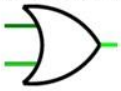
CSE306

Computer Architecture Sessional



Assignment 2: Floating Point Adder

Section – A2

Group - 03

Group members student ID:

1. 1605051
2. 1705040
3. 1705042
4. 1705048
5. 1705054

Submission Date :31 May,2021

Introduction:

A floating-point adder is a combinational-logic circuit that adds two signed real numbers. According to IEEE 754 floating-point standard, if a floating-point is represented in 32 bits, then the 1st bit indicates the sign, following 8 bits indicates the exponent and the remaining 23 bits indicates the fraction. If 64 bits structure is used then the 1st bit indicates the sign, next 11 bits indicates the exponent and the remaining 52 bits indicates the fraction. The representation for all floating-point numbers uses the form:

$$(-1)^{\text{Sign}} * (1 + \text{Fraction}) * 2^{(\text{Exponent-bias})}$$

Negative exponents pose a challenge to simplified sorting. Thus, the bias is used.

In our assignment, we designed a floating-point adder which takes two floating-point number in IEEE 754 standard format as input and gives us the summation of the two numbers. We took in account the cases where all 0 bits in exponent and fraction is used for floating point representation of 0 and the pattern of all 1 in the exponent is used for indicating values and situations outside the scope of normal floating-point numbers. We also detected if any overflow or underflow occurred during the process. The status of the overflow and underflow bits were set or cleared as follows:

1. When normalizing the sum, if the exponent becomes 0 or negative during left shift then underflow bit is set to 1. But if the answer is 0 then we deliberately made the output exponent 0. In such case underflow bit is set to 0. In all other cases, the underflow bit is set to 0.
2. When normalizing the sum, if the exponent becomes 1111 or greater than 1111 then we set the overflow bit to 1. Otherwise, it is set to 0.

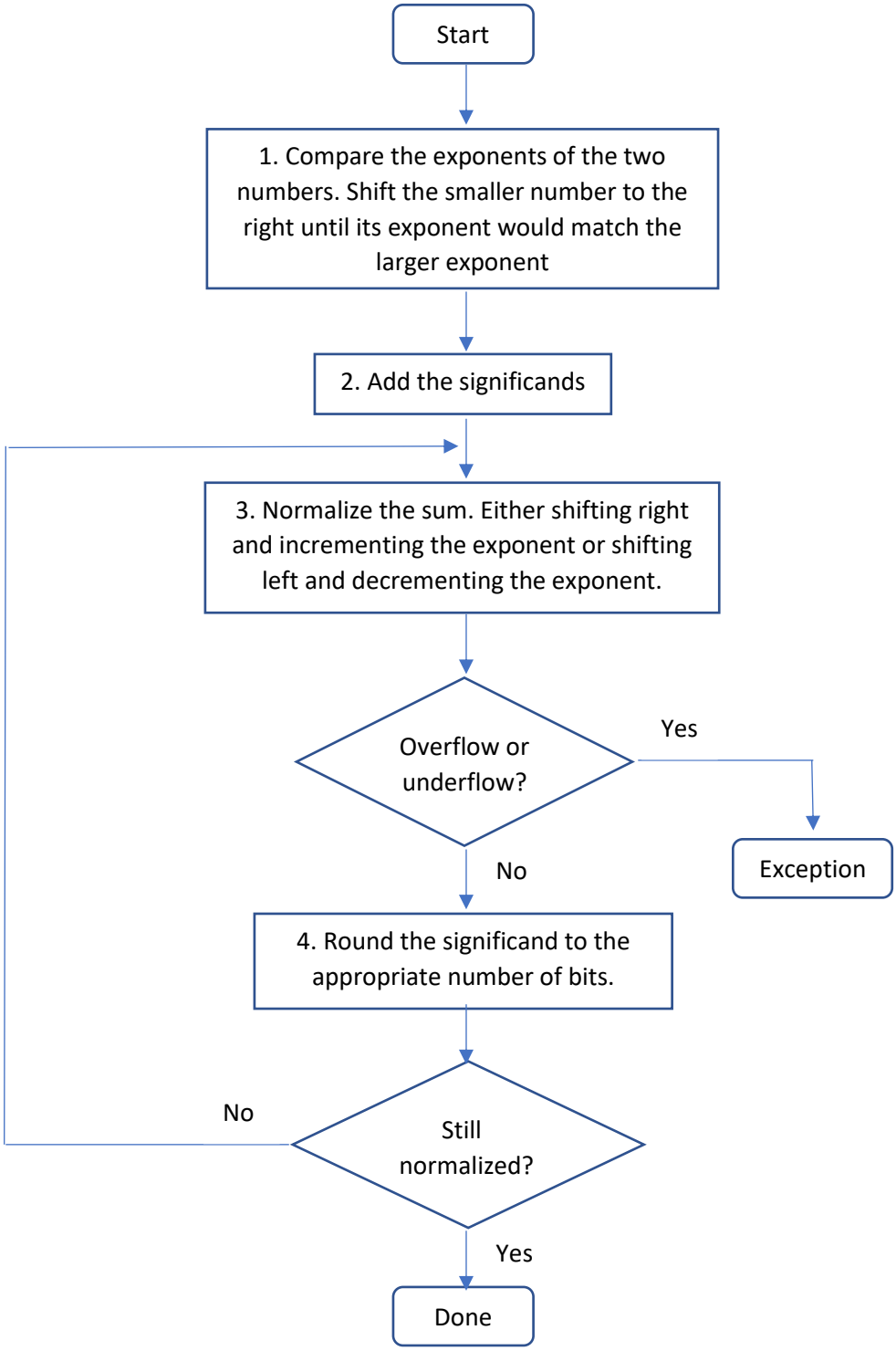
Problem specification:

In this assignment, we are required to design a floating-point adder circuit which takes two floating point s as inputs and provides their sum, another floating point as output. Each floating-point will be 16 bits long with following representation:

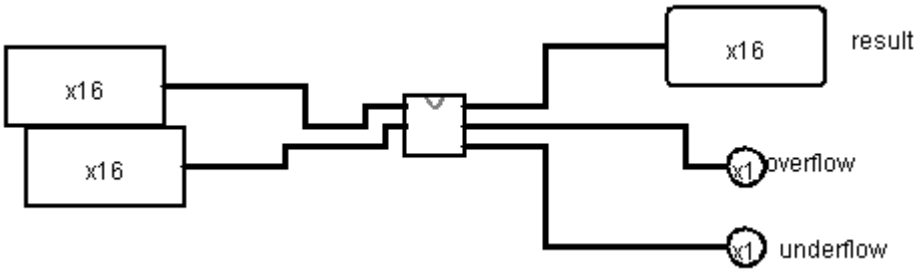
Sign 1 bit	Exponent 4 bits	Fraction 11 bits

As we have 4-bit long exponent, our bias value is 7.

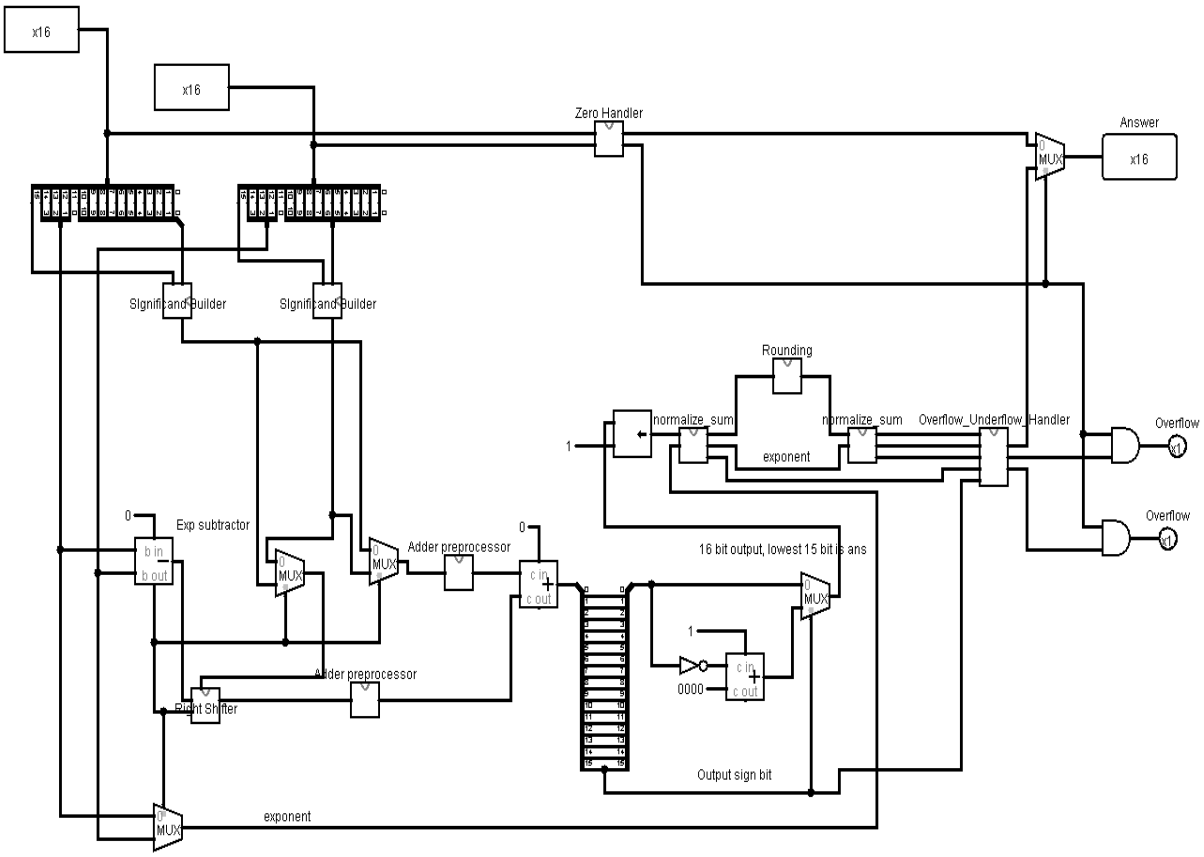
Flow chart of the addition algorithm:



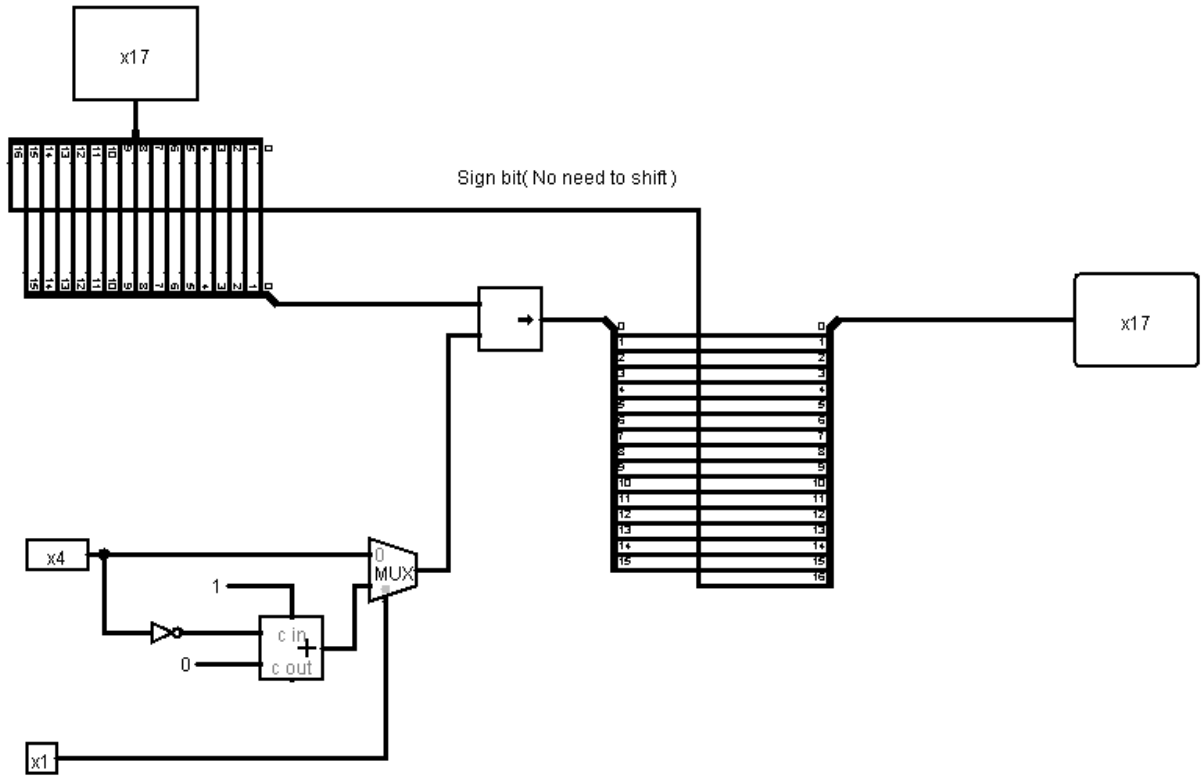
High level block diagram:



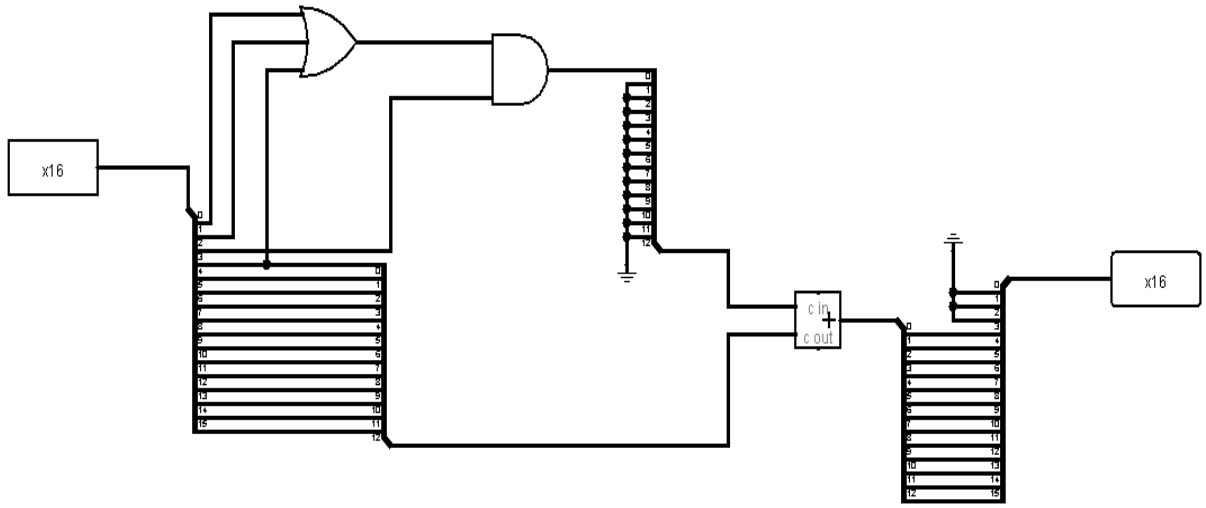
Floating-point adder circuit:



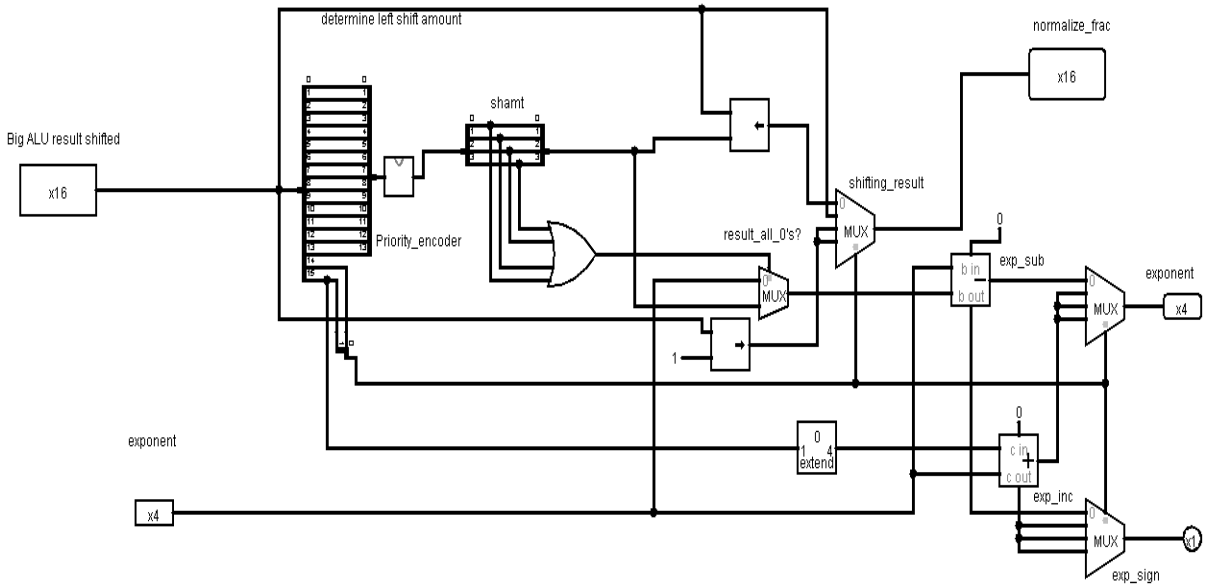
Right shifter circuit:



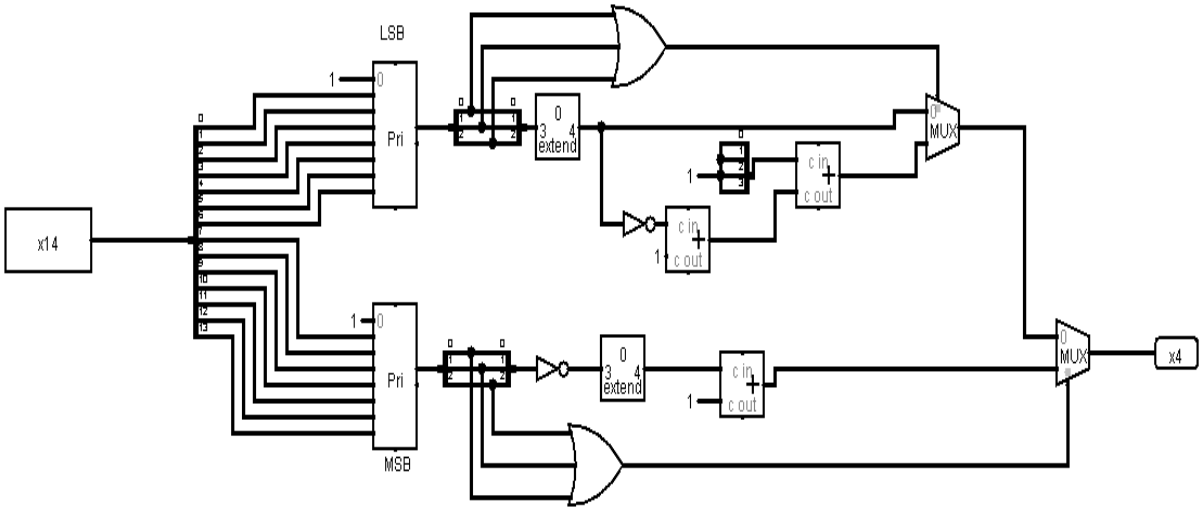
Rounding circuit:



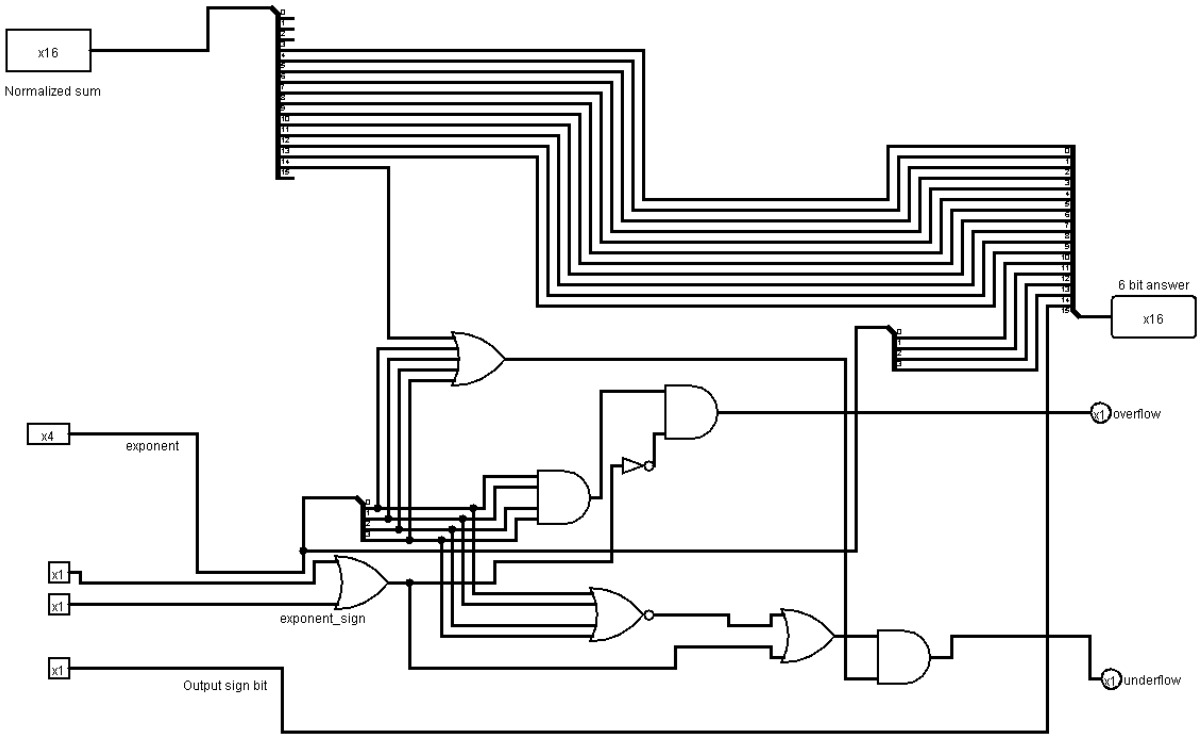
Sum normalizer circuit:



Bit finder:



Overflow, underflow detector and answer representor:



Required ICs and components:

IC number	IC type	Count
7432	Quad 2 input OR gate	6
7408	Quad 2 input AND gate	3
7404	Hex NOT gate	11
74153	4*1 MUX	29
74157	2*1 MUX	29
74148	8*3 priority encoder	4
7483	4-bit adder	30
4072	Dual 4 input OR gate	10

Component name	Count
16-bit right shifter	3
16-bit left shifter	3
Bit extender	6

Simulator used: Logisim-generic: 2.7.1

Discussion:

In this assignment, we implemented a floating-point adder for 16-bit floating-point number using AND gate, OR gate, NOT gate, Adder, Subtractor, Multiplexor and Priority encoder. While simulating the circuit diagram in logisim simulator, we set the input number of all gates conveniently for implementation purpose. But we counted the IC number according to the IC's available in real world and their input number. We also had to add 5 extra bits (3 in front and 2 in back) with the fraction part of the input to simplify the addition, normalization and rounding part of the circuit. Furthermore, we used Adder circuit instead of ALU as we only needed the addition functionality.