NCTU-EE IC LAB – Spring 2018

Lab04 Exercise

Design: ALU

Data Preparation

1. Extract test data from TA's directory:

% tar xvf ~iclabta01/Lab04.tar

Design Description

We are implementing a calculator with special operating policy. Input and mode will be sequentially given respectively, and the calculator should output the answer whenever it has finished calculating. Note that the output should be sequentially given as well. The calculating operation is in the following way.

The first output will be related to four zeros and the first input, and the second output will be related to three zeros, the first input and the second input, and so on. We reorder the related inputs into numerically descending manner. Then do the arithmetical operation below.

- Mode0

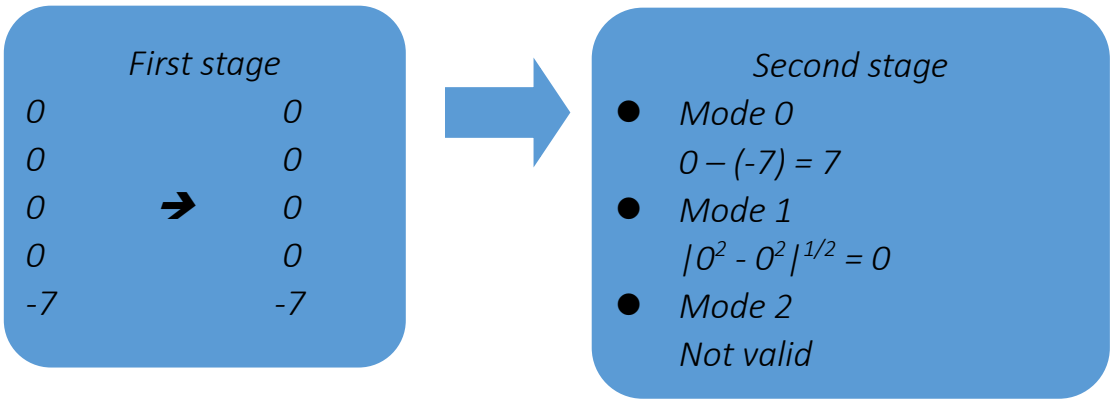
  Subtract the first number by the fifth number.
- Mode1

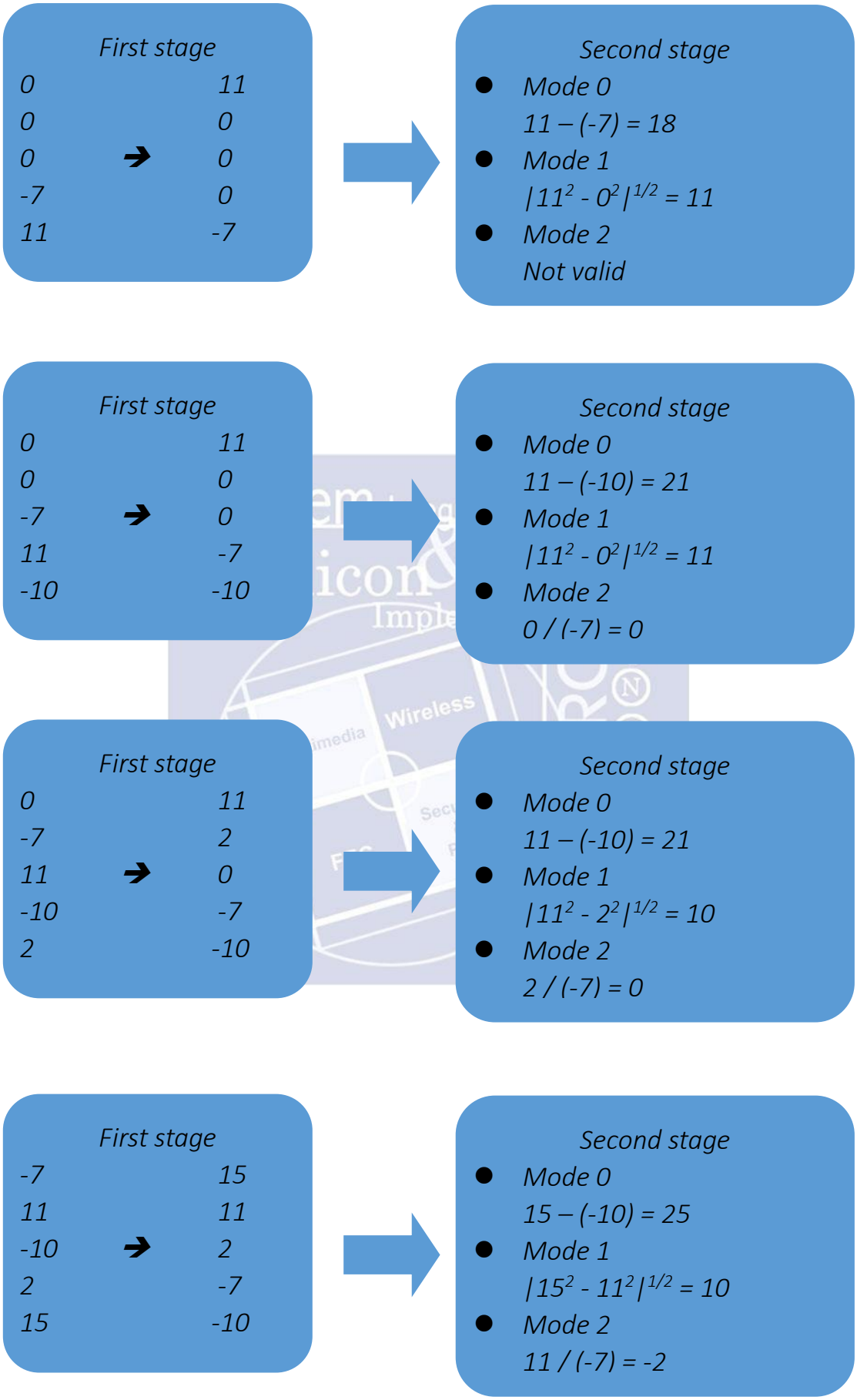
  Take a square root of the absolute value of the square difference of the first and the second number.
- Mode2 (the fourth number won't be zero)

  Divide the second number by the fourth number.

