

Computer-Aided VLSI System Design

Verilog HW 1

Problem 1.

Image processing is an important part in multimedia. We explore how ALU can help process image. Based on the ALU in Lab 1, special instructions for image processing are needed to accomplish for our new design – ALU with I/O register. The input registers prevent signal glitches directly transmitting to the combinational circuits, and the output register can prevent signal glitches influencing on the next circuits. Based on the ALU of Lab 1, please design an ALU with I/O register, which conforms to the specification with the block diagram of an 8-bit ALU with I/O register given in Fig. 1 and the 3-bit instruction set in the table below.

inst_i [2:0]	Operation	Notes
000	Unsigned Addition	$data_o = data_a_i + data_b_i$
001	Unsigned Subtraction	$data_o = data_b_i - data_a_i$
010	Unsigned Multiplication	$data_o = data_a_i * data_b_i$
011	NOT	$data_o = \sim data_a_i$
100	XOR	$data_o = data_a_i \oplus data_b_i$
101	Absolute Value	$data_o = data_a_i $
110	Subtraction & Divide by 2	$data_o = (data_b_i - data_a_i) \gg 1$
111	Unused	Unused

Note that the output signal “data_o” has 16 bits and uses 2’s complement representation. The input data (a & b) are unsigned for instruction 000/001/010/110, and signed for instruction 101. You have to do zero-extension when output is always positive and do sign-extension when output may be negative (e.g., instruction 001/110). Please do zero-extension for “NOT” and “XOR” operations.

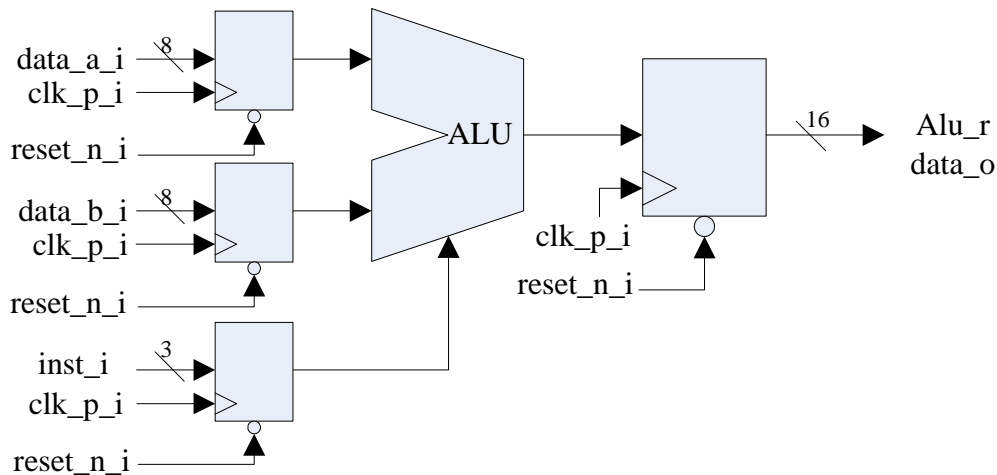


Figure 1. 8-bit ALU with I/O register

- 1.1 An unfinished Verilog RTL template program is provided. Please complete the program to achieve the above functions. It is recommended to verify one instruction at the same time. (Verify 000. If correct, then 001, 010, ...) That will speed up your debugging process. The signal “test_instruction” of the provided testbench is useful to test your codes.
- 1.2 Please use the provided testbench to test the basic functions of your ALU with I/O register. We’ll score your ALU using this testbench.
- 1.3 Please note that the instructions from 000 to 100 are basic operations, while 101 and 110 are more difficult ones.

Electronic Submission

Please submit your design in one file with the naming convention: ***StudentID_HW1_alu.v*** (e.g., R98943001_HW1_alu.v). Please follow the electronic submission guideline (available on website).

Note:

1. In your design, please follow the port name and port order as used in the given Verilog template. Otherwise your design may fail our test for grading.
2. If you want to modify your code, please submit the new file with the name ***StudentID_HW1_alu_v1.v***.

Appendix [Naming Rules]

Please apply the following rules to name wires in your Verilog RTL codes.

- (1) It is okay for one wire to have several names. Verilog compiler will regard them as the same one. For clarity, it is suggested to give one wire several names.
- (2) “_w” should be added to the wire which is the input of a register.
- (3) “_d1” should be added to the wire which is the output of the first stage register, and “_d2”, “_d3” to the output of the second stage register, third stage register.
- (4) “_r” should be added to the wire which is the output of a register.
- (5) “_p” should be added to the clock or reset signal with positive edge trigger.
- (6) “_n” should be added to the clock or reset signal with negative edge trigger.
- (7) “_i” should be added to the wire which denotes the input of the top module.
- (8) “_o” should be added to the wire which denotes the output of the top module.

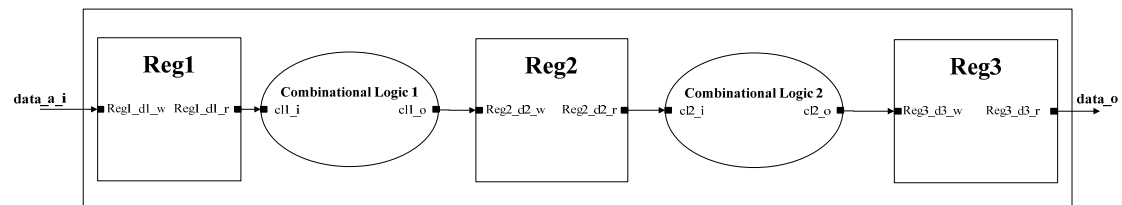


Fig. 2 Naming rules of wires

Using several names for one wire makes a design more readable. Fig. 2 shows an example with three stages of registers, including an input stage, one pipeline state, and an output stage. As can be seen, there are two names for a wire. One is for the previous stage, and the other is for the next stage. Naming wires clearly help you understand which part you are writing in a large design. With the above naming rules, you can identify the input/output signals and the internal signals in your circuit more easily.