中国科学院大学计算机组成原理实验课

实 验 报 告

学号：2018K8009929046　　姓名：何咏哲　　专业：计算机科学与技术

实验序号：1　　实验名称：基本功能部件设计——Register File & ALU

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：2018K8009929000-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：2018K8009929000-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，并通过 git push 推送提交。

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）

（一）Register File

```
always @(posedge clk) begin
        if(rst) begin
                REG_FILE[0] <= 32'b0;
        end
        if(wen && !rst) begin
                REG_FILE[waddr] <= wdata;
        end
end

assign rdata1 = (raddr1==5'b0)?32'b0:REG_FILE[raddr1];
assign rdata2 = (raddr2==5'b0)?32'b0:REG_FILE[raddr2];
```

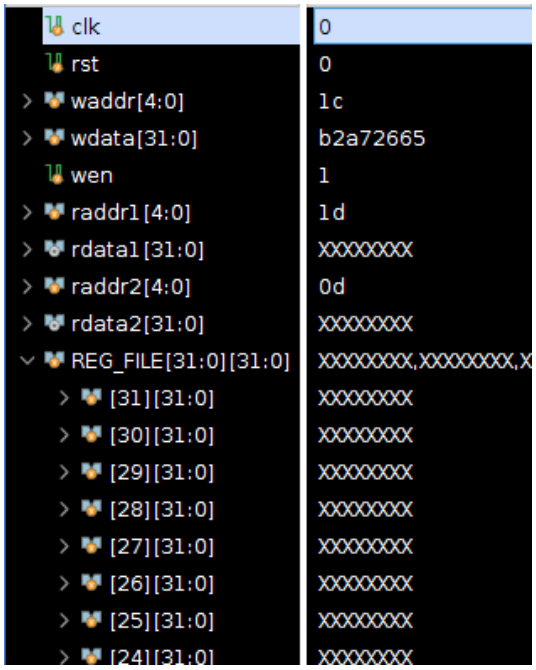
可以看到，写入操作是由时钟控制的，并且仅当写使能信号有效以及 rst 信号无效时才可以进行写操作，实现了同步写（reset 时不写入）的功能。

