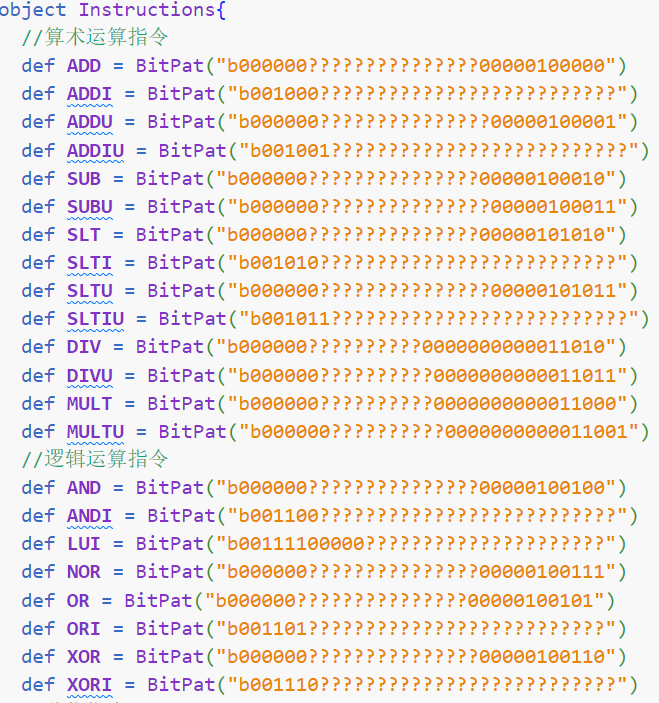
**RISC-V更改**

更改主要围绕Control.scala进行，目的是为了使其适用于MIPS指令集，在更改过程中需要对其他模块进行改进，遂有此篇文章来简单介绍目前的更改以及尚未解决的问题。

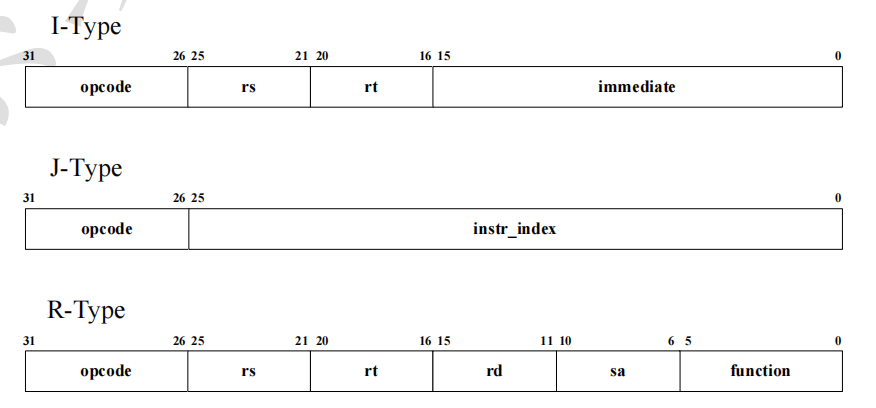
1. **一些变化**

**1.1 指令格式的变化**

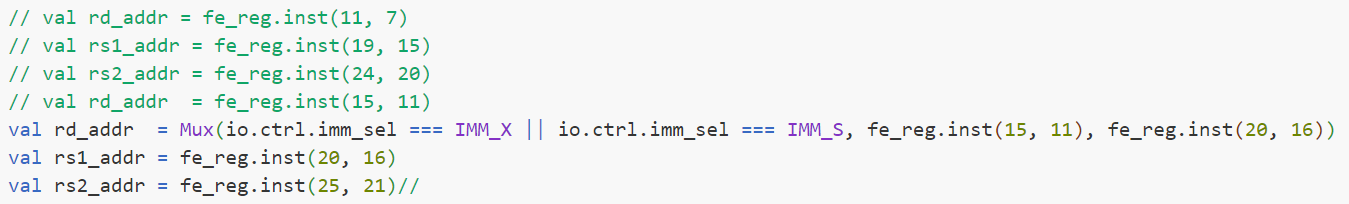
RISC-V和MIPS的指令格式并不完全相同，为了程序可以正确识别MIPS指令，所谓的识别功能Instructions.scala中进行实现，因此需要将Instructions.scala中的内容作出改动，改动如下：



