实验 3~5 报告

学号 2018K8009929030
姓名 热伊莱·图尔贡
箱子号 74

一、实验任务（10%）

1. lab3：对已经给出的流水线 CPUdebug。
2. lab4：对写后读数据相关产生的流水级冲突，进行阻塞。
3. lab5：对写后读数据相关产生的流水级冲突，进行前递 。

二、实验设计（40%）

（一）总体设计思路

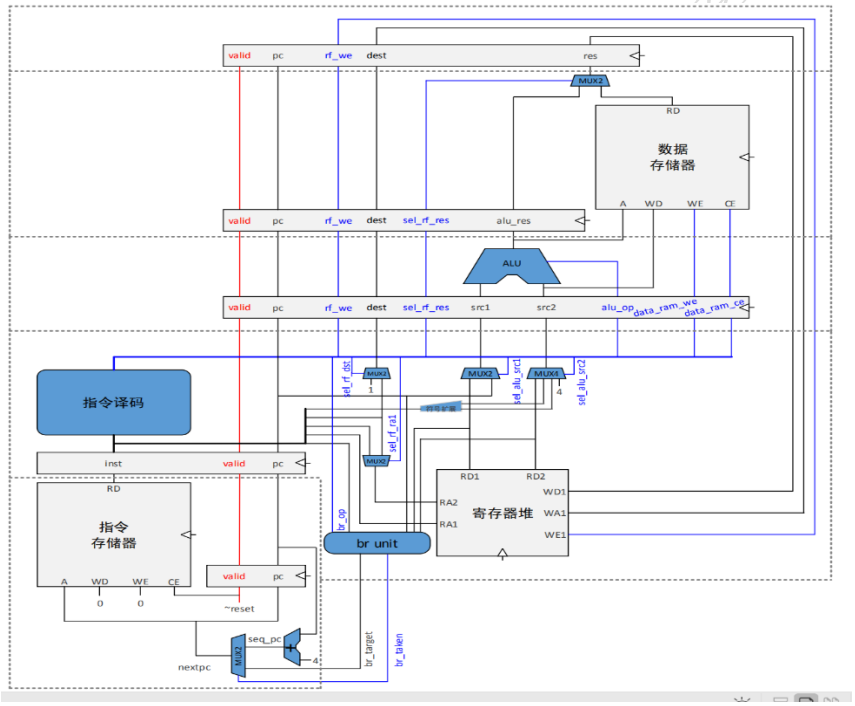

图 1 简单 CPU 结构设计图

Lab4 和 lab5 中的处理写后读的方案：

用阻塞的手段处理指令相关：在 es，ms 和 ws 三个流水级各自设计一个 bus，装载该流水级的 rf_wen 信号和 rf_dest 送至 ds 流水级。

用前递的手段处理指令相关：与 lab4 相似，只需要 es 和 ms。同时设计前递 bus。

（二）重要模块 1 设计：DS 模块

该流水级负责分析 FS 阶段取进来的指令的功能，同时将译码出的信息传递给下一个流水级或者 fs 流水级。其中，部分信息需要从寄存器堆中取出。

1、工作原理

通过译码器对读进来的指令进行解析，并将解析出的信息装载入 ds_to_es_bus 或者直接从寄存器中读出信息后装入 ds_to_es_bus，此外，若译码信息是分支跳转指令，则将此信息传递回 fs 流水级。

2、接口定义

表 1　DS 模块接口

名称	方向	位宽	功能描述
es_allowin	IN	1	当此信号拉高时，表示 es 流水级允许数据传入
ds_allowin	OUT	1	当此信号拉高时，表示 ds 流水级允许数据传入
fs_to_ds_valid	IN	1	当此信号拉高时，表示 fs 流水级已经准备好发送信号
fs_to_ds_bus	IN	[`FS_TO_DS_BUS_WD -1:0]	装载 fs 流水级向 ds 流水级传递的信息
ds_to_es_valid	OUT	1	当此信号拉高时，表示 ds 流水级已经准备好发送信号
ds_to_es_bus	OUT	[`DS_TO_ES_BUS_WD - 1:0]	装载 ds 流水级向 es 流水级传递的信息
br_bus	OUT	[`BR_BUS_WD :0]	装载需要传递给 fs 流水级的译码出的分支跳转信息

3、功能描述

通过译码阶段译码中的信息在寄存器中读出需要向下一级流水级传递的信息，还有部分则是通过从寄存器中读出的信息进行运算后得到。译码译出的信息中跳转部分则要装载入另一个 bus 传递给 fs 流水级。