读出操作不受时钟的限制，故为异步读。

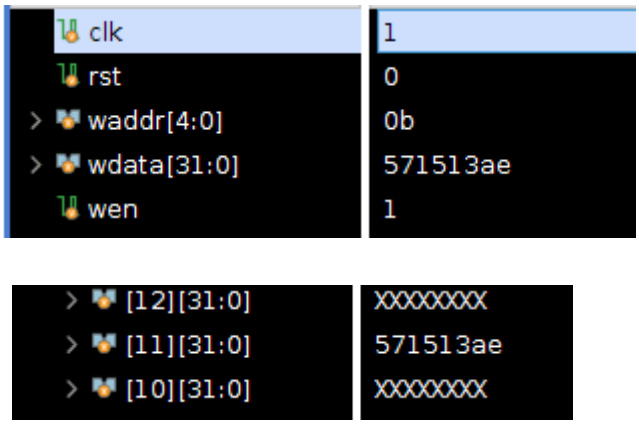
0 号寄存器在时钟控制下，复位信号有效时进行清零，实现了同步复位。

在读取数据时，若目标寄存器是 0 号寄存器，则始终输出 0。

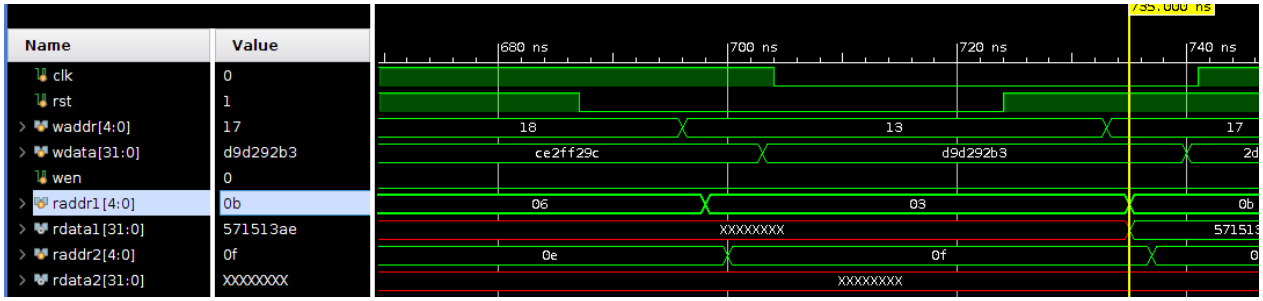
能够反映上述功能的波形图如下：



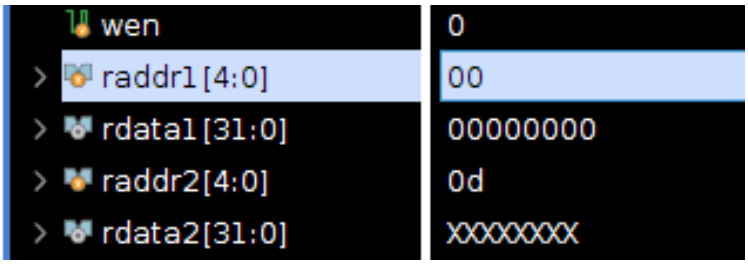
时钟无效，不能写入。



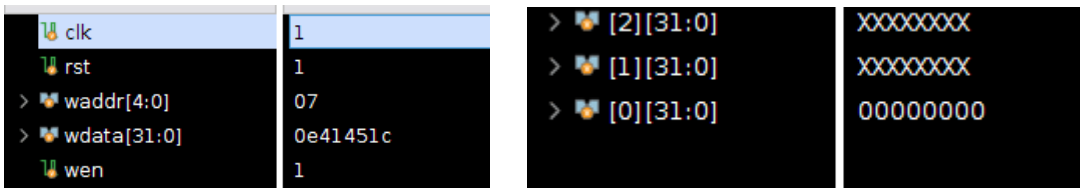
写入允许，11 号寄存器的值被更改。



1 号读端口在非时钟控制下实现读操作，即异步读。

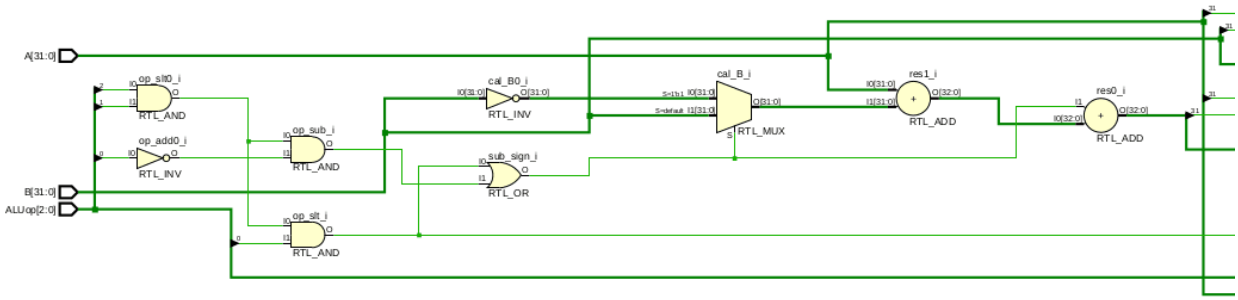


0 号寄存器总是读出 0。



时钟信号与复位信号有效时，0 号寄存器清零。

（二）ALU



如图所示，加法操作、减法操作及有符号数比较操作使用同一套加法器逻辑进行运算，符合实验要求。实现此功能的代码如下：

```
assign cal_B = (sub_sign)?~B:B;
assign plus = (sub_sign)?32'b1:32'b0;
assign {cf,res} = A + cal_B + plus;
```

