**­­­实验6报告**

学号 2017K8009929034

姓名 杨宇恒

箱子号 15

一、实验任务

增加算术逻辑运算类指令、乘除运算类指令、乘除法配套的数据搬运指令，增加相应的HI、LO寄存器和数据通路、控制信号。

二、实验设计

（一）总体设计思路

对于算术逻辑运算指令，尽量复用现有数据通路并调整控制信号。

对于乘除指令，新增与alu并列的乘法部件和除法部件，通过IP核实现，并且在执行级增加HI、LO寄存器。

对于乘除法配套的数据搬运指令，在执行级读取、写入HI、LO寄存器。

（二）重要控制与数据通路设计：基本算术逻辑运算指令

1、工作原理

通过对新的指令译码，之后对alu输入多选器和输出多选器进行新的控制。同时增加立即数非符号拓展数据通路，用src2\_is\_uimm作为多选器控制信号。

2、重要wire定义

下面依次是多选器的控制信号和alu的控制信号修改。



图1多选器控制信号修改

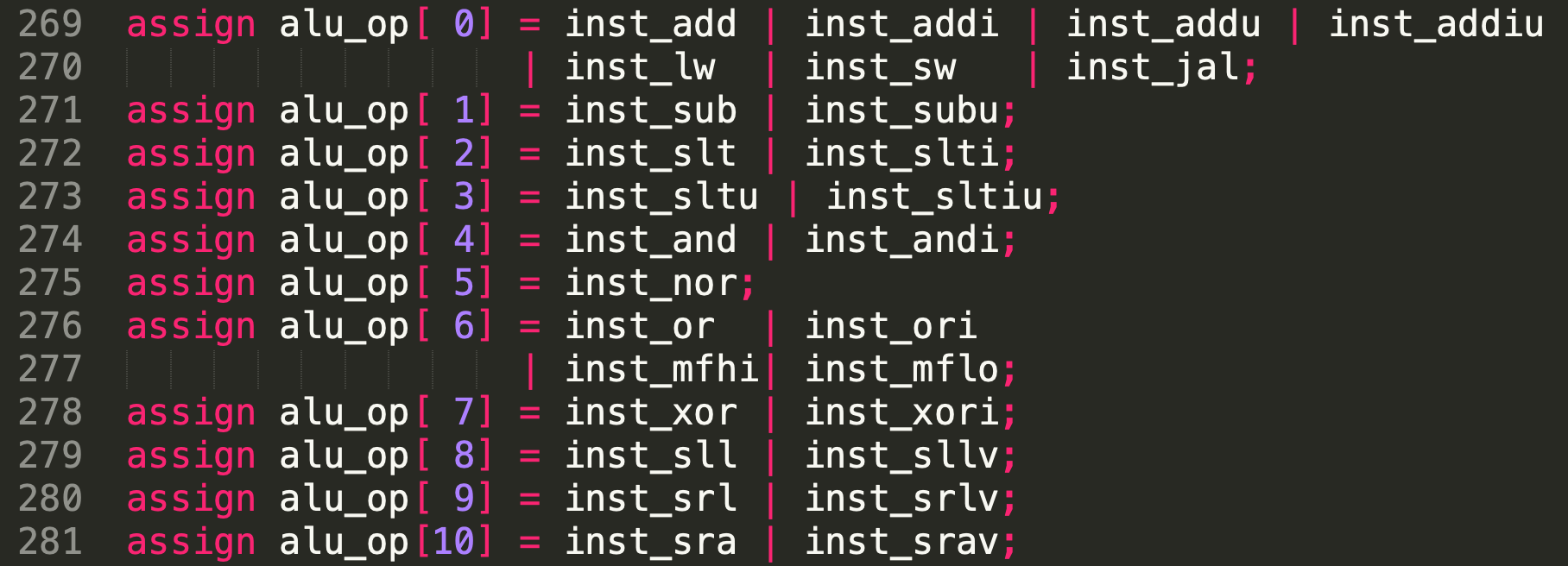


图2 alu控制信号修改

（三）重要模块设计：除法部件

1. 工作原理

封装除法器IP核，并实现功能1）为除法器提供握手信号valid，2）在上一个运算结果没有出来之前，不会把新的valid从封装外部传递给IP核，这和静态流水相符。3）把IP核输出的valid维持住，直到mem流水级可以进入。

