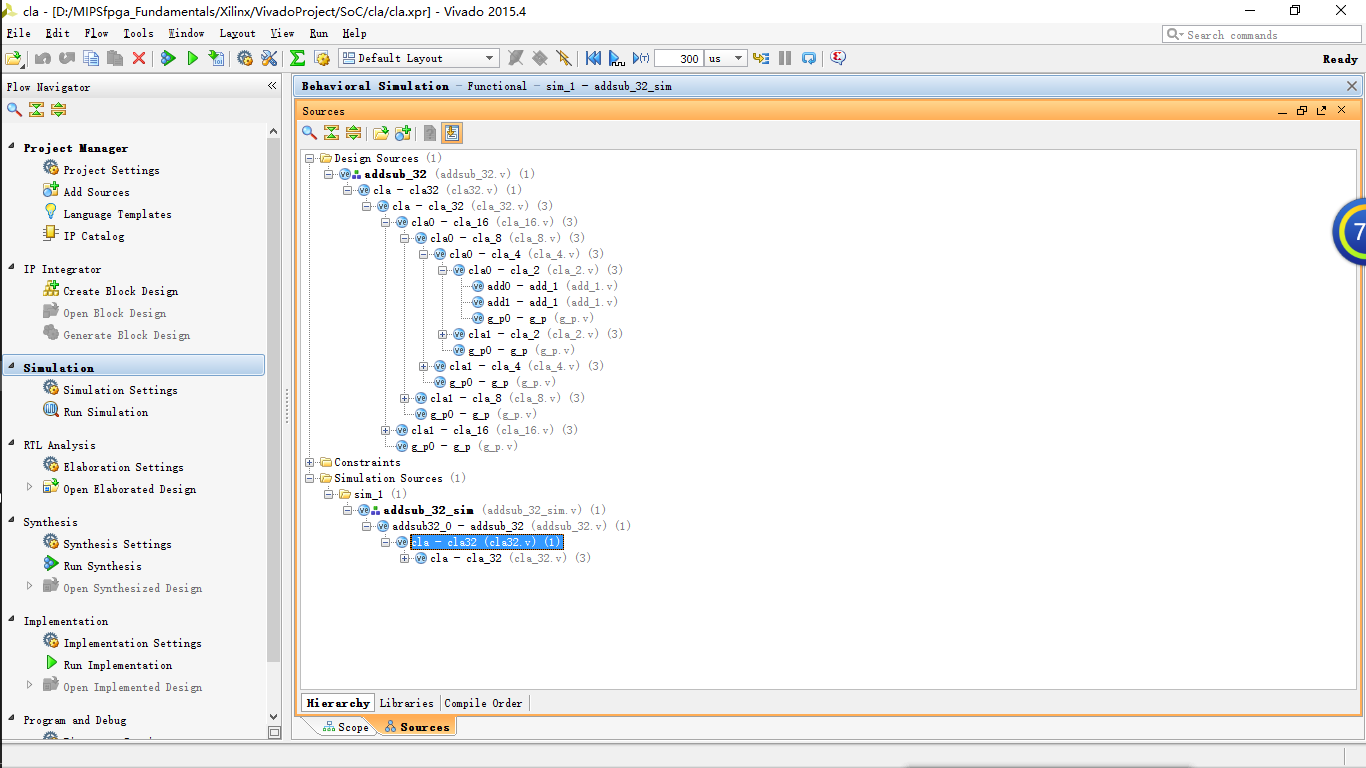
**2017.10.14**

**① 32位超前进位加减法器（已完成）**

## 一、32位超前进位加减法器

***1. 文件结构（详细设计见参考书）***

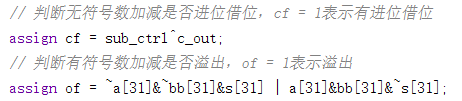


***2. 遇到的问题***

①如何判断无符号数是否有进位和借位？

②如何判断有符号数是否溢出

在addsub\_32顶层文件中添加如下代码：



因为of的判断需要用第二运算数经过补码转换之后的值，所以需要提前用bb保存b求补的结果；一开始我们直接用的b参与判断，导致of输出错误。

**2017.11.04**

**① 32位超前进位乘除法器（已完成）**

**② 指令译码器（完成了一半）**

## 二、32位乘除法器

***1. 文件结构***



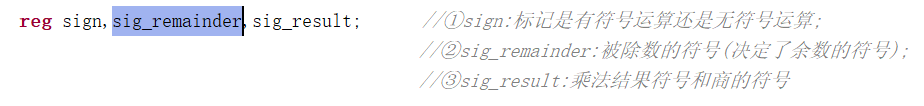
***2. 详细设计思路***

将32位操作数适当分组，构建一个状态机，完成无符号乘、有符号乘、无符号除和有符号除四种运算。

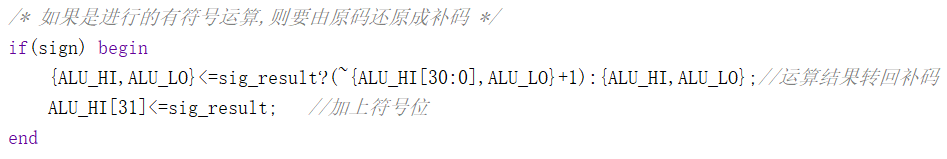
状态机一共有13个状态，S0进行各信号线的初始化，S1~S3完成无符号乘法和有符号乘法运算，S5~S13

***3. 遇到的问题***

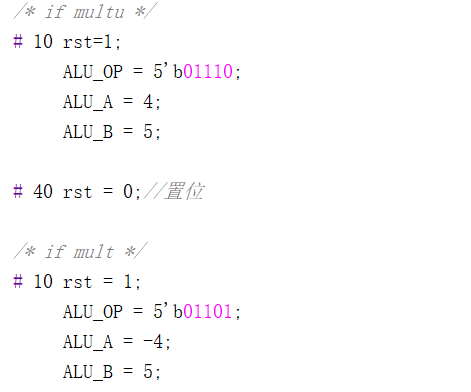
