Lab2 Report

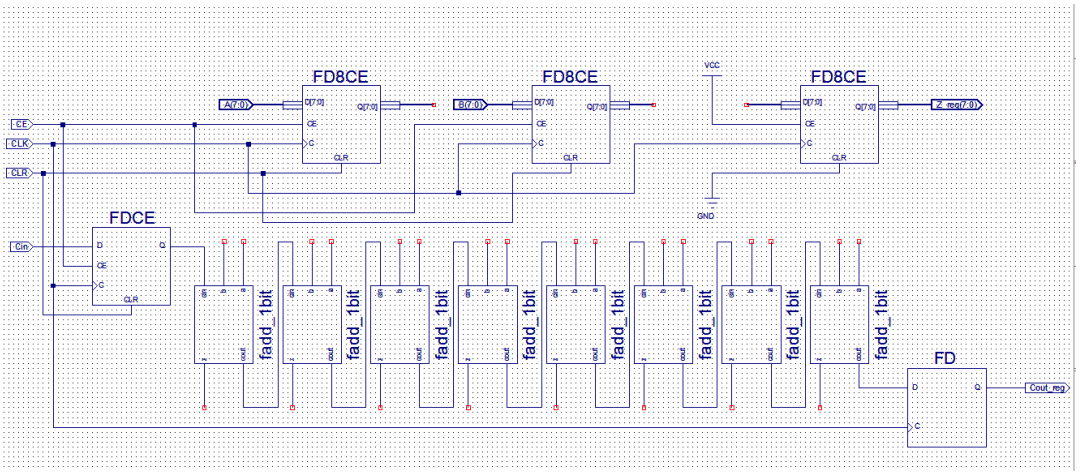
Name: Haoyang Zhang

Email: hzhang11@usc.edu

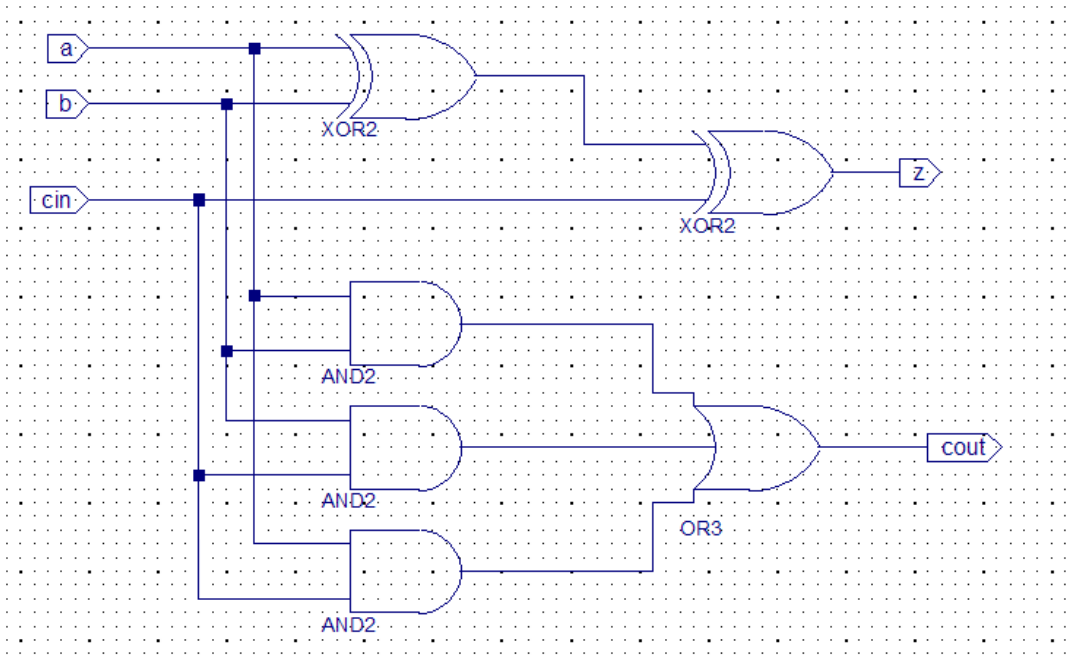
Remote repository address: https://github.com/hzhang2422/EE-533

Part 1: Designing and Simulating Synchronous 8-bit Adder

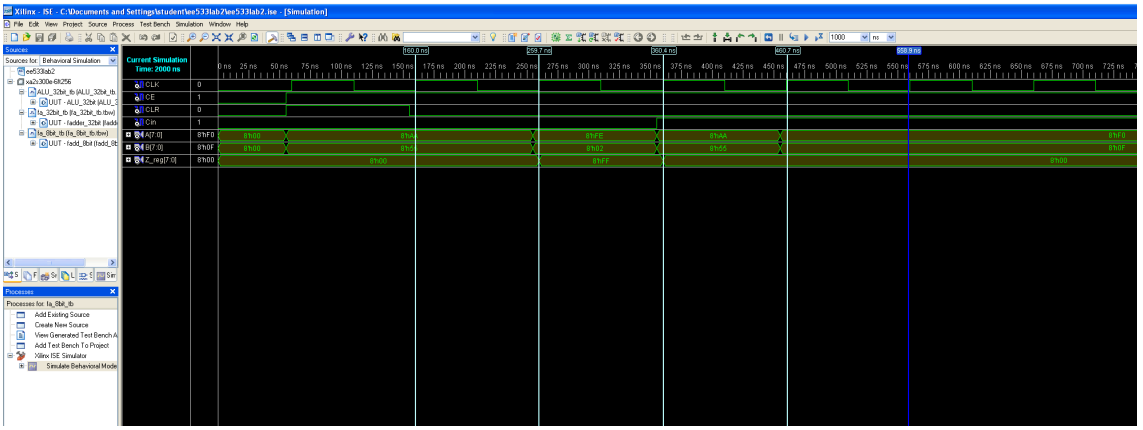
1. Screen capture of schematics
    - 8-bit Synchronous Full Adder



        - 1-bit Full Adder



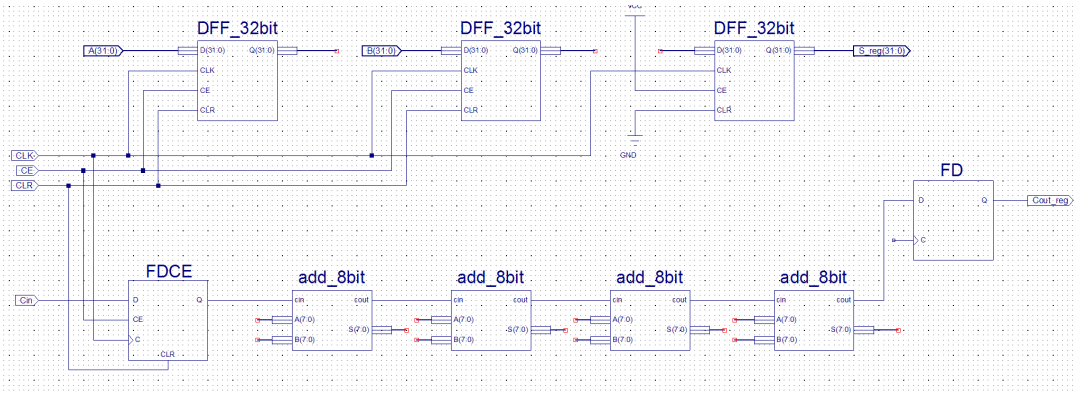
2. Screen capture of the waveforms generated by the behavioral simulation tools