有关MIPS指令集的格式，详情参见龙芯杯大赛资料nscscc-group\doc\_v0.01\A03\_“系统能力培养大赛”MIPS指令系统规范\_v1.01.pdf,此处给出三种指令格式截图：



除了在指令识别处的变化，由于指令格式的变化，我们获得立即数、源寄存器、目的寄存器的方法也应该有所变化，因为它们在指令中的位置、长度均有所不同，这一变动主要在Data.scala中体现：



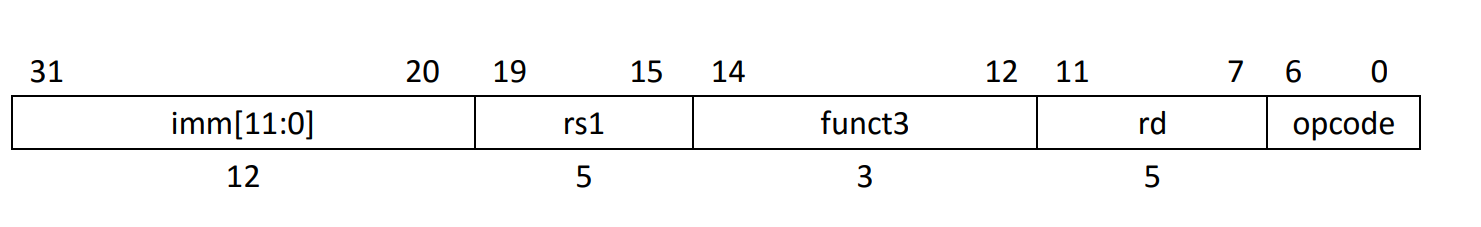
其中注释部分为原代码，下面为更改后的代码；上述仅展示了一处更改，在文件中仍有其他更改，如有遗漏or错误欢迎指出

