Computer Organization

Lab 1: 32-bit ALU

Due: 2018/04/06

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. HW Requirement

a. Please use Xilinx or ModelSim as your HDL simulator.(ModelSim ispreferred)

b. Please attach student IDs as comments at the top of each file.

Please zip the archive and name it as "ID.zip" (e.g., 0516XXX.zip or 0516XXX_0516XXX.zip) before uploading to e3

c. Testbench module is provided.

d. Any work by fraud will absolutely get a zero point.

e. 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 bitssource 1 (input)
        src2,           // 32 bits source2 (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".

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

ALU action	Name	ALU control input
And	**And**	**0000**
Or	**Or**	**0001**
Add	**Addition**	**0010**
Sub	**Subtract**	**0110**
Nor	**Nor**	**1100**
Slt	**Set less than**	**0111**

zcv three control signal : zero、carry out、overflow (30%)

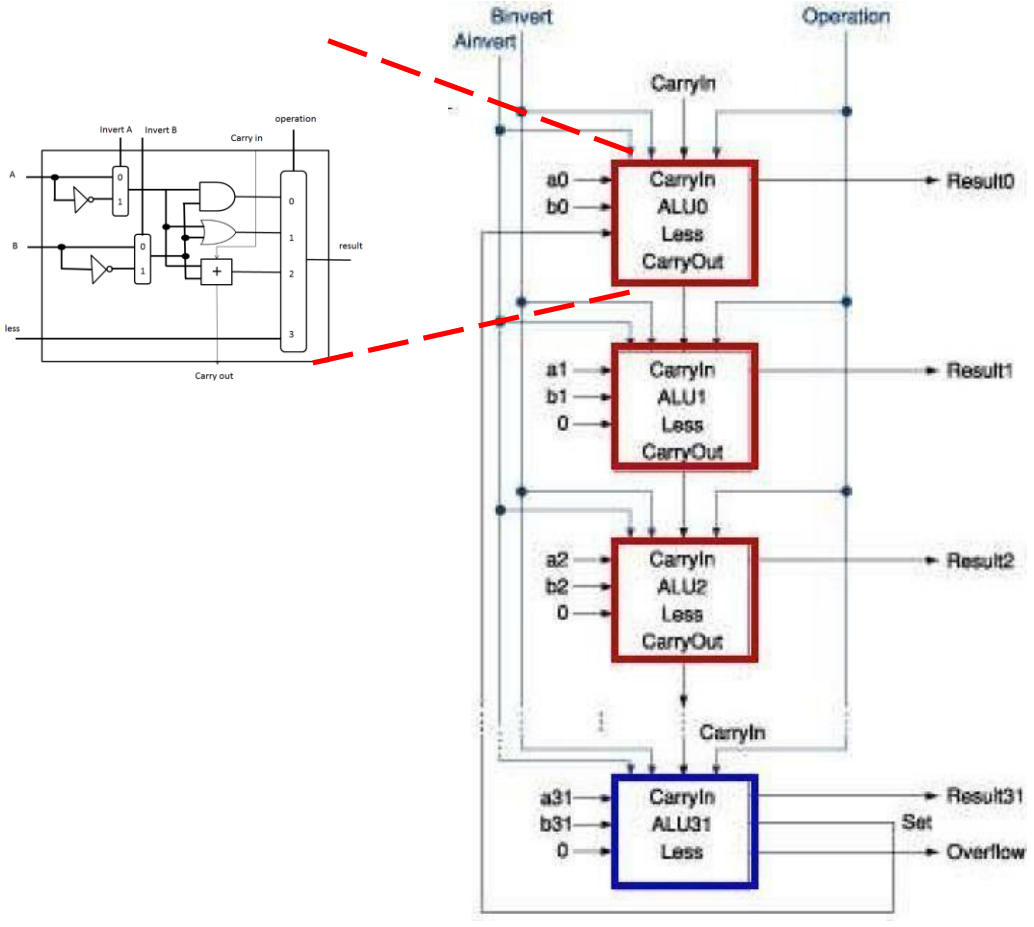
zero must be set when the result is 0.

cout must be set when carry out.

overflow must be set when overflow.

3. Architecture diagram

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



Blue frame is 1-bit ALU (Bottom)