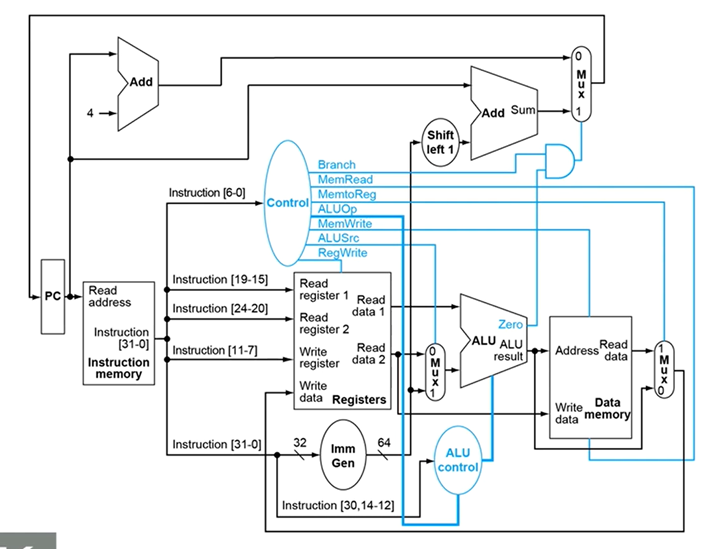
**计组小组学习——RISC-V 4.1~4.4**

**实验：指令译码器**

**一、 实验目的与任务**

实现RISC-V单周期CPU的指令译码模块。包括控制单元、立即数生成器、两个源寄存器端口、一个目标寄存器端口。



具体包括Control模块、Imm Gen（立即数生成器）模块和输入输出寄存器。

通过本实验我们可以：

1. 理解RISC-V的指令结构，理解RISC-V中常用指令的功能和编码。

2. 进一步了解单周期CPU的结构与设计。

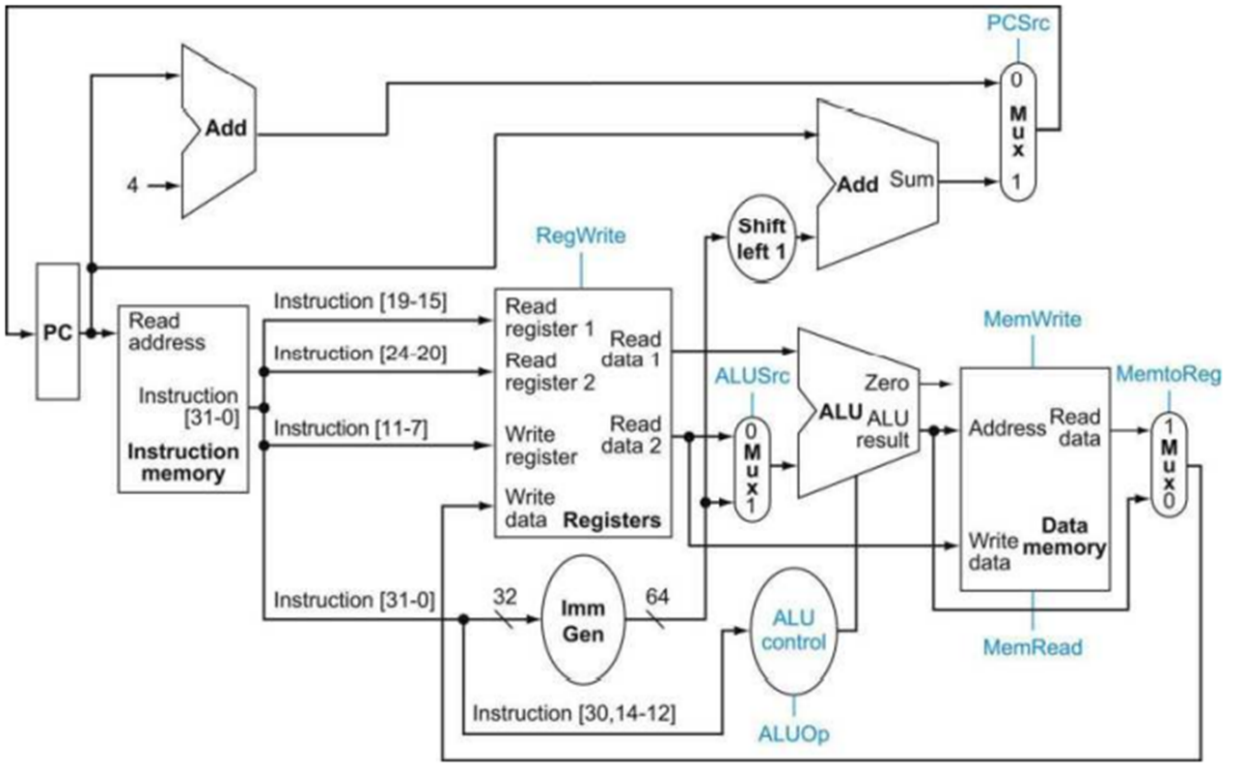
3. 深入理解计算机如何处理指令，加强我们对数据通路以及CPU内控制单元部分的理解。

