**BANGLADESH UNIVERSITY OF**

**ENGINEERING AND TECHNOLOGY**

**Course No**: CSE306

**Course Name**: Computer Architecture Sessional

**Name of the Experiment**:

Assignment on Floating Point Adder

**Level/Term**: 3-1

**Section**: B2

**Group No**: 02

**Student ID**: 1705092

1705111

1705116

1705118

1705121

Date of Submission: 31-05-2021

**Introduction**

A floating point number has three parts, sign, exponent and fraction. A Floating Point Adder is a digital circuit used to add two floating point numbers.

The IEEE-754 standard defines a floating point number as,

For a 16 bit floating point number, the representation is,

|  |  |  |
| --- | --- | --- |
| *Sign*  *1 bit* | *Exponent*  *4 bit* | *Fraction*  *11 bit* |

And,

**Problem Specification:**

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

|  |  |  |
| --- | --- | --- |
| *Sign*  *1 bit* | *Exponent*  *4 bit* | *Fraction*  *11 bit* |

**Flow Chart of the Addition Algorithm**

Start

Compare the exponents of the two numbers. Shift the smaller number to the right until its exponent would match the larger exponent.

Add the significands

Normalize the sum, either shifting right and incrementing the exponent or shifting left and decrementing the exponent.

Yes

Set Flag

Overflow or Underflow?

Round the significand to the appropriate number of bits and renormalize if needed.

Output

Figure: Flowchart of Floating Point Addition algorithm

**High-Level Block Diagram of Floating Point Adder**

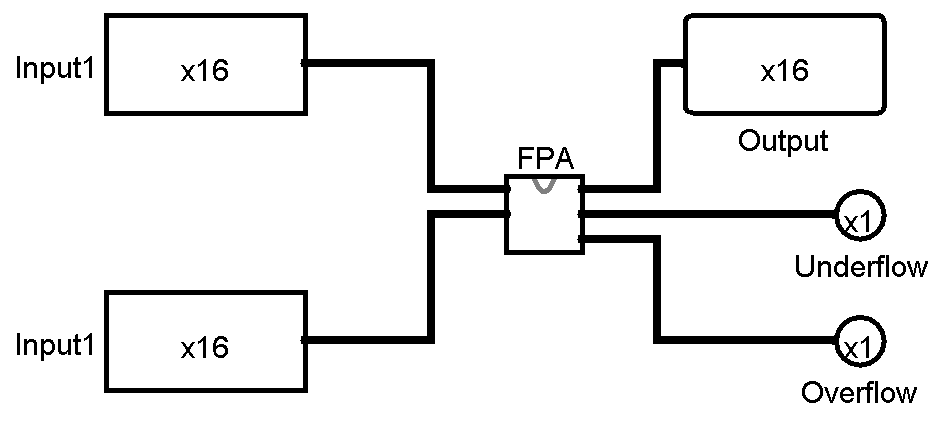


Figure: High-Level Block Diagram of Floating Point Adder

**Circuit Diagram**

Floating Point Adder:

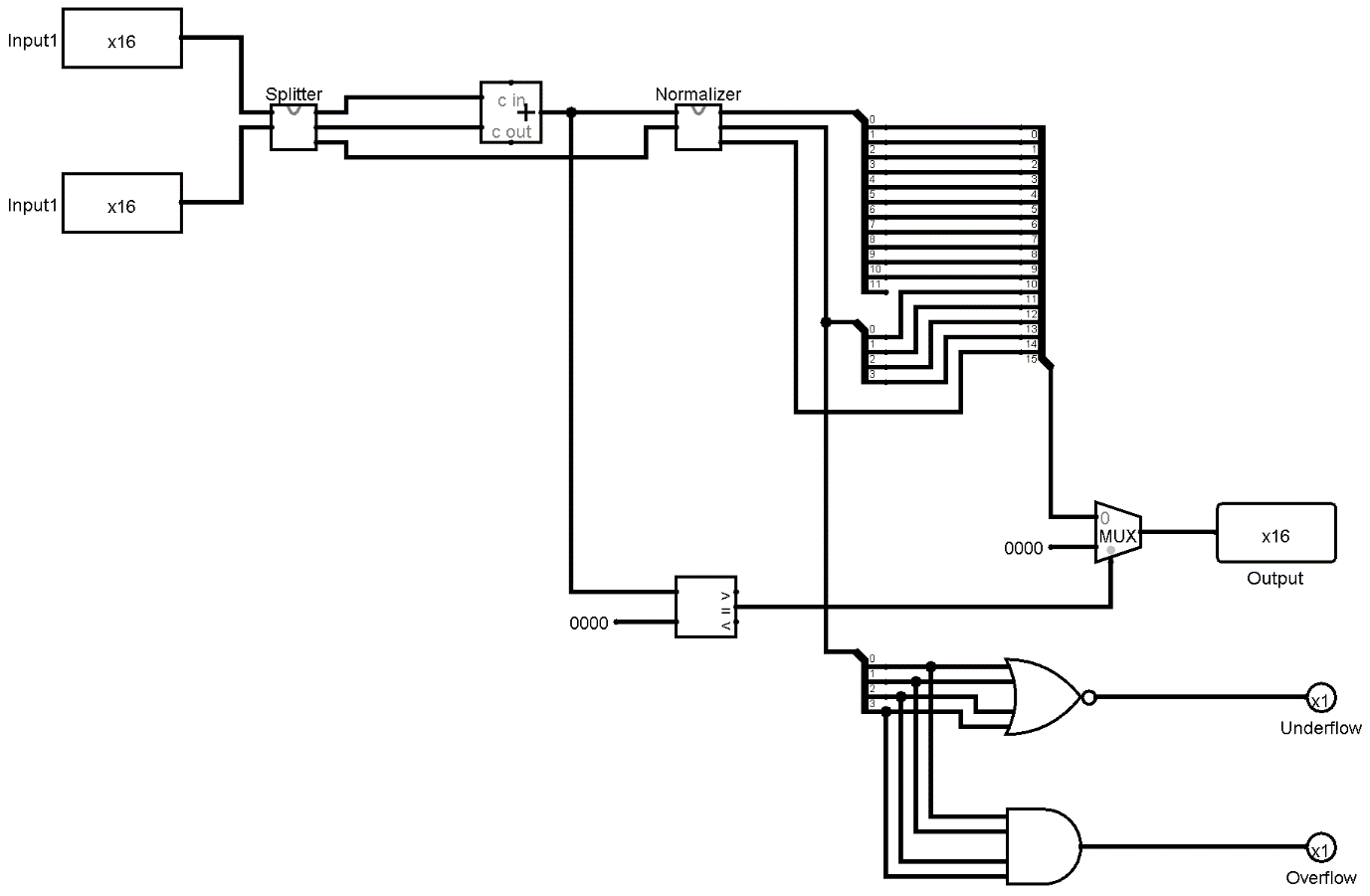
****

Figure: Circuit diagram of Floating Point Adder

Splitter:

Splitter splits the input numbers into significand and exponent parts.

