

中国科学院大学计算机组成原理（研讨课）

实 验 报 告

学号：2021K8009925006 姓名：冯浩瀚 专业：计算机科学与技术

实验序号：1 实验名称：处理器基本部件单元设计

- 注 1：撰写此 Word 格式实验报告后以 PDF 格式保存 SERVE CloudIDE 的 /home/serve-ide/cod-lab/reports 目录下（注意：reports 全部小写）。文件命名规则：prjN.pdf，其中“prj”和后缀名“pdf”为小写，“N”为 1 至 4 的阿拉伯数字。例如：prj1.pdf。PDF 文件大小应控制在 5MB 以内。此外，实验项目 5 包含多个选做内容，每个选做实验应提交各自的实验报告文件，文件命名规则：prj5-projectname.pdf，其中“-”为英文标点符号的短横线。文件命名举例：prj5-dma.pdf。具体要求详见实验项目 5 讲义。
- 注 2：使用 git add 及 git commit 命令将实验报告 PDF 文件添加到本地仓库 master 分支，并通过 git push 推送到 SERVE GitLab 远程仓库 master 分支（具体命令详见实验报告）。
- 注 3：实验报告模板下列条目仅供参考，可包含但不限定如下内容。实验报告中无需重复描述讲义中的实验流程。

- 一、 逻辑电路结构与仿真波形的截图及说明（比如关键 RTL 代码段{包含注释}及其对应的逻辑电路结构图{自行画图，推荐用 PPT 画逻辑结构框图，复制到 word 中}、相应信号的仿真波形和信号变化的说明等）

● Reg_File:

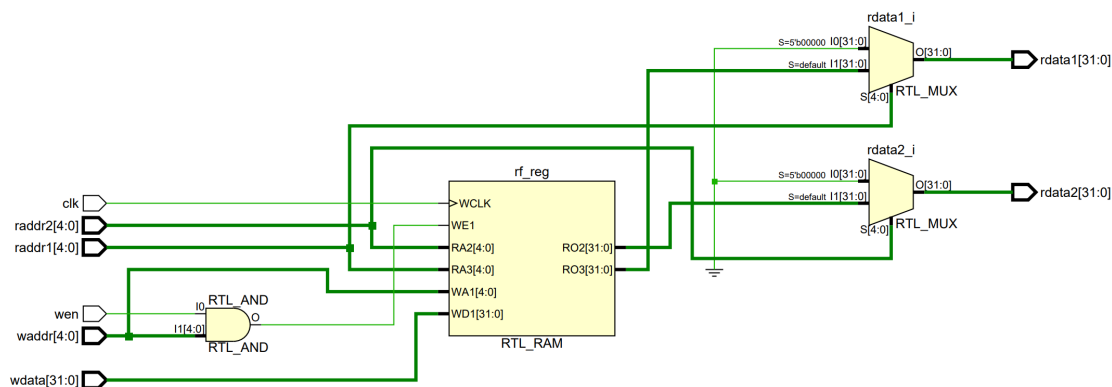
■ 关键 RTL 代码段:

```
reg [`DATA_WIDTH - 1:0] rf [`DATA_WIDTH - 1:0];    // declaration

always @(posedge clk) begin    // write using sequential logic
    if (wen && waddr) rf[waddr] <= wdata;
end

assign rdata1 = (raddr1 == 5'b0)? 32'b0: rf[raddr1];    // read using combinational logic
assign rdata2 = (raddr2 == 5'b0)? 32'b0: rf[raddr2];
```

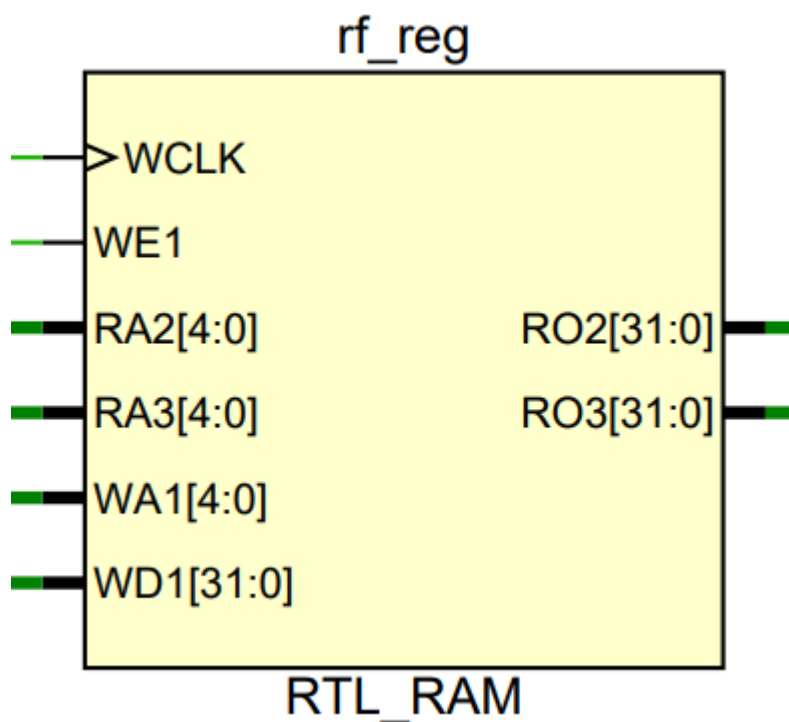
■ 逻辑电路结构图:



