CSE 306 Computer Architecture Sessional Assignment on **Floating Point Adder**

Lab Group: B1 Group No: 06 Group Members:

1705077

1705085

1705087

1705088

1705089

May 30, 2021

Contents

1	Introduction	3
2	Problem Specification	3
3	Flowchart	4
4	High-level Block Diagram	5
5	Circuit Diagram	6
6	Used ICs	14
7	About the simulator	15
8	Discussion	15

1 Introduction

In this task, we have designed a floating point adder circuit that can take floating point inputs which can be either positive or negative and will generate their sum as output. The output will be in form of a floating point too.

In order to implement this circuit, we compared the exponents of the given inputs and right shifted the smaller value to match its exponent with the larger one. Then we performed the addition operation and checked if the output result is in normalized form. We also checked overflow/underflow of the exponent. If not then we normalize the result.

We will see this process in details in the following sections.

2 Problem Specification

There will be two floating point inputs in this circuit. Input will be provided in IEEE 754 Standard format of binary representation of the floating point number (sign bit, exponent bit, fraction bits). There will be 1 bit for the sign, 4 bits for the exponent and 11 bits for the fraction.

i Two 16 bit inputs A and B

The output will be in the same format too.

3 Flowchart

Flowchart of the addition algorithm is given below.

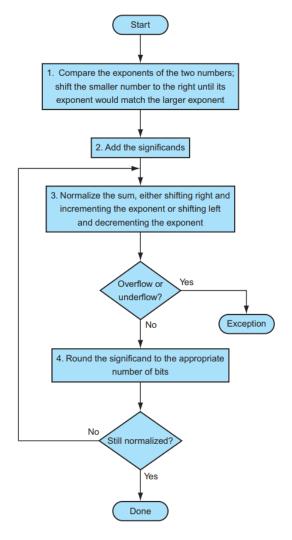


Figure 1: Floating point addition

4 High-level Block Diagram

The high level block diagram of the architecture

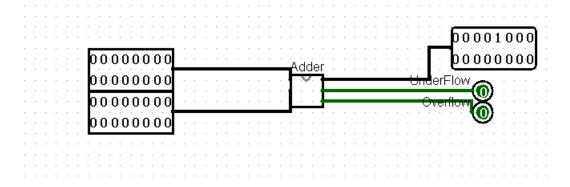


Figure 2: Block diagram of floating point adder.

5 Circuit Diagram

Detailed circuit diagram of important blocks are given below.

- 1. Preprocessor
 - Bit splitter

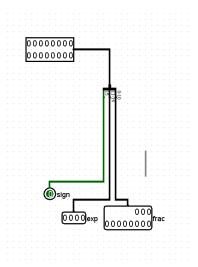


Figure 3: Diagram of bit splitter.

• Comparer

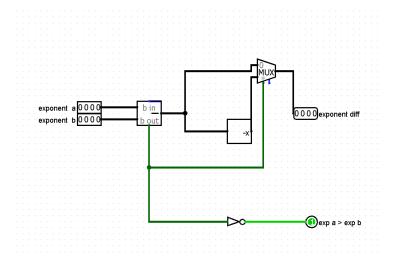


Figure 4: Diagram of comparer.

• Fraction shifter

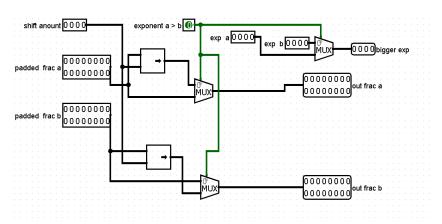


Figure 5: Diagram of fraction shifter.

• Padder

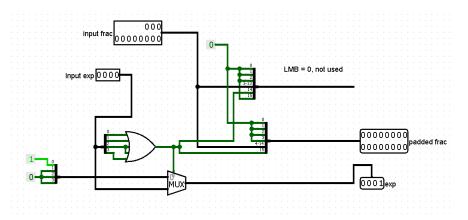


Figure 6: Diagram of padder.