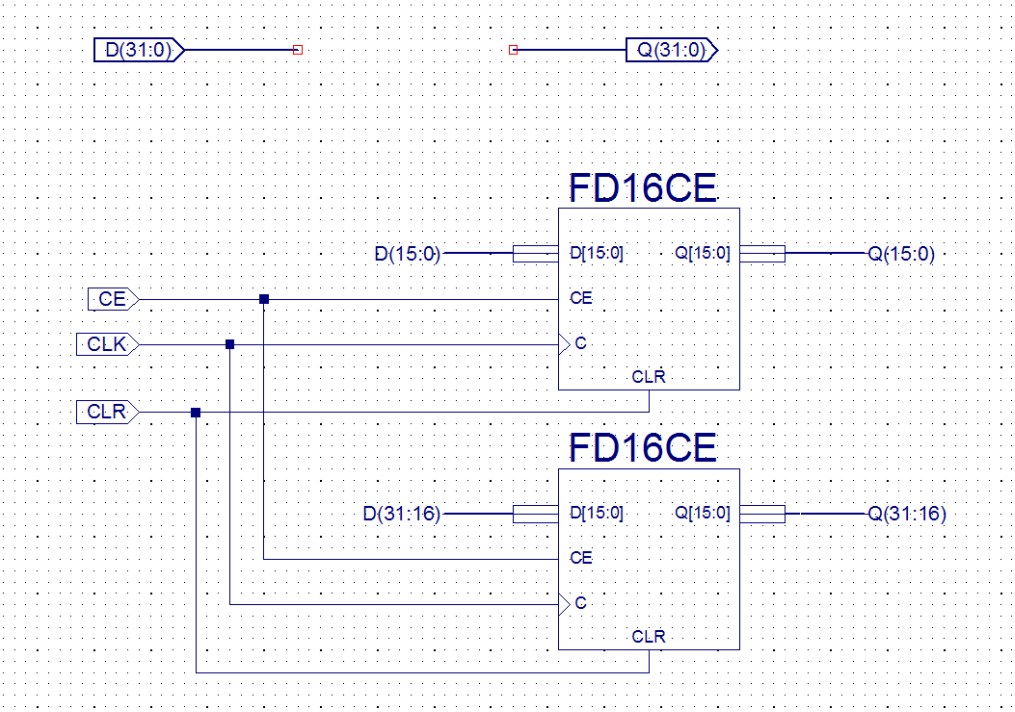
Part 2: Extending Adder into 32-bit ALU

A : Extend the 8-bit Adder into 32-bit Adder by instantiating and connecting 4 adders

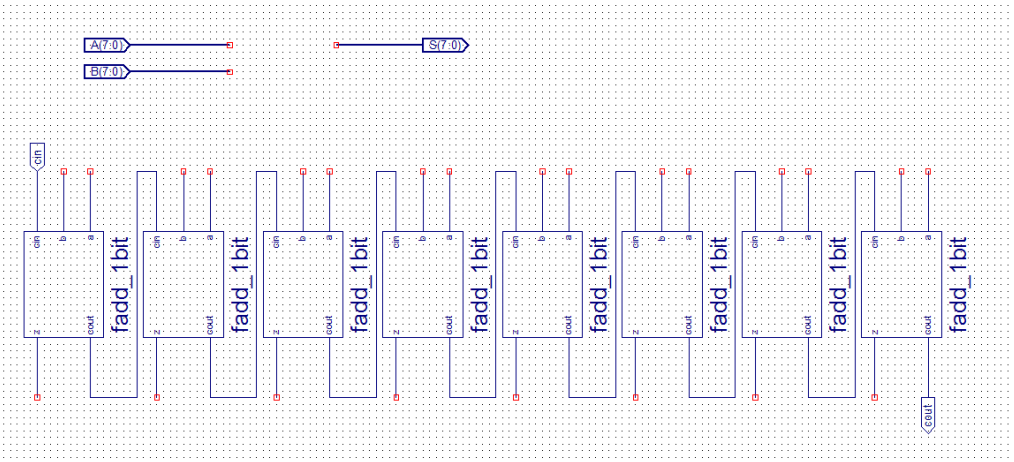
1. Screen capture of schematics
    - 32-bit Synchronous Full Adder