■ 代码与逻辑电路结构图的对应：

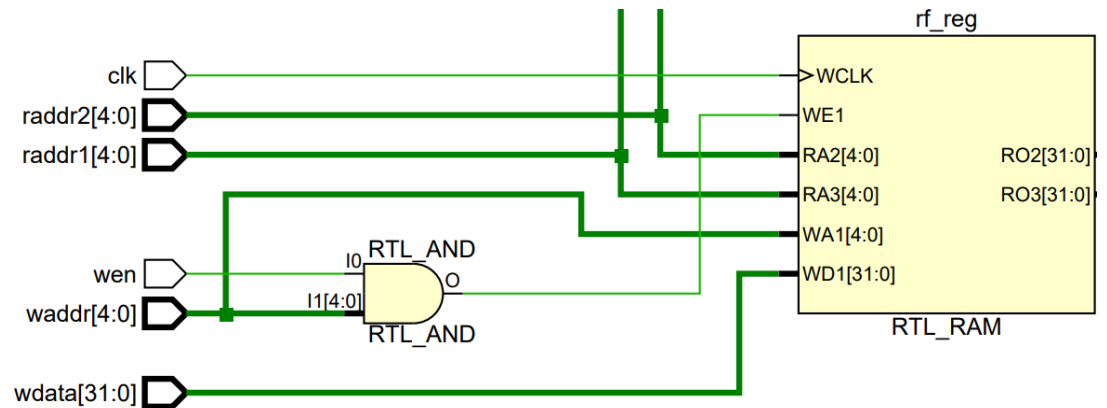
- i. 生成 32 个 32 位寄存器作为一个寄存器堆

```
reg [`DATA_WIDTH - 1:0] rf [`DATA_WIDTH - 1:0];    // declaration
```



- ii. 使用时序逻辑描述写操作

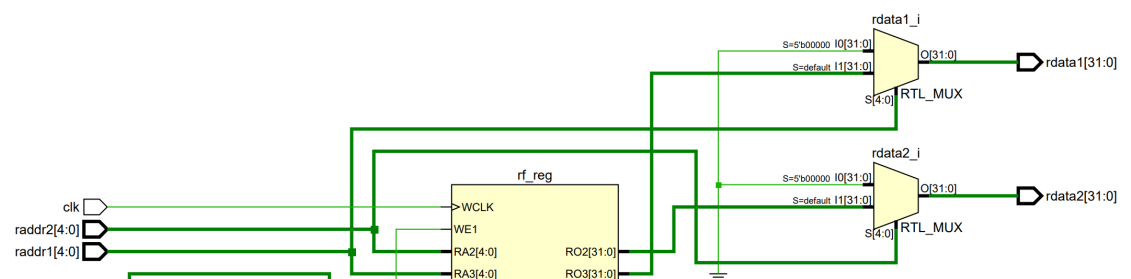
```
always @(posedge clk) begin    // write using sequential logic
    if (wen && waddr) rf[waddr] <= wdata;
end
```



使用与门控制写入的使能信号，即当 wen 和 waddr 均不为 0 时才
能进行写入；clk 信号体现时序逻辑中的同步写入。

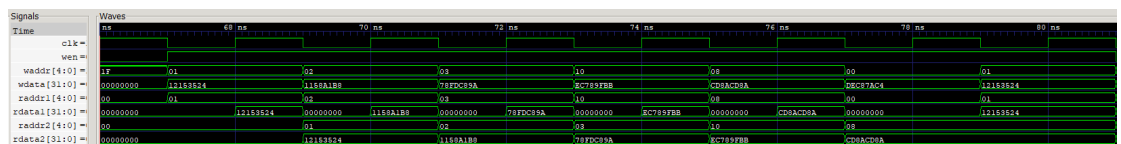
iii. 使用组合逻辑描述读操作

```
assign rdata1 = (raddr1 == 5'b0)? 32'b0: rf[raddr1]; // read using combinational logic
assign rdata2 = (raddr2 == 5'b0)? 32'b0: rf[raddr2];
```



