中国科学院大学计算机组成原理实验课

实 验 报 告

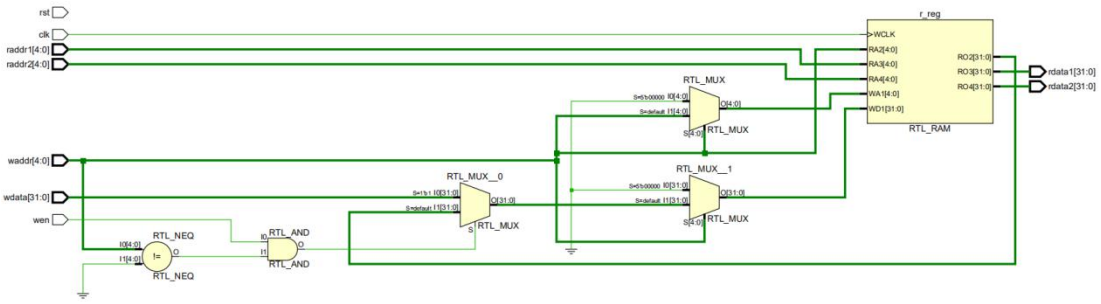
学号：_2018K8009929030_　姓名：_热伊莱·图尔贡_　专业：_计算机科学与技术_

实验序号：_1_　实验名称：_CPU 的基本功能部件设计_

注 1：请在实验项目个人本地仓库中创建顶层目录 doc。撰写此 Word 格式实验报告后以 PDF 格式保存在 doc 目录下。文件命名规则：学号-prjN.pdf，其中学号中的字母“K”为大写，“-”为英文连字符，“prj”和后缀名“pdf”为小写，“N”为 1 至 4 的阿拉伯数字。例如：2019K8009929000-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：学号-prj5-projectname.pdf，例如：2019K8009929000-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 doc 目录下的实验报告 PDF 文件添加到本地仓库 master 分支，并通过 git push 推送到 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、相应信号的仿真波形和信号变化的说明等）

- Register File



【Register File 原理图】

```
reg [31:0] r [31:0];
always @ (posedge clk) begin
    if (waddr == 5'd0) begin
        r[waddr] <= 32'b0;
    end
    else if (wen && waddr != 5'd0) begin
        r[waddr] <= wdata;
    end
end
assign rdata1 = r[raddr1];
assign rdata2 = r[raddr2];
```

