Computer Organization
Lab 1: 32-bit ALU

Due: 2022/4/5

1. Goal

The goal of this LAB is to implement a 32-bit ALU (Arithmetic Logic Unit). ALU is the basic computing component of a CPU. Its operations include AND, OR, addition, subtraction, etc. This LAB will help you understand the CPU architecture. LAB 1 will be reused; you will use this module in later LABs. The function of testbench is to read input data automatically and output erroneous data. Please unzip the files in the same folder.

2. Homework Requirement

a. Please use Xilinx Vivado or ModelSim as your HDL simulator. (Xilinx Vivado is preferred)
b. Please **attach student IDs as comments** at the top of each file.
c. Please zip the archive and **name it as "ID.zip"** (e.g., 109XXXXX.zip) before uploading to e3
d. Testbench module is provided.
e. **Any work by fraud will absolutely get a zero point.**
f. The names of top module and IO ports must be named as follows:

Top module: alu.v

```
module alu(
    clk, // system clock (input) rst_n,
    // negative reset (input)
    src1, // 32 bits source 1 (input)
    src2, // 32 bits source 2 (input)
    ALU_control, // 4 bits ALU control input (input)
    result, // 32 bits result(output)
    zero, // 1 bit when the output is 0, zero must be set (output)
    cout, // 1 bit carry out (output)
    overflow // 1 bit overflow(output)
);
```

ALU starts to work when the signal rst_n is 1, and then catches the data from src1 and src2.

In order to have a good coding style, please obey the rules below:

- One module in one file.
- Module name and file name must be the same.

For example: The file "alu.v" only contains the module "alu".

g. instruction set: basic operation instruction (60%)

ALU Action	Name	ALU Control Input
And	And	0000
OR	Or	0001
Add	Addition	0010
Sub	Subtraction	0110
Nor	Nor	1100
Slt	Set less than	0111

h. zcv three control signal: zero, carry out, overflow (30%)

1. **“zero”**must be set when the result is 0.
2. **“cout”**must be set when there is a carry out.
3. **“overflow”**must be set when overflow.

Only "add" and "sub" operations need to handle carry out and overflow flag

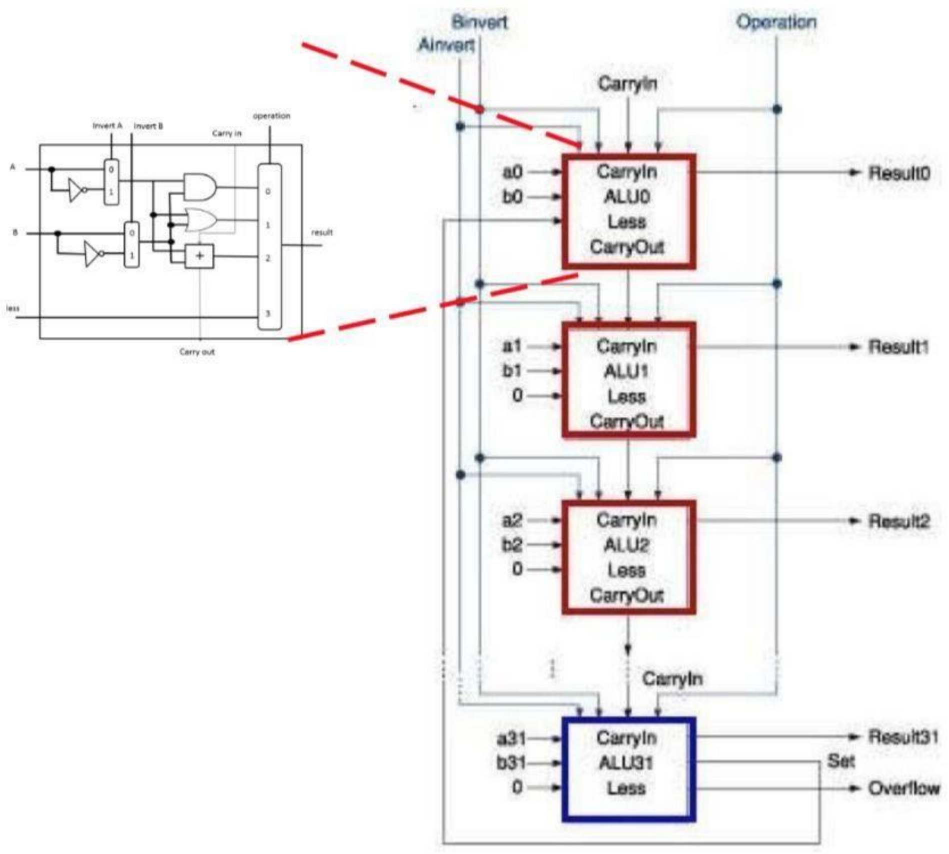
i. About alu_top.v

alu_top.v is an 1-bit ALU module. We don’t force you to use alu_top.v to implement this homework. However, we still strongly recommend that you implement this homework by using alu_top.v like the architecture diagram below. The design will become more complex in the following homework, so implementing your design by connecting multiple modules is an essential skill.