• Fp preprocessor

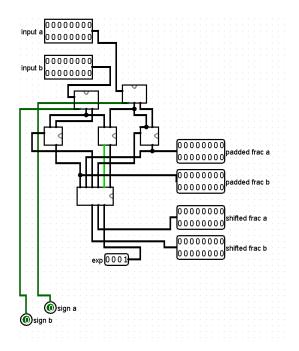


Figure 7: Diagram of fp preprocessor.

2. Adder

• 2s complement

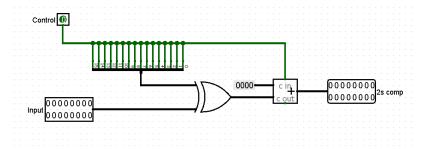


Figure 8: Diagram of 2s complement.

• IEEE adder

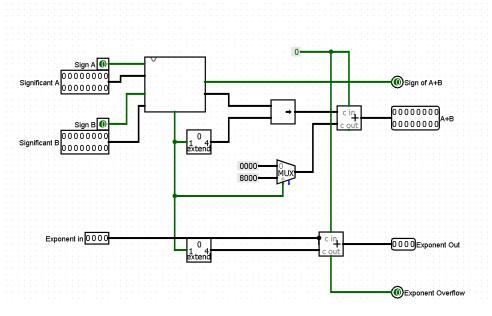


Figure 9: Diagram of IEEE adder.

• Magnitude comparator

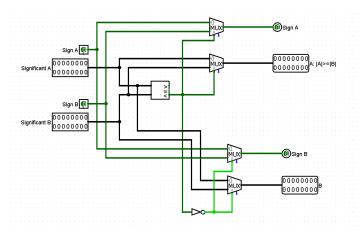


Figure 10: Diagram of magnitude comparator.

• Significant adder

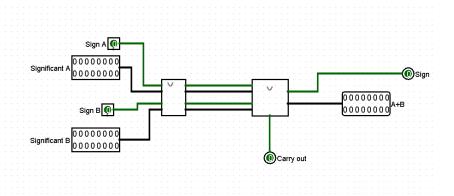


Figure 11: Diagram of significant adder.

• Special adder

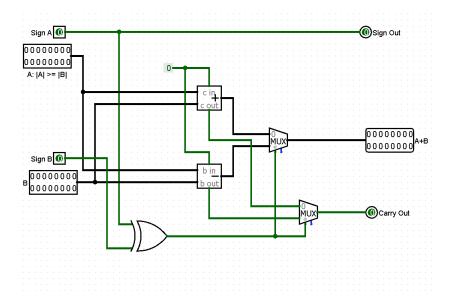


Figure 12: Diagram of special adder.

\bullet Adder

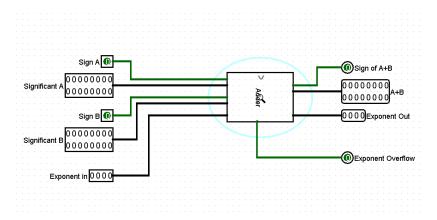


Figure 13: Diagram of adder.

3. Final circuit

• Normalizer

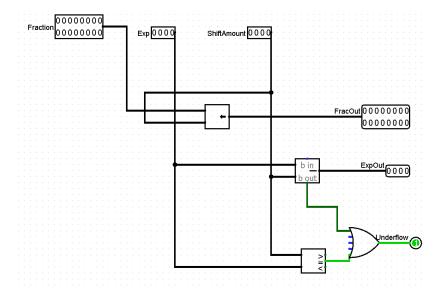


Figure 14: Diagram of normalizer.

• Shift amount

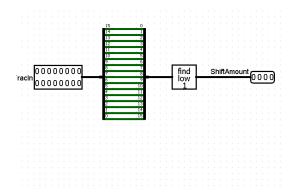


Figure 15: Diagram of shift amount.

