Comparch Lab01

PB18111757 陈金宝

1

XOR属于R类型指令。在IF段根据PC从指令内存中取出对应的指令。在ID段将取出的指令的OP(instr[0:6]),func3(instr[12:14]),func7(instr[25:31])送入control unit,产生控制信号:`RegWriteD`,`RegReadD`,`AluControlD`,`Alusrc1D`,`Alusrc2D`。寄存器根据`RegRead`信号以及指令中的`rs`和`rt`，读出对应的寄存器数据(`RegOut1D`,`RegOut2D`)。根据`rd`字段将写入的寄存器号读出。之后将信号以及读出的寄存器数据向后传到EX段。在EX段根据传来的`AluControlE`,alu决定进行xor操作。根据`AluSrc1E`,`AluSrc2E`,`RegOut1E`,`RegOut2E`选取操作数，alu对指定的数据进行xor操作。计算后的结果`AluOutE`以及`RegWrite`，`rdE`信号继续向后传到M段。M段继续将该信息向后传到W段。W段将`rdW`,`ResultW`,以及`RegWrite`信号传给寄存器，向指定寄存器写入数据。

2

beq属于B类指令。只有IF,ID和EX段。在IF段取指后进入ID段。ID段指令送入Control unit。产生BranchType，Alucontrol,alusrc1,alusrc2信号。同时寄存器取出对应的源寄存器值以及offset并进行符号扩展并计算出跳转地址(JalNpc)。这些值和信号送入EX段。在EX段,Branch Decision接收从ID段接受的BranchType以及REG1,REG2信号。决定是否跳转。跳转则产生BrE信号送入NPC GEN。BrNPC送入NPC GEN的BrT口。

3

LHU属于I类指令。IF段取指后进入ID段。ID段从寄存器取出RS1，对offset进行扩展，control unit产生RegWrite,Memtoreg,RegRead,Alucontrol,alusrc1,alusrc2信号。之后rd,rs1,offset和相应信号被送入EX段。EX段ALU计算出目标地址，ALuOutE(rs1 + sext(offset))。之后将该值以及rd,RegWrite,Memtoreg信号送入MEM段。MEM段根据地址取出数数据RD，将其以及信号rd,RegWrite,Memtoreg送入WB段。WB段根据相应寄存器目的地址，数据，控制信号等进行写回。

4

增加CSR寄存器组，同时增加CSR寄存器的读写信号，读地址信号，写地址信号，寄存器输入、输出选择信号（选择CSR或寄存器组），扩展ALUControl信号。类似于寄存器文件的读写操作，ID段读出对应寄存器值，EX段进行操作，WB进行写回。

5

伪代码（k是要扩展的位数,top为要扩展的数的最高位）

```
case opcode
 零扩展:assign out = {k{0},offset}
 符号扩展:assign out = {k{offset[top]},offset}
endcase
```

6

数据线依旧为32位，此时：
load:byte和half word时进行符号扩展到32位。
store:增加控制信号，控制memory写入的byte(如00:byte 01:half word,10:word)

7

默认无符号

8

表示branch命中。此时NPC GEN会进行跳转同时清空对应流水段。

9

有优先级。后来的指令先跳转。同时，若修改数据通路，使得br,jal,jalr均在EX段跳转，则不会有冲突

10

load后立即使用，此时会有冲突。需要插入1个气泡。
if id ex me wb
  if id ex me wb
    if id ex me wb

11

branch在EX段跳转。branch命中后需要控制flush信号来清空跳转后的ID和EX段。

12

会产生影响。涉及到x0寄存器时就不需要forward（x0恒为0）