**1.2 立即数生成方式的变化**

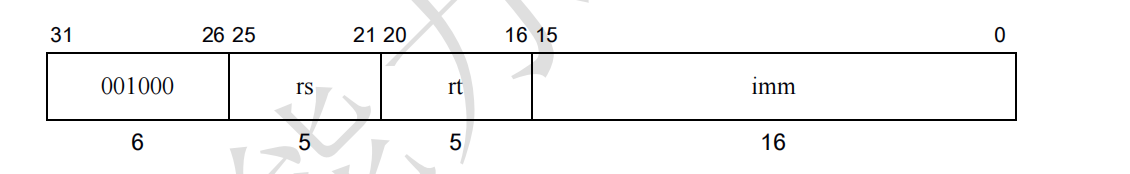
这一变换产生的原因很大一部分也是指令格式变换所导致的。

举个栗子：立即数加法指令

RISC-V指令格式如下：立即数为高12位



MIPS指令格式如下：立即数为低16位

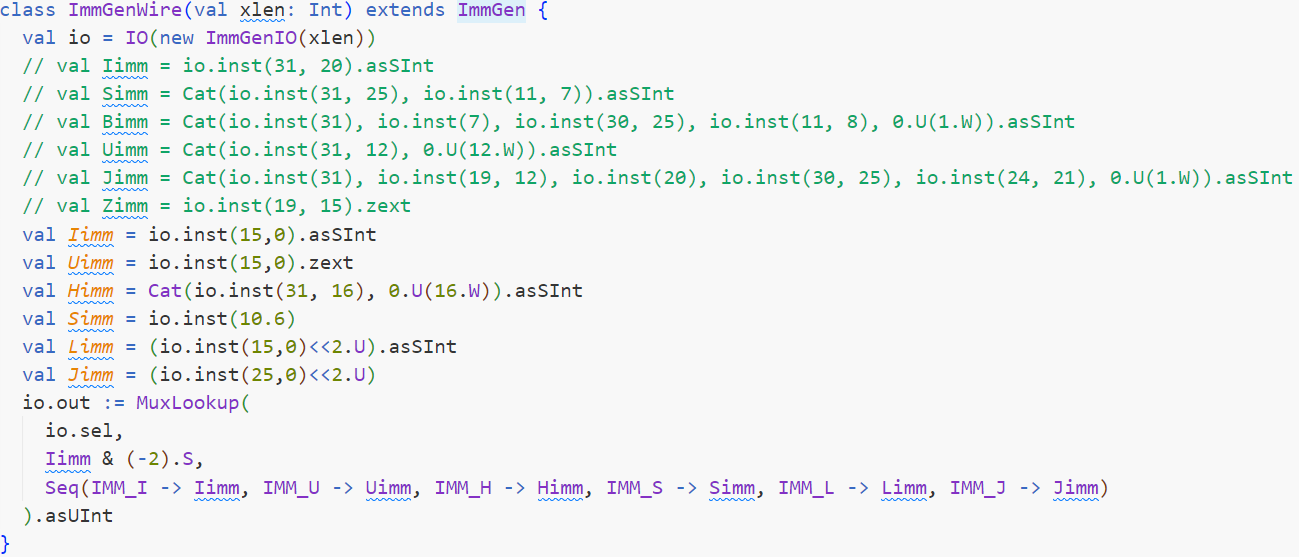


除了指令格式变化之外，还存在着同样的指令扩展方式不一样（零扩展与符号扩展）等问题，为了解决上述问题，我将原有的立即数格式作了重写，主要体现在ImmGen.scala与Control.scala模块，在后者中给出了指令格式的解释：

|  |
| --- |
| Plaintext  val *IMM\_X* = 0.U(3.W)//不构造立即数  val *IMM\_I* = 1.U(3.W)//有符号扩展至 32 位的立即数  val *IMM\_U* = 2.U(3.W)//零扩展至 32 位的立即数  val *IMM\_H* = 3.U(3.W)//16位高位为该16位立即数 16位低位填零  val *IMM\_S* = 4.U(3.W)//用于移位指令中的SA立即数  val *IMM\_L* = 5.U(3.W)//左移 2 位并进行有符号扩展  val *IMM\_J* = 6.U(3.W)//26位立即数左移两位 留作转移地址的处理 |

上述立即数格式可以参照nscscc-group\doc\_v0.01\A03\_“系统能力培养大赛”MIPS指令系统规范\_v1.01.pdf中指令格式进行理解。

