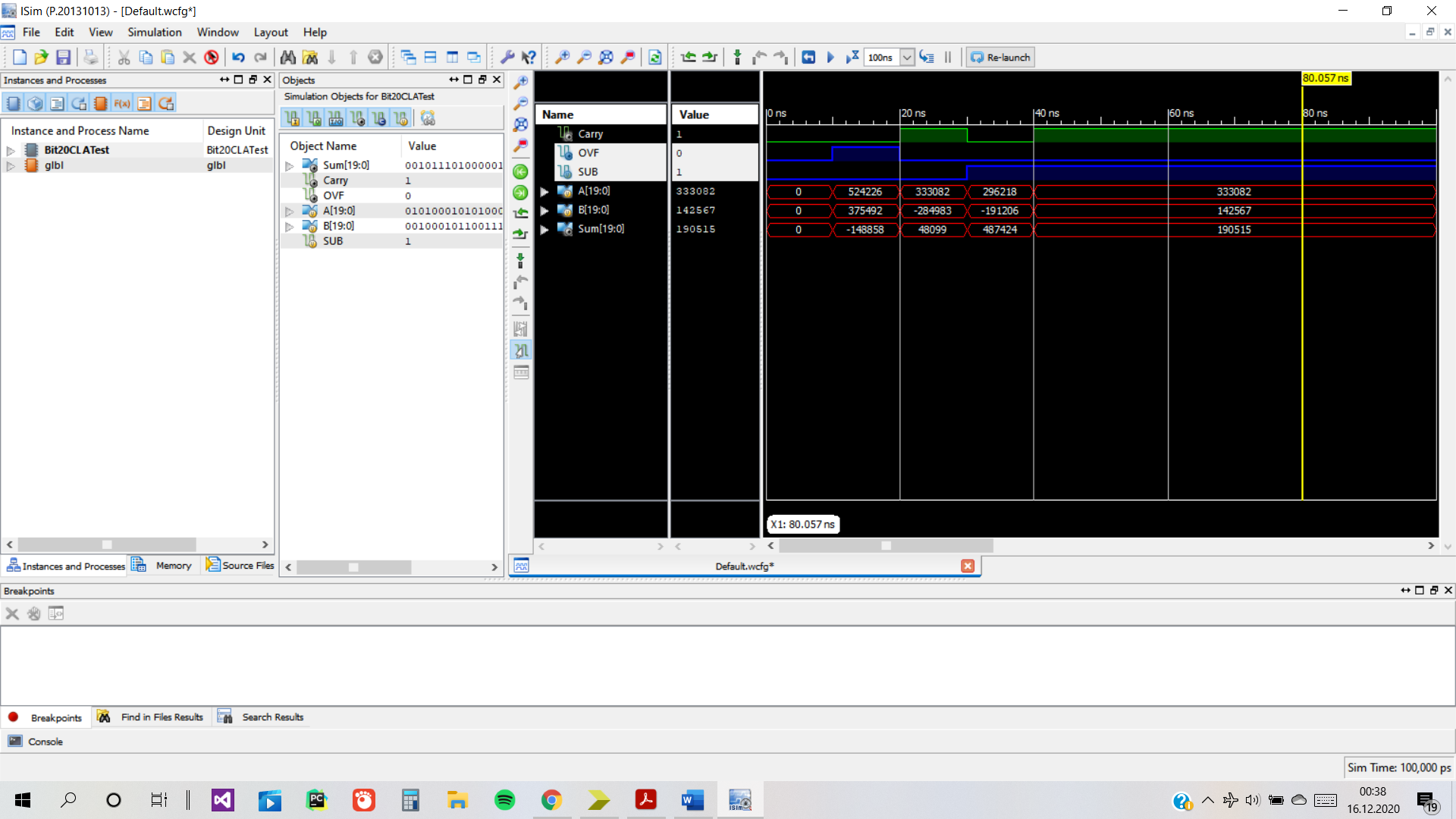
assign Sum[4]=P[4]^C[3];

assign C[4]=G[4]|(P[4]&G[3])|(P[4]&P[3]&G[2])|(P[4]&P[3]&P[2]&G[1])|(P[4]&P[3]&P[2]&P[1]&G[0])|(P[4]&P[3]&P[2]&P[1]&P[0]&Cin);

assign Carry=C[4];

