P5 设计文档

在流水线设计过程中采用分布式译码，分为了五级，使用了 IF/ID，ID/EX，EX/MEM，MEM/WB 四个流水寄存器。下面是遇到的冲突以及解决方案。

1. 遇到的冲突

在课下所需支持的指令集中，所有的冲突本质都是某条指令需要用到的寄存器值本应在前方某一条指令进行更新但因为流水线的缘故尚未写入寄存器。

解决方法优先使用转发：将已经算出但未写入的值进行转发取代掉现在取出的寄存器值。

如果在需要使用寄存器值但前指令尚未算出该值（只可能出现在 lw），则暂停插入空指令。

2. 转发

转发粗略可分为两种：ID 级的转发，EXT 级的转发。本来应该还有 MEM 级的转发，sw 指令在 memory 主存存入值来自 rt，但是我们发现可以在 EXT 就进行转发。

指令类型	指令
Cal_r	addu,subu
Cal_i	lui,ori
b	beq
load	lw
save	sw
j	j,jal,jr

在这里做一些对指令的处理：EXT 级的 imm 只会在对应 rt 的 input 输入，对所有会在 grf 写入的指令，我们将写入地址记为 rd（有的 cal_r 指令写入了 rt，在这里将 rt 改为 rd），将 rs，rt，rd 不存在的记为 0。

这样，我们就可以将转发简化为以下两种情况：

(1) rs/rt=rd.mem (2)rs/rt=rd.wb

在分别进行转发即可。

对于 ID 级，还需要增加一个 rd.ext 判断，转发内容为 alu 算出的值，这样可以减少暂停。

3. 暂停

暂停是在将要使用某会被更新的寄存器值但是却未被算出时，采用 Tuse，Tnew，Tnew-Tuse>0 且 rd==rt/rs 时。当没有写入时，Tnew=0，当不用寄存器值时，Tuse=10。

```
if($signed(Tnew_EX-Tuse_ID)>0&&(rd_EX==rt_ID||rd_EX==rs_ID)&&rd_EX!=0)
s=1;
```

指令类型	Tuse	Tnew
Cal_r	1	2
Cal_i	1（特殊的，lui 为 0）	2
b	0	0
load	1	3
save	1	0
j	10	0
jal	10	1
jr	0	0

在这里，j，jal，jr 各有不同，jr 可以转发，jal 我们希望将跳转与链接分离。

四. 思考题

```
addu $3,$1,$3
addu $3,$1,$3
addu $3,$1,$3
```

类型：rt.ext=rd.mem，转发

```
lw $1,0($3)
addu $3,$1,$0
```

类型：rs.ext=rd.mem，需要暂停，然后转发

```
lui $1,1
Lui $2,1
```

Beq $1,$2,a

暂停，然后转发

Sw $1,0($0)

Lw $1,0($0)

无冲突