深 圳 大 学 实 验 报 告

课 程 名 称：计算机系统(3)

实验项目名称：新增指令实验

学　　院：计算机与软件学院

专　　业：计算机与软件学院所有专业

指 导 教 师：刘刚

报告人：　学号：　班级：

实 验 时 间：2025 年 12 月 17 日

实验报告提交时间：2025 年 12 月 26 日

教务处制

一、 实验目标：

了解 RISC-V mini 处理器架构，在其基础之上新增一个指令，完成设计并观察指令执行。

二、实验内容

1) 修改数据通路，新增指令 comb rs1,rs2,rd 采用 R 型指令格式，实现将 rs1 高 16 位和 rs2 低 16 位拼接成 32 位整数，并且保存到 rd 寄存器。

2) 在处理器上执行该指令，观察仿真波形，验证功能是否正确。

3）自行设计其他功能指令，并验证设计是否正确

三、实验环境

硬件：桌面 PC

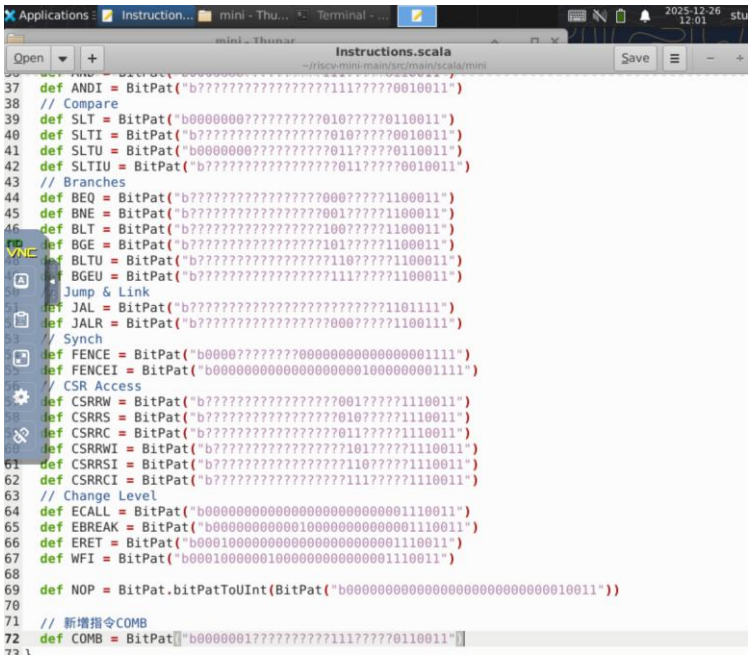
软件：Chisel 开发环境

四、实验步骤及说明

学习 Chisel 数据通路的 Chisel 描述，特别是指令译码部分和 core 核心代码。然后按照下面操作完成指令译码器的修改，以及数据通路的修改，

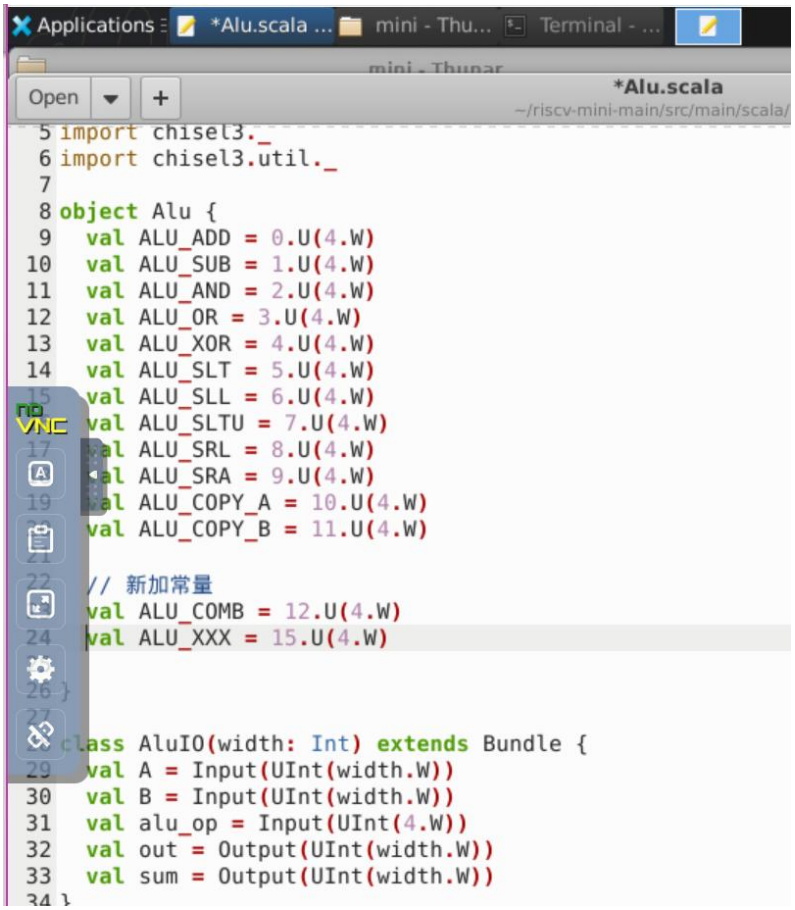
具体操作如下：按照参考文档完成 comb 指令的实现，自行设计新指令实现其功能并验证。