在ImmGen.scala中给出了相应格式立即数的生成方式

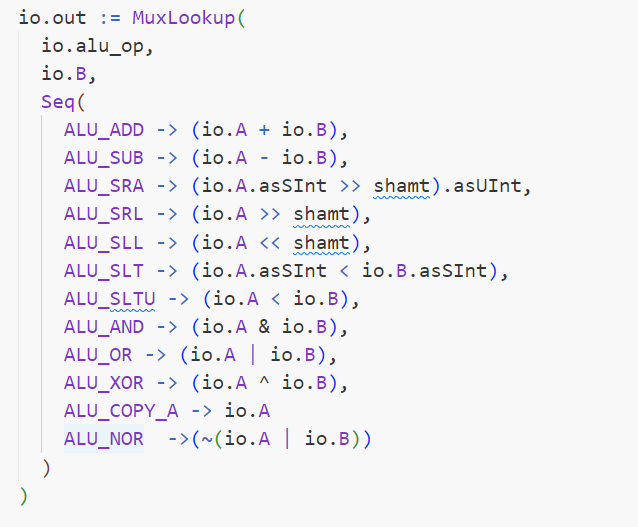


注：此处仅重写了一种方式，下面的ImmGenMux尚未作更改

**1.3 ALU功能的补充**

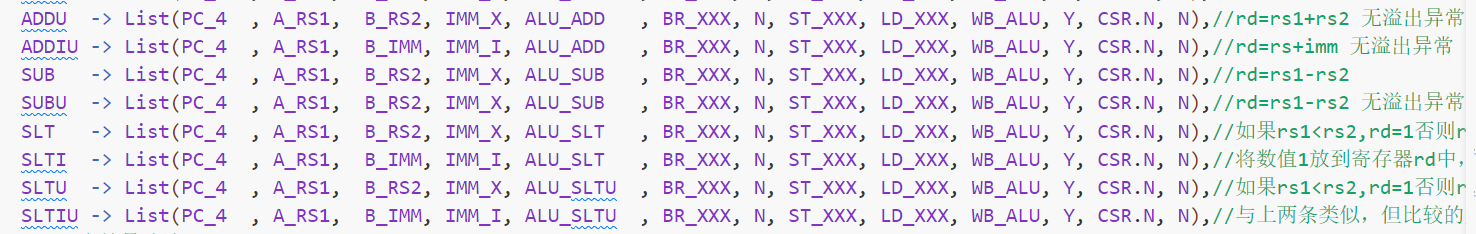
为ALU新增了NOR运算，为了适应NOR指令





这里也只是重写了AluSimple里的方法