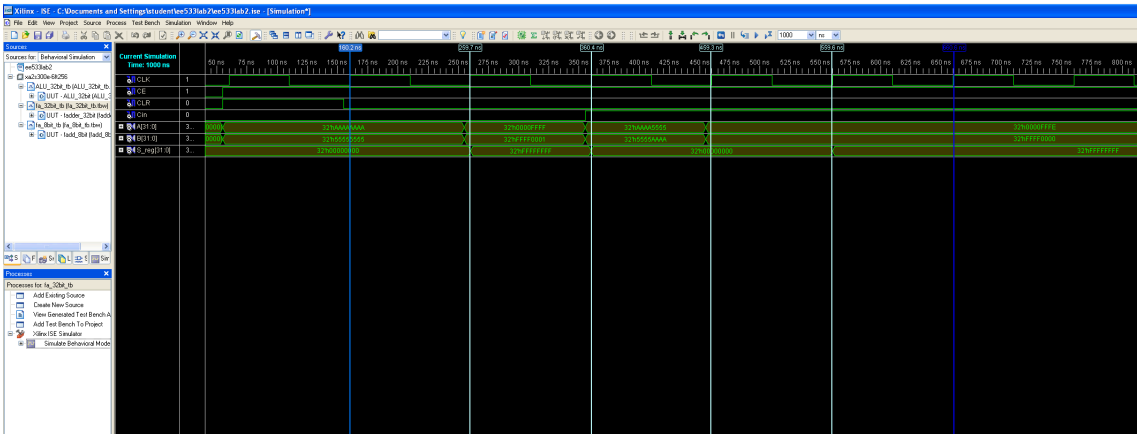
    - 32-bit DFF



    - 8-bit Full Adder



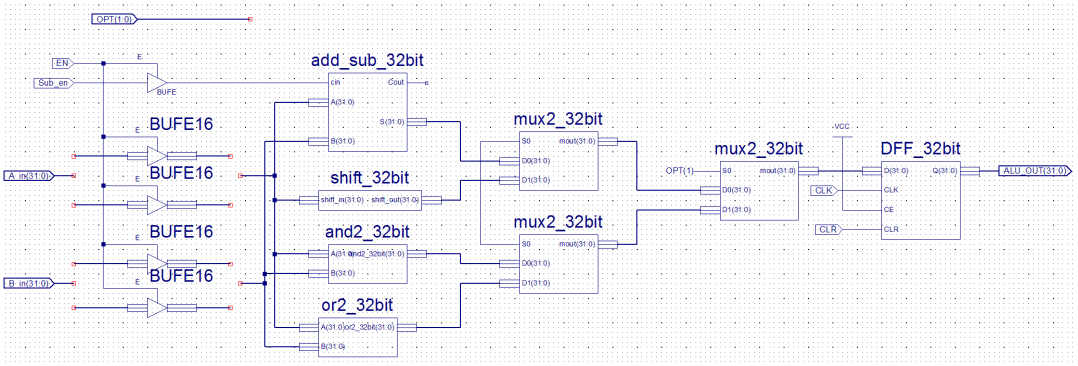
2. Screen capture of the waveforms generated by the behavioral simulation tools



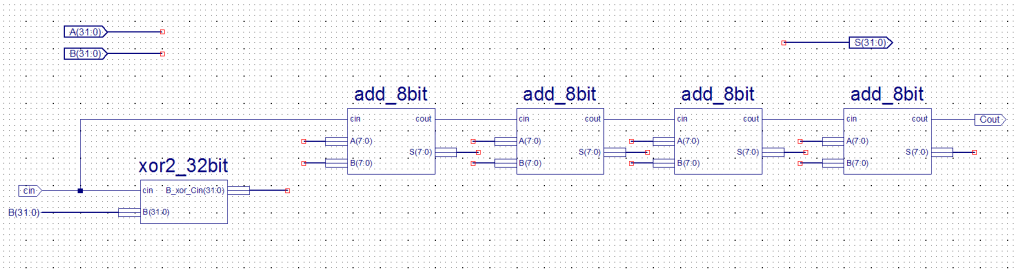
B : Extend the 32-bit Adder to have other functions including subtractor, shifter, and two other functions(AND & OR) of your choice. Go through the mapping process of the tools to get the gate counts such as number of D-FF and LUTs.

1. Brief description of all of the functions
    - Addition or Substract: When option is '00', the first function is active. A(31:0) plus B(31:0) when Sub_enable is 0, and 1 is for A(31:0) minus B(31:0).
    - Left shift for 1 bit on A.
    - Bitwise AND by A(31:0) and B(31:0).
    - Bitwise OR by A(31:0) and B(31:0).

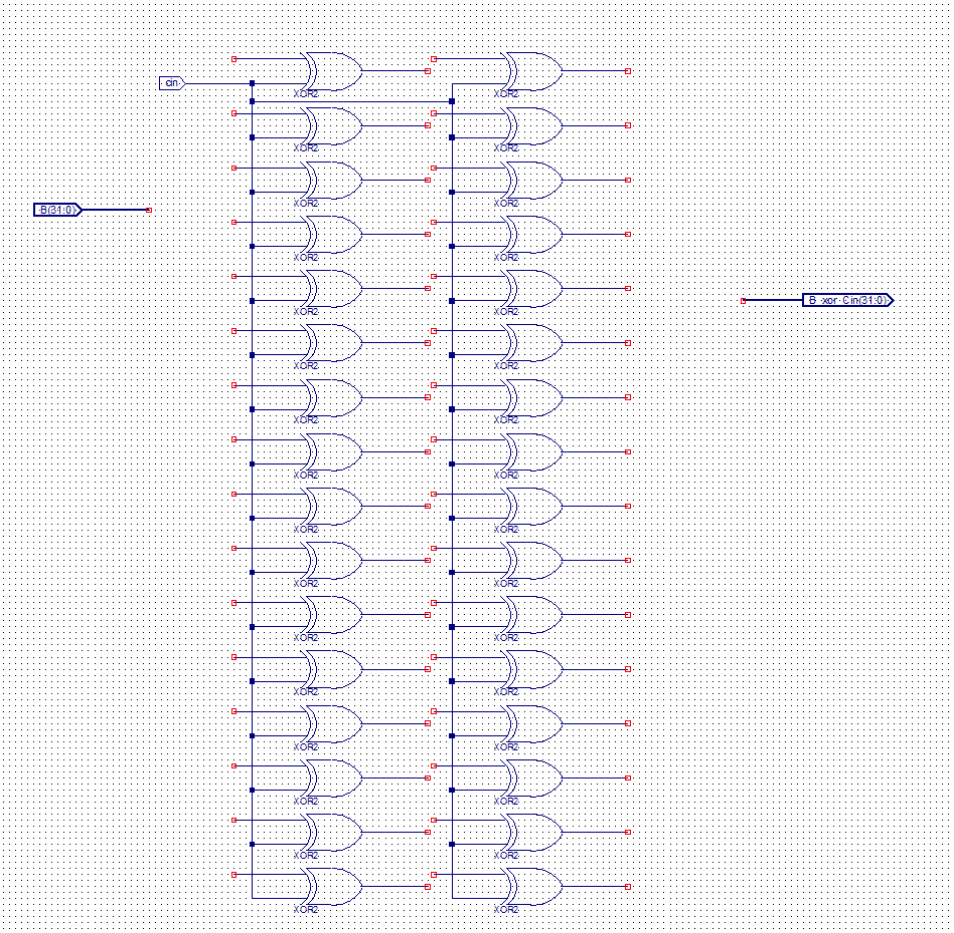
2. Screen capture of schematics
    - 32-bit ALU



