Project 4: IPs for Floating Point Operations

Tasks:

1. Design a floating point adder/subtractor
    - Module name: adder_fp
    a. Input: clk, start, op (0 for addition, 1 for subtraction)
        [31:0] A, B
    - Output: ready, busy,
        [31:0] Y
2. Design a floating point multiplier
    - Module name: multiplier_fp
    - Input: clk, start
        [31:0] A, B
    - Output: ready, busy,
        [31:0] Y
3. Write a test bench for each module
    - Give at least 4 test cases for each module
    - Submit a screenshot of the console output
        - Display results using hex
4. Synthesize each module separately
    - Use **ZYNQ-7 ZC702 Evaluation Board**
    - Submit a screenshot of the utilization table of post-implementation
    - Make a table to sum the utilization percentage of both modules together
        - Make sure the numbers aren't greater than 100%

Note:

- This is a group project with **two people per group.**
- Use 1ns timescale; **10 ns clock cycle** (i.e. 5ns high, 5ns low)
- Start should only be high for one clock cycle; busy goes high right after start goes low; ready goes high for one clock cycle while giving result; busy and ready signal go low together at the clock cycle after giving result.
- When busy is high, ignore start.
- Your module will be tested against edge cases (i.e. infinite, 0, NaN)
    - Infinite has positive and negative, NaN does not
    - With precedence
        i. Any input with NaN should give NaN output
        ii. Infinity multiply by 0 gives NaN
        iii. Positive infinity minus positive infinity (or similar combinations) gives NaN
        iv. Any other input with infinite should give infinite as output with corresponding sign
- Objective is to optimize your speed in terms of clock cycle, use whatever method you can think of.

- Keep in mind project 4, 5 are going **into** project 6, so if you're using 50% of utilization, for this project alone, something is wrong.

Submission:

- Please only submit one project per group, specify your partner's full name, PID in screenshot.pdf
- The module name, inputs and outputs are all provided in the write-up. Please do name your module accordingly.
- Your file name should be exactly same as the module name, with all the submodules within the same file.
- Each test bench should be named with _tb as extension (i.e. adder_tb.sv for adder.sv)
- Put all your screenshots into screenshot.pdf, (one picture per page if you use Word to create it). And include corresponding name immediately preceding the picture.
- Please make sure the codes you turn in are compilable.
- Please only submit the required files specified, all other files will be ignored.