哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY

实验报告

开课学期：2025 春季

课程名称：计算机组成原理（实验）

实验名称：浮点运算器设计

实验性质：设计型

实验学时：4 地点：T2506

学生班级：20233117

学生学号：2023311709

学生姓名：宁中昊

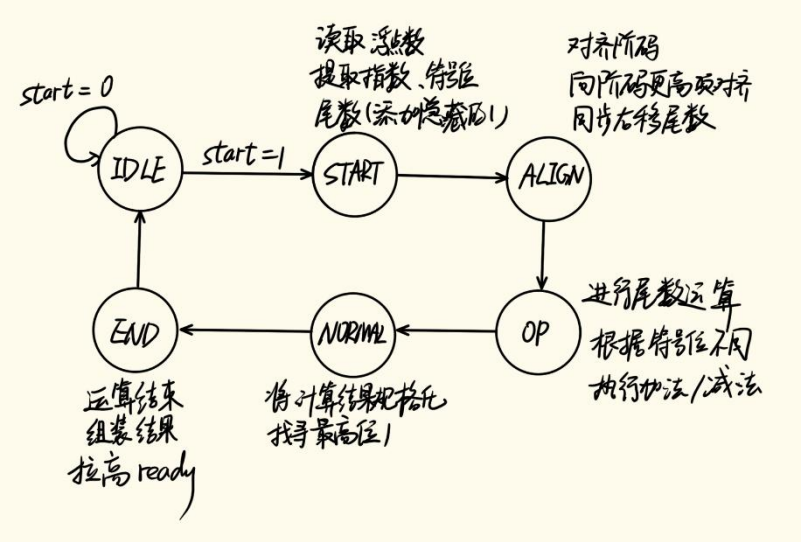
作业成绩：

实验与创新实践教育中心制

2025 年 4 月

1、设计与实现

要求：结合状态图、流程图、电路框图或时序图等工具，阐述你的浮点运算器是如何设计与实现的。必要时可结合代码说明，但不能大段粘贴代码。



状态图如上，通过空闲、开始、对阶、运算、规格化、结束六个状态实现了规格化浮点数运算器。

空闲状态（IDLE）等待 start 信号拉高，start=1 时进入开始状态，否则保持空闲状态；

在开始状态（START）中，读取浮点数，提取两个浮点数的符号位（减法时减数符号位取反）、指数、尾数，并为规格化数的尾数添加隐含的 1；

对阶状态（ALIGN）中将指数较小的向指数较大的数对齐，并同步右移对齐数的尾数；

运算状态进行尾数的运算，由于在提取符号时已经考虑到加减法，因此在这里符号位相同时尾数做加法，符号位相反时尾数做减法即可。最终的符号与尾数较大的一项相同；

规格化状态（NORMAL）中，如果尾数有溢出则需要向前进位，直接截取并将指数加一即可，没有溢出是需要找到最高位的 1，代码如后图。

最后是结束状态，组装结果并拉高 ready 信号。

```
// 组合逻辑部分，寻找高位 1
always @(*) begin
    high_one = 0;
    if      (man_sum[23]) high_one =  0;
    else if (man_sum[22]) high_one =  1;
    else if (man_sum[21]) high_one =  2;
    else if (man_sum[20]) high_one =  3;
    else if (man_sum[19]) high_one =  4;
    else if (man_sum[18]) high_one =  5;
    else if (man_sum[17]) high_one =  6;
    else if (man_sum[16]) high_one =  7;
    else if (man_sum[15]) high_one =  8;
    else if (man_sum[14]) high_one =  9;
    else if (man_sum[13]) high_one = 10;
    else if (man_sum[12]) high_one = 11;
    else if (man_sum[11]) high_one = 12;
    else if (man_sum[10]) high_one = 13;
    else if (man_sum[ 9]) high_one = 14;
    else if (man_sum[ 8]) high_one = 15;
    else if (man_sum[ 7]) high_one = 16;
    else if (man_sum[ 6]) high_one = 17;
    else if (man_sum[ 5]) high_one = 18;
    else if (man_sum[ 4]) high_one = 19;
    else if (man_sum[ 3]) high_one = 20;
    else if (man_sum[ 2]) high_one = 21;
    else if (man_sum[ 1]) high_one = 22;
    else if (man_sum[ 0]) high_one = 23;
    else                  high_one =  0;

    temp_sum = (man_sum << high_one);
    temp_man = temp_sum[23:0];
    temp_exp = exp_res - high_one;
end
```