**1.4 控制模块**

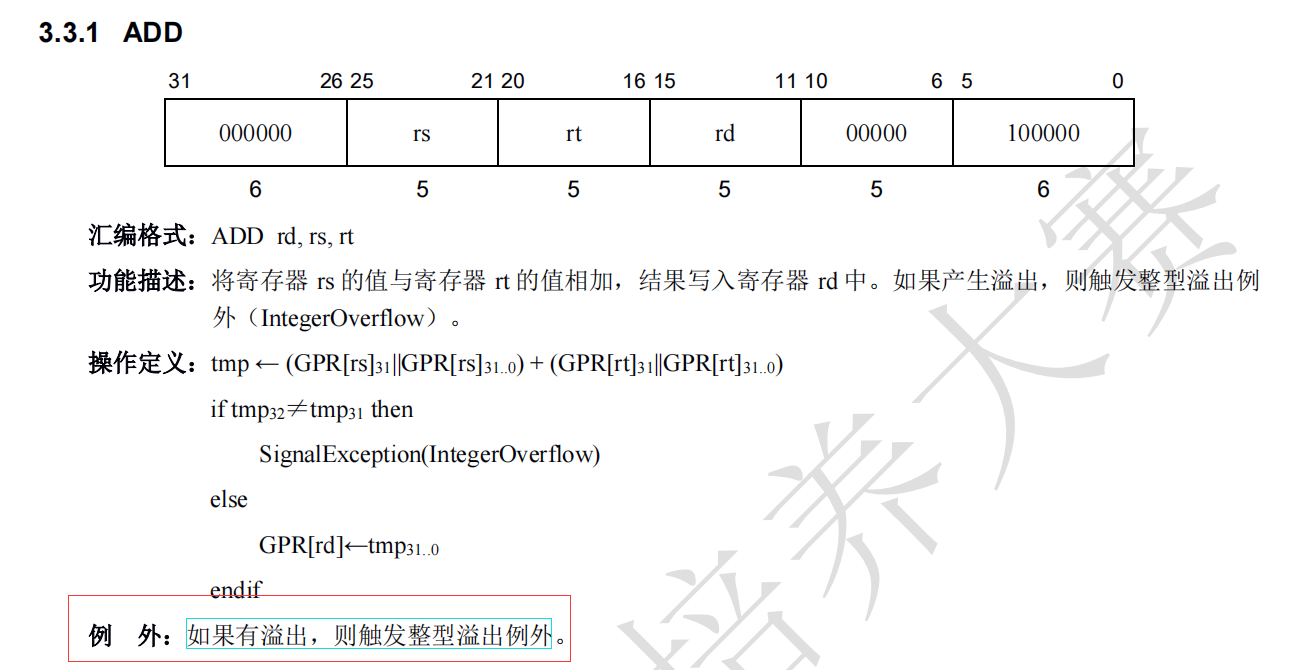


对控制模块指令发出的信号做了相应更改，由于后面的问题，并未完成全部改写，一些指令也只是搬了过来，尚未更改

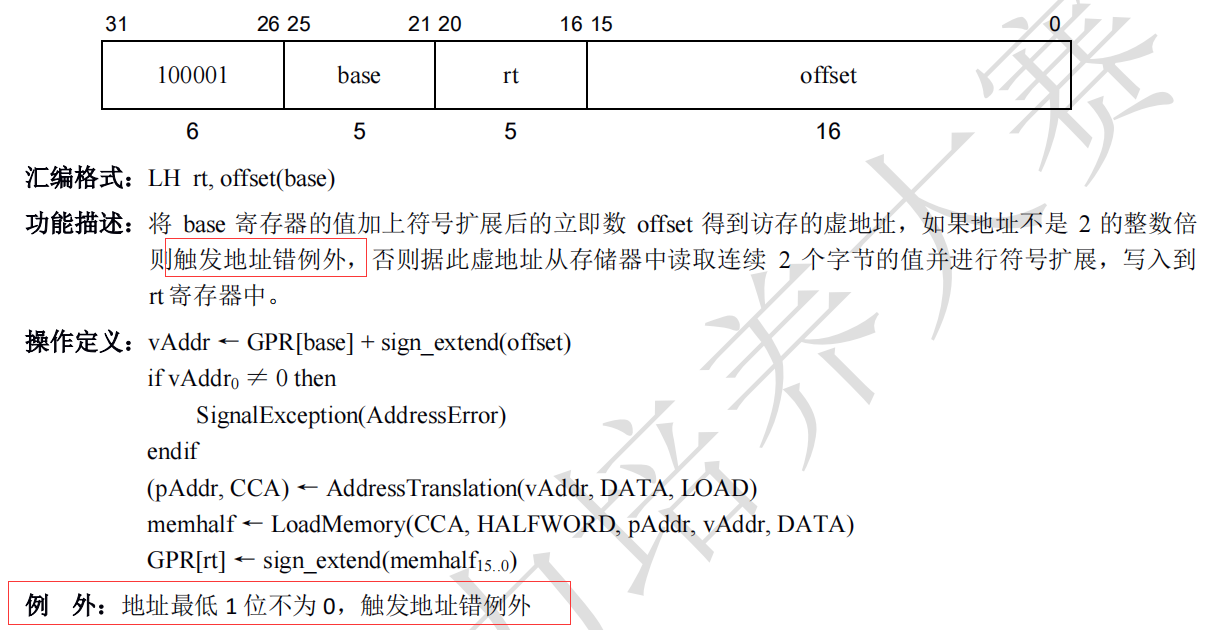
**2.一些问题**

**2.1 自陷**

部分指令带有例外情况，如ADD指令可能出现运算溢出例外



访存指令同理

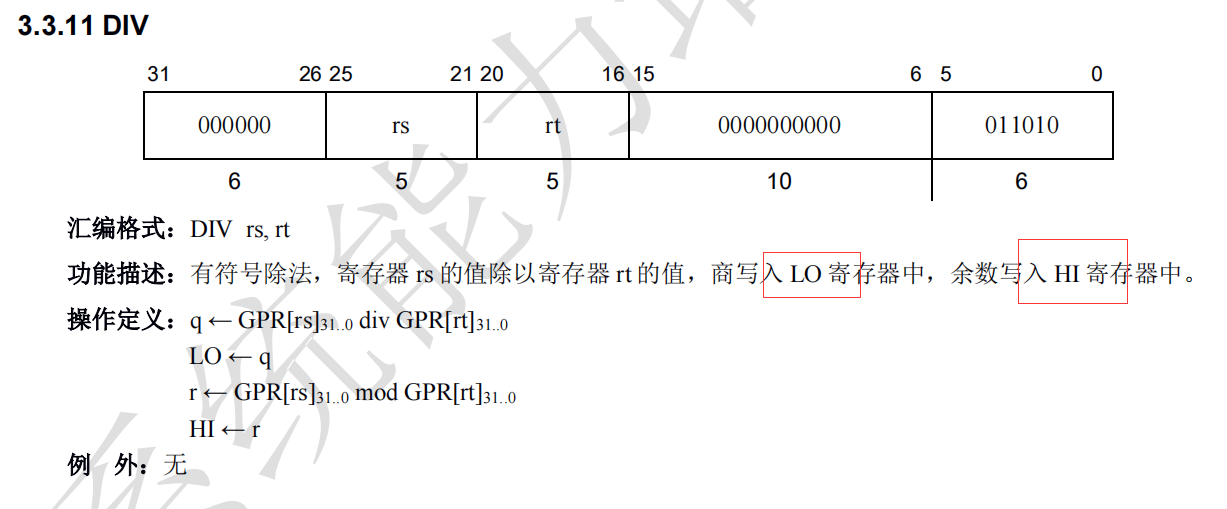