①余数的符号位由被除数的符号位决定的，一开始我们忘记了单独保存被除数的符号位，直接用AB两个操作数最高位的异或值来代替，导致最后运算结果出错。单独设置了sig\_remainder之后解决了这个bug；



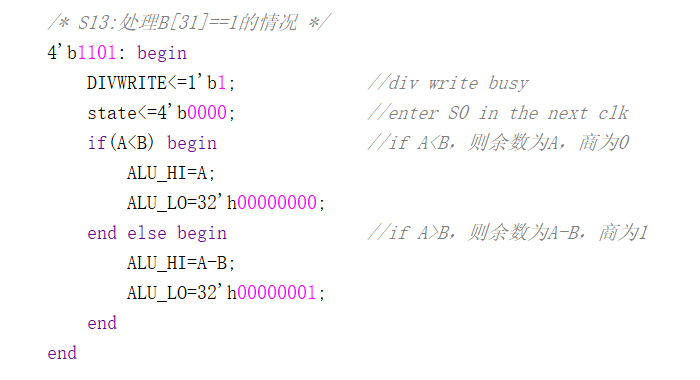
②因为我们是采用的原码一位乘法和原码一位除法，所以对于有符号运算需要先将操作数转化为原码，计算完毕之后再将结果转回为补码；



③编写仿真文件的时候，关于clk的翻转和延时（clk究竟取#5还是#10，2倍关系），并且每次进行新的运算之前都要reset一次；



④因为是整数除法，所以在无符号除法的最开始要判断一下被除数ALU\_B的最高位是否为1。因为若ALU\_B[31]为1且A≥B，则A/B的商只能是1，此时余数为A-B；ALU\_B[31]为1且A＜B，则A/B的商只能是0，此时余数为A；若ALU\_B[31]为0，则进一步使用加减交替法求解。



## 三、指令译码器

***1. 文件结构***



***2. 详细设计思路***

将32位操作数适当分组，构建一个状态机，完成无符号乘、有符号乘、无符号除和有符号除四种运算。

状态机一共有13个状态，S0进行各信号线的初始化，S1~S3完成无符号乘法和有符号乘法运算，S5~S13

***3. 遇到的问题***

①余数的符号位由被除数的符号位决定的，一开始我们忘记了单独保存被除数的符号位，直接用AB两个操作数最高位的异或值来代替，导致最后运算结果出错。单独设置了sig\_remainder之后解决了这个bug；