Note: You can add any module into your design if you want, but the top module must be alu in the alu.v defined in section f.

3. Architecture Diagram

1-bit ALU (Top) **32-bit ALU**

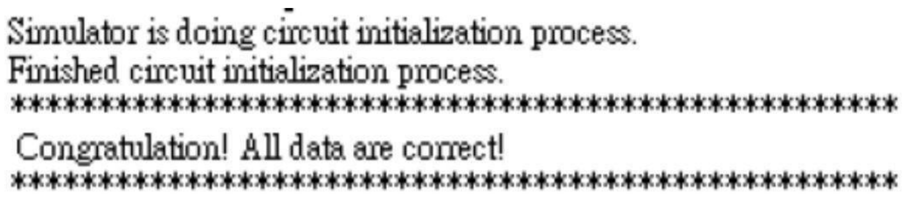


Blue frame is 1-bit ALU (Bottom)

4. Grade

a. **Total:**100 points (hidden cases will be evaluated)
b. **Report:** 10 points
c. **Late submission:** Score * 0.8 before 4/11. After 4/11, you will get 0.
d. **Format:** Put all your design sources(alu.v and alu_top.v in this lab) into one directory named "your_student_id" and zip the directory named "your_student_id.zip". You only need to submit "your_student_id.zip". If your submission doesn't meet the required format, you'll get 10 points punishment.
e. Plagiarism will get 0 point

Please add all the .txt files into the project (please refer to section 5), after simulation finishes, you will get some information.

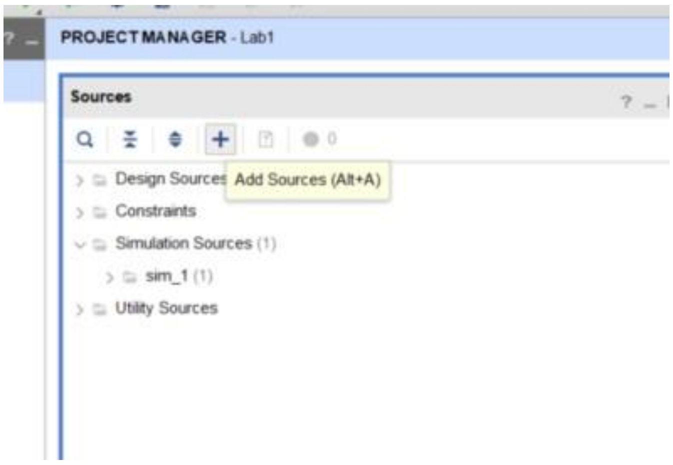


```
Simulator is doing circuit initialization process.
Finished circuit initialization process.
*****************************************************
Congratulation! All data are correct!
*****************************************************
```

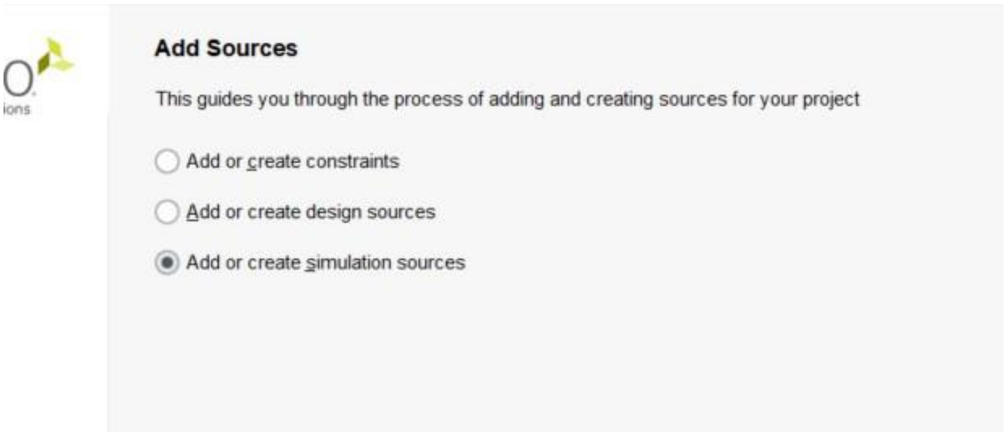
5. Add Simulation Source

Before running simulation, you need to add testcases into your project.