4. 加强对Verilog语言的应用能力。

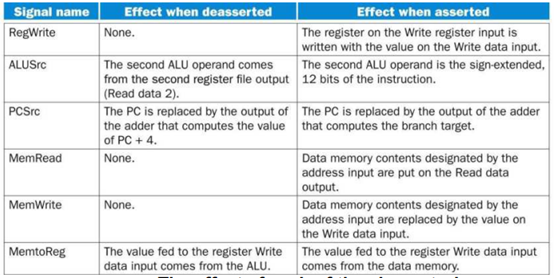
**二、实验原理**

前置知识已在课堂讲授，在此不再罗列。

如图所示的数据通路只剩下最后一步，即主控制单元的设计（图中蓝色标明的信号）



根据每个信号的功能分析，需要实现的功能如下：



现在请利用你学到的知识，实现译码和符号拓展功能。

**三、实验要求**

1.使用Verilog，根据真值表设计对应组合逻辑电路，编写Control模块，实现指令到控制信号的转换。

2.使用Verilog，编写Imm Gen模块，实现符号拓展。

3.编写testbench进行测试，验证以上模块是否正确。

4.撰写实验报告，记录实验过程及实验结果，分析指令译码器在CPU结构中的作用。

**四、注意事项**

1.存/取指令和条件跳转指令的立即数在指令中的字段不同，立即数生成器要对不同类型的指令分别处理。

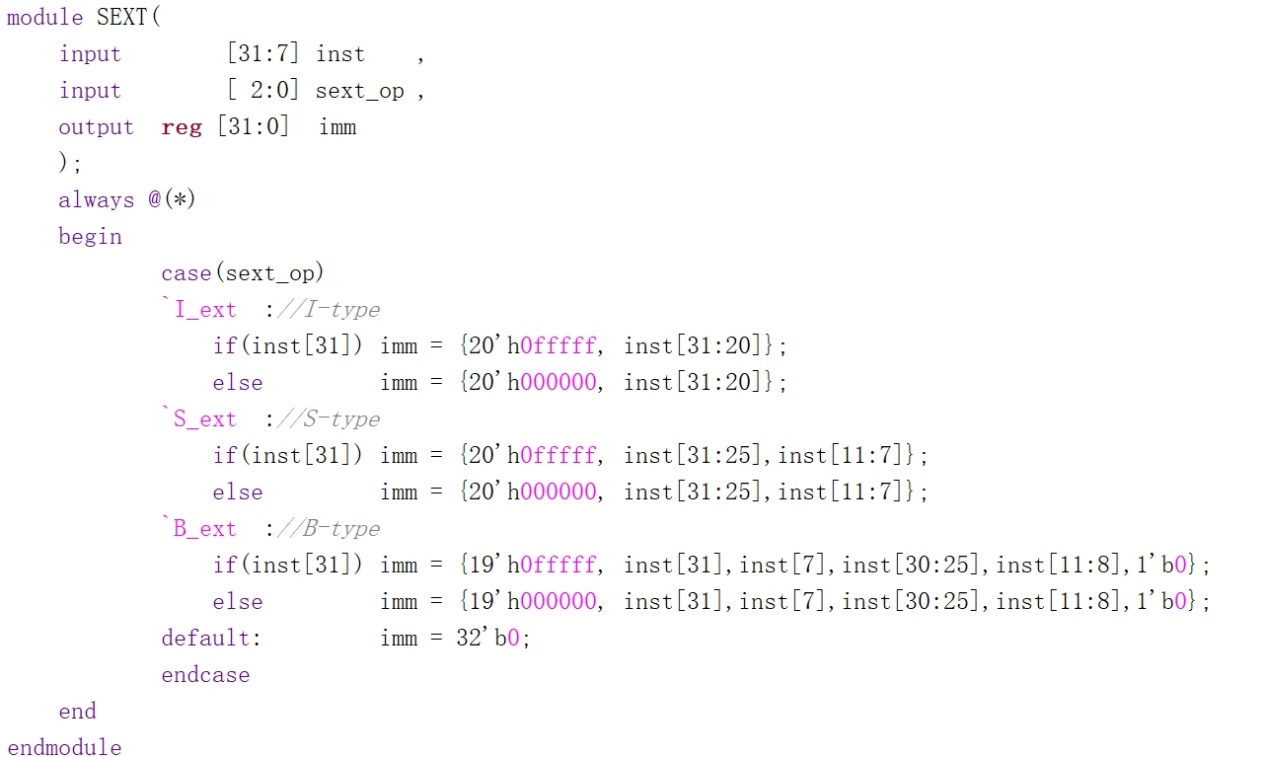
2.本实验不需要实现寄存器堆、数据存储器等模块，只需要输出对应的信号值、寄存器操作数的值、符号拓展后的值即可。

**五、参考思路**

1.对于每个信号，可以根据真值表建立逻辑门电路实现。

2.对于寄存器操作数，根据其在不同指令中的字段直接输出即可。

3.对于符号拓展，需要设计相应的模块，可以参考如下代码：



另外，我们附上了一种chisel实现，详见文件中的ID.scala。