此外还包括中断指令，也尚未理解和解决

**2.2 出现了特殊的寄存器**

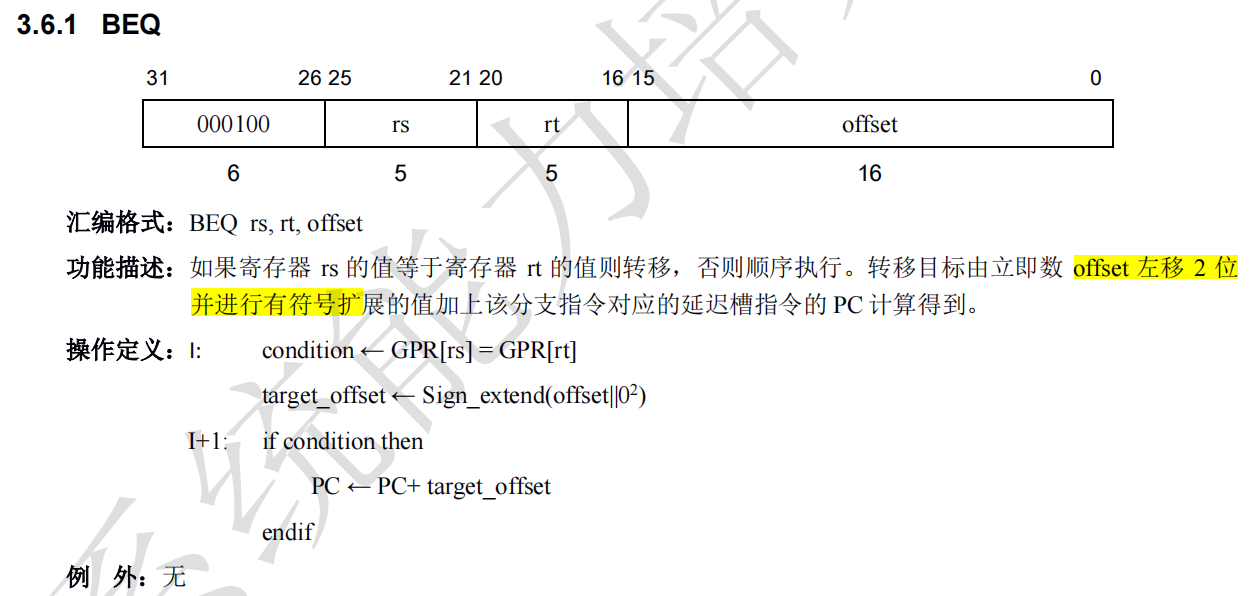
在乘法与除法指令中，出现了HI与LO两种特殊寄存器，在RISC-V中并未得到体现

（数据移动指令中也用到了这两种寄存器）



**2.3 延迟槽问题**

在分支跳转指令中，几乎所有指令都需要用到延迟槽



**2.4 特权指令**

特权指令尚未完成