1. Add sources



2. Add simulation sources

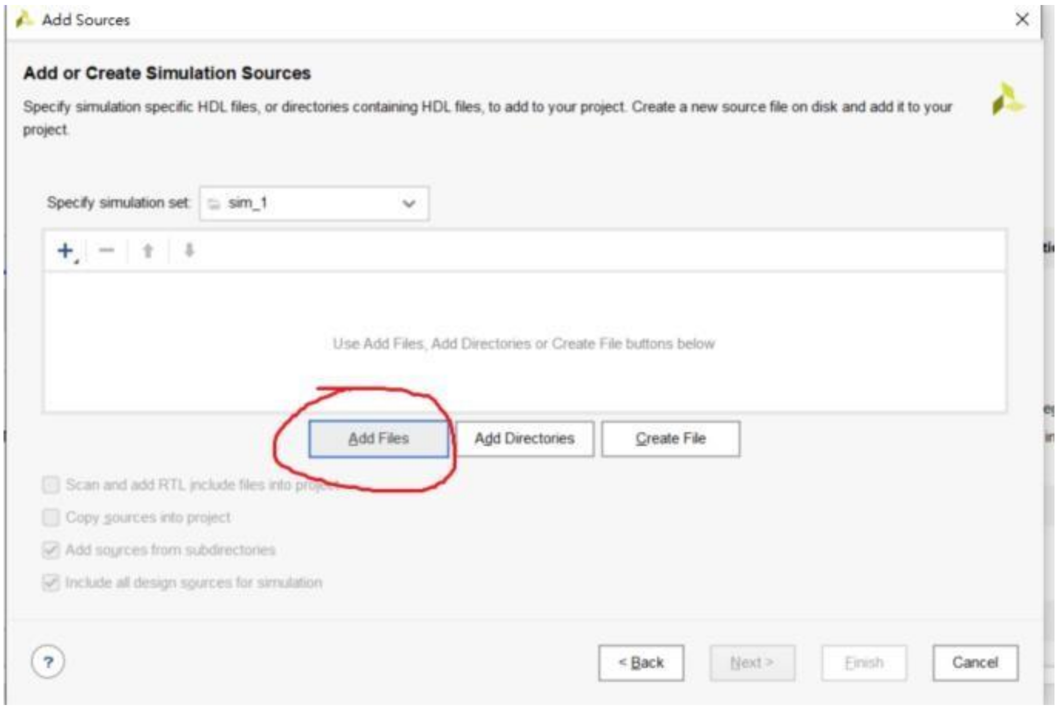


Add Sources

This guides you through the process of adding and creating sources for your project

- ○ Add or create constraints
- ○ Add or create design sources
- ● Add or create simulation sources

3. Add files



Add Sources

Add or Create Simulation Sources

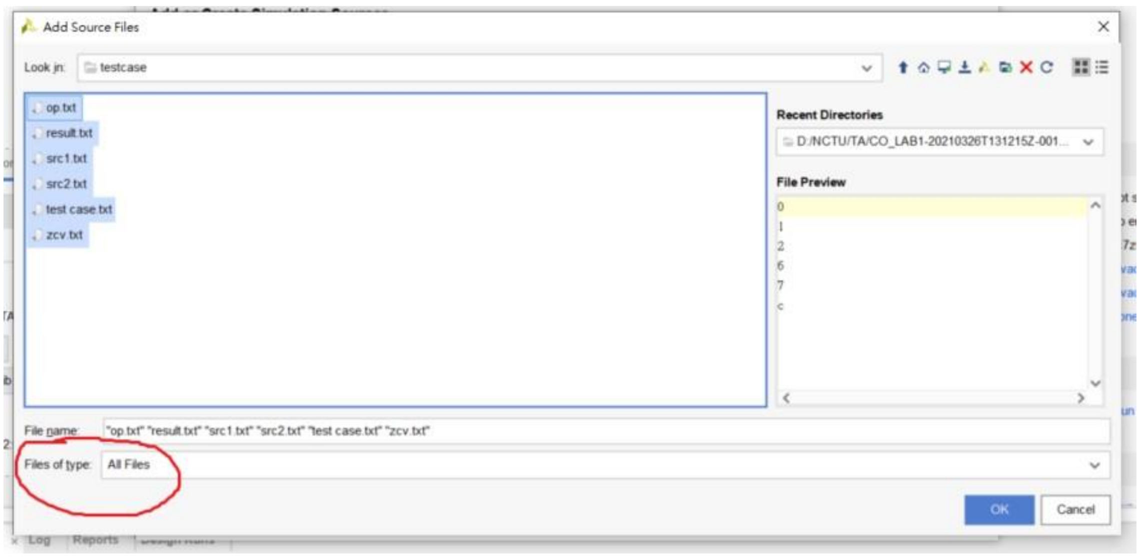
Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