使用数据选择器从寄存器中读出若 raddr 不为 0 时的数据；没有
clk 信号，体现异步读出。

■ 仿真波形图：



同一地址中读出的数据与写入的数据相同，证明了代码的正确性

● ALU

■ 关键 RTL 代码段：

```

`define ALUOP_AND 3'b000
`define ALUOP_OR 3'b001
`define ALUOP_ADD 3'b010
`define ALUOP_SUB 3'b110
`define ALUOP_SLT 3'b111 // use `define to delete "magic numbers"

wire op_and = ALUop == `ALUOP_AND;
wire op_or = ALUop == `ALUOP_OR;
wire op_add = ALUop == `ALUOP_ADD;
wire op_sub = ALUop == `ALUOP_SUB;
wire op_slt = ALUop == `ALUOP_SLT; // decode the alu_op signals

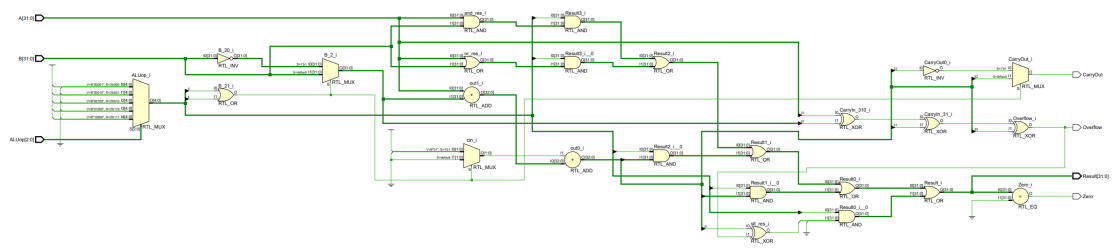
wire [`DATA_WIDTH - 1:0] and_res = A & B; // calculate the result of and
wire [`DATA_WIDTH - 1:0] or_res = A | B; // calculate the result of or
wire [`DATA_WIDTH - 1:0] B_2 = (op_sub || op_slt)? ~B : B;
wire cin = (op_sub || op_slt)? 1: 0;
wire [`DATA_WIDTH - 1:0] out;
wire cout;
assign {cout, out} = A + B_2 + cin;
assign CarryOut = (op_sub || op_slt)? !cout: cout; // calculate the CarryOut flag
wire [`DATA_WIDTH - 1:0] add_res = out; // calculate the result of add
wire [`DATA_WIDTH - 1:0] sub_res = out; // calculate the result of sub
wire slt_res = out [`DATA_WIDTH - 1] ^ Overflow; // calculate the result of slt

assign Result = {`DATA_WIDTH{op_and}} & and_res | // choose the final Result
                {`DATA_WIDTH{op_or}} & or_res |
                {`DATA_WIDTH{op_add}} & add_res |
                {`DATA_WIDTH{op_sub}} & sub_res |
                {`DATA_WIDTH{op_slt}} & slt_res;

wire CarryIn_31 = A [`DATA_WIDTH - 1] ^ B_2 [`DATA_WIDTH - 1] ^ out [`DATA_WIDTH - 1];
//out = a ^ b ^ cin, so cin = a ^ b ^ out
assign Overflow = CarryIn_31 ^ cout; // calculate the Overflow flag
assign Zero = Result == 32'b0; // calculate the Zero flag