1.在 Instrutcions.scala 文件中添加 comb 指令比特模式串

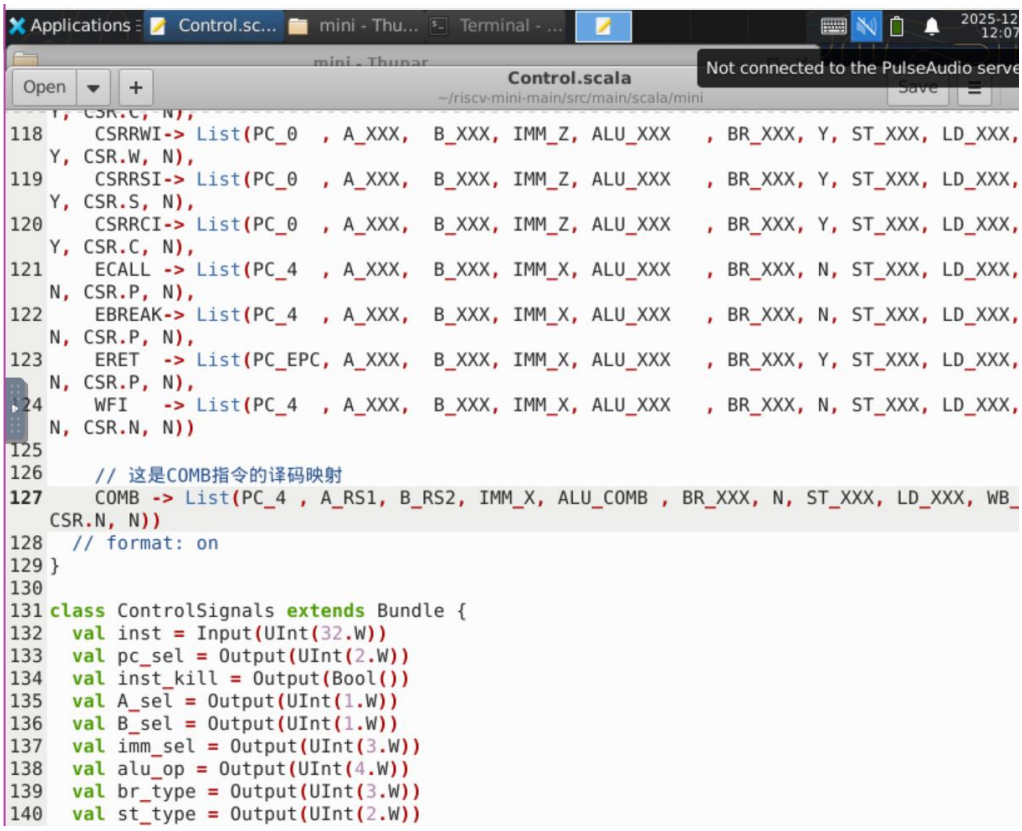


2. 添加 comb 指令的译码

在 Alu.scala 文件中添加常量 ALU_COMB ，让译码器可以译码出正确的信号

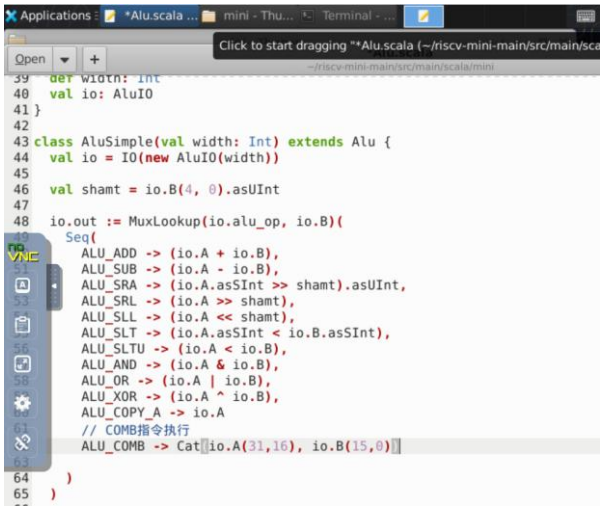


在 Control.scala 文件中添加代码

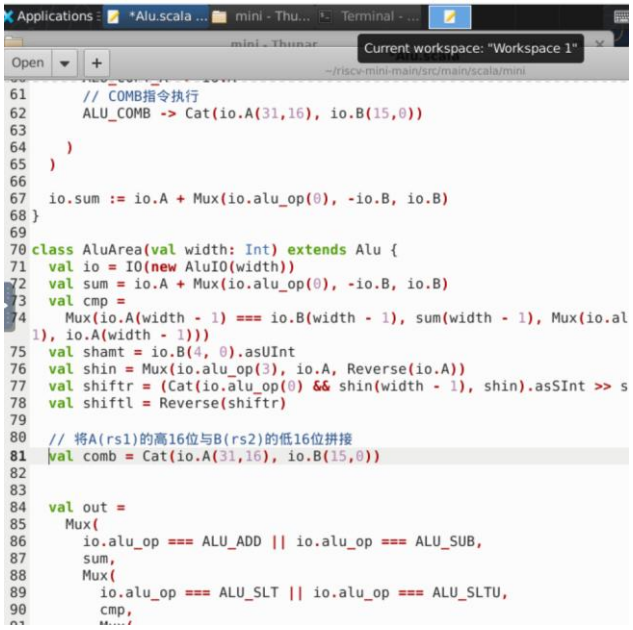


3. 实现 comb 指令的执行操作

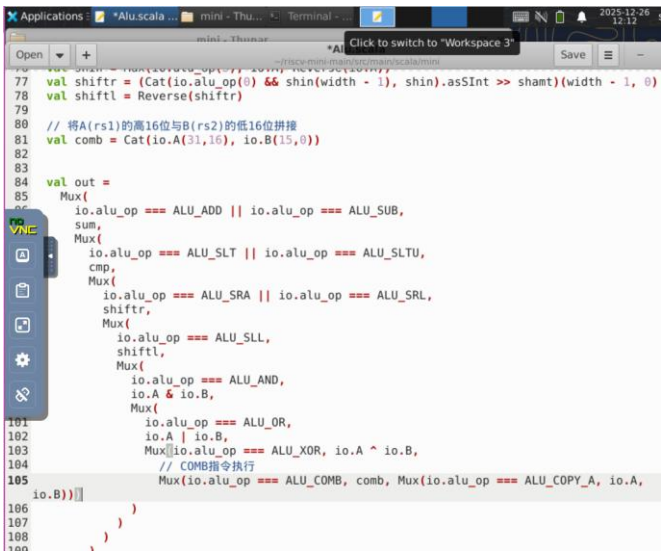
在 Alu.scala 文件添加将 rs1 高 16 位和 rs2 低 16 位拼接成 32 位整数的操作



```
39  def width: Int
40  val io: AluIO
41 }
42
43 class AluSimple(val width: Int) extends Alu {
44   val io = IO(new AluIO(width))
45
46   val shamt = io.B(4, 0).asUInt
47
48   io.out := MuxLookup(io.alu_op, io.B)(
49     Seq(
50       ALU_ADD -> (io.A + io.B),
51       ALU_SUB -> (io.A - io.B),
52       ALU_SRA -> (io.A.asSInt >> shamt).asUInt,
53       ALU_SRL -> (io.A >> shamt),
54       ALU_SLL -> (io.A << shamt),
55       ALU_SLT -> (io.A.asSInt < io.B.asSInt),
56       ALU_SLTU -> (io.A < io.B),
57       ALU_AND -> (io.A & io.B),
58       ALU_OR -> (io.A | io.B),
59       ALU_XOR -> (io.A ^ io.B),
60       ALU_COPY_A -> io.A
61       // COMB指令执行
62       ALU_COMB -> Cat(io.A(31,16), io.B(15,0))
63
64     )
65   )
```



