Lab1 实验报告

PB18051098 徐碧涵

实验说明

本次实验是 RV32I Core 设计的铺垫。参考提供的 RV32I Core 设计图，理解每条指令需要的数据通路，以及相应的控制信号并回答相应问题。

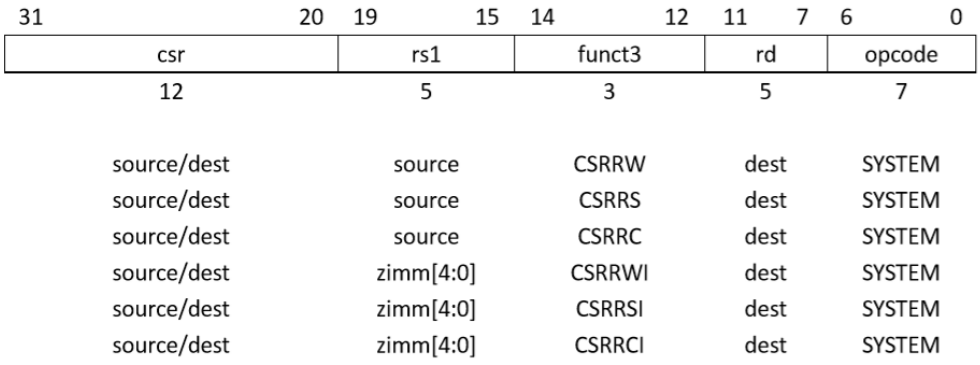
实验内容

1. 描述执行一条 XOR 指令的过程（数据通路、控制信号等）。
   控制信号：Alusrc1 和 Alusrc2 均选通寄存器的值，Alucontrol 选择 xor 操作，LoadNpc 选择 AluOutM（alu 计算结果），MemToReg 无效使得 ResultM（alu 计算结果）写回寄存器，RegWrite 信号有效，其他信号无效

   数据通路：在 ID 段得到寄存器 rs1 和 rs2 存储的值，进入 EX 段选择两个寄存器的值进行 xor 操作得到计算结果，在 MEM 和 WB 段均选择计算结果最后写回 rd 寄存器。
2. 描述执行一条 BEQ 指令的过程（数据通路、控制信号等）。
   控制信号：ImmType 为 SB-Type，BranchType 信号有效，Alusrc1 和 Alusrc2 均选通寄存器的值进入 Branch Decision 产生 BranchE 信号，其余控制信号均无效

   数据通路：在 ID 段得到寄存器 rs1 和 rs2 存储的值，并通过 PC + 4 + Imm 得到分支跳转地址 BrNPC，进入 EX 段对两个寄存器的值进行比较得到信号 BranchE 表示分支是否发生，利用 BranchE 控制 NPC Generator，如果 BranchE 有效，那么将选择 BrNPC 作为 npc。
3. 描述执行一条 LHU 指令的过程（数据通路、控制信号等）。
   控制信号：ImmType 为 I-type，Alusrc1 选通寄存器值，Alusrc2 选通立即数 Imm，Alucontrol 选择加法操作，LoadedBytesSelect 选择 16 位（0 扩展到 32 位），MemToReg 信号有效选择 memory 读出的数据写入目的寄存器，RegWrite 有效。

   数据通路：在 ID 得到 rs1 和 imm 的值，进入 EX 段经过 ALU 计算（（rs1）+imm）得到 memory 地址，进入 MEM 段根据计算得到的地址去 memory 取出值，然后进入 WB 段经过 DataExt 进行位数选择并扩展后，写回寄存器
4. 如果要实现 CSR 指令（csrrw，csrrs，csrrc，csrrwi，csrrsi，csrrci），设计图中还需要增加什么部件和数据通路？给出详细说明。



