组原课设中遇到的问题

1. 立即数的移位位数问题



1. PC和IR不一致

PC以字节计数，IR以双字编址

1. LEDDATA输出时a0和a7的地址找不到

逐步调试指令，找到含有a7和a0的指令，在该指令下找到这两个寄存器对应的线路，用阅读模式的小手点击线路找到地址。

1. Leddata的值不对

采用寄存器锁存

1. 两个指令的译码信号相同

添加一个新的控制信号来区分

1. 理想流水线pc值不对的问题

Pc+4需要在IF阶段完成，其他的跳转指令改变的pc需要在EX阶段完成。保证每一个阶段都有一条指令在执行，否则的话在每五个周期中有三个周期的IF/ID两个阶段的指令是不变的，因此需要先把pc+4传进去，保证每个周期都有一个pc值传入pc寄存器中。

1. 理想流水线中halt的位置

应该在WB阶段将halt的值传递给IF阶段的halt，因为需要用halt暂停整个流水线，所以halt应该在最后一个阶段的位置，否则halt后面的阶段在halt信号传递给IF阶段的时候不会暂停，只有halt前面的阶段才会暂停。

1. 单周期cpu在运行基础代码时总周期数不是1546，而是1659

开始时采用halt的非和clk的与信号得到计数器的时钟端信号，ecall信号的产生和a7与34的比较之间时有延迟的，因此halt会出现信号为1的毛刺，在clk为1的半周期里面会进行一次额外的计数，导致计数增加。本地出错而educoder上通过的原因是educoder采用的自己的周期计数方法，因此clk并未发生错误，其他的电路皆为正确的。

解决办法：将halt的非接入使能端而非原本的与门接入时钟端。

1. 打开rars



1. 在气泡流水线中，需要分支结构检测来产生时钟敏感的同步清零信号

无法使用置零位直接将其置零，因为这个是电平敏感的异步清零信号，需要使用多路选择器来选择输入的信号和0来进行同步清零。

1. 升级了流水寄存器后保存的值一直为0

寄存器使能端高电平有效

1. 气泡流水线的总周期在3623的时候并未停止

原因：在ecall指令后面还有指令，ecall在EX阶段同时满足a7==34的时候会使得halt为1，此时EX阶段的halt为1 ，但是前面的IF、ID阶段还存在指令，此时设置的流水寄存器的使能端都是常量，因此ecall在传入下一阶段的时候ecall后面的指令就会向后传，这样的话ID、IF阶段向后传了指令以后halt会再次变为0，EX-MEM-WB阶段的halt由100变为010变为001变为000，使得halt在3623时候为1之后又变成了0，从了pc只暂定了一个周期就继续再运行了。

解决办法：将halt信号与流水寄存器的使能端相联系起来，在halt为1以后流水寄存器不再向后传，这样就会停止。

1. 重定向流水线0x378周期出现问题，ID阶段的PC值不对

使能端出错，在loaduse为1的时候，需要插入气泡，这个时候需要前面IF阶段和ID阶段暂停，因此IF/ID流水寄存器的使能端应该为暂停，因此该处的逻辑需要loaduse。

1. 单周期CPU单级中断时的PC不对

所要保存的断点应该是下一条指令的地址，而不是当前指令的地址。

1. 单周期CPU单级中断时的中断请求不对

单级中断指的是无法嵌套，但是会响应所有的中断请求，因此在执行过程中，需要记录所有的中断请求，此时先考虑时间先后上的请求信号，在同一时间的中断请求再考虑优先级，那么在中断请求信号清除的时候也是同样的要求，因此需要记录当下正在执行的中断请求的信号，在后续清楚的过程中应该按照该信号进行清除，而非优先级。

错误原因：清除中断请求信号时按照优先级进行清除的。

更改：利用中断响应信号保存当前正在进行中断服务的编号。

1. 多级中断中中断嵌套返回原中断程序出问题

与单级中断不同，无法利用优先编码器简单的选择返回的断点位置，可以采用保存断电地址的压栈方式把中断号也压栈，返回的时候出栈即可。

1. 多级中断的中断使能位的问题

多级中断在中断发生后的所要保存的数据保存了以后需要开中断，并且在后面恢复数据的时候要关中断，因此需要在中断执行的过程中保持一段时间的开中断，在单级中断的基础上需要增加一些操作使得中断使能位在这段过程中保持，如果只采用CSRRCI和CSRRSI这两个信号的号只能维持一个周期，但是需要保持多个周期让中断处理程序执行完，因此可以采用寄存器并用CSR信号作为使能端，便可以在CSRRCI和CSRRSI之间保持信号，但是这样的话开中断会有一个周期的延迟，因此需要寄存器保存和该信号直接传递两种方式结合采用或门完成。同样的对于关中断也有一个延迟，此时清除中断使能位可以采用与门清空。