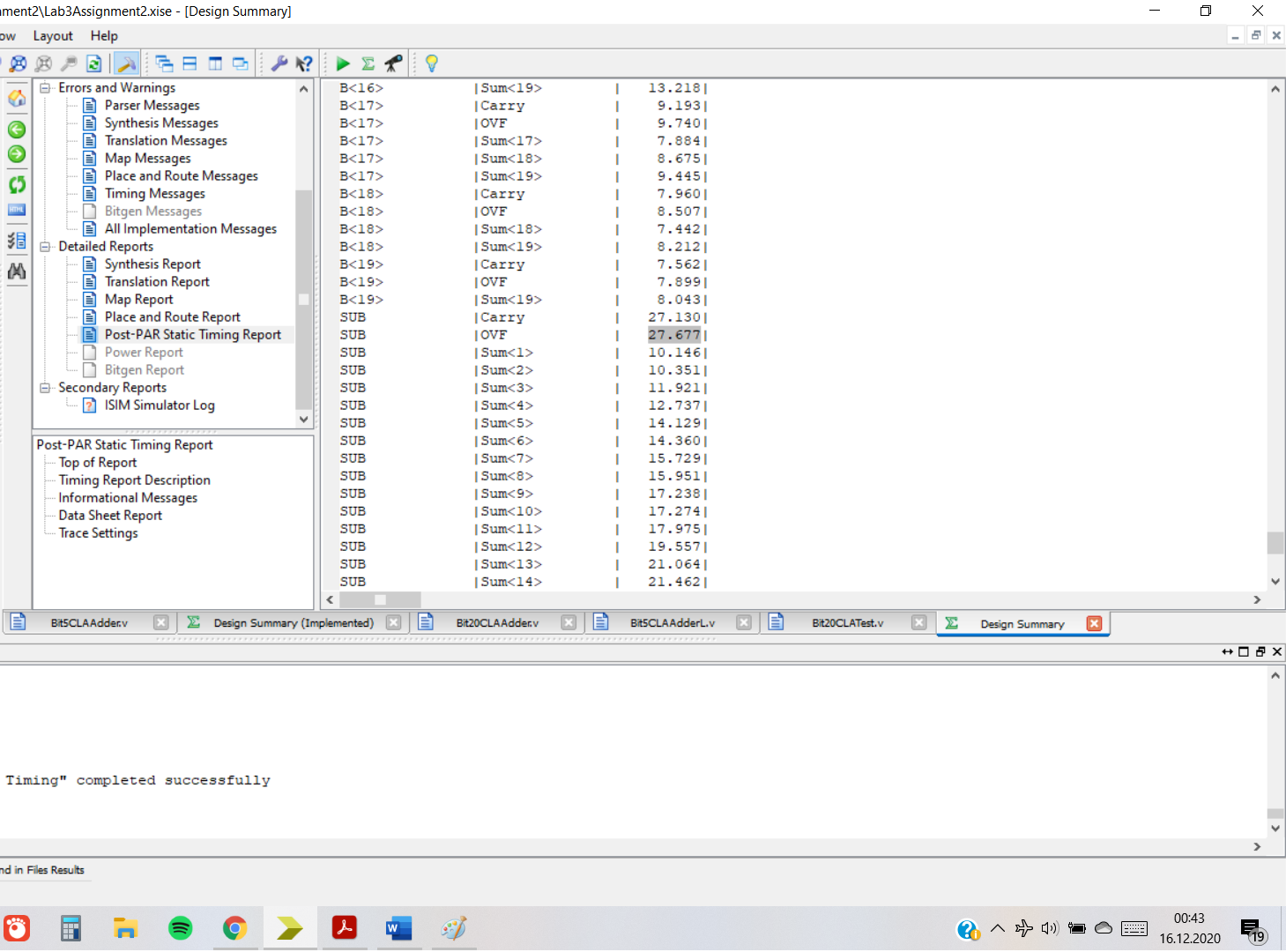
endmodule

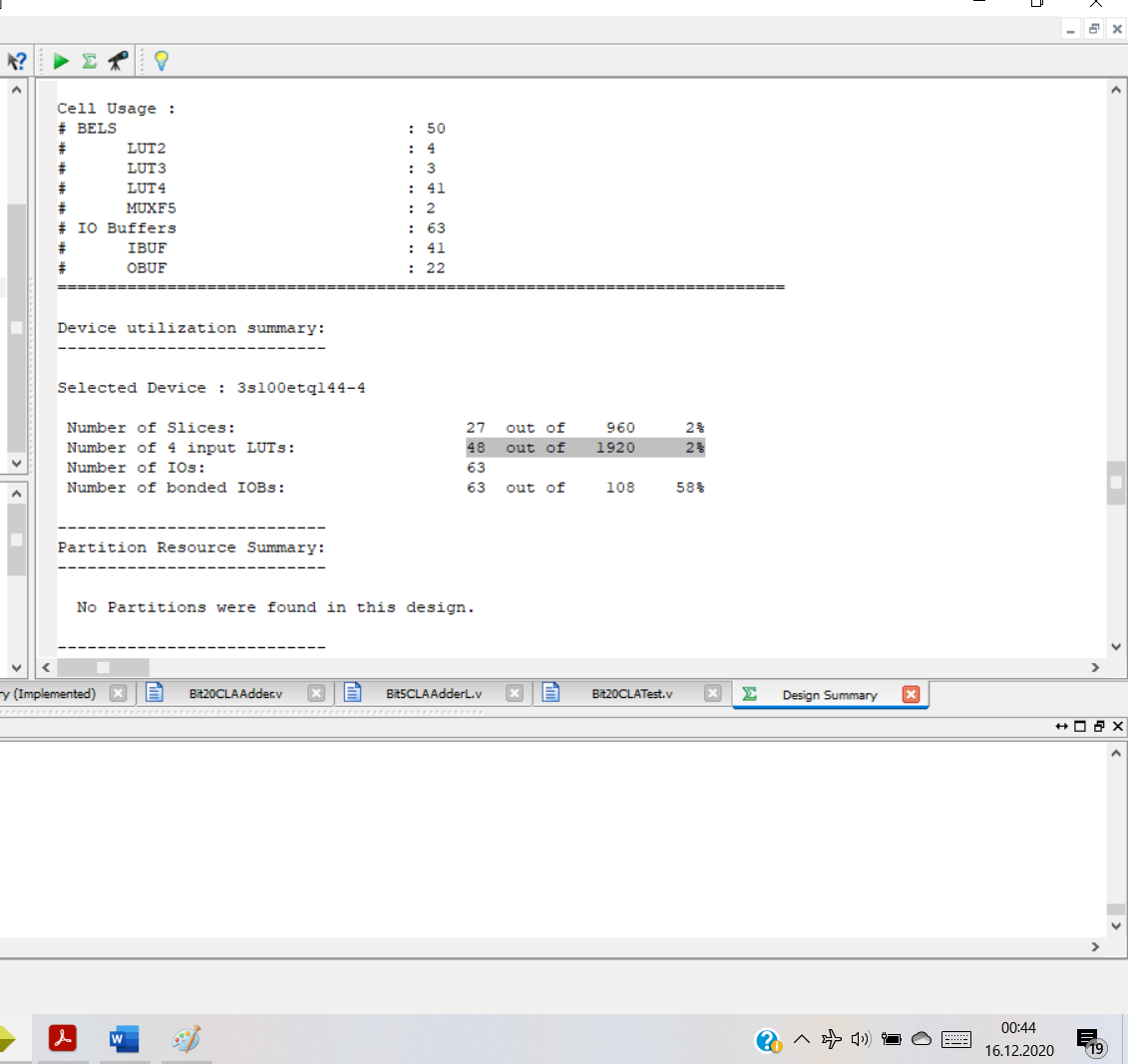
**3.2.Simulation**

****

**3.3.Implementation Results**

It uses 48 LUTs and the maximum critical delay is 27.677ns.

****

****

**4.Discussion**

1.Synthesis results give us the area and implementation results give us the speed of the design.

2.Ripple Carry Adder design using Full Adders is better in terms of area with 41 LUTS vs. 48 LUTS.

3. Ripple Carry Adder design using Full Adders is slightly better in terms of speed with 27.582 ns delay vs. 27.677 ns delay.

4.If there was such a metric RCA would score 27.582\*41=1130,862.

CLA design would score 27.677\*48 = 1328,496.

Which would mean that RCA design using full adders is better in terms of the new metric.