```

■ 逻辑电路结构图：



■ 代码与逻辑电路结构图的对应：

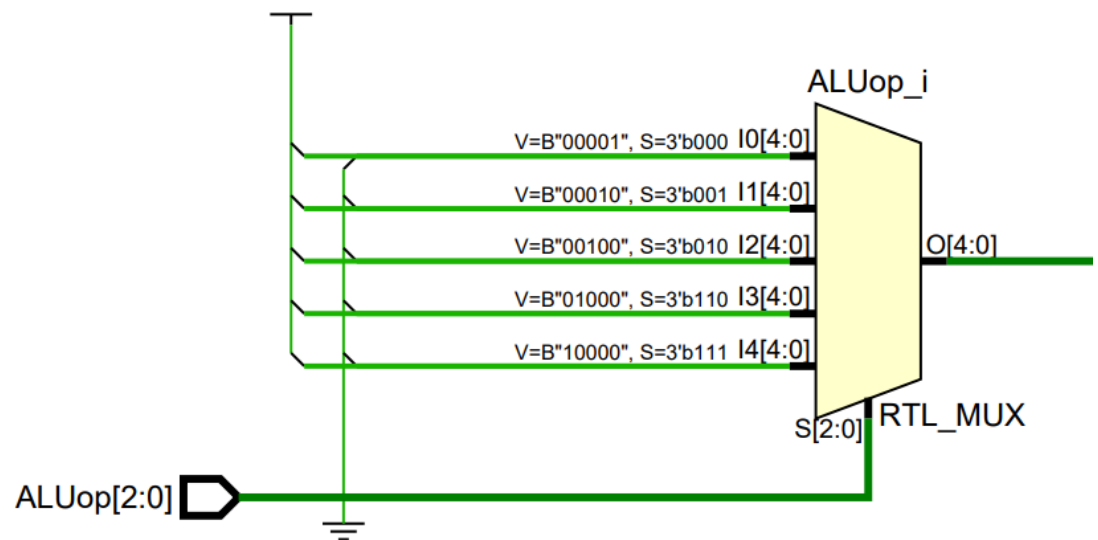
i. 将 ALU_op 译码

```

`define ALUOP_AND 3'b000
`define ALUOP_OR 3'b001
`define ALUOP_ADD 3'b010
`define ALUOP_SUB 3'b110
`define ALUOP_SLT 3'b111 // use `define to delete "magic numbers"

wire op_and = ALUop == `ALUOP_AND;
wire op_or = ALUop == `ALUOP_OR ;
wire op_add = ALUop == `ALUOP_ADD;
wire op_sub = ALUop == `ALUOP_SUB;
wire op_slt = ALUop == `ALUOP_SLT; // decode the alu_op signals