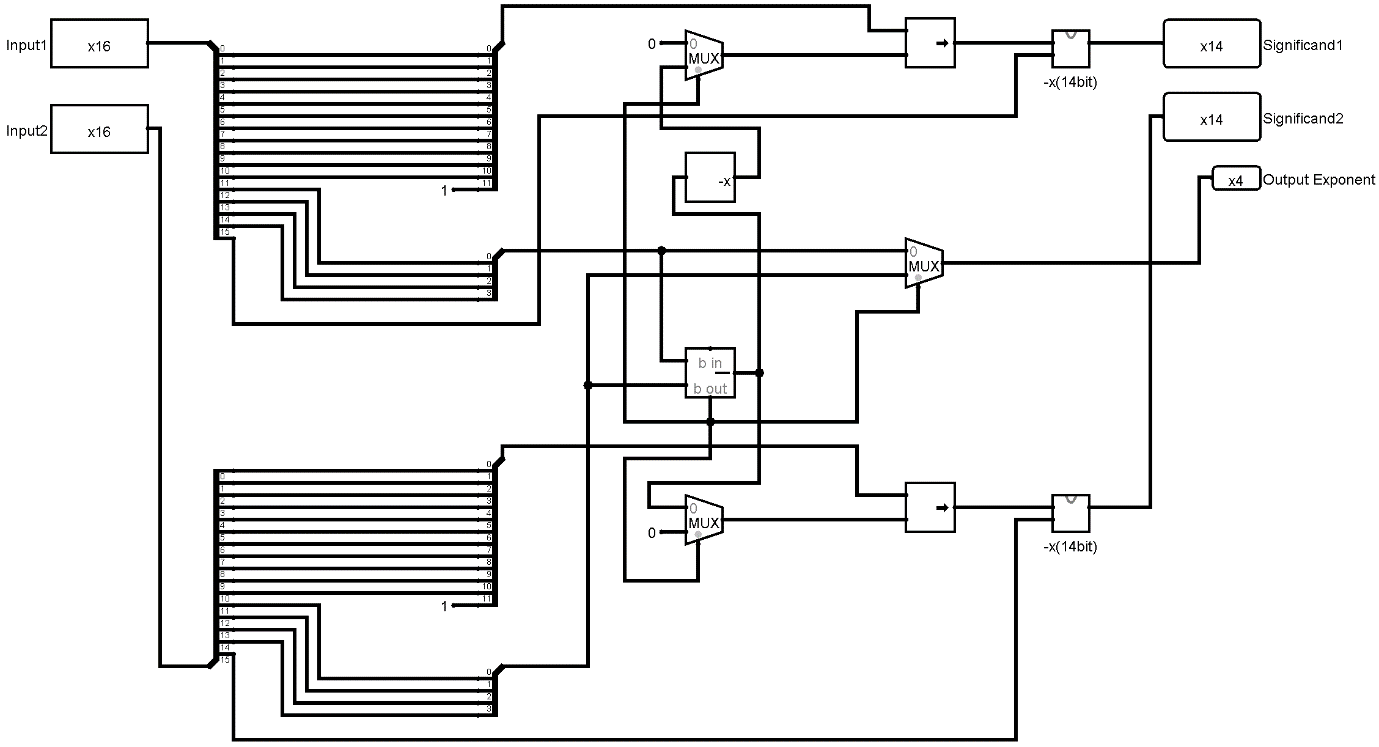


Figure: Circuit diagram of Splitter

Negator:

Negator negates a number checking its sign bit.

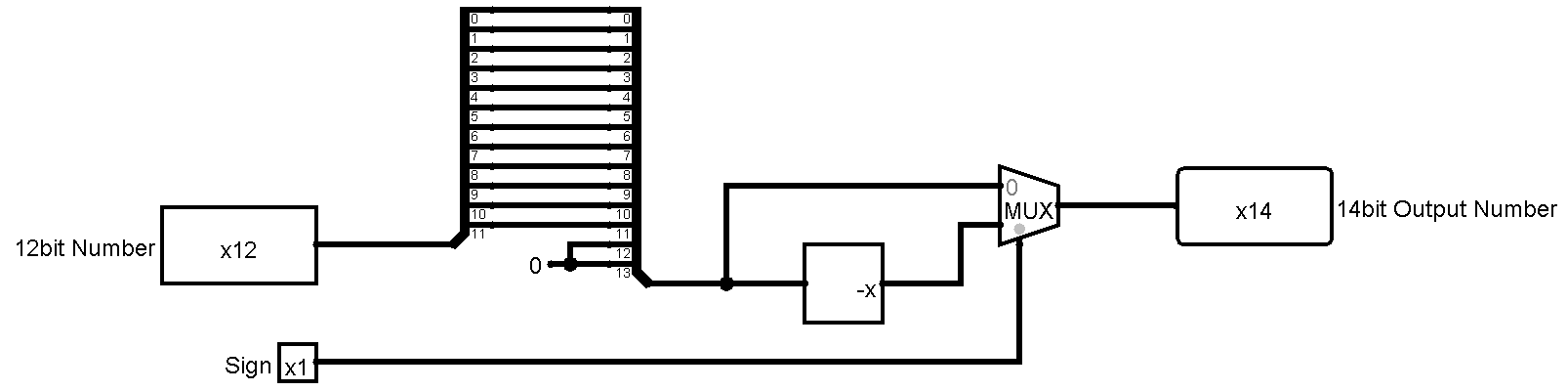


Figure: Circuit diagram of Negator

Normalizer:

Normalizes a number.