其中 cal_B 是 B 在计算前的处理量，当执行加法时，cal_B 与 B 相等；当执行减法时，cal_B 为 B 的按位取反。plus 为相加操作的附加量，因为在进行减法操作时结果等于 A+~B+1。

执行 slt 时，其结果可以规约为“减法结果的最高位与减法溢出的亦或”，可以看到减法是前提操作，而在使用了同一套加法器之后我们并不能保证减法结果的正确性，因此需要在执行计算前进行判断，如果 slt 的信号为 1，则将减法的信号也置为 1，其实现方式为：

```
assign sub_sign = (op_slt | op_sub)?1:0;
```

在进行溢出判断时，判断的依据如下：

1、同号相加，结果最高位与被加数不同

2、异号相减，结果最高位与被减数不同

代码如下：

```
assign add_of = (A[`DATA_WIDTH - 1]^!B[`DATA_WIDTH - 1])&(A[`DATA_WIDTH - 1]^res[`DATA_WIDTH - 1]);
assign sub_of = (A[`DATA_WIDTH - 1]^B[`DATA_WIDTH - 1])&(A[`DATA_WIDTH - 1]^res[`DATA_WIDTH - 1]);
```

在进行进/借位判断时，判断依据如下：

1、对于加法，使用位拼接运算符，最高位即为 CarryOut 的值

2、对于减法，使用位拼接运算符，最高位的取反即为 CarryOut 的值

其中对减法的 CF 进行说明，由于减法是通过加上被减数的补码实现的，因此这样得到的 CF 相比于真正的 CF 有一个翻转。

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、本地上板及云平台调试过程中的难点等）

1、在多个 assign 语句中给同一个 wire 型变量赋值。此时 rtl_chk 并未报错，直到生成比特流与二进制文件时报错。由于并没有发现出错的原因，因此并未引起太多重视，直接进行云平台上板测试。而由于生成比特流失败，此时上板的是之前生成的二进制文件，导致反复尝试了很多次、修改了很多次代码之后得到的是一样的结果。直到最后查阅日志文件，发现了 multi-driver 报错，才排查了该错误。

2、没有使用“同一套加法器”。按照之前的理解，我认为都使用同样位宽的加法就算是使用了同一套加法器，因此将加法与减法分开写了。直到进行课堂验收的时候助教老师告诉我这样不符合要求，我才开始去思考这个问题。原来使用“同一套加法器”是要求加法和减法要用同一个加法式子完成，于是考虑在进行加法操作之前先对第二个操作数以及相加的常数 1（或 0）进行处理，虽然这样会导致不能同时正确地得出加法与减法的结果，但由于我们只选择需要的结果进行使用，所以对其他运算的正确性没有要求。在经过了反复修改之后终于达到了要求。

3、寄存器堆的初始化问题。出于习惯与波形图的美观，我总会在声明 reg 变量时对它们进行初始化操作，一开始在这次寄存器堆的设计中就把所有寄存器的值都赋为 0。而验收的时候助教老师告诉我，这样初始化可能会遭受到黑客的攻击，导致安全隐患。在老师提出这个问题之前我还没有对安全进行过考虑，以后会尽可能全面地测试和优化自己的设计的！

三、 对讲义中思考题（如有）的理解和回答

多出的三种情况可以当作未定义的输入，在这样的输入下，所得到的结果都是未定义的、无效的。可以添加一个 ERROR 信号，当输入为这三种未定义的操作时，将 ERROR 置为高电位，以示警告。后续若有需求可以将这三个未定义的操作改造为需要的操作。

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）

我觉得本次实验的难度尚可，但是在与同学的讨论交流过程中我发现，对于未接触过相关设计或者接触较少的同学来说，在实验开始阶段会感到不知所措，不知道应该从哪里入手。建议老师前期增加一些对实验设计的大致框架与实现方式的讲解，让同学们知道在代码的每一部分有些什么具体的任务需要完成，后期熟练以后再给同学们更多自由发挥的空间。