Lab 3:
Arithmetic Logic Unit
(ALU)

Outline

- Introduction
- Hardware Description
  - Instructions
  - I/O Information
- Lab 3 Implementation
  - Data Format
  - Saturation and Rounding
  - Multiplication, Division and Sorting
- Criteria
  - Grading Policy
  - Simulation Result
  - Requirement & File Format
  - Deadline
  - Commands in Makefile

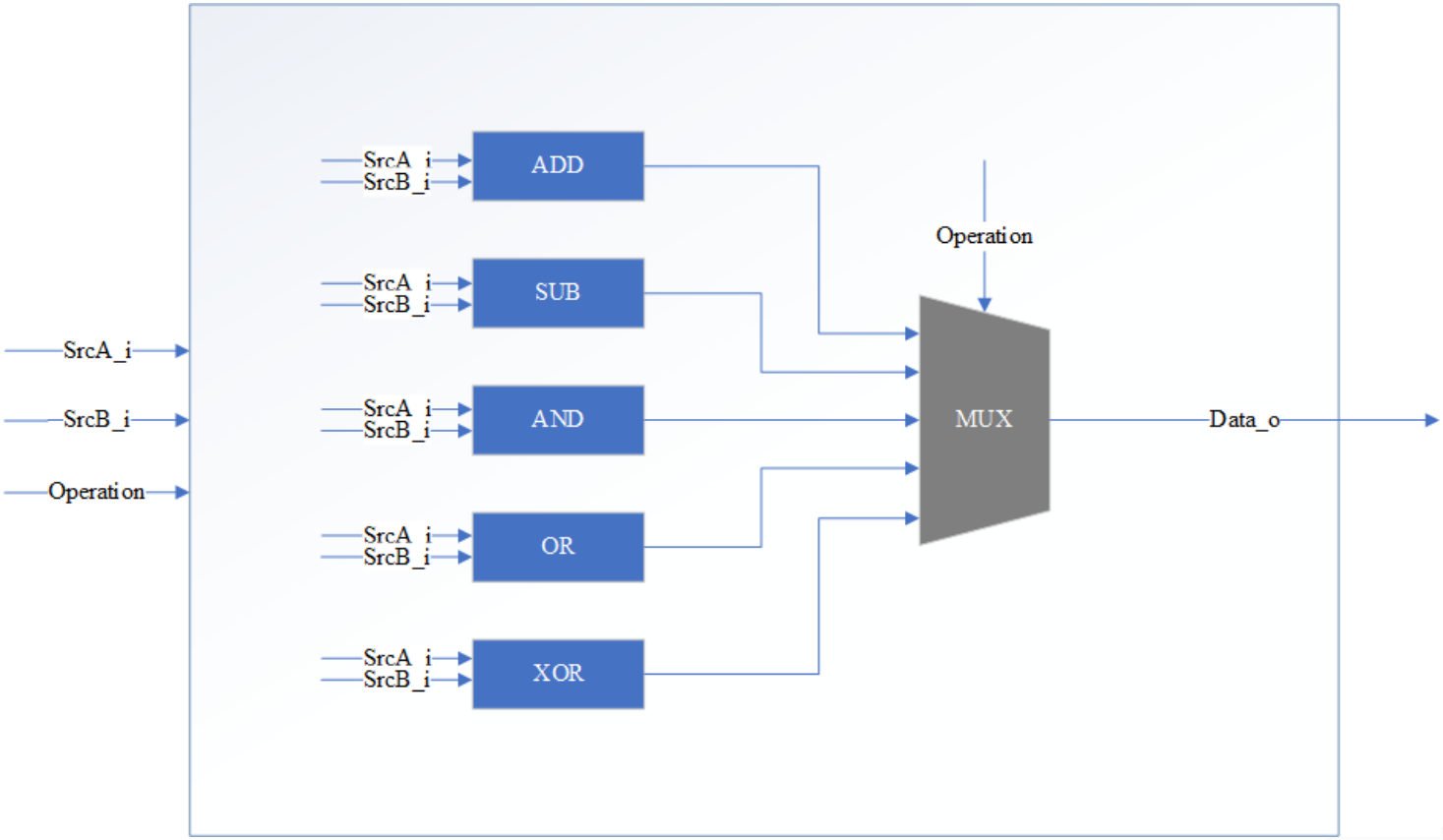
Introduction

- An arithmetic logic unit (ALU) is a digital circuit used to perform arithmetic and logic operations.

- In this lab, you will design a fixed-point ALU that supports signed addition, signed subtraction, signed multiplication, signed division and unsigned sorting.