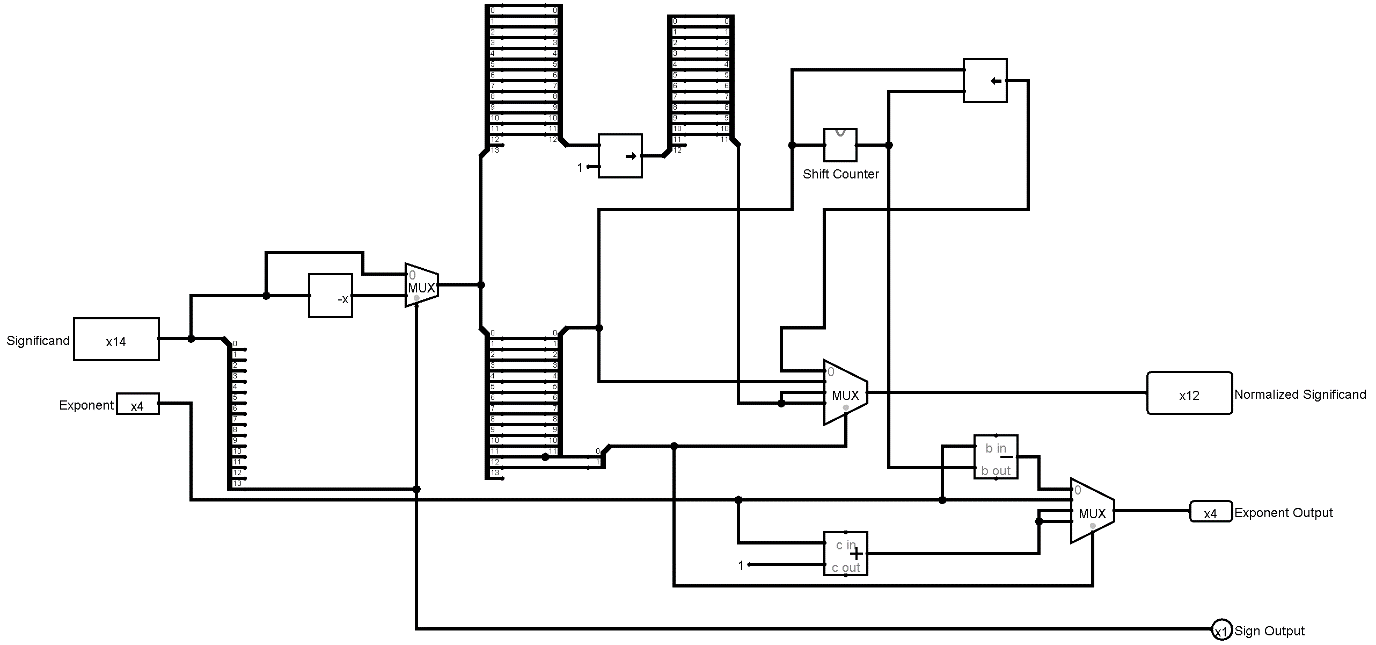


Figure: Circuit diagram of Normalizer

Shift Counter:

Finds the first high bit from most significant bit to determine the number of times the number needs to be shifted.

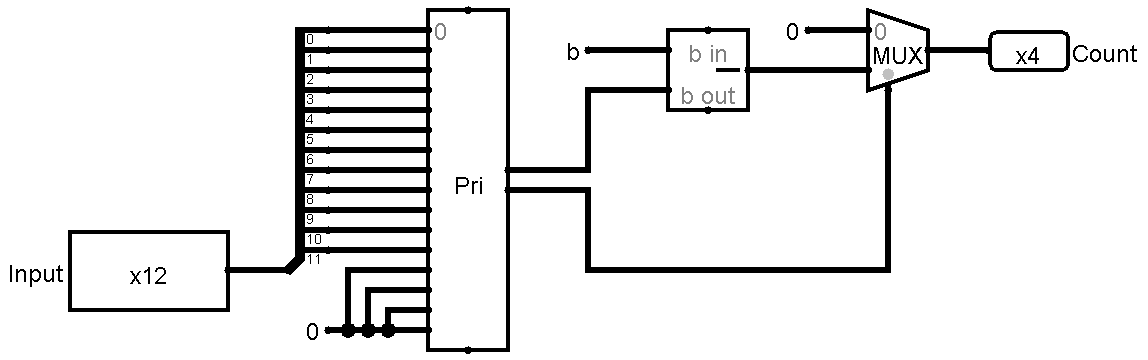


Figure: Circuit diagram of Shifter

**ICs used with count**

|  |  |  |
| --- | --- | --- |
| **Components** | **IC Number** | **Count of ICs** |
| 16 bit Right Shifter |  | 3 |
| 16 bit Left Shifter |  | 1 |
| 16 bit Comparator |  | 1 |
| Negator |  | 3 |
| 4 bit Subtractor |  | 3 |
| 4 bit Adder | IC7483 | 5 |
| 16 to 4 Priority Encoder |  | 1 |
| 2x1 MUX | 74157 | 7 |
| 4x2 MUX |  | 2 |
| AND | IC74LS08 | 1 |
| NOR | IC74LS02 | 1 |
|  |  |  |

**Simulator used**

The simulation software used in this Floating Point Adder experiment is ***logisim-win-2.7.1.exe.***

**Discussion**

The input numbers are considered to be limited in Normalized form.

For rounding, truncation is done as per the instruction of the respected course teacher.

In the case of a zero result, i.e., if two equal numbers with opposite signs are given in inputs, the output bits are all made zero as per the zero representation in the binary floating point numbers.

As truncation is done, renormalizing was not necessary and is not implemented.