1. 重要wire和reg定义

| 组别 | 名称 | 功能描述 |
| --- | --- | --- |
| 向IP输入的输入有效握手信号 | div\_in\_valid\_r | 在！div\_busy\_w的时候根据部件输入有效置位。 |
| div\_in\_valid\_w | 在div\_in\_valid\_r被置位的前一个周期就输出将要被置位的值，其余周期与div\_in\_valid\_r相同。 |
| 部件内部的判断是否接受新输入的信号 | div\_busy\_r | 与div\_in\_valid\_r同时被置1，与div\_out\_valid\_r同时被置0。 |
| div\_busy\_w | 在div\_busy\_r被置位的前一个周期就输出将要被置位的值，其余周期与div\_busy\_r相同。 |
| 向部件外部输出的得到计算结果信号 | div\_out\_valid\_r | 除法器IP计算完成时置1，部件和外部输出握手后或新的输入到来时被置0。 |
| div\_out\_valid\_w | 在div\_out\_valid\_r被置位的前一个周期就输出将要被置位值，其余周期与div\_out\_valid\_r相同。 |
| 除法操作码 | Div\_op[1:0] | Div\_op[0]表示有符号，Div\_op[1]表示无符号 |

3、需要引入新的valid寄存器信号的原因

我们在设计过程中遇到的问题是如何给除法器IP核输入有效信号。

这是否可以复用es\_allowin、ds\_to\_es\_valid呢？我们在设计时进行了如下分析。

在不修改这两个信号的情况下，如果dividerIP没有readyin，他们依旧会置仅仅高一个周期后变为零。这就需要修改allowin，让它等待dividerIP的readyin置高之后再拉低，但这不满足allowin本身让上一个流水级数据进入的语义。实际上，allowin只能拉高一个周期，allowin需要在除法器ready之后再置高。因此如果触发器的握手规则是valid拉高后ready才拉高，就会发生死锁。这时，就不能复用es\_allowin、ds\_to\_es\_valid。

那么，如果触发器的握手规则不要求valid先拉高呢？这要求allowin在除法器ready之后再置高，然而如果触发器的握手规则中ready有可能在置高后，并且没有valid之前自己拉低，在保证allowin只拉高一个周期的条件下，那么会出现除法器没有成功输入信号的错误。

现在我们假设触发器的握手机制满足1）ready不会等候valid拉高才拉高，2）ready不会在valid到来之前自己降低。此时，allowin需要等待除法器ready再变成1。但allowin不能对任何指令都等待，而只在下一条指令用到除法器的时候才等待。这样，就需要在ds\_to\_es\_valid的时候去判断exe的下一条指令是否用到除法器。下面我们分析这是否会形成逻辑环。

既然我们希望添加allowin由ds\_to\_es\_valid控制的组合逻辑，那么就需要判断是否有相反的组合逻辑依赖。值得注意，id级的readygo有对exe级valid的前递逻辑依赖，但不会形成逻辑环，然而正如下面的分析这会导致流水级间的死锁。

对于死锁，我们先分析让allowin等ds\_to\_es\_valid，除了会推迟流水级寄存器的更新，会带来什么负面影响。这就是es\_valid不会及时更新而是等ds\_to\_es\_valid为1之后再更新，这样如果ds\_to\_es\_valid某些条件下需要等es\_valid，就会发生死锁。实际上，考虑阻塞的情况，只有es\_valid为0，ds\_ready\_go才为1，ds\_to\_es\_valid才为1，然而只有在ds\_to\_es\_valid变为1之后allowin才会变为1，allowin才会改变es\_valid的值为0，因此出现死锁。