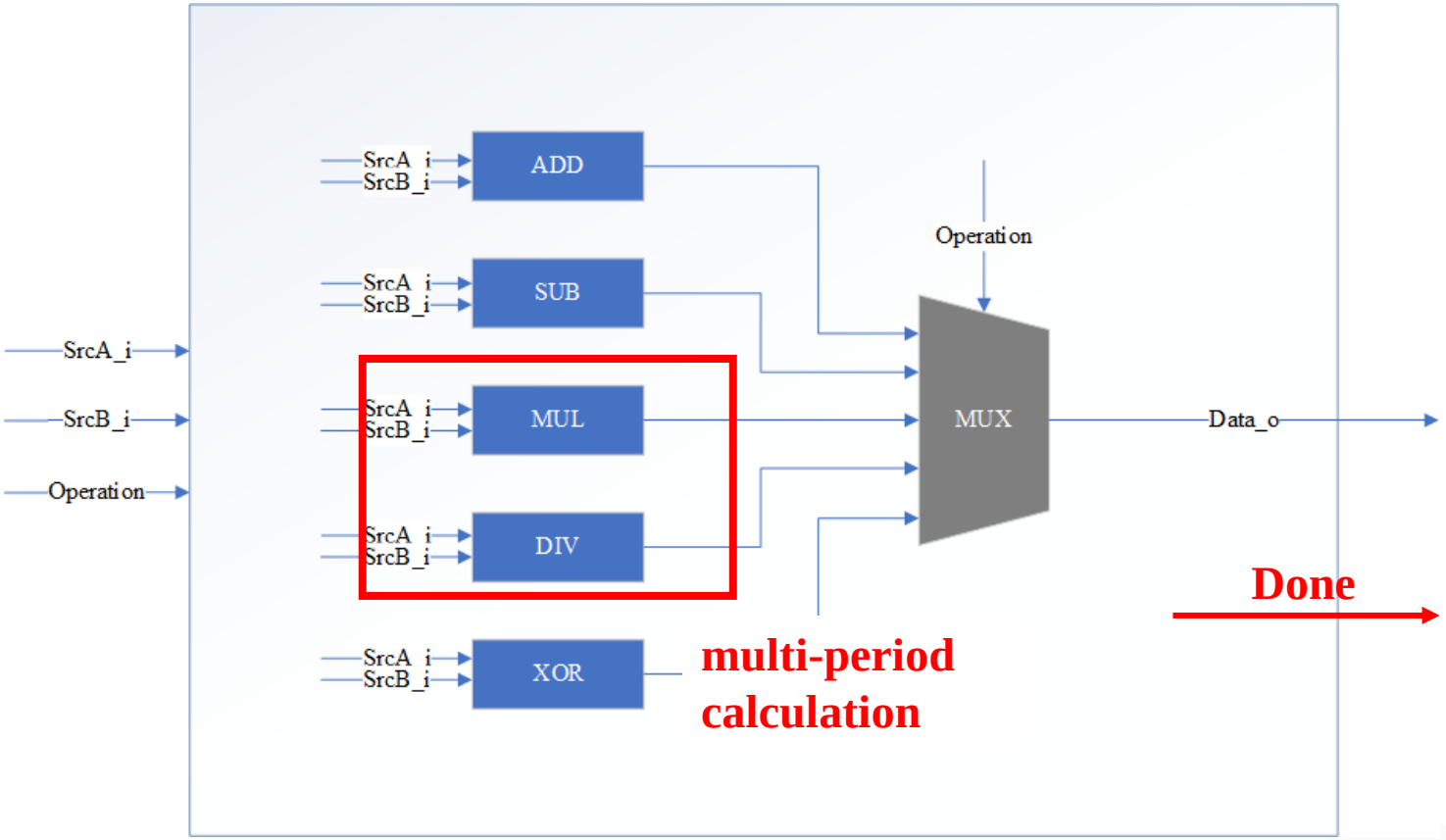
Introduction

- This is a basic ALU that performs simple and logic operations.



Introduction

- This is an advanced ALU that includes multi-period calculation.



Hardware Description

■ Instruction

Fixed-Point Operation	Operation Code	Description	Note
Signed Addition	3'b000	data_o = srcA_i + srcB_i	Output saturation is needed.
Signed Subtraction	3'b001	data_o = srcA_i – srcB_i	
Signed Multiplication	3'b010	data_o = srcA_i * srcB_i	Output saturation and rounding are needed.
Signed Division	3'b011	data_o = srcA_i / srcB_i	The format of data_o is signed $Q_{6.10}$ fixed point.
Sorting Nine Unsigned Numbers	3'b100	Sort nine numbers in ascending order.	The format of numbers are unsigned $Q_{6.10}$ fixed point.