For example, in this case the output sequence will be like, 0, 18, 11, 21, 10, …

Cycle :	1	2	3	4	5	6	7	8	9
Input :	-7	11	-10	2	15	-9	-7	3	4
Mode:	1	0	1	0	1	0	2	1	0




First stage
0 11
0 0
0 ➔ 0
-7 0
11 -7
Second stage
Mode 0
11 – (-7) = 18
Mode 1
|11² - 0²|^1/2 = 11
Mode 2
Not valid
First stage
0 11
0 0
-7 ➔ 0
11 -7
-10 -10
Second stage
Mode 0
11 – (-10) = 21
Mode 1
|11² - 0²|^1/2 = 11
Mode 2
0 / (-7) = 0
First stage
0 11
-7 2
11 ➔ 0
-10 -7
2 -10
Second stage
Mode 0
11 – (-10) = 21
Mode 1
|11² - 2²|^1/2 = 10
Mode 2
2 / (-7) = 0
First stage
-7 15
11 11
-10 ➔ 2
2 -7
15 -10
Second stage
Mode 0
15 – (-10) = 25
Mode 1
|15² - 11²|^1/2 = 10
Mode 2
11 / (-7) = -2

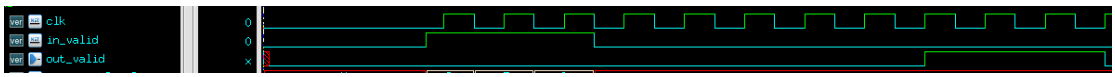
Input and Output

Input signal	Bit width	Definition
clk	1	Clock
rst_n	1	Asynchronous active-low reset
in_valid	1	Enable input signal
mode	2	Calculation mode (0, 1, 2)
in_number	5	Input number

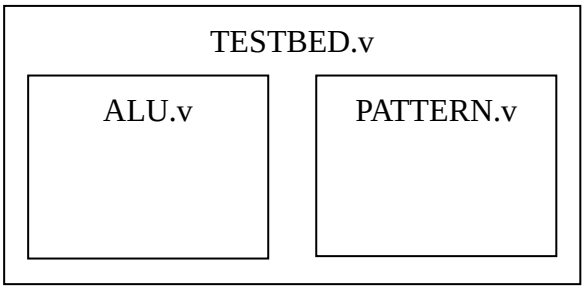
Output signal	Bit width	Definition
out_valid	1	Enable output check
out_number	7	Output number

Specification

1. Top module name: **ALU**(design file name: **ALU**.v)
2. It is **asynchronous** reset and **active-low** architecture.
3. The reset signal would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.
4. The clock period of the design must be shorter than **5ns**.
5. Input data will be given continuously when in_valid is pulled up.
6. out_number should output **continuously** when out_valid is pulled up.
7. The synthesis result of data type **cannot** include any latches.
8. After synthesis, you can check ALU.area and ALU.timing. The area report is valid when the slack at the end of timing report is **non-negative**.
9. You must output your answer within **50 cycles**.
10. In mode 2, the **quotient** will be **rounded to the nearest integer**. For -2.49 and 2.5, they will be -2 and 3 respectively.
11. The **square root** result will be **rounded down**. For 2.51, it will be 2.
12. Out_valid should be pulled up for the same time as in_valid.



Block Diagram



Note

1. **Grading policy:**

   RTL and Gate-level simulation correctness: 70%

   Performance: 30%

   - Simulation time 15%
   - Area 15%

2. **Please upload the following file on e3 platform before 12:00 p.m. on Oct. 23:**

   - ALU_iclabXX.v (XX is your account number)
   - OO_iclabXX.txt (OO is the clock period of your design, like 4_iclab99.txt)

3. **Template folders and reference commands:**

   01_RTL/ (RTL simulation) **./01_run**

   02_SYN/ (Synthesis) **./01_run_dc**

   (Check the design if there's latch or not in ***syn.log*****)**

   (Check the design's timing in /Report/ **ALU.*timing***)

   03_GATE / (Gate-level simulation) **./01_run**

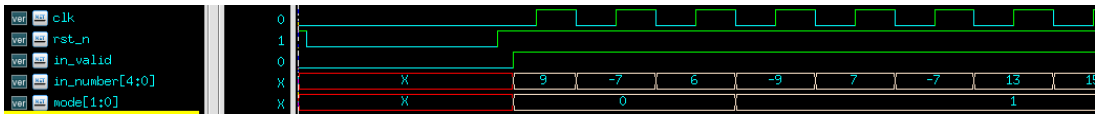
4. **Note:**

   Run 09_clean_up every time before you run 01_run.

   Clock period can change by one step of 0.1ns. For instance, 4.9ns is valid, and 4.95ns is not valid.

Example Waveform

Input signal



Output signal