这样，我们需要创建新的valid信号作为除法器IP的输入，我们下面把这个包含除法器IP的模块封装成除法部件。

4、功能描述

（1）输入有效：需要创建一个valid寄存器和除法器IP输入ready进行握手。

（2）如果除法器IP在运算过程中除法部件模块还有输入valid信号怎么处理/避免？这可以通过限制向除法器输入避免，保证只输入一拍valid，这可以通过把es\_allow\_in作为valid信号产生的一部分实现。另一方面，这也可以在除法器中实现忽略这些valid信号，除法部件调用除法器IP核，使用busy信号在上一个除法没有完成前，忽略除法部件新的输入valid，而不会向除法器IP转发新的valid，这样的除法器模块更完善，不会对输入有太严格的要求。

（3）对于输出，我们实例化了非阻塞的IP核，但如果我们考虑mem级可能会阻塞，这时即便我们可以立刻将hilo寄存器写入，但readygo需要在识别到输出valid之后一直不依赖后续的valid而保持为1。这实际上和输出的握手信号类似，即输出valid要和mem级allowin握手。因此除法部件中也封装了类似输出控制信号握手的规则，和除法部件外部握手，当然只需要进行控制信号握手，而不需要在握手时输出数据，因为第一拍输出valid的时候输出数据已经被写到hilo寄存器。

