lab01实验报告

1. 描述执行一条XOR指令的过程（数据通路，控制信号等）**。
    - IF阶段：根据PCF在Instruction Memory 中取出xor指令。
    - ID阶段：instr[24:20],inst[19:15], 翻译为rs2和rs1传到Register FileA1，A2端口，读出RegOut1D,RegOut2D 。 instr[11:7] 翻译为rd 。 Instr[31:25] , instr[14:12] ,instr[6:0] . 传到control unit 。

      产生控制信号 。(RegWrite = 1 有效 , Alusrc1D=1 , Alusrc2D = 00,AluControlD = 100)
    - EX阶段： 执行的是 AluOutE = RegOut1E XOR RegOut2E 运算 。
    - WB阶段： RdW接 A3端口 。 RegWriteW接RegWrite实现。 将ResultW写入RdW位置

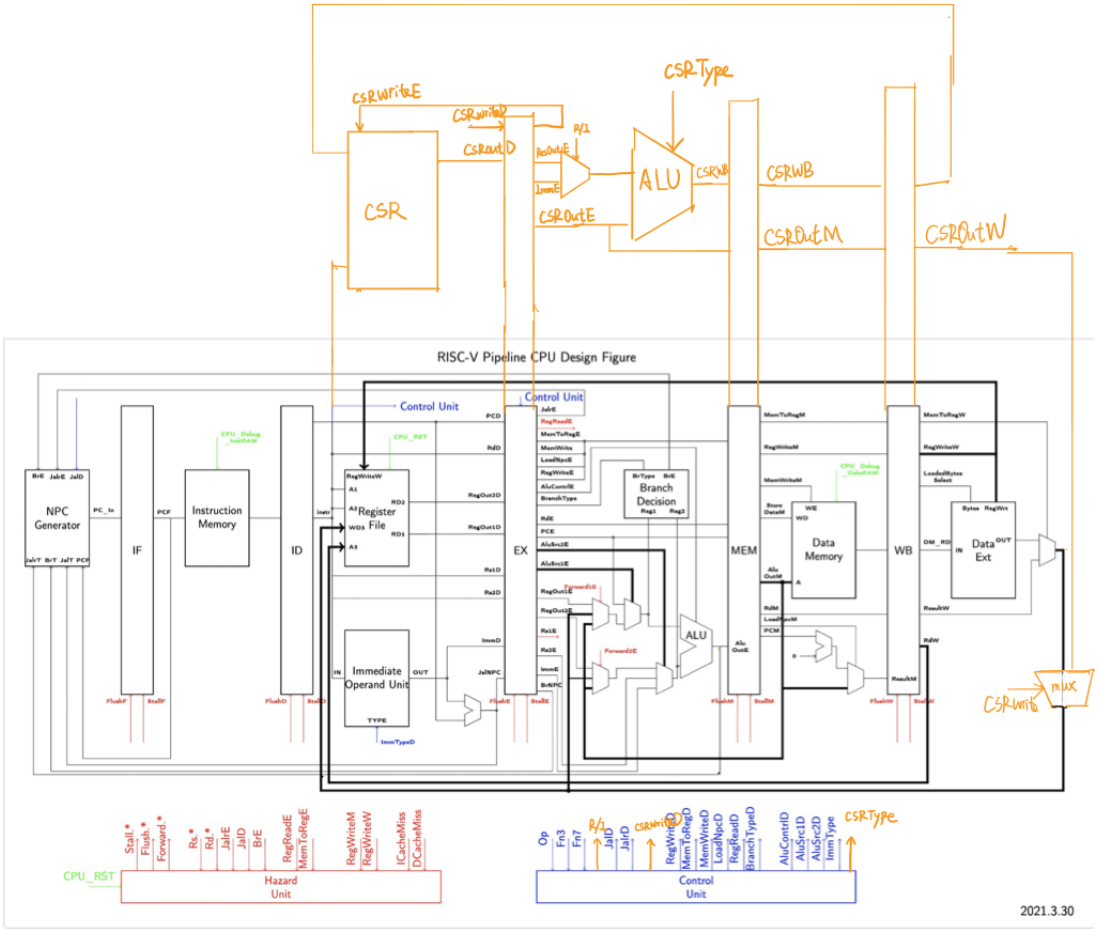
2. **描述执行一条BEQ指令的过程（数据通路，控制信号等）。**
    - IF阶段：根据PCF在Instruction Memory 中取出beq指令。
    - ID阶段：instr[24:20],inst[19:15], 翻译为rs2和rs1传到Register FileA1，A2端口，读出RegOut1D,RegOut2D 。 instr[14:12] ,instr[6:0] . 传到control unit 。 并将立即数由Immediate Operand Unit处理好，计算出JalNPC。