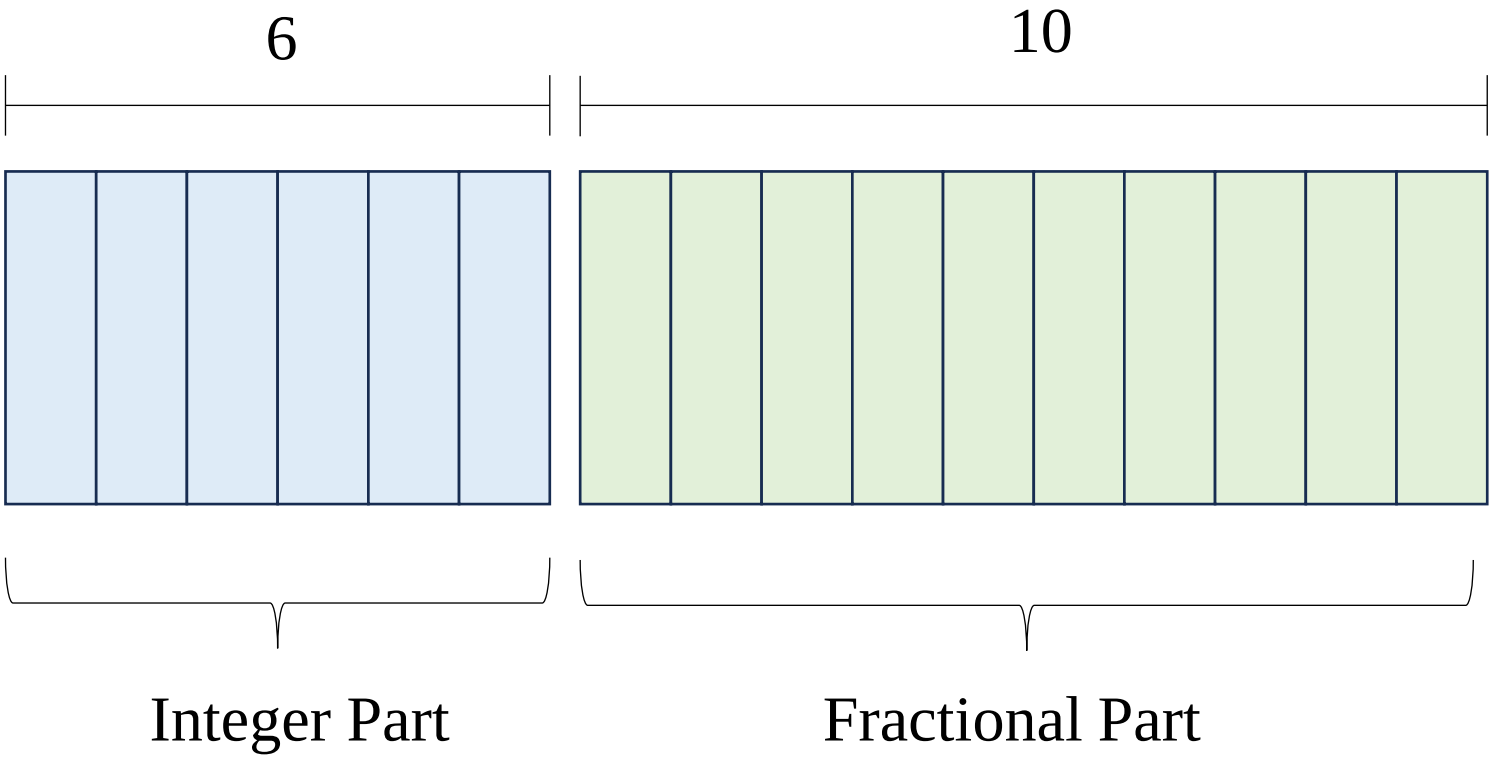
Hardware Description

I/O Information

Signal	I/O	Width	Description
clk	I	1	clock signal
rst	I	1	Active-high reset
operation	I	3	Operation code
done	O	1	Indicates ALU operation is complete
srcA_i	I	16	Signed $Q_{6.10}$ fixed-point number
srcB_i	I	16	
sortNum0~8_i	I	16	Unsigned $Q_{6.10}$ fixed-point number
sortNum0~8_o	O	16	
data_o	O	16	Signed $Q_{6.10}$ fixed-point number