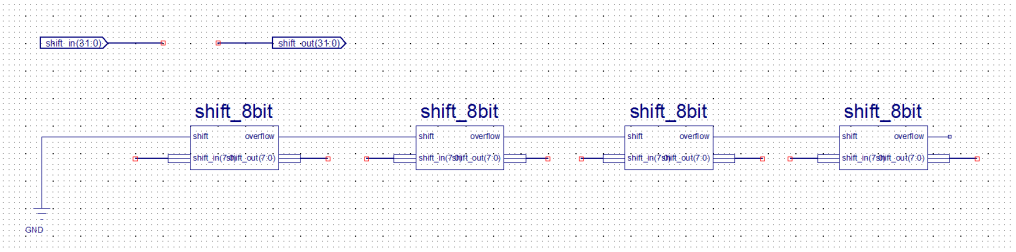
- 32-bit Add_Sub



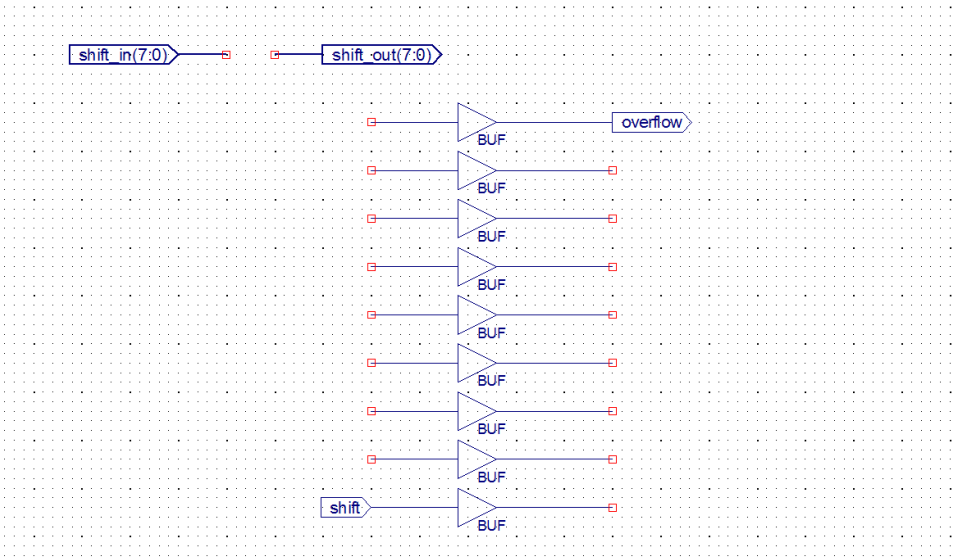
- 32-bit Bitwise XOR



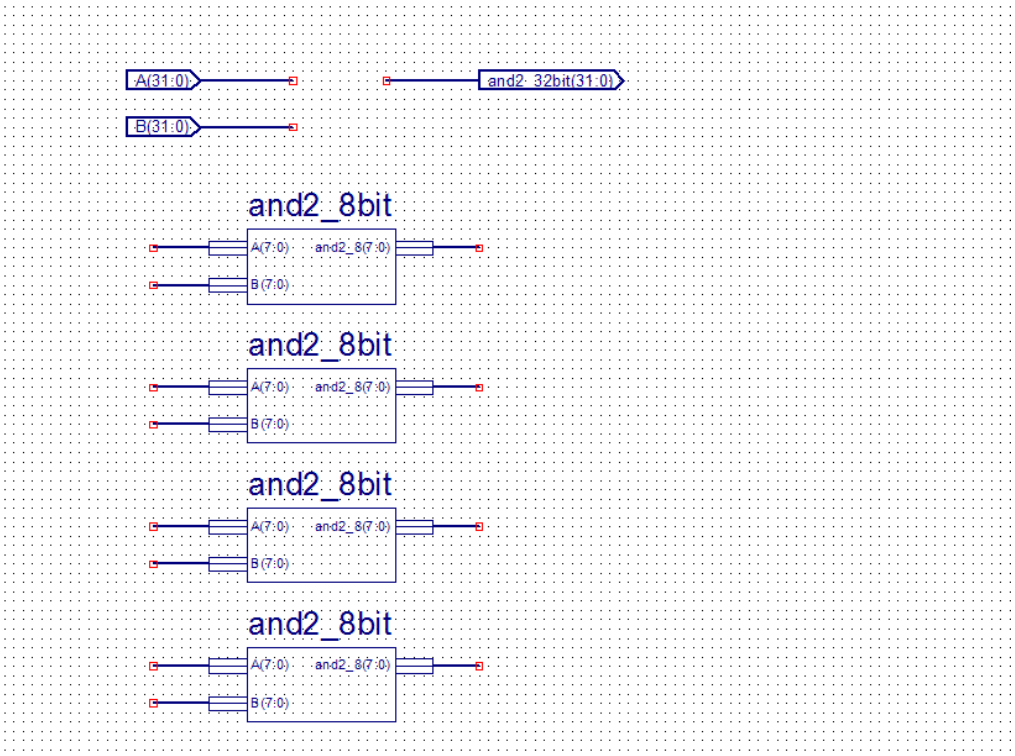
- 32-bit Shift



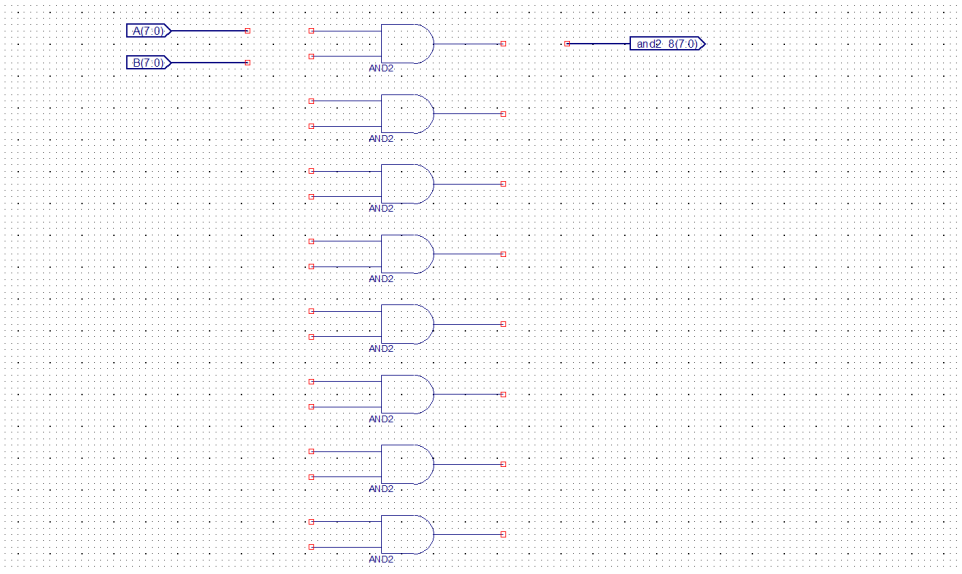
- 8-bit Shift



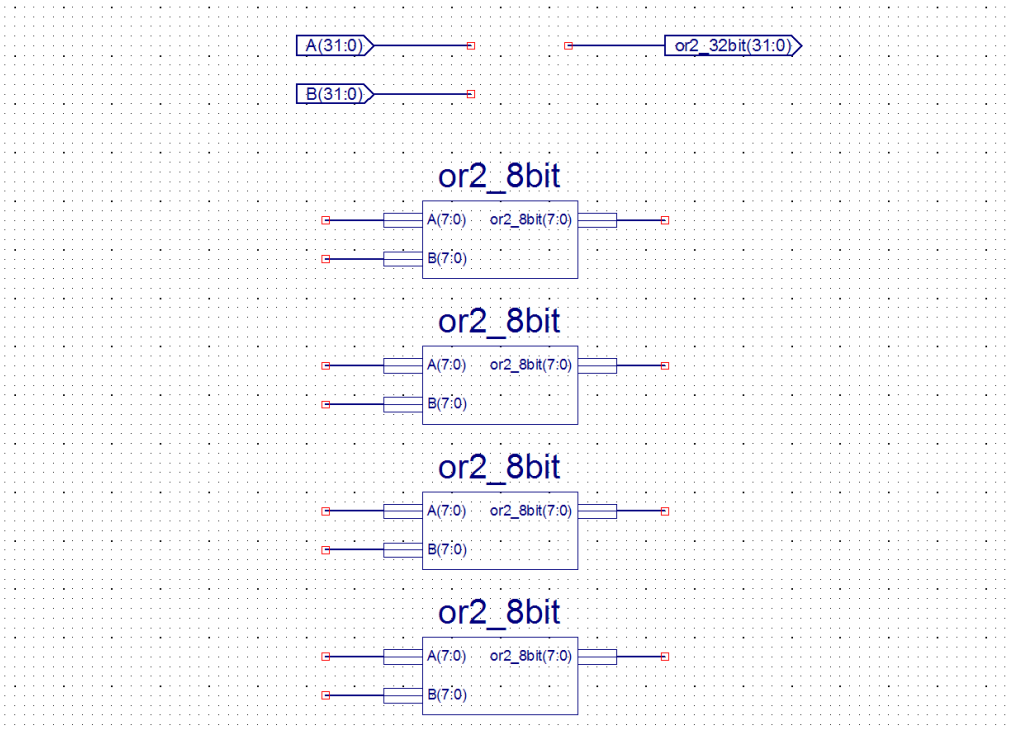
- 32-bit Bitwise AND



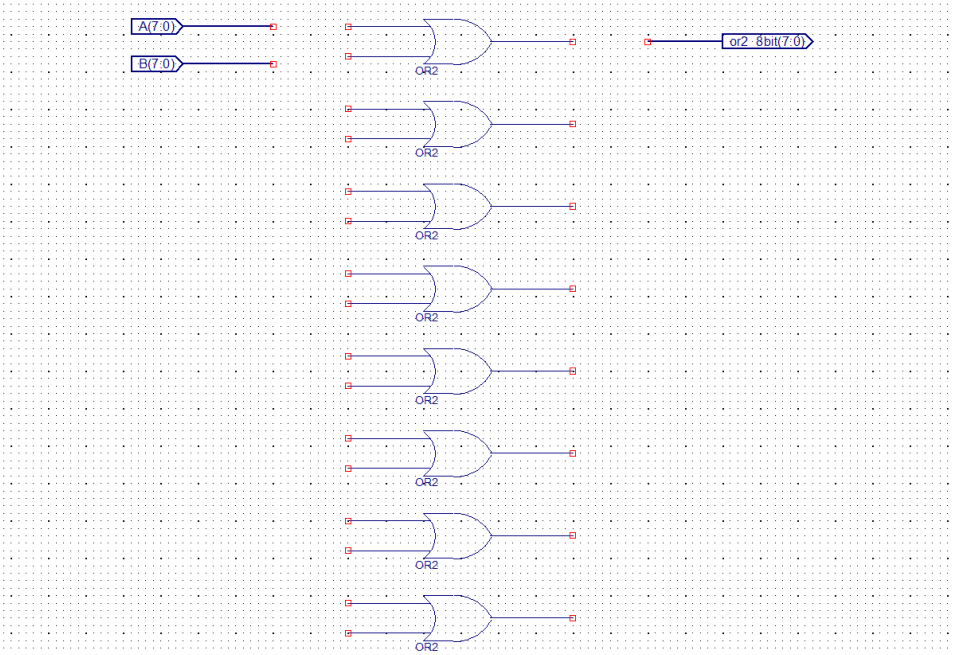
  - 8-bit Bitwise AND



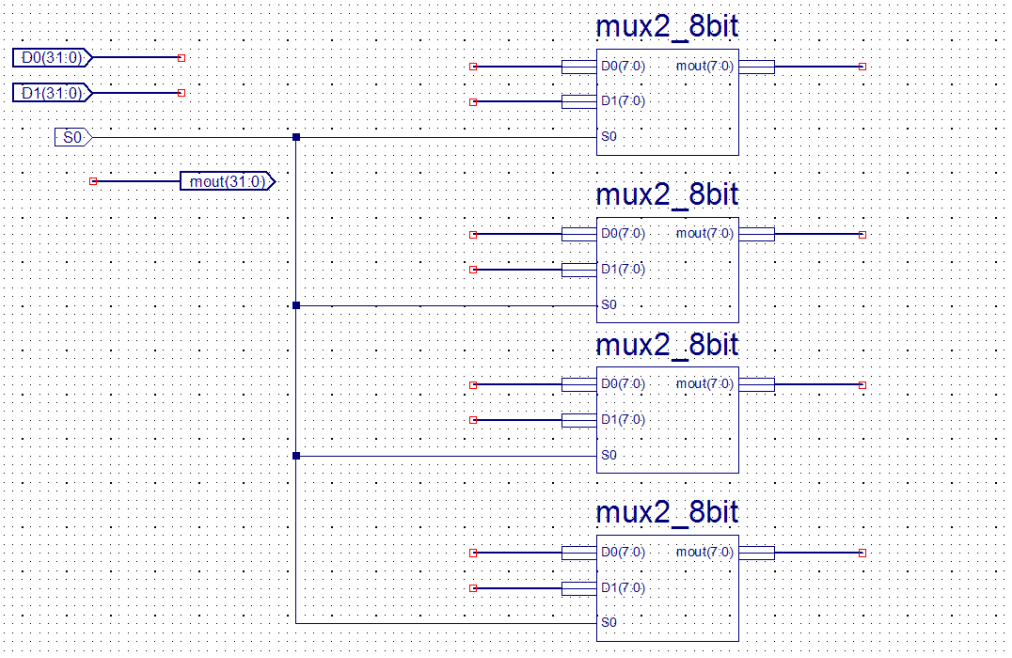
- 32-bit Bitwise OR



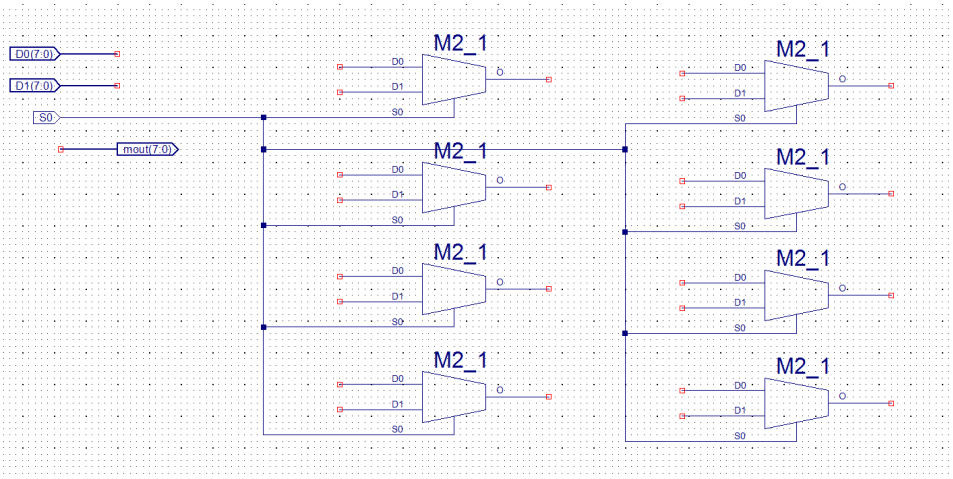
- 8-bit Bitwise OR