      产生控制信号 : (BranchTypeD = 000 , Alusrc1D=1 , Alusrc2D = 00 ,)
    - EX阶段： 执行的是 由BrType信号 和Reg1和Reg2 在 Branch Decision单元决定是否跳转。同时BrNPC连接到BrT。

3. **描述执行一条LHU指令的过程（数据通路，控制信号等）。**
    - IF阶段：根据PCF在Instruction Memory 中取出beq指令。
    - ID阶段：inst[19:15], 翻为rs1传到Register File 的A1端口，读出RegOut1D。 instr[11:7] 翻译为rd 。并将instr[31:20]作为立即数，由 Immediate Operand Unit处理好。

      instr[14:12] ,instr[6:0] . 传到control unit 。

      产生控制信号 : (RegWriteD = 1 有效 ,MemToRegD = 1 有效， Alusrc1D=1 , Alusrc2D = 10 ,AluControlD = 000)
    - EX阶段： 执行的是 AluOutE = RegOut1E + ImmE 运算 。
    - MEM阶段： 读出内存中地址为AluOutM的数据
    - WB阶段： DM_RD 写回到寄存器RdM中 。
4. **如果要实现CSR指令（csrrw，csrrs，csrrc，csrrwi，csrrsi，csrrci），设计说明还需要增添什么部件和数据通路？可以进行详细说明。**



- 增加数据通路大致如上图橙色部分。
- Control Unit添加控制信号：`R/I` 区分是以寄存器为源，还是以立即数为源。`CSRType`：2'b00：表示非CSR指令。2'b01：表示CSRRW系列， 2'b10:表示CSRRS系列，2'b11表示CSRRC系列 。`CSRwriteD` ：是CSR的写使能信号。

工作原理：

- **ID阶段：** 产生控制信号， 并取出两个源操作数，CSROutD， 和Regout1或immD 。
- **EX阶段**： 根据控制信号， 在计算单元计算要写回CSR值CWD。
- **WB阶段：** 将 并且将读出的CSR值写回到目标寄存器， 将CWD写回CSR 。

5. **Verilog如何实现立即数的扩展?**

```
always@(*)
begin
    case(Type)
        ITYPE: Out<={ {21{In[31]}}, In[30:20] };
        STYPE: Out<={ {21{In[31]}}, In[30:25], In[11:7]};
        BTYPE: Out<={ {20{In[31]}}, {In[7]}, In[30:25], In[11:8], {1'b0} };
        UTYPE: Out<={ In[31:12], {12{1'b0}} };
        JTYPE: Out<={ {12{In[31]}}, In[19:12], In[20], In[30:21], {1'b0} };
        RTYPE: Out<=32'hxxxxxxxx;
        default:Out<=32'hxxxxxxxx;
    endcase
end
```

6. **如何实现数据存储器的非字对齐的加载和存储?**

   加载:

   - 先按字加载数据。DataMemory addra 传入 `{A[31:2],{2'b00}}` 。
   - AluOut低两位存入 LoadedBytesSelect，用于决定需要写回的数据的对齐方式 。
   - 再在 DataExt 模块选择写回的数据。

   存储:

   - DataMemory addra 传入 `{A[31:2],{2'b00}}`。对字寻址。
   - A[1:0] 决定非字对齐的偏移

7. **ALU模块中，默认wire变量是有符号数还是无符号数?**

   默认的wire变量是无符号数。

8. **简述BranchE信号的作用。**

   用于决定NPC Generator的源操作数。当 BranchE 为 1 时, PC_In = PCF + BrT 。

9. **NPC生成器中对于不同加速度target的选择有没有优先级?**

   有优先级。

   JalD 优先级小于JalrE 和BrE，因为信号产生的阶段不同。 若有如下指令:

   ```
   beq a1,a2,4  IF ID EX(BrE) MEM WB
   jal a3,4        IF ID(JalD)     EX  MEM WB
   ```

   同时产生了`BrE` 和`JalD` 信号。则需要先执行BrE有效 。

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?**

一类。

load 指令与紧跟它的指令有写后读相关。

停等一个周期, 再通过转发消除解决冲突。

11. **Harzard模块中采用静态静态预测器，即默认不复位，遇到分支指令时，如何控制flush和stall信号?**

    如果默认不跳转。

    若遇到Branch指令：若 branchE==1, 此时需要跳转 。 将已经读入的Branch后的指令flush掉 。 具体操作，将 IF/ID、ID/EX 段寄存器的 Stall 置0、Flush 置 1。

12. **0号寄存器值始终为0，是否可以转发的处理产生影响?**

    在 z0 的写后读相关上会产生影响。

```
add Z0,a1,a2              IF ID EX(产生z0!=0中间值) MEM WB(Z0=0)
sub a3,Z0,a2                 IF ID             EX(使用Z0)  MEM WB
```

    如果进行转发处理 ，转发的中间值不是零 。 产生错误结果。