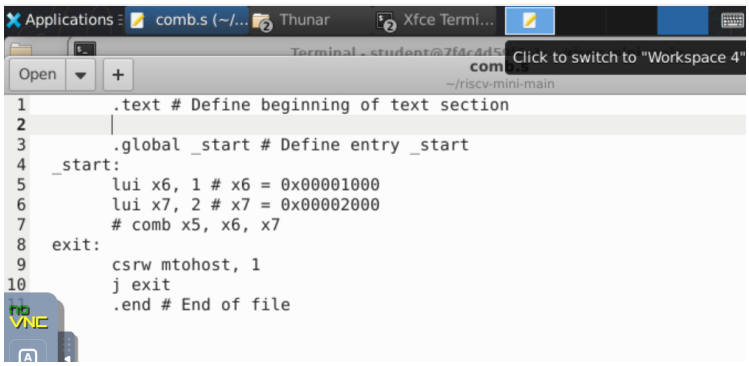
```
61       // COMB指令执行
62       ALU_COMB -> Cat(io.A(31,16), io.B(15,0))
63
64     )
65   )
66
67   io.sum := io.A + Mux(io.alu_op(0), -io.B, io.B)
68 }
69
70 class AluArea(val width: Int) extends Alu {
71   val io = IO(new AluIO(width))
72   val sum = io.A + Mux(io.alu_op(0), -io.B, io.B)
73   val cmp =
74     Mux(io.A(width - 1) === io.B(width - 1), sum(width - 1), Mux(io.al
   1), io.A(width - 1)))
75   val shamt = io.B(4, 0).asUInt
76   val shin = Mux(io.alu_op(3), io.A, Reverse(io.A))
77   val shiftr = (Cat(io.alu_op(0) && shin(width - 1), shin).asSInt >> s
78   val shiftl = Reverse(shiftr)
79
80   // 将A(rs1)的高16位与B(rs2)的低16位拼接
81   val comb = Cat(io.A(31,16), io.B(15,0))
82
83
84   val out =
85     Mux(
86       io.alu_op === ALU_ADD || io.alu_op === ALU_SUB,
87       sum,
88       Mux(
89         io.alu_op === ALU_SLT || io.alu_op === ALU_SLTU,
90         cmp,
```



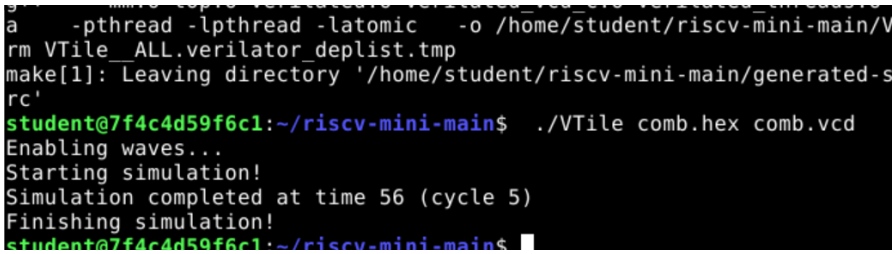
```
77   val shiftr = (Cat(io.alu_op(0) && shin(width - 1), shin).asSInt >> shamt)(width - 1, 0)
78   val shiftl = Reverse(shiftr)
79
80   // 将A(rs1)的高16位与B(rs2)的低16位拼接
81   val comb = Cat(io.A(31,16), io.B(15,0))
82
83
84   val out =
85     Mux(
86       io.alu_op === ALU_ADD || io.alu_op === ALU_SUB,
87       sum,
88       Mux(
89         io.alu_op === ALU_SLT || io.alu_op === ALU_SLTU,
90         cmp,
91         Mux(
92           io.alu_op === ALU_SRA || io.alu_op === ALU_SRL,
93           shiftr,
94           Mux(
95             io.alu_op === ALU_SLL,
96             shiftl,
97             Mux(
98               io.alu_op === ALU_AND,
99               io.A & io.B,
100              Mux(
101                io.alu_op === ALU_OR,
102                io.A | io.B,
103                Mux(io.alu_op === ALU_XOR, io.A ^ io.B,
104                  // COMB指令执行
105                  Mux(io.alu_op === ALU_COMB, comb, Mux(io.alu_op === ALU_COPY_A, io.A,
   io.B))
106            )
107          )
108        )
```