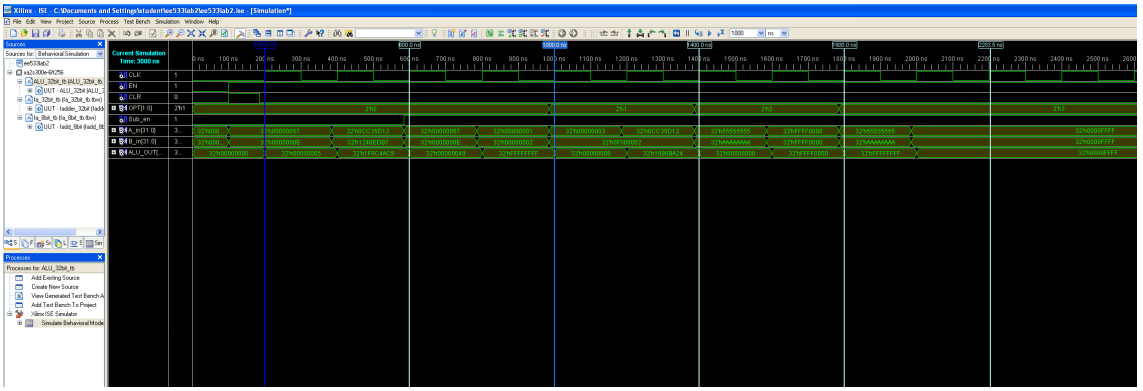
- 32-bit Bitwise 2 to 1 MUX



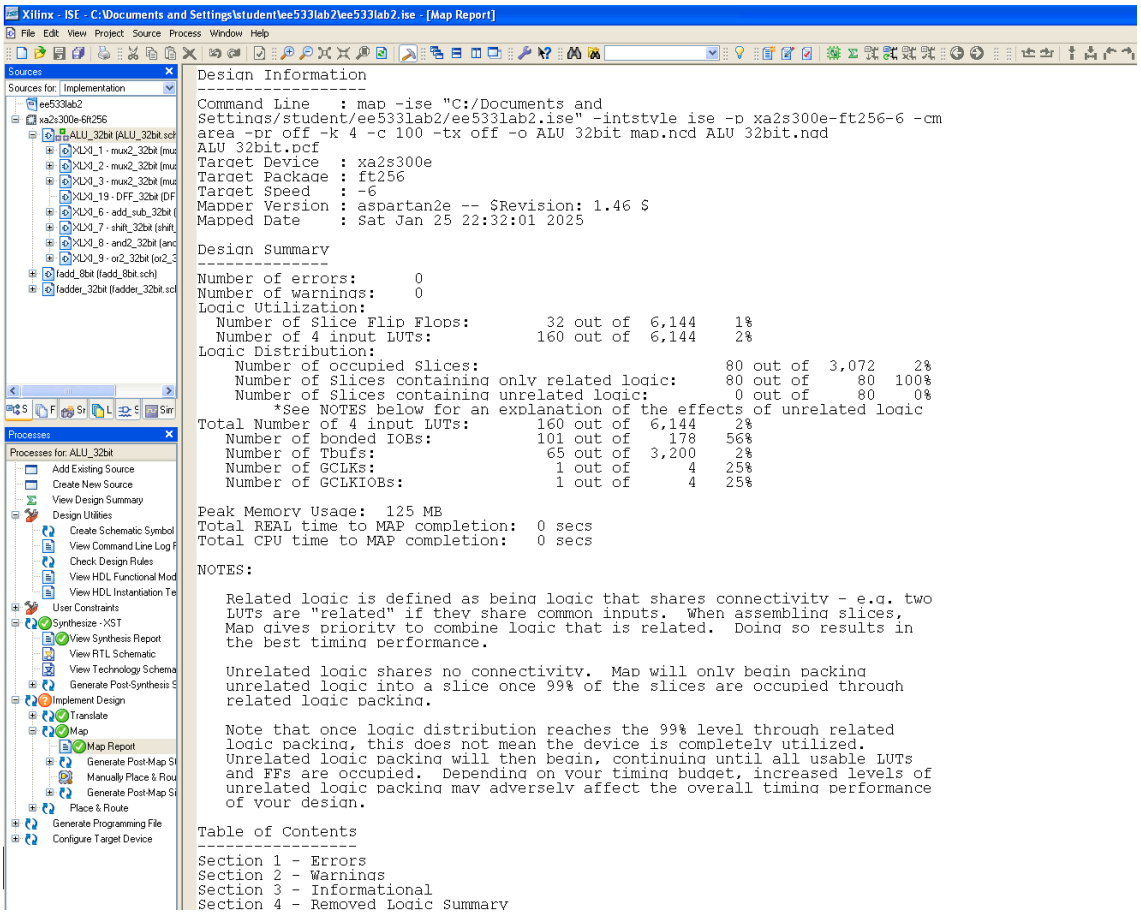
- 8-bit Bitwise 2 to 1 MUX



3. Screen capture of the waveforms generated by the behavioral simulation tools



4. Log file of the mapper

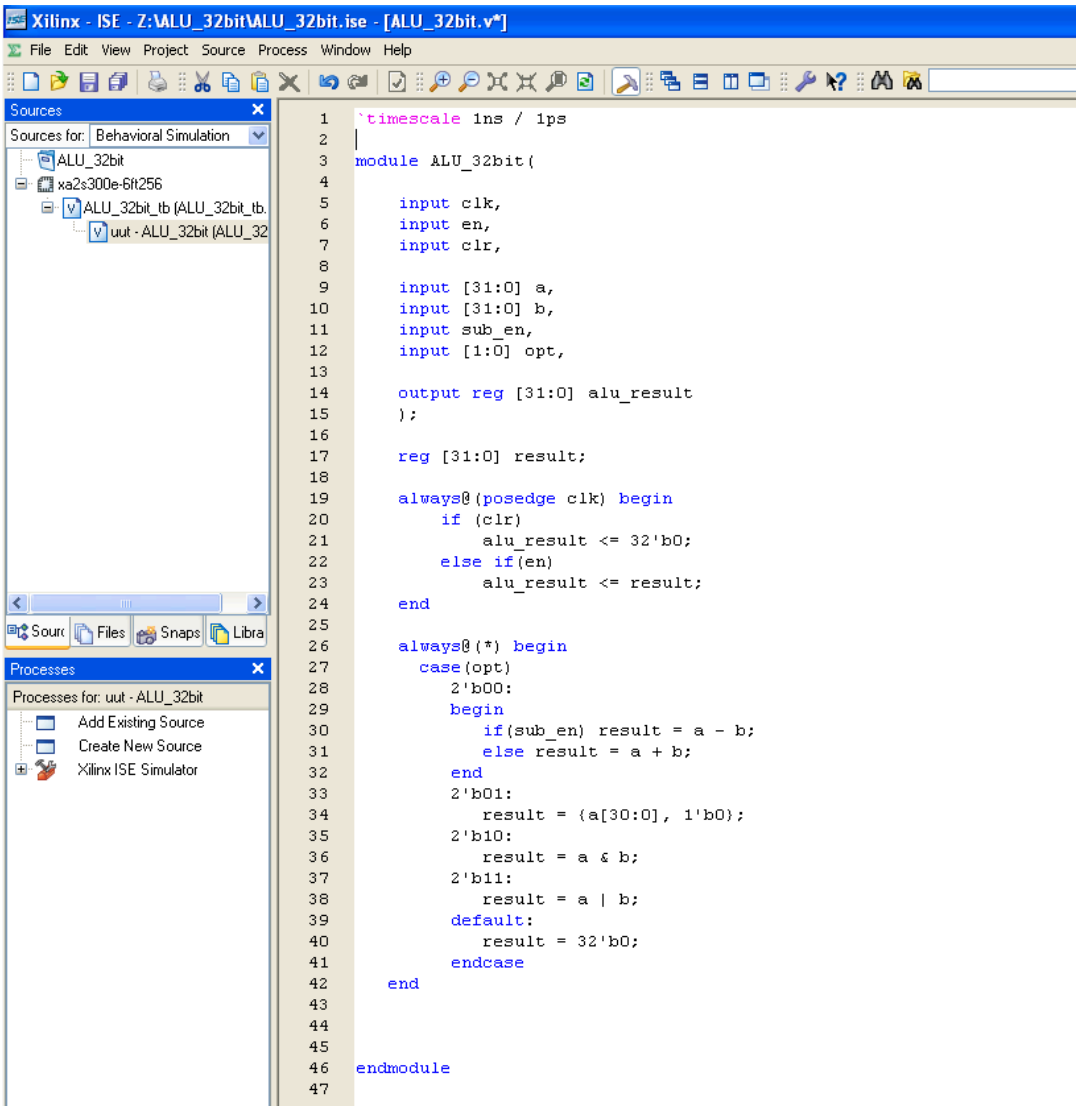


The number of D-FF: 32

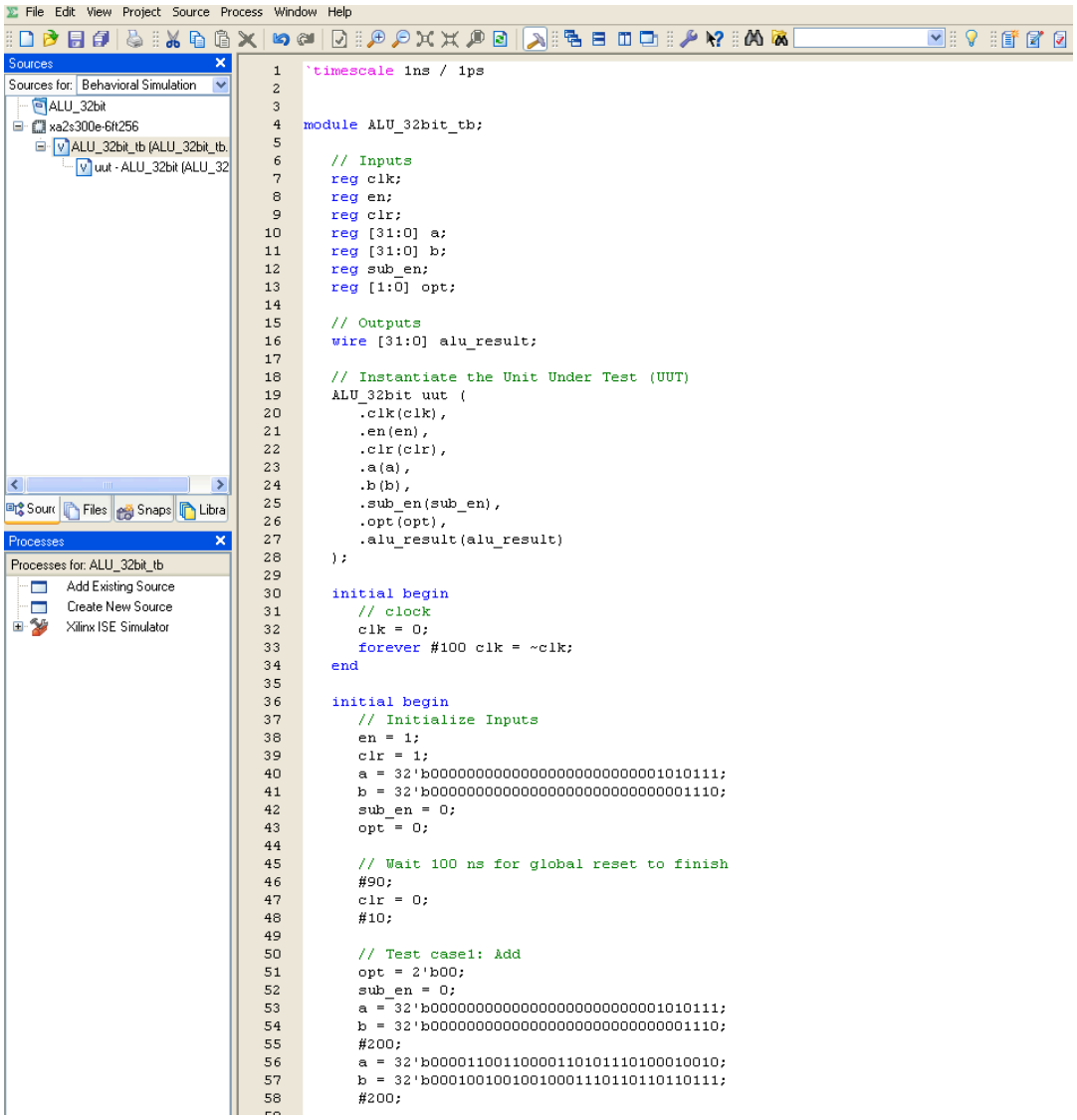
The number of LUTs: 160

C : Write a Verilog equivalent of 32-bit ALU

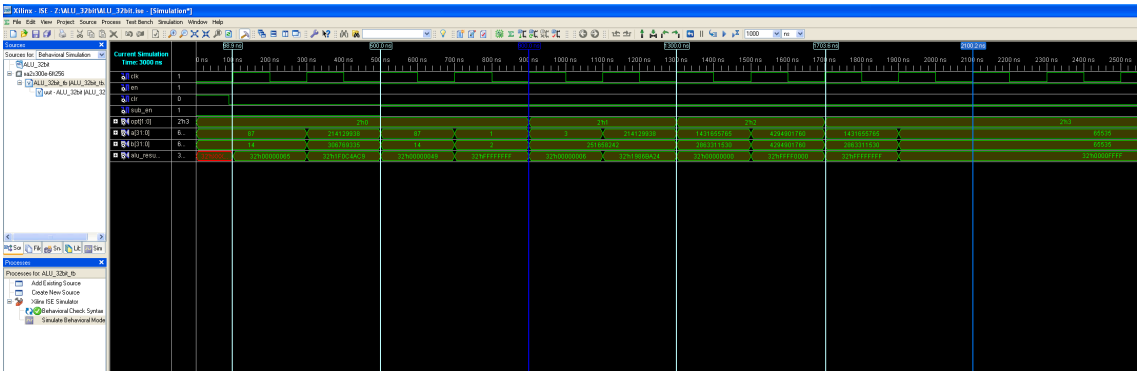
1. Code for logic and testbench
