Expo\_sign\_4

Expo\_numo\_4

Frac\_numo\_4

isZero\_4

Expo\_numo\_3

Expo\_sign\_3

isZero\_3

isZero\_2

Unum1\_shift

Unum2\_shift

isZero\_1

temp2

Temp1

NaN\_4

isInf\_4

NaN\_3

isInf\_3

Frac\_numo\_3

NaN\_2

isInf\_2

Expo\_num1

Expo\_num2

temp1\_2

temp2\_2

isInf\_1

num2

num1

check whether the input number is special situations: zero and Inf. If the number is nagative, change it from 2's complement to original. Calculate the number of leading zero/one of the number (by using LZC module).

Left shift the temp so that the exponent bits, sign bit and fraction bits will in the certain positions. Calculate the value of regime bits (in expo\_num). If one of the number is Inf, also make isZero 0 (which means one of the number is zero) because the manipulation of zero and Inf is very similar.

Multiple two fractions, add two exponent values.

Normalize the result of fraction multiplication.

Internal multipliers of FPGA are used to do the multiplication of fraction bits.

LZC(leading zero counter) module is designed based on MODULAR DESIGN OF FAST LEADING ZEROS COUNTING CIRCUIT (http://iris.elf.stuba.sk/JEEEC/data/pdf/6\_115-05.pdf)

check overflow, underflow, rounding and negative number/

Right shift the exponent bits and fraction bits following the regime value (in order to adding regime bits on the left side)

round

Runumo\_6

NaN\_6

overflow

underflow

NaN\_5

Runumo\_5

NaN\_4

isInf\_4

isZero\_4

Frac\_numo\_4

Expo\_numo\_4

Expo\_sign\_4