Use Add Files, Add Directories or Create File buttons below

Add Files
Add Directories
Create File
Scan and add RTL include files into project
Copy sources into project
Add sources from subdirectories
Include all design sources for simulation
< Back
Next >
Finish
Cancel

4. Add all files in testcase directory



Add Source Files

Look in: testcase

- op.txt
- result.txt
- src1.txt
- src2.txt
- test case.txt
- zcv.txt

Recent Directories: D:/NCTU/TA/CO_LAB1-20210326T131215Z-001...

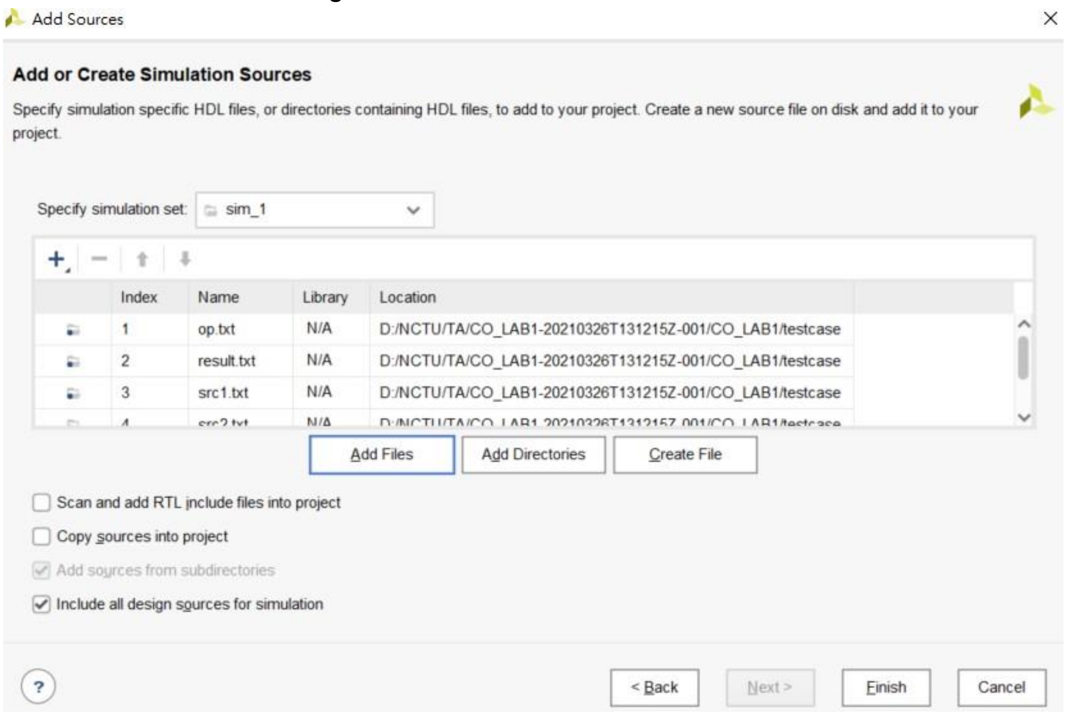
File Preview:

```
0
1
2
6
7
c
```

File name: "op.txt" "result.txt" "src1.txt" "src2.txt" "test case.txt" "zcv.txt"

Files of type: All Files

5. Finish and start simulating



Add Sources

Add or Create Simulation Sources

Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

	Index	Name	Library	Location
	1	op.txt	N/A	D:/NCTU/TA/CO_LAB1-20210326T131215Z-001/CO_LAB1/testcase
	2	result.txt	N/A	D:/NCTU/TA/CO_LAB1-20210326T131215Z-001/CO_LAB1/testcase
	3	src1.txt	N/A	D:/NCTU/TA/CO_LAB1-20210326T131215Z-001/CO_LAB1/testcase
	4	src2.txt	N/A	D:/NCTU/TA/CO_LAB1-20210326T131215Z-001/CO_LAB1/testcase

- [ ] Scan and add RTL include files into project
- [ ] Copy sources into project
- [x] Add sources from subdirectories
- [x] Include all design sources for simulation