（三）重要模块 2 设计：ES 流水级

1、工作原理

对写后读数据相关产生的流水级冲突，进行前递。

2、接口定义

```
//lab4
//assign ds_ready_go    = 1'b1;
wire rk_read = inst_add_w | inst_sub_w | inst_slt | inst_sltu
             | inst_and | inst_or | inst_nor | inst_xor;
wire rj_read = ~inst_lu12i_w & ~inst_b & ~inst_bl;
wire rd_read = src_reg_is_rd;
//lab5
wire rj_fwd_es = es_ok && (es_rf_dest == rj) && (rj != 0) && rj_read;
wire rj_fwd_ms = ms_ok && (ms_rf_dest == rj) && (rj != 0) && rj_read;
wire rj_fwd_ws = rf_we && (rf_waddr == rj) && (rj != 0) && rj_read;
wire rkd_fwd_es = es_ok && (es_rf_dest == rk) && (rk != 0) && rk_read||es_ok && (es_rf_dest == rd) && (rd != 0) && rd_read;
wire rkd_fwd_ms = ms_ok && (ms_rf_dest == rk) && (rk != 0) && rk_read||ms_ok && (ms_rf_dest == rd) && (rd != 0) && rd_read;
wire rkd_fwd_ws = rf_we && (rf_waddr == rk) && (rk != 0) && rk_read||rf_we && (rf_waddr == rd) && (rd != 0) && rd_read;
wire rk_conf = (rk != 0) && rk_read && (
    es_ok && es_rf_dest == rk
    || ms_ok && ms_rf_dest == rk
    || rf_we && rf_waddr == rk
);
wire rj_conf = (rj !=0) && rj_read && (
       es_ok && es_rf_dest == rj
    || ms_ok && ms_rf_dest == rj
    || rf_we && rf_waddr ==  rj
);
wire rd_conf = (rd !=0) && rd_read && (
       es_ok && es_rf_dest == rd
    || ms_ok && ms_rf_dest == rd
    || rf_we && rf_waddr ==  rd
);
```

图 2 前递 bus

3、功能描述

负责对 ID 生成的各种数据进行运算，完成绝大多数运算指令的操作、并生成访存指令的地址。

三、实验过程（50%）

（一）实验流水账

2021.9.16 之前完成了 lab3；

2021.9.28 之前完成了 lab4；

2021.11.2 之前完成了 lab5。

（二）错误记录

1、错误 1：信号为 X

（1）错误现象

debug_wb_pc 信号出现 X 波形

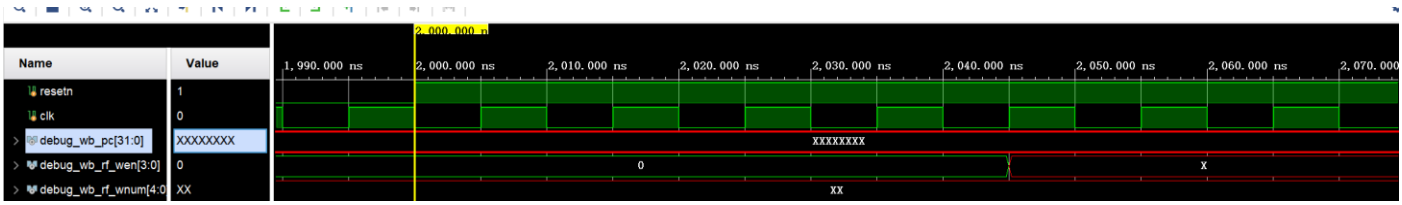

图 3 错误波形

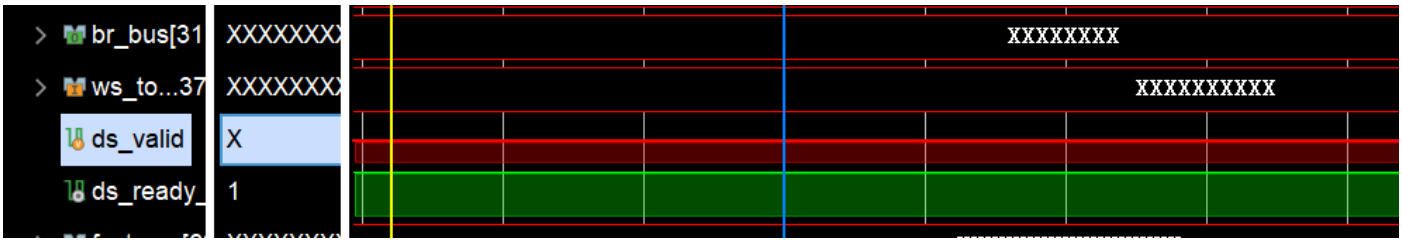

图 4 错误波形

（2）分析定位过程

X 信号表示出现不定值，这种问题的出现主要有两种情况：声明为 wire 型的变量从未被赋值和写成了多驱动的代码。

debug_wb_pc 是从 IF 阶段一步一步传递到 WB 阶段得到的变量，途径中没有出现多驱动的情况，于是问题可能出在没有对其赋值。逐步向上一层流水线追溯，直至 if_stage 文件中的 ds_valid 信号，发现其并未被赋值。