4. 对 comb 指令进行测试

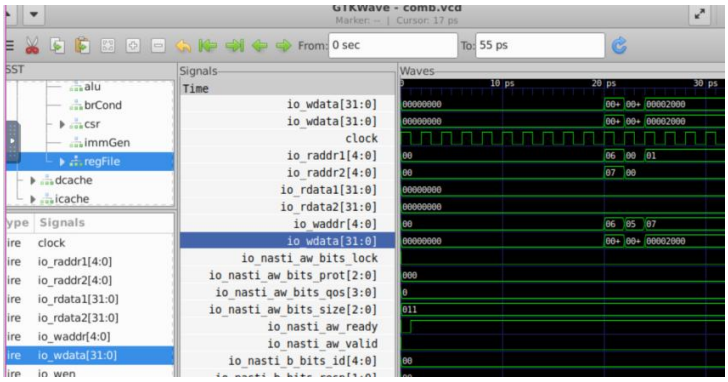
创建 comb.s 文件



编译并转换 hex 文件，按要求进行修改后进行 make，产生 vcd 文件

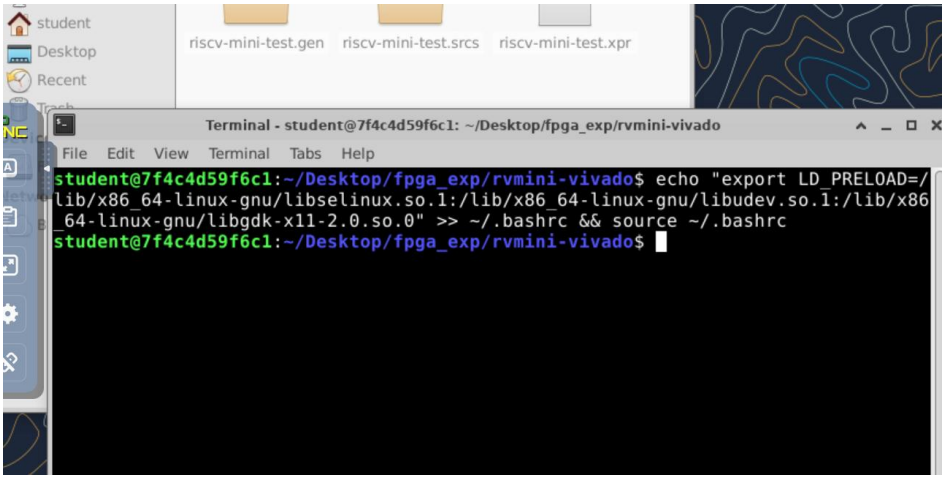


查看波形文件

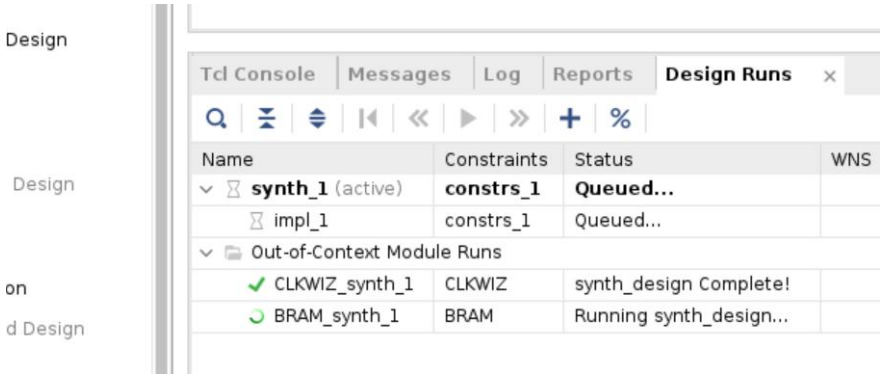


从波形图中可以看出， comb 指令将拼接后的结果 0x00002000 写回到了 5 号寄存器中，故该指令执行正常。

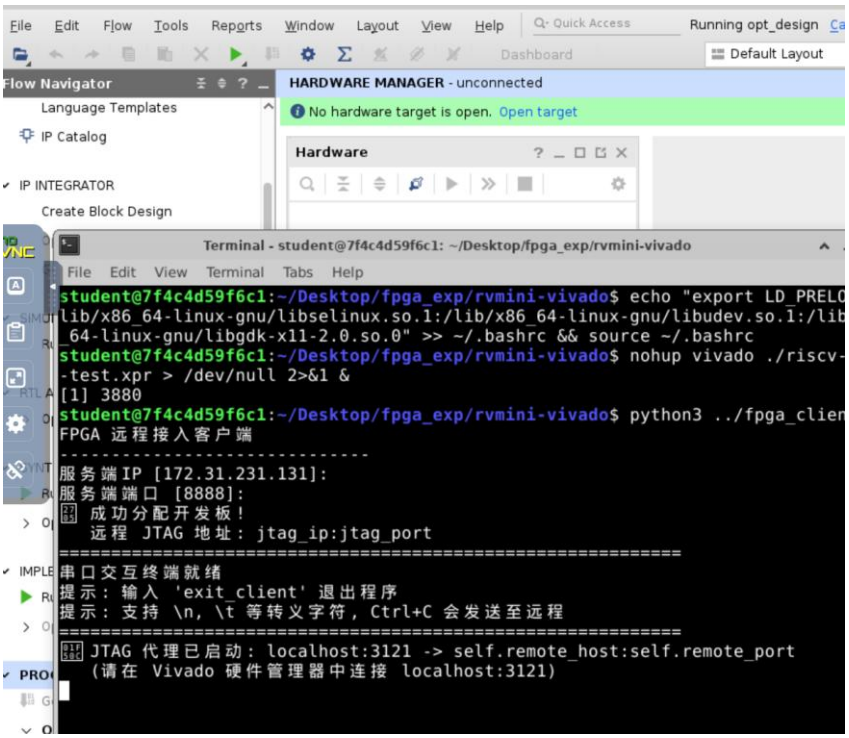
串口实验环境设置