（规格化结果，寻找最高位 1）

2、调试报告

要求：仿真截图及时序分析。列举 2 个测试用例进行分析。分析时，需**把浮点运算器内部的关键信号添加到仿真波形并进行相应分析**。*若实现了非规格化数据的运算，则还需再列举 4 个非规格化测试用例分析：

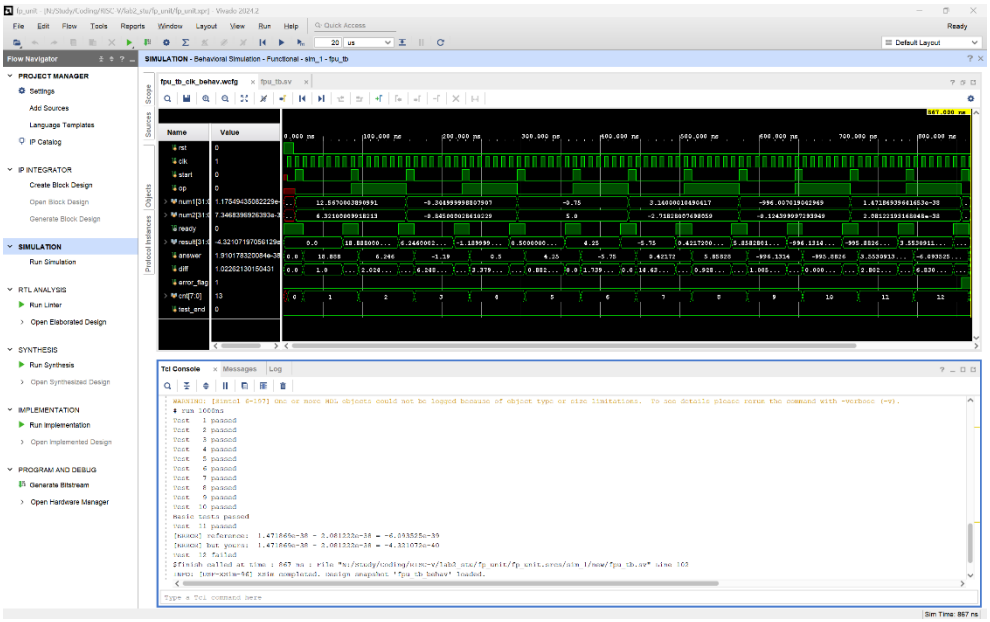
用例 1：输入的 A 和 B 是规格化数，但运算结果 C 是非规格化数

用例 2：A 和 B 其中一个是非规格化数，另一个是规格化数，C 是非规格化数

用例 3：输入的 A 和 B 都是非规格化数，运算结果 C 也是非规格化数

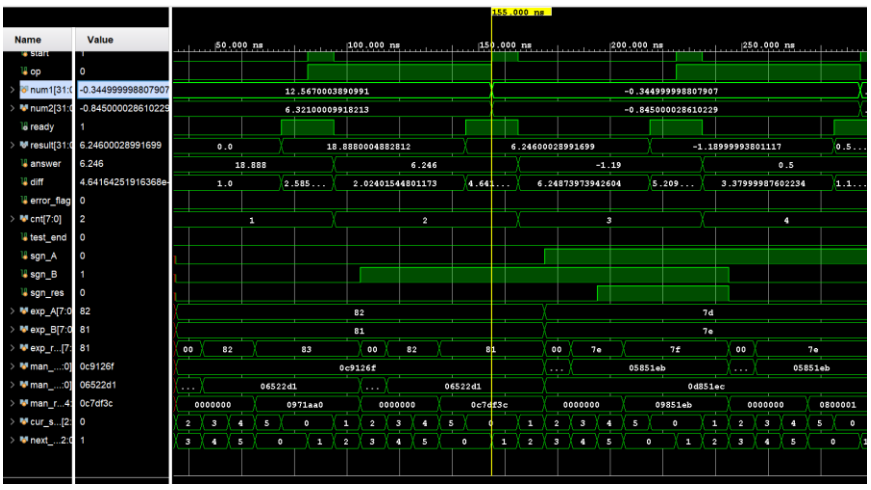
用例 4：输入的 A 和 B 都是非规格化数，但运算结果 C 是规格化数

浮点数运算器仿真截图：



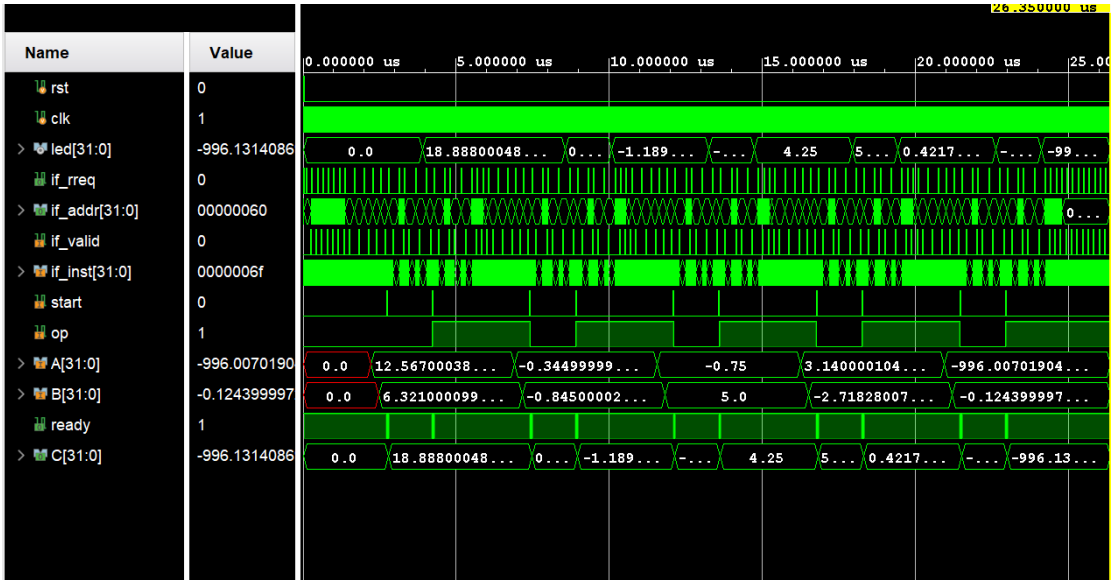
可以看到前 10 个测试用例均已通过。

下图是前两个测试用例的波形图：



可以看到三段式的状态正常更新，在对应的状态中顺利实现了符号、指数、尾数的提取，完成了指数的对齐与尾数的右移，也正常完成了尾数的加减法，以及规格化和组装。

以下是在 SoC 模板工程中的仿真截图：



3、思考与讨论