• Truncate

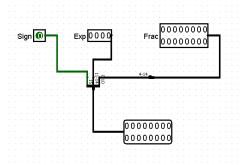


Figure 16: Diagram of truncate.

• Final adder

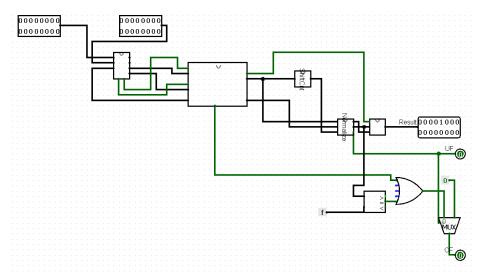


Figure 17: Diagram of final adder.

6 Used ICs

Types and counts of gates used for designing this circuit:

Section Sect	Oper	Sp	Signi	2s	Mag	IE	Com	Pa	fp	Bit	Fr	Nor	Shif	Tru	Fi	IC
Add Add						ı	1				I			ı		
Add of er Property Property Add of er Property Property Property Property Property Property	duon				ı	ı						l				
Decomposition Color Colo			_	1			Ι'	-	ı			C1				
COUA 1				""					01	ļ ·						
QUA D 2- XOR 1 1 1 1 1 74 86 QUA D 2- OR NOT 1 1 1 1 1 1 1 4 74 32 NOT NOT 1 1 1 1 1 1 1 74 64 bit Adde r 4 bit Adde r 1 1 1 1 1 1 1 74 83 16 bit Subtr actor 4 bit Subtr actor 4 bit Subtr actor 4 4 bit Subtr actor 4 2 1 1 1 1 1 1 1 2 1 1 2 1 1 2 1 1 2 1 1 2 1 1 7 7 83 2 4 2 2 2 2 4 1 7 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1					ı	Ι΄.					Ι΄.					
D 2- XOR QUA D 2- OR NOT D 1 D 1 D 1 D 1 D 1 D 1 D 1 D 1 D 1 D 1	QUA	-	1	1	-											74
XOR																86
QUA D2- OR NOT 1																
D 2- OR								1	2			1			4	74
OR I	D 2-															32
16 1																
16 bit Adde r 1 1 1 2 1 2 1 74 83 4 bit Adde r 1 1 1 1 1 1 74 83 16 bit Subtr actor 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 1 1 1 2 1 1 1 2 1 1 1 2 74 15 1	NOT		1		1	1	1		1						1	74
bit Adde r 4 bit Adde r 1 1 1 74 83 83 83 83 83 83 83 83 83 83 83 83 83																04
Adde r		1	1	1		2									2	
r Image: contraction of the bit of th																
4 bit Adde r 1 1 1 1 74 83 16 bit Subtr actor 2 1 1 1 1 1 2 1 4 bit Subtr actor 2 4 1 1 1 1 2 2 2 4 1 1 2 2 4 15 1 1 2 74 15 1 1 2 74 15 1 1 2 74 15 15 3 2 7 74 15 3 15 1 <	Adde															
Adde r 16																
r I						1									1	
16 bit Subtr actor 4 bit Subtr actor Quad 16:1 MUX Dual 4:1 MUX A:1 MUX 1 1 2 4 5 6 74 15 00 MUX MUX 1 1 2 4 6 74 15 00 MUX MUX 1 1 1 2 7 1 74 15 70 MUX Splitt er 2 4 5 7 1 7 1 7 1 7 1 7 1 7 1 7 1 7 1 7 1 7	Adde															83
bit Subtr actor 4 bit Subtr actor 1 1 1 1 1 2 4 bit Subtr actor 2 4 2 2 2 6 74 15 00 00 00 00 00 00 00 00 00 00 00 00 00																
Subtr actor 4 bit Subtr actor 1 1 1 1 1 1 2 4 bit Subtr actor 1 1 1 1 1 2 4 bit Subtr actor 2 4 1 1 1 1 2 2 4 1 2 2 2 4 1 2 7 4 1 1 1 1 1 1 1 1 2 7 4 1 1 1 2 7 4 1 1 1 2 7 4 1 1 1 2 7 4 1		1														
actor 4 bit Subtr actor 1 1 1 1 1 1 1 1 2 3 2 4 1																