上面的三个功能分别需要一个寄存器，代码如下图。

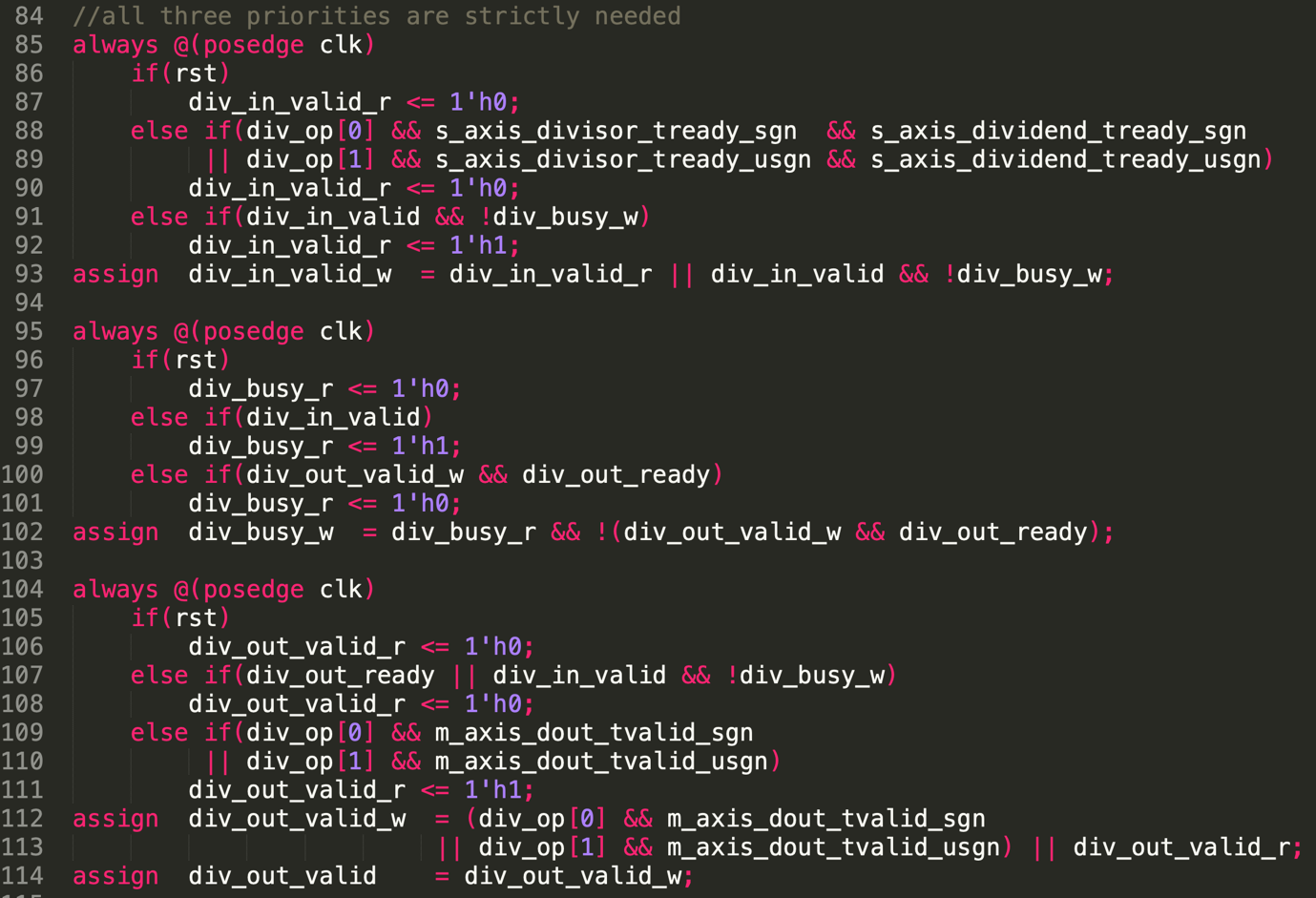


图3 实现三个功能的寄存器

三、实验过程（50%）

（一）实验流水账

10.14晚上，完成代码。

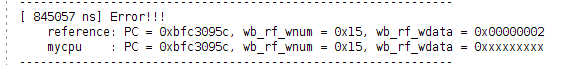
10.15上午，并调试通过。

（二）错误记录

1、错误1：MTLO指令取到X

（1）错误现象

仿真因为错误停止，如图X。



图X 仿真报错

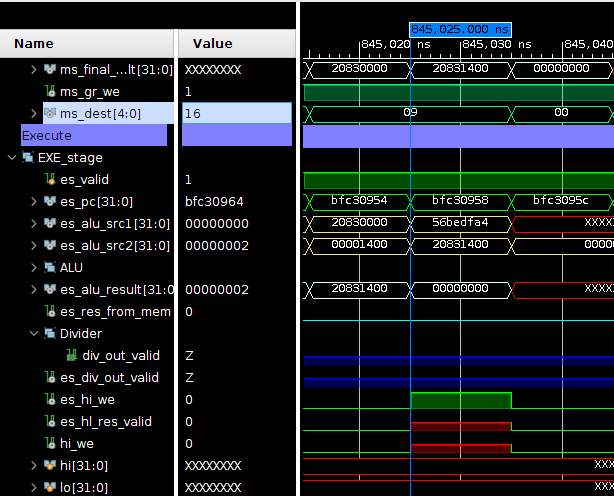
（2）分析定位过程

从反汇编文件中根据PC找到对应的指令，如图X:

https://qqadapt.qpic.cn/txdocpic/0/2d420784856592f18b47a1083bfa64e1/0

图X 出错的指令

这是一条MFLO指令，但根据图X显示的信息，写回s5寄存器的是不定值X。此处的波形如图X。在第一个Marker处，除法指令并没有向HI/LO寄存器写入结果就进入了下一条指令，见图X。所以，实际出错的指令是除法指令DIV。



图X 除法指令的波形

（3）错误原因

从图X中可以看出两点错误：第一，除法器模块的输出信号悬空，导致HI/LO寄存器的写使能信号为不定值；第二，执行流水级没有显然没有等待除法结果就继续前进了，es\_ready\_go信号一定有错误。

进一步检查（4可以发现，错误一的）修正原因效果模块代vider中忘记给d是除法器di\_valid信号赋值，错误二的原因是执行流水级ves\_read的y\_码io恒为1，没有添加\_g结果有效的除法o判断。ut

信号歩对于第一个错误，在除法器中添加了div\_out\_valid的赋值（见图）。对于第二个错误，调整了es\_ready\_go信号，需要阻塞的情况是指令为除法但结果还未算出，如图X。

https://qqadapt.qpic.cn/txdocpic/0/2e347bea9450dce9eac725b2a8aeb4bf/0

图X 调整后的执行流水级es\_ready\_go信号

作了这两处修改后，仿真

四、实验总结（可选）

1、在设计前需要仔细分析时序要求，进而可以确定使用新的控制信号还是可以复用已有的控制信号。

2、如果准备设计单独的模块，在小组合作时需先明确模块的功能，在实际功能不够明确的时候，模块的功能可以稍微高于实际要求进行设计。

3、在有较多控制寄存器转换的时候，应当考虑逻辑更加清晰的状态机设计。