    - Logic


Xilinx - ISE - Z:\ALU_32bit\ALU_32bit.ise - [ALU_32bit.v*]
File Edit View Project Source Process Window Help
Sources
Sources for: Behavioral Simulation
ALU_32bit
xa2s300e-6ft256
ALU_32bit_tb (ALU_32bit_tb.
uut - ALU_32bit (ALU_32
Sourc
Files
Snaps
Libra
Processes
Processes for: uut - ALU_32bit
Add Existing Source
Create New Source
Xilinx ISE Simulator
1 `timescale 1ns / 1ps
2
3 module ALU_32bit(
4
5 input clk,
6 input en,
7 input clr,
8
9 input [31:0] a,
10 input [31:0] b,
11 input sub_en,
12 input [1:0] opt,
13
14 output reg [31:0] alu_result
15);
16
17 reg [31:0] result;
18
19 always@(posedge clk) begin
20 if (clr)
21 alu_result <= 32'b0;
22 else if(en)
23 alu_result <= result;
24 end
25
26 always@(*) begin
27 case(opt)
28 2'b00:
29 begin
30 if(sub_en) result = a - b;
31 else result = a + b;
32 end
33 2'b01:
34 result = {a[30:0], 1'b0};
35 2'b10:
36 result = a & b;
37 2'b11:
38 result = a | b;
39 default:
40 result = 32'b0;
41 endcase
42 end
43
44
45
46 endmodule