Lab 3 Implementation

- Data Format : $Q_{6.10}$

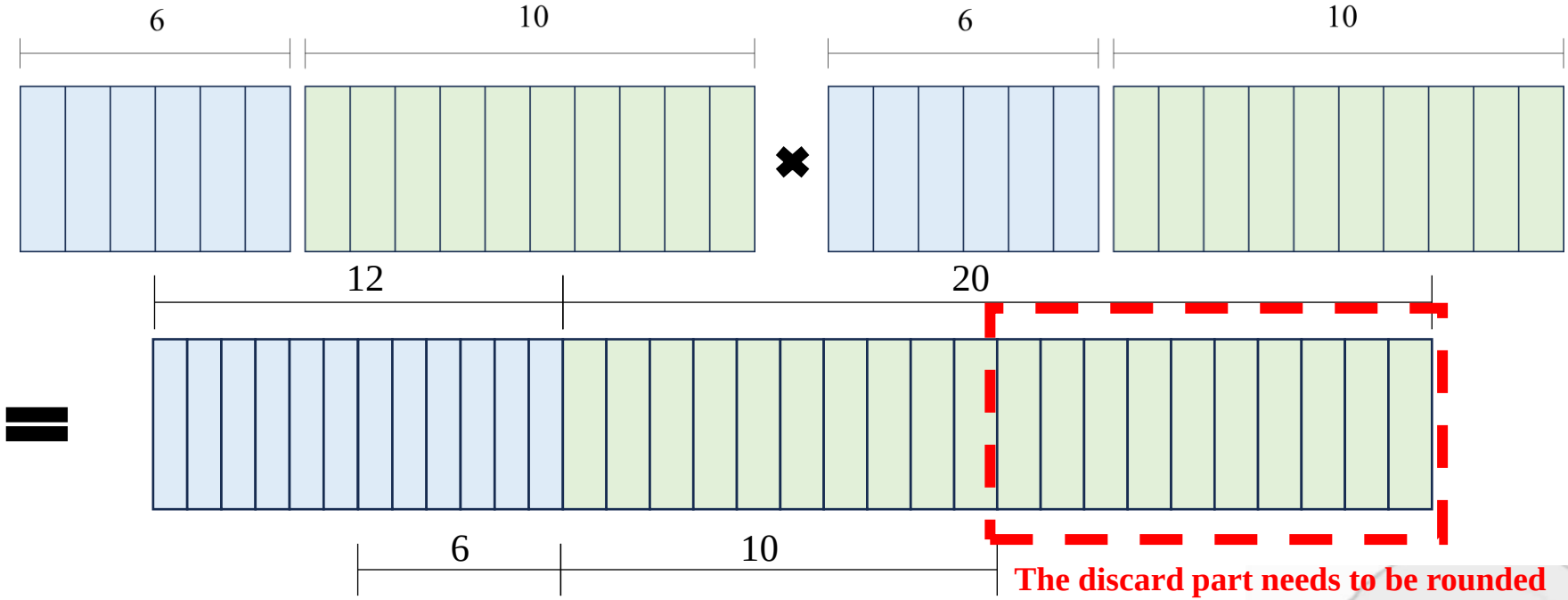


Lab 3 Implementation

- Saturation

  If the result exceeds the range of a 16-bit representation, clamp the values to the maximum or minimum.

- Rounding (round to nearest, ties to even)



Lab 3 Implementation

- Multiplication and Division

  You should design the multiplication and division hardware yourself and specify your design in the report.

- Sorting (ascending order)

  Avoid using a for-loop to solve this problem, and please specify which algorithm you have implemented.

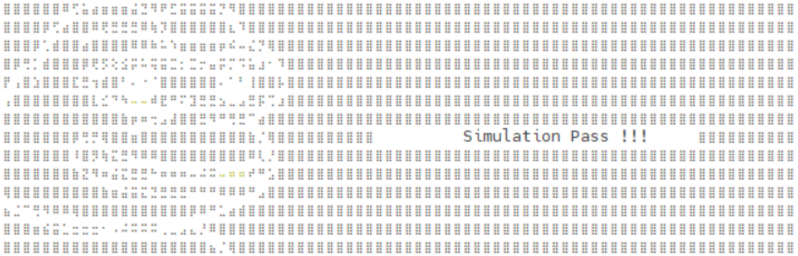
Criteria

- Grading Policy (100%)
  - Simulation Pass (90 %)
    - Signed Addition (18%)
    - Signed Subtraction (18%)
    - Signed Multiplication (18%)
    - Signed Division (18%)
    - Signed Sort (18%)
  - Report (10 %)

