Задача

С использованием открытой библиотеки, реализующей floating point арифметику, на SystemVerilog написать модуль для сложения ряда чисел с минимальной утилизацией аппаратных ресурсов.

Описание директорий

- hdl (файлы описания разработанного модуля и модуля подключаемой библиотеки);
- testbench;
- simulation (папка для файлов симуляции в ModelSim).

Ход решения

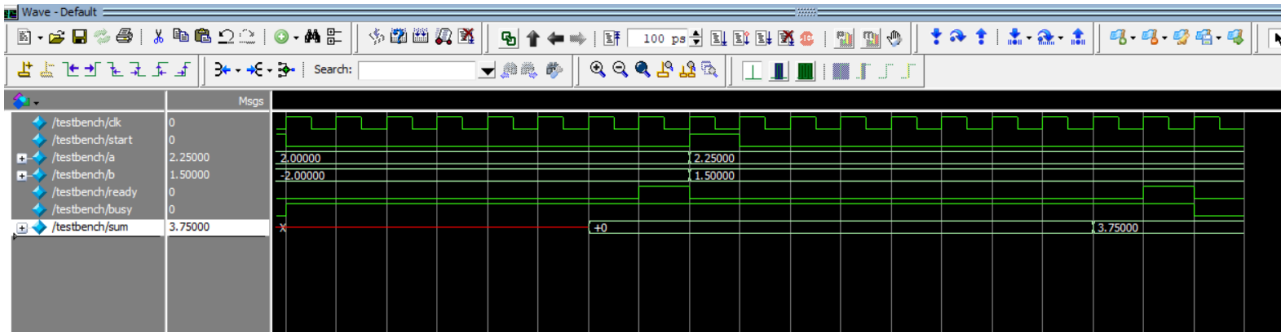
Выбор библиотеки для сложения

Я искала решения на github по ключевым словам.

Попробовала работать с двумя:

- GitHub - kyle-sit/Floating-Point-Modules: Floating point adder/subtractor and multiplier in SystemVerilog
  (с clk)
- 32-Verilog-Mini-Projects/Floating Point IEEE 754 Addition Subtraction at main · sudhamshu091/32-Verilog-Mini-Projects · GitHub
  (без)

В первой входными сигналами являются clk, start (сигнал начала операции, считывается только когда модуль не занят вычислениями, он не конвейеризован), op (номер операции: 0 для сложения, 1 для вычитания), A и B (слагаемые), выходными – ready, busy, Y (сумма). Вычисление суммы занимает до 10 тактов (меньше, если среди операндов есть nan или inf).



Вторая сделана без clk, использована только комбинационная логика. На вход подаются слагаемые a и b, сигнал выбора операции add_sub_signal (тоже 0 для суммы, 1 для вычитания), на выходе получаются res (сумма) и exception (этот флаг устанавливается при экспоненте 255). На сложных примерах у этого модуля появляется «погрешность», из-за чего на диаграмме ModelSim всё выглядит правильно, но testbench валится:

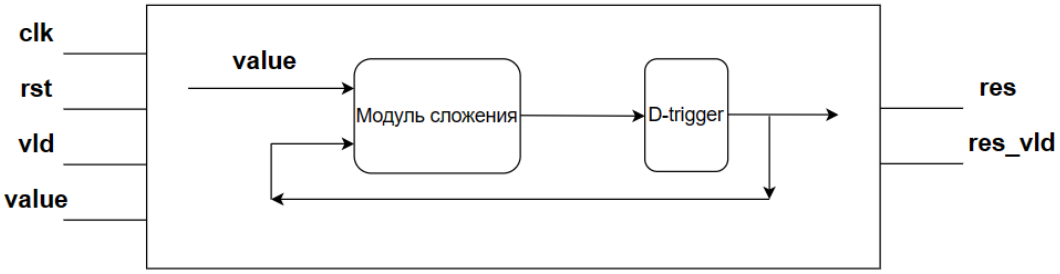
```
# FAIL: b(00110100000000000000000000000000) h(34000000) GOT, b(00000000000000000000000000000000) h(00000000) EXPECTED
# FAIL: b(00111111100000000000000000000000) h(3f800000) GOT, b(00000000000000000000000000000000) h(00000000) EXPECTED
# FAIL: b(01001011100000000000000000001001) h(4b800009) GOT, b(01000010000110000010100011110110) h(421828f6) EXPECTED
# FAIL: b(10111110101011100001010001111100) h(beae147c) GOT, b(10111110101011100001010001111011) h(beae147b) EXPECTED
# FAIL: b(01000001011011111101011100001100) h(416fd70c) GOT, b(01000001011011111101011100001010) h(416fd70a) EXPECTED
# FAIL: b(00111111000000000000000000000000) h(3f000000) GOT, b(00000000000000000000000000000000) h(00000000) EXPECTED
```

Сложение ряда чисел

Быстрее всего было бы использовать кучу экземпляров модуля-сумматора, разбив все операнды на пары и складывая получающиеся результаты (после чего объединяются в пары уже эти частичные суммы и тоже складываются, ит.д.), но для минимальной утилизации аппаратных ресурсов нужно иметь как можно меньше таких экземпляров.

В таком случае, нужен модуль, который потактово принимает новое слагаемое и суммирует его с уже образовавшейся суммой предыдущих. С использованием второй (комбинационной) библиотеки схема модуля будет выглядеть примерно так:

В итоге модуль реализован в виде



Для проверки его работы написан testbench.