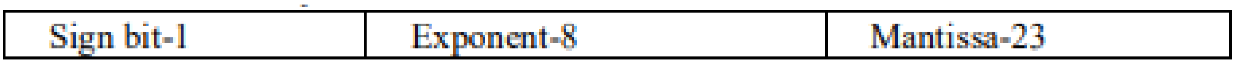
FLOATING POINT ADDITION

Abdelrahman Khaled Elsayed

The IEEE 754 Single Precision Binary Format is as shown below:-

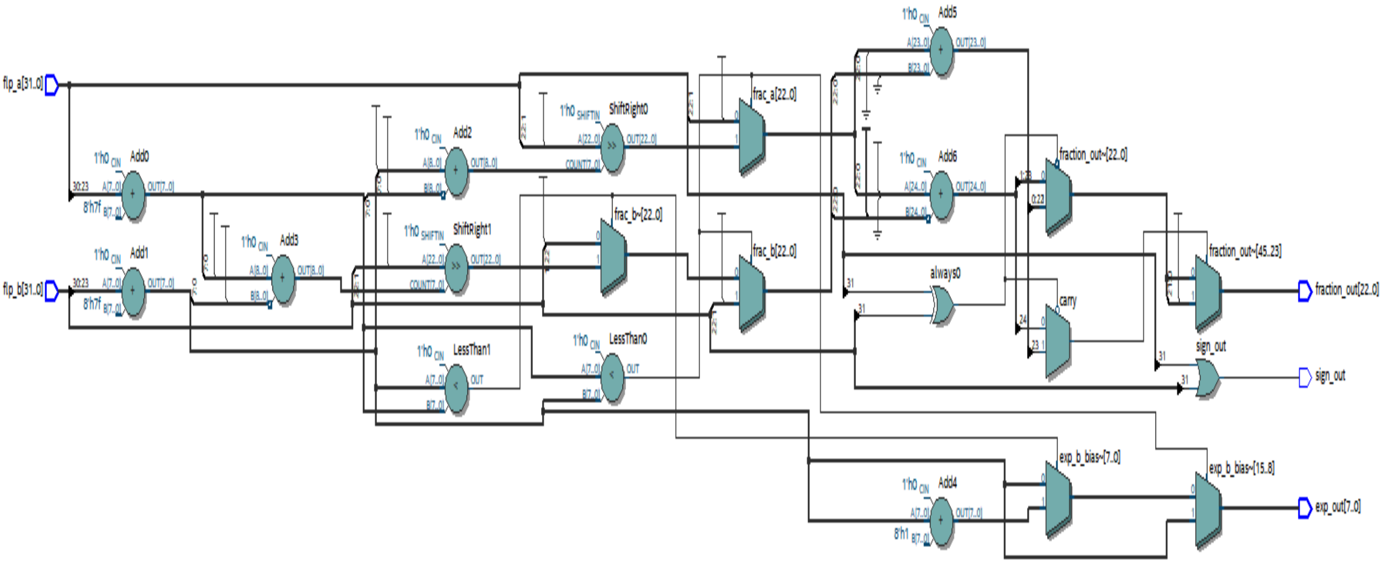


Sign bit-1	Exponent-8	Mantissa-23

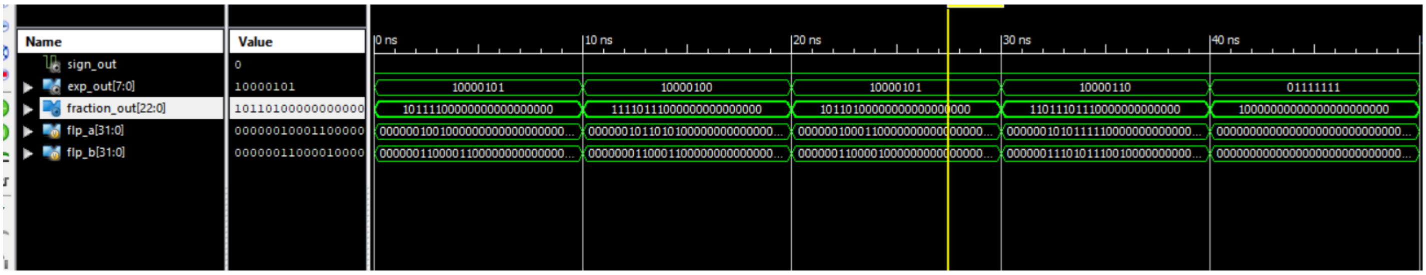
To perform floating-point addition using 32-bit binary representation, you typically need to follow the IEEE 754 standard for floating-point arithmetic. This standard defines the format for representing floating-point numbers and specifies the rules for arithmetic operations.

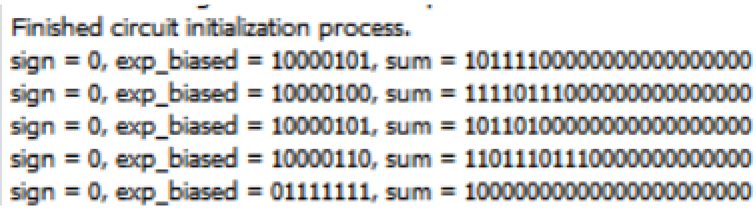
1. Check the signs of the two numbers:
    - If both numbers have the same sign, proceed to step 2.
    - If the signs are different, subtract the absolute value of the negative number from the absolute value of the positive number and assign the appropriate sign based on the result. Skip to step 4.
2. Align the exponents:
    - Compare the exponents of the two numbers.
    - Shift the number with the smaller exponent to the right by the difference in exponents, effectively aligning the decimal points.
3. Add the mantissas:
    - Add the aligned mantissas together.
    - If the sum exceeds the number of bits available for the mantissa (overflow), adjust the result and increment the exponent accordingly.
4. Normalize the result:
    - Adjust the result so that there is only one non-zero digit before the binary point.
    - If necessary, shift the result to the right or left and adjust the exponent accordingly.
5. Round the result:
    - Depending on the rounding mode specified, round the result to the nearest representable value.
6. Check for overflow or underflow:
    - If the result exceeds the range that can be represented by the 32-bit floating-point format, handle it as an overflow or underflow condition.
7. Assign the sign:
    - Apply the appropriate sign to the final result based on the signs determined in step 1.

Netlist:



Simulation:





```
Finished circuit initialization process.
sign = 0, exp_biased = 10000101, sum = 10111100000000000000000
sign = 0, exp_biased = 10000100, sum = 11110111000000000000000
sign = 0, exp_biased = 10000101, sum = 10110100000000000000000
sign = 0, exp_biased = 10000110, sum = 11011101110000000000000
sign = 0, exp_biased = 01111111, sum = 10000000000000000000000
```