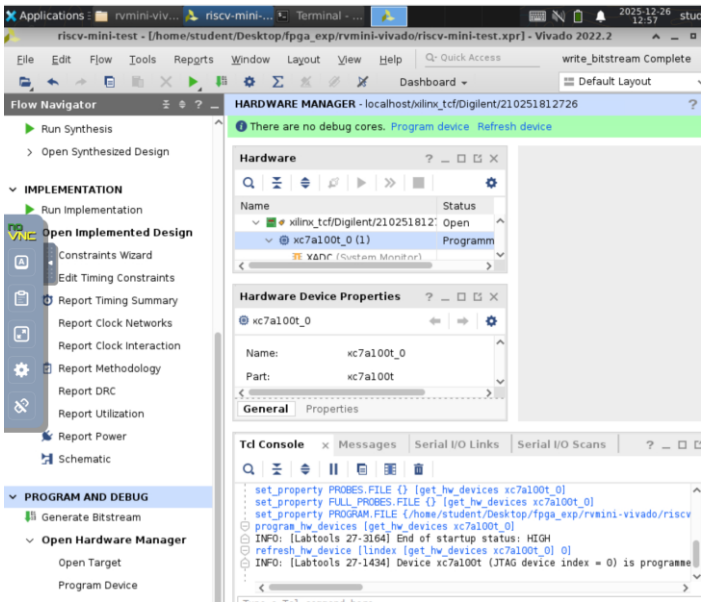


启动 vivado 打开项目，进行比特流生成

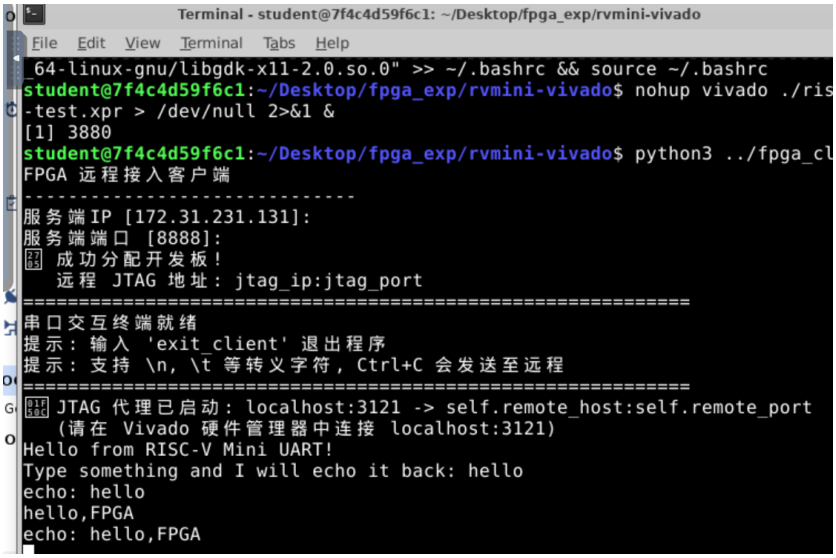


申请并连接开发板

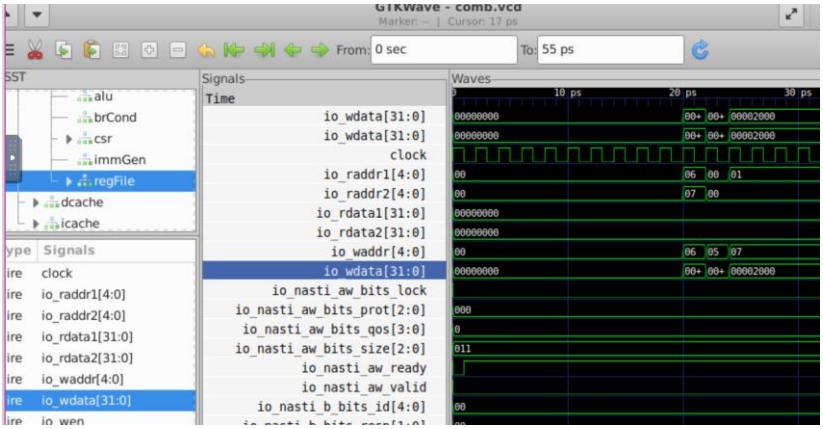




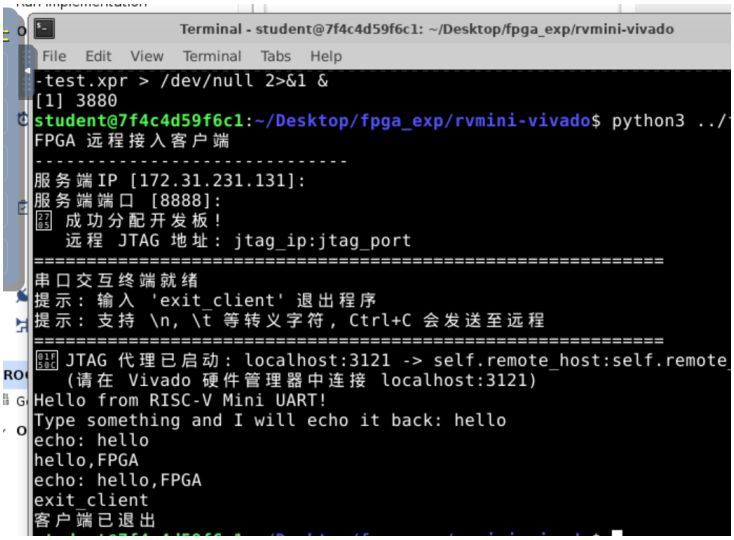
成功连接并可使用



五、实验结果



从波形图中可以看出， comb 指令将拼接后的结果 0x00002000 写回到了 5 号寄存器中，故该指令执行正常。



可成功使用开发板

六、实验总结与体会

通过本次实验，我深入了解了 RISC-V mini 处理器的架构和数据通路的设计，学会了如何新增指令并修改处理器的实现。通过对 comb 指令的设计与测试，掌握了 Chisel 中指令译码和数据通路的修改方法。实验中，成功地将两个寄存器的不同部分拼接成一个 32 位值，并验证了指令功能的正确性。通过调试和仿真波形的分析，进一步加深了对硬件设计的理解和掌握。

指导教师批阅意见： **成绩评定：** 指导教师签字： 年　月　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
　　2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。