IF：不需要增加部件

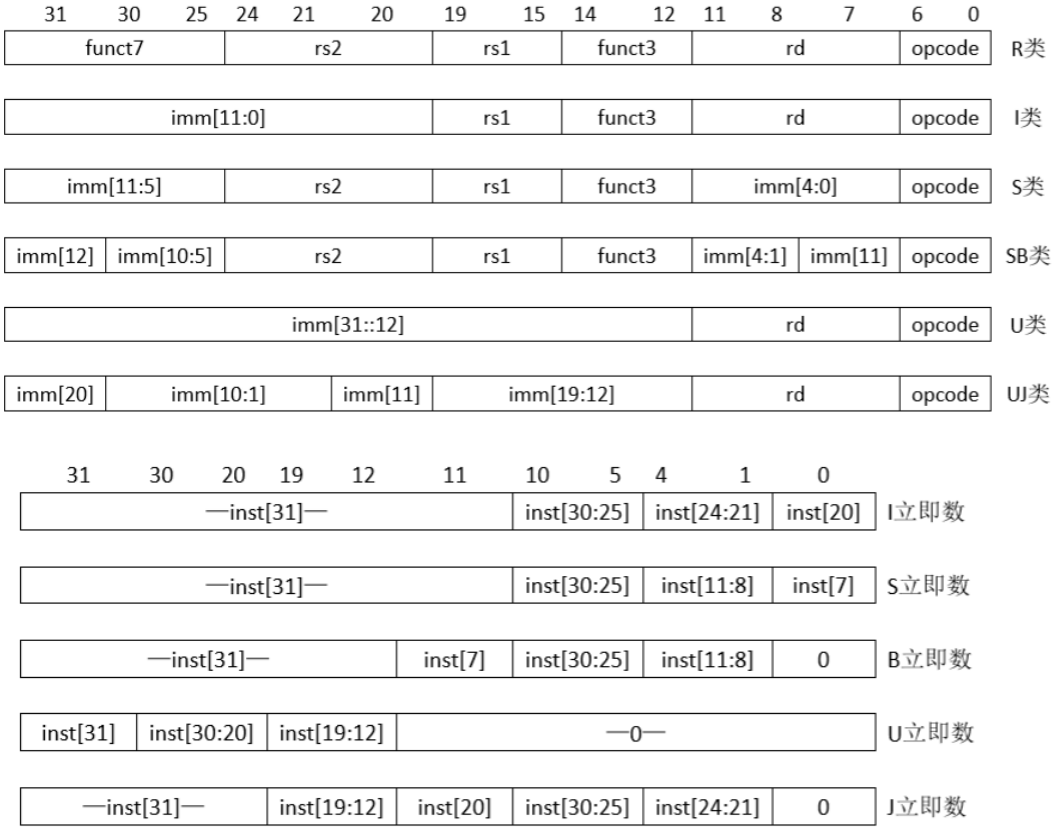
ID：添加 CSR 寄存器文件；完善立即数扩展模块使其能够将 csr 值 0 扩展到 32 位并且能够将处于 rs1 字段的 5 位立即数零扩展到 32 位的，控制单元需要额外生成 CSR 读写使能信号；将符号扩展后的 csr 值以及 rs1 寄存器中的值（或扩展后的 5 位立即数值）送入 EX 段寄存器；

EX：在 Alusrc2 控制的数据选择器处加入 csr 值，Alusrc1 控制的数据选择器添加 0 值与 csr 值相加（Alucontrol 选择加操作），将 ALU 运算结果送入 MEM 段寄存器

MEM：选择将运算结果送入 WB 段寄存器

WB：将 result 写回 rd 寄存器中，根据 funct3 字段将 rs1 中的值（或立即数）写入 CSR 或将对 result 值进行掩码操作后得到的值写入 CSR

5. Verilog 如何实现立即数的扩展？



由各包含立即数指令的格式来看，可知应该按如下方式进行扩展：

用 sigext（imm）表示对立即数按最高位进行符号扩展

I-Type：sigext（inst[31:20]）

S-Type: sigext({inst[31:25],inst[11:7]})

B-Type:sigext({inst[31],inst[7],inst[30:25],inst[11:8],1'b0})

U-Type:sigext(inst[31:12]<<12)

J-Type:sigext({inst[31],inst[19:12],inst[20],inst[30:21],1'b0})

SB-Type:sigext({inst[31],inst[7],inst[30:25],inst[11:8],1'b0})

UJ-Type:sigext({inst[31],inst[19:12],inst[20],inst[30:21],1'b0})

6. 如何实现 Data Memory 的非字对齐的 Load 和 Store？

Load：将非字对齐的地址向上按字对齐（按值更小的对齐地址）后，取出连续的两个字，然后根据实际地址从两个字中拼接出一个字

Store：将要存的字按地址（mod4 选择拆分位置）进行拆分并扩展成两个字，将两个字连续的存放到非字对齐向上对齐后的地址中

7. ALU 模块中，默认 wire 变量是有符号数还是无符号数？

无符号数

8. 简述 BranchE 信号的作用。

表示是否满足分支条件，若满足分支条件，那么 BranchE 信号选通，使得 NPC Generator 选择 branch 目标地址作为 npc

9. NPC Generator 中对于不同跳转 target 的选择有没有优先级？

优先级 Branch，jalr > jal， 因为 branch， jalr 控制 NPC Generator 时处于 EX 段，jal 指令处于 ID 段，故当 branch/jalr 指令与 jal 指令同时控制 NPC Generator 时，由于处于（不考虑乱序执行）EX 段的指令位于处于 ID 段指令之前，故应该考虑 branch 和 jalr 指令。

10. Harzard 模块中，有哪几类冲突需要插入气泡，分别使流水线停顿几个周期？

Load-use 冲突：停顿一个周期

无条件分支 jal：停顿一个周期

无条件分支 jalr：停顿两个周期（在 EX 段才完成跳转地址计算）

分支跳转指令分支 taken：停顿两个周期（在 EX 段才能判断分支是否发生）

11. Harzard 模块中采用静态分支预测器，即默认不跳转，遇到 branch 指令时，如何控制 flush 和 stall 信号？

Branch 指令在 EX 段才能够确定分支跳转是否 taken，此时其后两条指令正分别位于 ID，IF 段，若跳转发生，那么需要将其后两条指令 flush 掉，即（在下一个时钟上升沿使得无法写入 ID，EX 段寄存器，从而使得处于 IF，ID 段的指令无法继续执行）flush 掉 ID，EX 段寄存器，若分支跳转未发生，那么不需要进行 flush 或 stall。

12. 0 号寄存器值始终为 0，是否会对 forward 的处理产生影响？

在产生 forward 控制信号时需要判断如果前面指令的目的寄存器与当前指令的源寄存器是 0 号寄存器时，并不会选择转发 ALU 计算结果，而是选择 0（即 0 号寄存器的值 0）