Verilog-Lab1 Report

梁峻滔 PB19051175

Q & A

1. **描述执行一条 XOR 指令的过程**

   IF段：根据 IFPC 的值到 Instruction Cache 中取出 XOR 指令，存入IR

   ID段：Controller Decoder 根据指令的Opcode、funct3、funct7字段值设置以下信号：

ALU Func	WB Select	Op1	Op2	RegWrite
异或操作	选择ALU RESULT	选择Op1 Forwarding MUX	选择Op2 Forwarding MUX	有效

   同时根据指令src1、src2字段的地址从寄存器文件中读出相应寄存器的值，分别传递给Op1、Op2段间寄存器；同时将 dest 字段的值(目标寄存器的地址)向 REG ADDR 寄存器传递

   EX段：四个 MUX 选择器合理选择操作数传递给 ALU，ALU根据操作码(ALU Func)执行异或操作，ALU MUX 将结果存进 RESULT 寄存器

   MEM段：WB MUX选择要写回寄存器堆的值，存进 WBDATA 寄存器

   WB段：根据前面一直传递下来的寄存器写地址RegDstW，往相应寄存器中写入 WBDATA 的结果

2. **描述执行一条 BEQ 指令的过程**

   IF段：取指令，存入IR

   ID段：根据rs1、rs2字段读出两个寄存器的值，存到Op1、Op2；立即数生成单元根据指令的类型和该指令的立即数字段生成立即数，传递到一个加法单元与PCD相加得到分支目标地址，存到 Br Target 寄存器；Controller Decoder 根据指令的 Opcode 和 funct3 字段设置以下控制信号：

ImmType	BrType
控制立即数生成	beq

   EX段：Op1 MUX和 Op2 MUX 的值传递到 Branch Module，再根据控制信号BrType和两个操作数的值输出BR信号(有效或无效)，同时 Br Target 的值传递到 NPC Generator MUX

   为什么*branch*指令要在 *EX* 段而不是在 *ID* 段决定是否跳转？考虑以下指令序列

   ```
   lw x5, 0(x6)
   beq x5, x6, label1
   ```

   即使是通过旁路机制，*x5*的结果最早也只能在 *MEM* 段才出来，这种情况下*branch*指令是不可能在 *ID* 段就做出决策的

3. **描述执行一条 LHU 指令的过程**

   IF段：取指令，存入IR

ID段：根据 rs1 字段值取出相应寄存器的值，存到Op1；立即数生成单元根据指令类型和该指令的立即数字段(这里是imm[11:0])经符号扩展后产生立即数；将 rd 字段的值(目标寄存器的地址)向 REG ADDR 寄存器传递；Controller Decoder 根据指令的 Opcode 和 funct3 字段设置以下控制信号：

ALU Func	WB Select	Load Type	Op1	Op2	RegWrite	ImmType	Load NPC
加法操作	选择Data Extension	控制将16位无符号数值零扩展到32位	选择Op1 Forwarding MUX(即rs1)	选择Imm	有效	控制立即数生成	在ALU MUX选择ALU out存入RESULT

EX段：ALU执行加法操作计算要访问的内存地址，将 ALU Out 存入 RESULT 寄存器

MEM段：根据 RESULT 寄存器的值在 Data Cache 读取相应地址的单元，将该单元的值传递给 Data Extension 模块，Data Extension根据 Addr[1:0](即RESULT[1:0])和 Load Type 信号进行零扩展后将数据传递给 WB MUX，WB Select 信号选择Data Extension的结果作为 MUX 的输出 WBData，存入 WBDATA 寄存器

WB段：根据前面一直传递下来的寄存器写地址RegDstW，往相应寄存器中写入 WBDATA 的结果

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**

   CSR 指令的主要功能有：

   - 取 CSR 中的数据，进行扩展运算后写入一个通用寄存器rd
   - 取一个通用寄存器rs1的值写入 CSR 中
   - 根据rs1或者zimm的值修改 CSR 中的某些位

   部件和数据通路：增加 CSR 寄存器组；在ID段增加一个 CSR 译码器，用于对 CSR 指令的[31:20]字段进行译码，识别要访问的 CSR 寄存器、读写权限和访问权限等，并且需要在后面的各个段增加一些寄存器存储这些信息，以便检查是否有权限访问或读/写该 CSR 寄存器，同时在 ID 段读取 CSR 的值往后传递，增加在 WB 段写寄存器时可以选择输入源为 CSR 的值；在 MEM 段增加与 WB MUX、WBDATA 平行的写 CSR 寄存器的选择器和寄存器，存储要写入 CSR 的值，增加相应的控制信号(例如CSRWrite)

5. **Verilog 如何实现立即数的扩展?**

   先根据指令类型取出立即数字段，再根据指令类型、ImmType和最高位做符号扩展或零扩展以及拼接：

   - I 型

     ```
     assign Imm = {20{Inst[31]}, Inst[31:20]};
     ```

   - S 型

     ```
     assign Imm = {{20{Inst[31]}}, Inst[31:25], Inst[11:7]};
     ```

   - B 型

     ```
     assign Imm = {{20{Inst[31]}}, Inst[7], Inst[30:25], Inst[11:8], 1'b0}
     ```

   - U 型

```
assign Imm = {Inst[31:12], 12'b0};
```

- J 型

```
assign Imm = {{12{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：分多次读取后拼接

   Store：需要分多次先将涉及到的memory单元取出，修改相应的位后再写入

7. **ALU 模块中，默认 wire 变量是有符号数还是无符号数?**

   默认无符号数

8. **简述BranchE信号的作用**

   即 EX 段中的 BR 信号，作用是控制 IF 段的 NPC Generator MUX 选择 BR Target 作为输入；同时 Hazard Module 会根据 BR 信号生成 FlushF、FlushD 信号清空 IF、ID 段的寄存器

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

   branch 指令和 jalr 指令都是在 EX 段才能决定跳转(branch指令在 EX 段判断跳转条件是否成立，jalr 指令在 EX 段计算出跳转地址)，jal 指令在 ID 段就可以跳转，如果当前流水线中 EX 段是 branch 或 jalr 指令，且 ID 段是 jal 指令，按指令序列原本的顺序应该是先执行位于 EX 段的 branch 或 jalr 指令的，如果成功跳转了，则 jal 指令会被忽略，相当于优先级 branch = jalr > jal

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?**

    - 硬件相关可以在流水线的设计中避免
    - 控制相关通过 stall 或 flush 解决
    - 只有数据相关有可能需要插入气泡
      - ALU指令需要用到前一条 Load 指令的结果时需要插入一个气泡，使流水线停顿一个周期。因为 Load 指令的结果最早也要在 MEM 段才能出来，如果后面紧跟的 ALU 指令不停顿直接到了 EX 段的话，Load指令的结果不能及时送给 ALU 指令使用
      - 写后读相关可以使用旁路机制解决，不需要stall

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?**

    在 ID 段才能译码出是一条 branch 指令，branch 指令进入 EX 段时，原序列中的下一条指令存入 IR 并进入 ID 段。如果在 EX 段计算结果为不跳转，则没有 flush 信号；如果 EX 段计算结果为跳转，则 flush 掉 IF 和 ID 段的段间寄存器(即 IR、IDPC 和 Op1、Op2、EXPC等)

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

    会。例如给出如下指令序列

    ```
    add x0, x5, x6
    add x9, x0, x10
    ```

    如果不考虑0号寄存器始终为0的约束，则 forward 会将x5+x6的结果传递给第二条加法指令作为第一个操作数；但是由于存在这个约束，在 forward 机制中需要加入检查操作数是否为x0，如果是x0就不需要forward