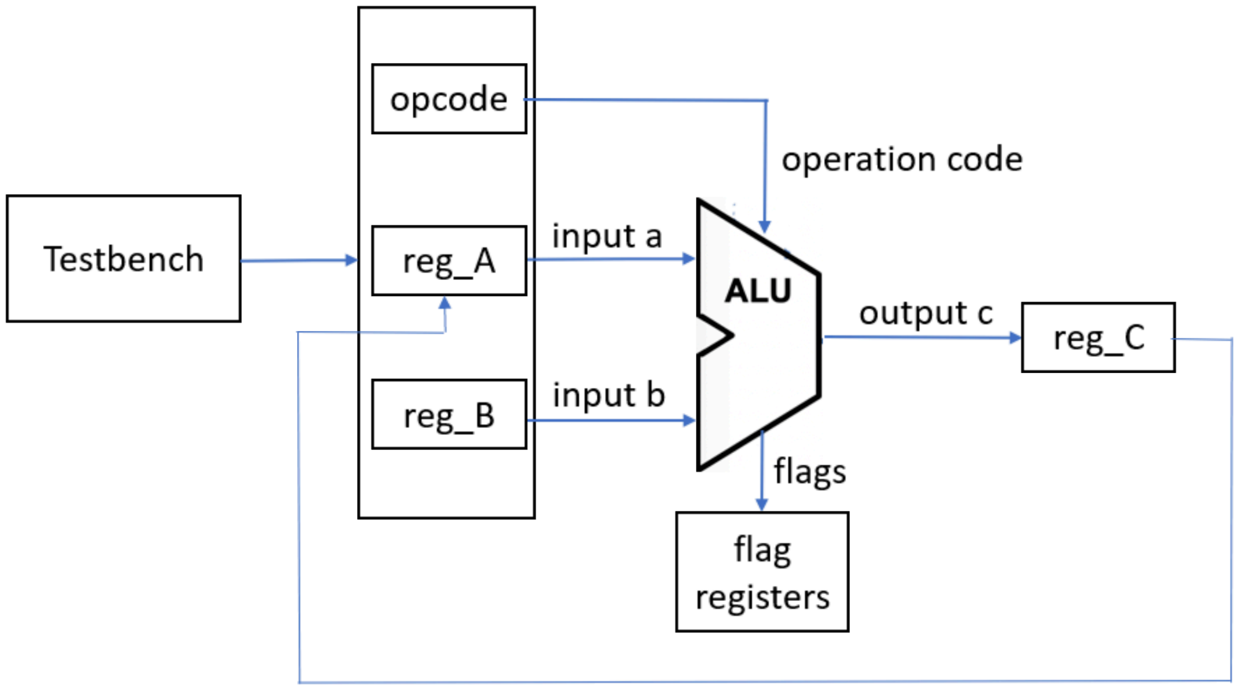
Programme 3 Instruction

Overview

In the class, the teacher introduced the Arithmetic and Logic Unit(ALU) to us. This module is to do math co-processing. In this programme, we will use verilog to complete it.
I will give you examples of verilog module file and testing file. You should add other operations into the code file and analyze the result.

Block Diagram



The ALU must support:
- add, sub, mult, div
- addi
- addu, subu, multu, divu
- addiu
- sqrt
- and, or, nor, xor, xnor
- andi, ori
- slt, slti

I will give you example of sla and srai.

In the diagram we can see that the reg_C is connected to reg_A. I use a thinner line to show you that it is sometimes not connected. For the instructions like addi, you should rewrite the result to reg_A; For other instructions like add, this relation should be cut.
The flag registers includes zero flag, negative flag and overflow flag. The negative flag will appear when you do subtraction with unsigned values or

other arithmetic with signed values. In the addition part, there may be overflow and the overflow detection will better improve our ALU module. The zero flag will be important in the next cpu programme because it decide the jump in a cpu. There are more instructions(bne, beq, etc) in a cpu related to the zero flag.

You should handle two verilog file:
- ALU.v
- test_ALU.v

And your project report.
I will give you example test bench and you should extend the test bench by yourself and analyze the final result in your report.

Grading

Support the Arithmetic/Logic operations - 60%

With special handling for flags - 30%
- Sign extend - 10%
- Zero extend - 10%
- Overflow detection - 10%

Project report - 10%