```

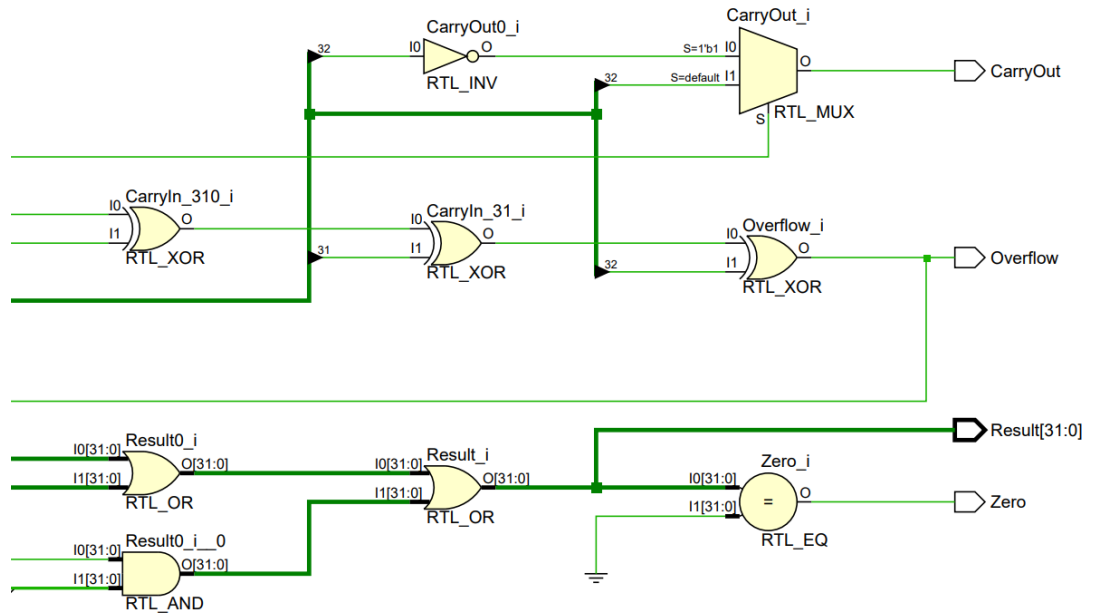


ii. 计算五种操作的结果

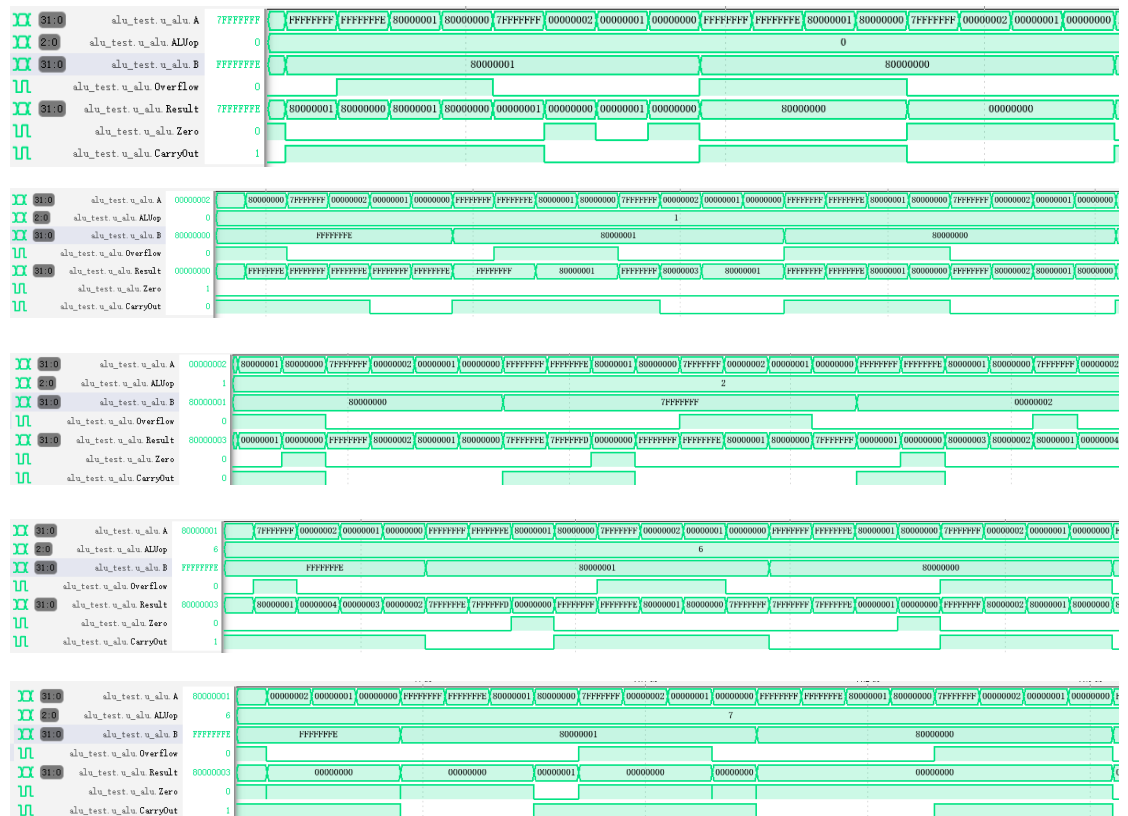
```

wire [`DATA_WIDTH - 1:0] and_res = A & B; // calculate the result of and
wire [`DATA_WIDTH - 1:0] or_res = A | B; // calculate the result of or
wire [`DATA_WIDTH - 1:0] B_2 = (op_sub || op_slt)? ~B : B;
wire cin = (op_sub || op_slt)? 1: 0;
wire [`DATA_WIDTH - 1:0] out;
wire cout;
assign {cout, out} = A + B_2 + cin;
assign CarryOut = (op_sub || op_slt)? !cout: cout; // calculate the CarryOut flag
wire [`DATA_WIDTH - 1:0] add_res = out; // calculate the result of add
wire [`DATA_WIDTH - 1:0] sub_res = out; // calculate the result of sub
wire slt_res = out [`DATA_WIDTH - 1] ^ Overflow; // calculate the result of slt

```

■ 仿真波形图：



二、 实验过程中遇到的问题、对问题的思考过程及解决方法（比如 RTL 代码

中出现的逻辑 bug，逻辑仿真和 FPGA 调试过程中的难点等)

- RF:

- i. bug: 处理 raddr 是否为 0 时候使用了 if-else 语句，导致输出出错
solution: 改用了 "assign data = (sel) ? a : b ;" 三目运算符

- ALU:

- i. bug: slt 计算的结果错误
solution: slt 对有符号数进行比较, 要将两数相减之后的 Overflow
- ii. bug: CarryOut 计算的结果错误
solution: 如果是减法模式, 要将 CarryOut 的结果取反才是争取结果

三、 在课后，你花费了大约____5____小时完成此次实验。

四、 对于此次实验的心得、感受和建议（比如实验是否过于简单或复杂，是否缺少了某些你认为重要的信息或参考资料，对实验项目的建议，对提供帮助的同学的感谢，以及其他想与任课老师交流的内容等）

- i. 感谢指导老师以及助教老师撰写的实验文档的指导尤其是对 Verilog 语法的讲解和简单 ALU 编写的实例。这些指导给了我极大的帮助。
- ii. 感谢李家驹老师在我遇到 bug 或不确定实验要求时候（比如怎么样才算“使用了一个加法器”）给我提供的帮助。
- iii. 该实验总体上较为简单，参考资料较为齐全。