1. 流水线中断中走马灯显示的不是0XFFFFFFF1而是0x00000001

跳转的指令地址有问题，导致有些指令没有执行使得最后的数据产生错误。在原本的选择保存的断点地址上，是利用loaduse信号进行判断的，但是有一个中断信号在第一个周期就产生了，此时loaduse信号为0并且IF、MEM、WB阶段的指令都尚未传递至该位置，因此就会造成错误判断，此时保存的应该是ID阶段的地址，因此可以采用EX阶段产生的信号作为多路选择器的选择位，这样既可以保存原有的功能，因为IR被清除了以后信号与要跳转的地址所需要的信号是相同的，还可以解决后面阶段尚未传达指令的问题。

1. 流水线中断跳转的地址有问题

在uret指令执行的同时给予了中断信号，在原有的电路逻辑中存在的问题为，断点地址的选择只从ID的pc和EX执行后的选择结果来选择了，但是uret指令并不会产生一般的跳转指令所产生的信号，并且此时ID阶段的值会被清空，因此要保存的断电地址就会发生错误变为0，此时需要保存IF的pc值作为断点地址，因此需要额外增加IF和ID的pc的选择的电路。

1. 流水线多级中断的过程中多次相同中断号产生中断信号但是只发生了一次中断

在1-2-3-3-1的多级中断信号产生时，测试的结果应该是3-2-1-3-1，但是在执行过程中，由于1级中断在执行过程中产生了时间间隔很近的3和1中断信号，会导致1级在执行过程中对3级进行响应，在3级中断信号处理的过程中，新的1级中断信号也产生了，这样的话在原有的中断信号处理逻辑下，3中断信号处理程序执行完了之后会返回原有的1号中断程序的断点处，在uret之后会清空1号中断信号，这样的话后来的新的1号中断信号也不会被响应和处理。

解决办法：对中断信号进行计数，产生中断信号加一，该信号对应的中断处理程序uret一次就减一，通过该计数器的计数值来确定该中断信号是否存在。

1. 动态分支预测中周期数并未明显减少

第一个问题是解复用器和译码器的问题，在选择位为xxx的时候对应的所有输出都是不确定的，这个时候就会造成判断错误，一个重要的错误是写使能受到了影响，导致分支指令并未正常的写入，使得后面的历史预测位始终都是0，造成了和不使用该技术一样的效果。

第二个问题是异步清零的问题，在清空置换位的时候，采用了异步清零的做法，使得该值在不正确的时候清零，造成了后面写入分支指令时的判断错误，使得频繁被使用的条目被置换，造成周期数极大增加的问题。解决办法是加一个D触发器使其同步。

1. 动态分支预测中pc不对

一开始是忘记考虑动态分支预测下的IF阶段的pc+4并非跳转指令后面的指令，因此在找跳转指令的顺序地址时出错了；另一个是只考虑了在不应该跳转的时候跳转了的错误行为，忘记考虑在应该跳转时没有跳转的错误行为，导致pc的下一个值出错了。

解决办法：改变原有的重定向的pc选择方法，重新加入BRANCHADDR找到下一条指令。

1. Logisim转换为verilog所遇到的问题

第一个：分线器必须是连续值

第二个：所有端口都要有输入，这很重要

第三个：分线器不要有空闲的端口

第四个：GO和RST最好设计成引脚而不是按钮，方便verilog中对应引脚

第五个：乘法器和除法器去除掉，寄存器文件也去除掉

第六个：标签命名一定要符合标准

第七个：每一个元件都要观察三态

1. Verilog中的问题
2. 寄存器初始化，在fpga平台上更好操作
3. CPU和fpga需要使用两个频率，CPU中的频率降低让指令执行的不那么快方便观察，fpga中的频率要快一些使得所有的数字同时显示。
4. 移位器的运算右移需要使用有符号数。
5. 减法器中的减法运算应该用补码而不是反码。
6. RAM的数据位宽错误
7. 约束文件的引脚问题，fpga自带的频率即可，不需要再加一个频率输入

解决办法：加一个测试信号testbench逐周期逐信号调试