47

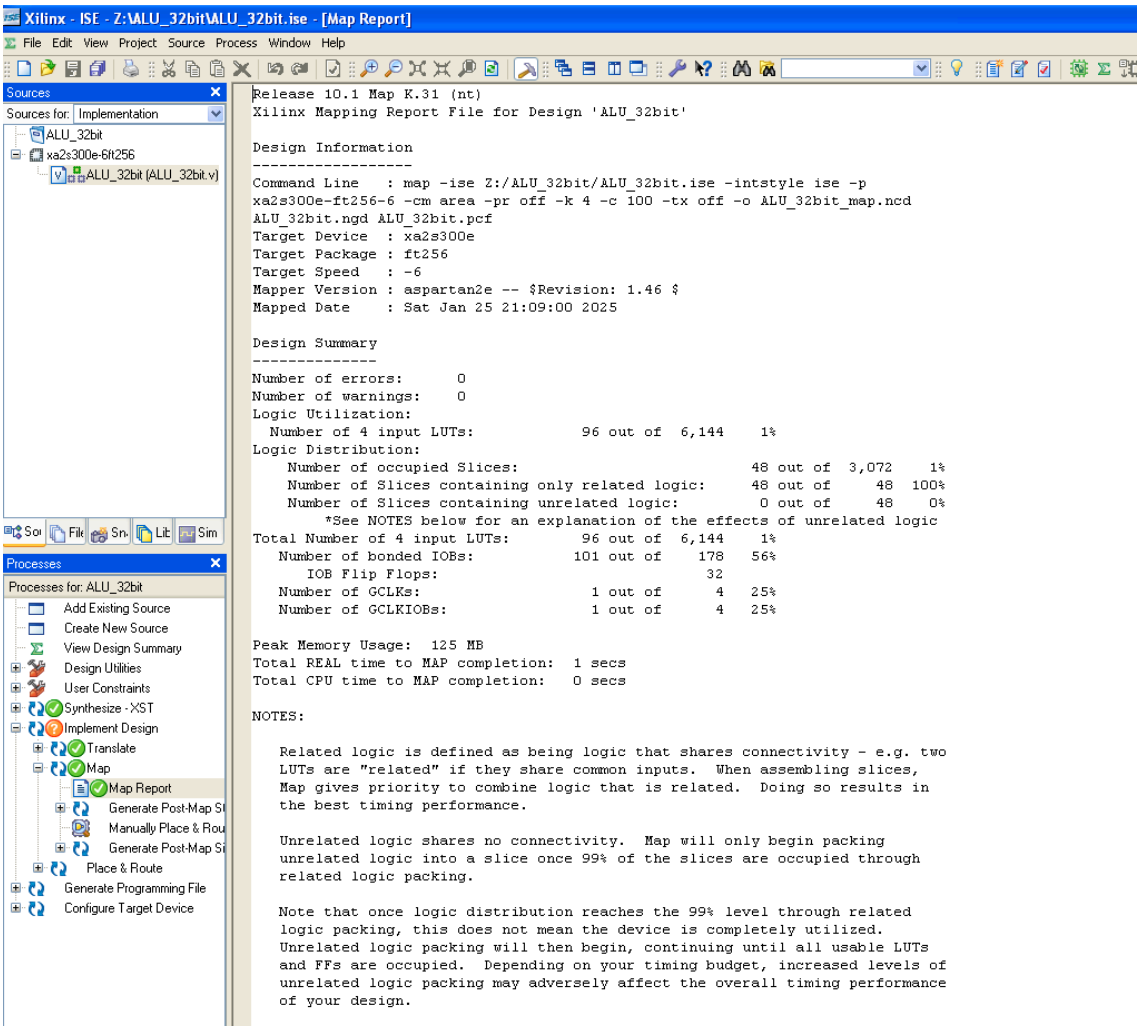
- Testbench



2. Screen capture of the waveforms generated by the behavioral simulation tools



3. Log file of the mapper



The number of D-FF: 32

The number of LUTs: 96

4. Brief comment on the number of gates as compared to the schematic version

   The number of gates generated by Verilog code is higher than that generated by schematics because redundant gates are produced when designing multi-layer nested wide-bit components using schematics.