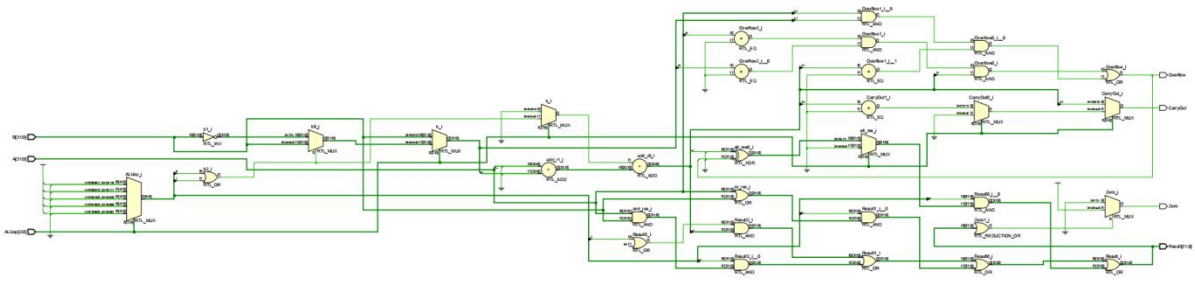
【Register File 代码】

· 对于“异步读，同步写”的实现：

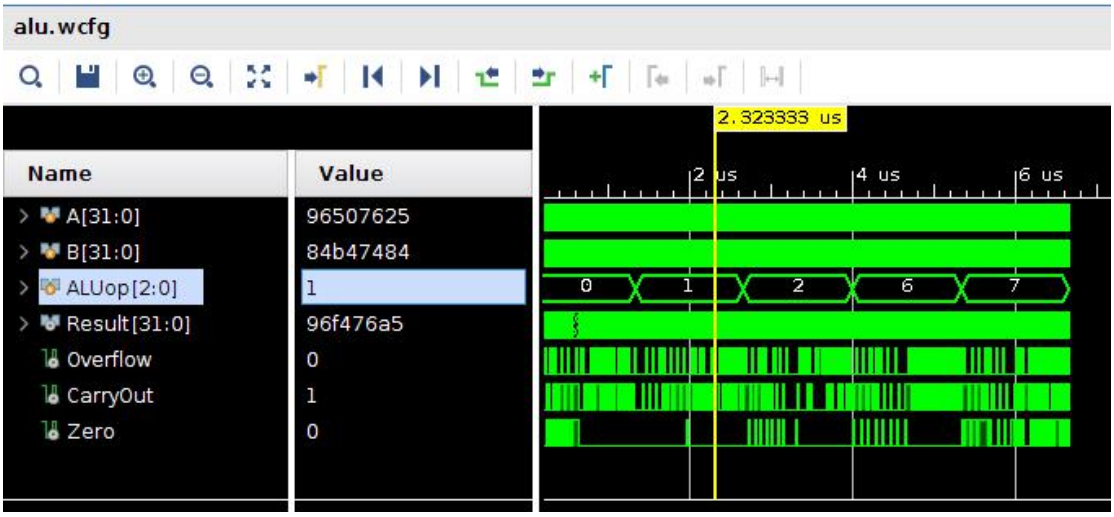
-异步读：读操作不受时钟约束，因此是组合逻辑。

-同步写：写操作受时钟约束，因此是时序逻辑。

- ALU



【ALU 原理图】



【ALU 波形图】

```
parameter AND =3'b000;
parameter OR  =3'b001;
parameter ADD =3'b010;
parameter SUB =3'b110;
parameter SLT =3'b111;
```

【ALU 代码图——ALUop】

✓ 对于 ALU 加(ADD)减(SUB)，有符号整数比较(SLT）运算复用一套加法器的代码实现：

1. 加（ADD）减（SUB）运算：

加法运算：A+B

减法运算：A+(-B)+1

因此，可以得出（不考虑进位/借位）复用一套加法器的加减法运算的代码，即：

ADD: assign result = a + b + 0;

SUB: assign result = a + b + 1;

由【ALU 代码图——ALUop】图中 ADD(010)和 SUB(110)的 ALUop 可以看出 ADD（加法）运算的 ALUop 最高位是‘0’，SUB（减法）运算的 ALUop 最高位是‘1’，与上述代码中的末尾加法操作数一致，因此可以方便实现加法器的减法运算。

由于 SLT（111）运算的 ALUop 最高位是‘1’，与 SUB 运算一致，因此在进行有符号整数的比较运算时，可以保证加法器按减法进行运算。

```
wire [`DATA_WIDTH - 1:0] add_r;
wire [`DATA_WIDTH - 1:0] a;
wire [`DATA_WIDTH - 1:0] b;
wire s;
wire carry;
//ADD:a+b+0; SUB:a+~b+1
assign a = A;
//assign b = (ALUop ==ADD)? B : ((ALUop ==SUB || ALUop == SLT)?(~B):B);
assign b = (ALUop ==ADD)? B : ~B;
//assign s = (ALUop ==ADD)? 0 : ((ALUop ==SUB || ALUop == SLT)?1:0);
assign s = ALUop[2];
assign {carry,add_r} = {0,a} + {0,b} + s;
assign add_sub_res = add_r;
```

【ALU 代码图——ADD ，SUB】(此为优化后的代码，注释部分为源代码)

2. Carry Out:

```
//assign CarryOut = (ALUop == ADD) ? ((carry==1)?1:0) : ((ALUop ==SUB)?((carry==0)?1:0):0);
assign CarryOut = carry ^ s;
```

【ALU 代码图——CarryOut】(此为优化后的代码，注释部分为源代码)

在加减法运算时增加一位用于捕捉进位，

即:

· assign {carry,ADD-result}={0,A}+{0,B}+0;

· assign {carry,SUB-result}={0,A}+{0,~B}+1;

进位：carry=1；

借位：carry=0；

3. Overflow:

溢出判断可做枚举:

· 正数+正数=负数；

· 负数+负数=正数；

```
//Overflow: A(+)+B(+)=R(-); A(-)+B(-)=R(+);
assign Overflow = ((a[31]==0 && b[31]==0 && add_r[31]==1) || (a[31]==1 && b[31]==1 && add_r[31]==0))?1:0;
```

【ALU 代码图——Overflow】

4. 有符号整数比较（SLT）运算:

由【ALU 代码图——ALUop】图可以看到由于 SLT（111）运算的 ALUop 最高位是‘1’，与 SUB 运算一致，因此在进行有符号整数的比较运算时，可以保证加法器按减法进行运算。

```
//A-B-->(-)--->slt=1
//assign slt_res  = (ALUop == SLT)?add_r[31] ^ Overflow:0;
assign slt_res = add_r[31] ^ Overflow;
```

【ALU 代码图——SLT】（此为优化后的代码，注释部分为源代码）

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，仿真、云平台调试过程中的难点等）

问题：在 SLT 运算中，忽略了同号整数溢出问题。

错误代码：assign slt_res = add_r[31];

解决方法：判断 A-B 是否溢出，如果溢出则 A-B 结果为正数时，A<B；如果不溢出则 A-B 结果为负数时，A<B。

正确代码：assign slt_res = add_r[31] ^ Overflow;

三、 在课后，你花费了大约____12_____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）

我认为此次实验难度适中，老师提供的信息完整。

在这次实验中非常感谢高云聪老师在验收时提供的思路，让我可以优化代码，以及王嵩岳老师和热依汉同学在提交流程出现问题时提供的帮助。