Criteria

- Simulation Result

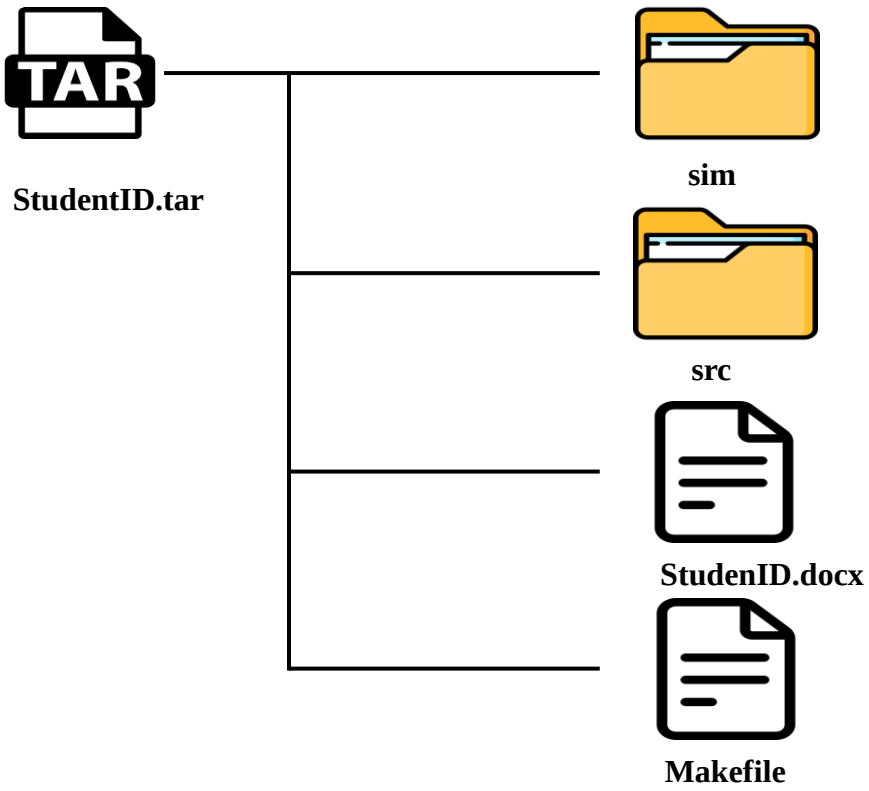
```
*************************
** Simulation Start!!! **
*************************

Addition Scenario 0: Pass
Addition Scenario 1: Pass
Addition Scenario 2: Pass
Addition Scenario 3: Pass

Subtraction Scenario 0: Pass
Subtraction Scenario 1: Pass
Subtraction Scenario 2: Pass
Subtraction Scenario 3: Pass

Multiplication Scenario 0: Pass
Multiplication Scenario 1: Pass
Multiplication Scenario 2: Pass
Multiplication Scenario 3: Pass

Division Scenario 0: Pass
Division Scenario 1: Pass
Division Scenario 2: Pass
Division Scenario 3: Pass

Sorting Scenario 0: Pass
Sorting Scenario 1: Pass
Sorting Scenario 2: Pass
Sorting Scenario 3: Pass
```




```
========================== Your Score is 90.00 / 90.00 ==========================

$finish called from file "/vlsisoc-data/ms113/113NCKU/113joechen777/hdl_lab3/sim/lab3_tb.sv", line 526.
$finish at simulation time              4300000
           V C S   S i m u l a t i o n   R e p o r t
Time: 4300000 ps
CPU Time:      0.620 seconds;       Data structure size:   0.0Mb
Thu Mar  6 01:09:31 2025
CPU time: .723 seconds to compile + .720 seconds to elab + .663 seconds to link + .678 seconds in simulation
```

Criteria

- Requirement & File Format



Points will be deducted for not following the file submission format.

Criteria

- Deadline: 2025/04/01 (Tue) 14:00
  - Late submissions will receive a partial score as follow:
    - 1 day late -> 80 %
    - 2 day late -> 50 %
    - 3 day late -> 20 %
    - Over 3 days late - > 0 %

Criteria

- Commands in Makefile

Situation	Command
RTL simulation	make vcs
Dump Waveform (no array)	make vcs WV=1
Dump Waveform (with array)	make vcs WV=2
Launch nWave	make wave
Delete waveform files	make clean
Compress homework to tar format	make tar