4 bit Subtr actor 1 1 1 1 1 2 2 3 2 4 1																
Subtr actor 2 4 2 2 2 6 74 15 00 00 00 00 00 00 00 00 00 00 00 00 00																
actor Image: square of the content of the							1		1			1			2	
Quad 16:1 MUX 1 3 2 4 2 2 2 6 74 15 00 Dual 4:1 MUX 1 1 1 1 2 1 2 74 15 3 MUX 1 1 1 1 1 1 74 15 3 MUX 1 1 1 2 4 2 4 16 to 1 bit Splitt er 1 2 4 2 4 16 to 3 bit Splitt er 2 1 2 1 2 4 to 1 2 1 2 2																
16:1 MUX			_		_	_		_	_		_				_	
MUX Image: control or cont		1	3		2	4			2		2				6	
Dual 4:1 MUX 1 1 1 2 1 1 15 3 3 Quad 2:1 MUX 1 1 1 1 1 1 74 15 7 MUX 1 1 1 2 4 2 4 16 to 1 bit Splitt er 1 2 4 2 4 2 4 16 to 3 bit Splitt er 2 1 2 1 2 2 4 2 4 4 2 4 2 4																
4:1 MUX							1	1	2		1				2	
MUX 3 Quad 2:1 MUX 1 1 1 1 74 15 7 16 to 1 bit Splitt er 2 4 2 4 2 4 2 4 1 2 4 2 4 2 4 2 4 4 2 4 2 4							1	1	2		1				2	
Quad 2:1 MUX 1 1 1 1 1 1 15 7 16 to 1 bit Splitt er 2 4 2 4 2 4 16 to 3 bit Splitt er 2 1 2 2 3 4 to 1 2 2 2 3	ı															
2:1 MUX		1	1		1	1									1	
MUX		1	1		1	1									1	
16 to 1 bit Splitt er 2 4 2 4 2 2 4 2 2 4 4 1 2 2 4 4 4 4 4 4																
1 bit Splitt er				1				2	4				2		4	,
Splitt er 2 1 2 1 2 2 1 2 4 to 2 1 2 2 2 2 2 3 bit Splitt er 2 2 2 3 bit Splitt er 3 3 3 3 4 4 to 1 2 2 2 3 4 <td></td> <td></td> <td></td> <td> *</td> <td></td> <td></td> <td></td> <td>_</td> <td> -</td> <td></td> <td></td> <td></td> <td>-</td> <td></td> <td> -</td> <td></td>				*				_	-				-		-	
er																
16 to 3 bit Splitt er 2 1 2 2 4 4 to 2 2																
3 bit Splitt er									2	1					2	
Splitt er 1 2 2									-	-					-	
er																
4 to 1 2 2																
								1	2						2	
	1 bit															

Splitt er											
3 to 1 bit Splitt er									1	1	
16 bit comp arato r	1	1	1							1	
4 bit comp arato r							1			2	74 85
16 bit Shift er			1		2	2	1			4	
1 to 4 bit exte nder			2							2	
16 bit Bit finde r								1		1	
4 bit Nega tor				1	1					1	
										To tal = 40	

7 About the simulator

The simulator used for making this design is Logisim 2.7.1. It is a free software for simulating digital logic circuits. It was downloaded from https://sourceforge.net/projects/circuit/.

8 Discussion

Here we checked overflow and underflow of the output. To get a normalized form, we truncated the output. To make the build up easier, We broke the problem into smaller problems and built circuit for that. And finally we combined everything to make the final circuit.