Lab1 实验报告

1. XOR指令的过程
    1. IF段：Institution Cache 里取指令。
    2. ID段：General Register File 里取两个源操作数。
    3. EX段：两个源操作数放入ALU计算。`OP1 sel` 选择`Reg1`， `Op2 Sel` 选择`Reg2`，`Op1`和`Op2`都选择MUX的输出，ALU Func是`XOR`，Load NPC 选择`ALU Out`. 计算结果放入RESULT.
    4. MEM段：RESULT中的结果放入WBDATA。WB Select选择 `RESULT`。
    5. WB段：WBDATA写回General Register File，写回地址在ID阶段从指令中取出并逐步传递。`RegWrite`信号此时为`true`。
2. BEQ指令的过程
    1. IF段：Instruction Cache 里取指令
    2. ID段：General Register File 里取两个源操作数；同时取出指令里的立即数，左移一位和PC相加得到`Br Target`。
    3. EX段：`OP1 sel` 选择`Reg1`， `Op2 Sel` 选择`Reg2`，Op1和Op2都选择MUX的输出。两个操作数放入Branch Module中计算，`Br Type`信号的值为`EQ`，得到结果`BR`。`BrType`信号来自ID段对指令的分析。
    4. `BR Target` 作为NPC的一个输入，`BR` 作为NPC的一个信号，和其他信号共同决定`NPC`的取值。
3. LHU指令的过程
    1. IF段：Instruction Cache 里取指令
    2. ID段：General Register File里取第一个操作数，通过Immediate Generate 生成立即数作为第二个操作数。
    3. EX段：两个操作数在ALU里运算，`OP1 sel` 选择`Reg1`， `Op2 Sel`选择`Reg2`，`Op1`选择寄存器，`Op2`选择立即数，ALU Func是`ADD`，Load NPC选择`ALU Out`。结果放入RESULT中。
    4. MEM段：RESULT结果作为`Addr`送入 Data Cache， `Cache Write`信号为`false`，表示读数据（而不是写）。读出的数据放入Data Extension 中进行符号位扩展，`Load TypeM`信号为`LHU`。`WB Select`信号选择`Data Extension`，结果放入WBDATA。
    5. WB段：WBDATA写回到General Register File中，`RegWrite`信号为`true`，写回地址在ID阶段从指令中取出并逐步传递。
4. 在和General Register File并列的位置增加CSR寄存器文件，增加对CSR寄存器文件的CSR_RegWrite信号，Addr，WBAddr，WBDATA输入，输出CSR_reg和General Register File的Reg1输出进入一个选择器，选择其一作为Op1.
5. 零扩展

```
//假设立即数imm共k位，要符号位扩展增加t位
assign out = {t{0}, imm}
```

符号扩展

```
//假设立即数imm共k位，要符号位扩展增加t位
if (imm[k-1] == 0)
    assign out = {t{0}, imm}
else
    assign out = {t{1}, imm}
```

6. Load：多次读取后进行拼接。首先读取访问地址的第一个字节，然后读取下一个字节，并将其和前一个字节合并。这两个字节的顺序和机器的大端/小端存储有关。

   store：将数据按字节拆分，然后逐个按序写入到地址中。

7. 无符号数

8. BR 和 JALR， JAL 一起作为NPC Generator MUX的选择信号，决定选择哪一个来源作为下一条指令的地址(NPC)。

   同时BR也是Hazard Module的输入，根据BR可能会产生相应的停顿。

9. 有。因为这些信号产生的周期不一样，如果JALR和JAL同为`true`，或JAL和BR同时为`true`，其实JALR/BR在JAL之前。因为JALR/BR 在EX段生成，而JAL在ID段就生成了。应当按照前两个进行跳转。但是JAR和BR不可能同时为`true`，二者的优先级应当是一样的。

10. 数据相关：前一条是`load`类型的指令，后一条需要使用`load`出来的数据。

    插入一个气泡，停顿一个周期。

11. branch不跳转那么不进行操作。

    branch跳转，那么IF和ID段的flush全都置为true，表示需要清空后面的指令，前面的指令继续执行。一个周期之后flush置false，流水线继续执行。

12. 不会，因为0号寄存器只可以读不可以写，所以不会涉及到数据相关forward处理。