要求：实验 1 采用纯软件实现浮点运算，而本实验采用软硬件结合实现浮点运算。请尝试从灵活性、设计复杂度、运行效率等多个角度对比和分析两种实现方法的优缺点。

优缺点：

1. 灵活性

RISC-V 纯软件实现灵活性强，可以通过简单修改代码实现不同的浮点格式，易于扩展，代码复用性也更好。但是能够实现的功能受限，难以实现复杂功能。

Verilog 软硬件结合实现灵活性较差，依赖于过分笨重的开发环境，语法使用限制多；代码兼容性不好。但是可以针对特定场景设计专用的电路，增强模块化代码复用能力，开发环境中集成了更多工具，能够直接实现复杂功能。

2. 设计复杂度

RISC-V 纯软件实现开发门槛低，仅需掌握编程语言语法即可，无须额外的硬件知识；且修改快，调试高效，有报错信息更加详细全面的编译器，能快速定位错误所在。但是代码冗长，且性能较差。

Verilog 软硬件结合实现性能可控，能更好地进行优化。但是设计复杂，需要考虑不少问题如阻塞赋值、时钟周期等，且调试不方便，需要构建测试平台 testbench，需要直接描述时钟信号并枚举常见的输入组合，效率低且复杂。

3. 运行效率

RISC-V 纯软件实现无须消耗额外硬件资源，但需要串行计算。

Verilog 软硬件结合实现通过集成的工具使得运行效率更高，但是本身开发环境的笨重导致占用资源更多。