（3）错误原因

信号被声明后未被赋值。

（4）修正效果

该信号的意义在于判断 if_stage 这一层流水级是否已经储存有效数据。

2、错误 2：信号为 X

（1）错误现象

PC 的值归于正常，但写回数据信号出现 X 波形

图 5 错误波形

（2）分析定位过程

基于与上一个 bug 类似的 debug 方式，从 top 层开始追索，直至发现其 ds_to_es_bus 信号存在一位 Z，即 load_op 被声明了但未赋值。

（3）错误原因

信号被声明后未被赋值。

（4）修正效果

assign load_op[0] = inst_ld_w;

3、错误 3：加法出错

（1）错误现象

加法指令，返回值出错。

```
------------------------------------------------------------
[   2067 ns] Error!!!
    reference: PC = 0x1c000000, wb_rf_wnum = 0x0c, wb_rf_wdata = 0xffffffff
    mycpu    : PC = 0x1c000000, wb_rf_wnum = 0x0c, wb_rf_wdata = 0xfffffffe
------------------------------------------------------------
```

图 6 数据返回错误

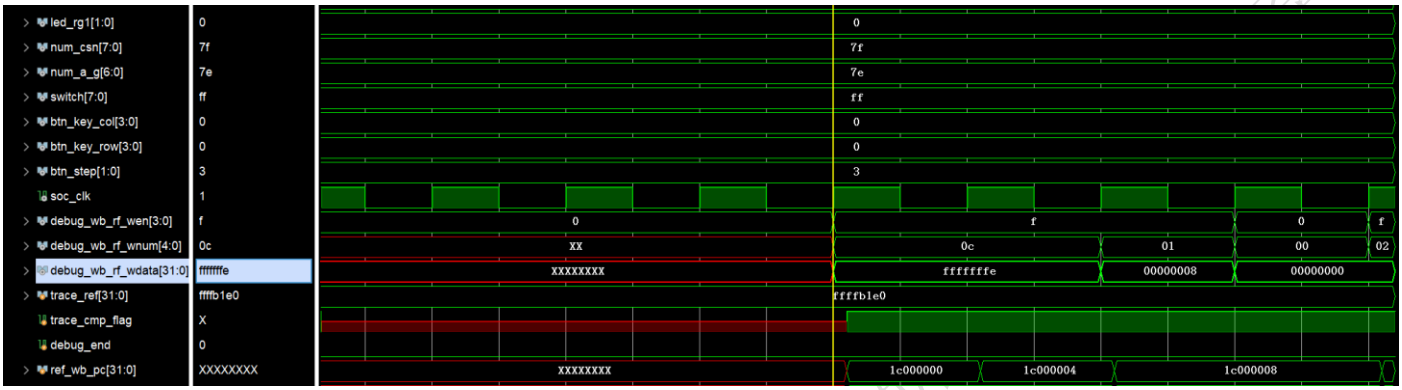

图 7 错误波形

（2）分析定位过程

在定位到 MEM 流水级时，发现写回的是 alu 运算的结果，遂去检查 alu 模块，alu_op 为 001，是加法。alu_src1 和 alu_src2 都是 ffff，所以实际计算写回的并没有错误。因此检查 alu_src1 和 alu_src2，发现错误在 alu 的输入中。

（3）错误原因

向 alu 模块 src1 和 src2 接口输入的都是 src2。

（4）修正效果

接口重连，把 scr1 接到 src1。

4、错误 4：跳转错误

（1）错误现象

```
2107 ns] Error!!!
reference: PC = 0x1c00070c, wb_rf_wnum = 0x02, wb_rf_wdata = 0x00000000
mycpu    : PC = 0x1c000010, wb_rf_wnum = 0x0c, wb_rf_wdata = 0x80000000
```

图 8 返回数据错误

（2）分析定位过程

找到 IF 流水级，此处的跳转信号异常。

（3）错误原因

由于 br_bus 高位是决定是否发生跳转的控制信号，而由于其位宽被少赋值了一位，故该信号处于异常态，进而导致 nextpc 对跳转的判断出现问题。

（4）修正效果

```
`define BR_BUS_WD      33
```

5、错误 5：跳转错误

（1）错误现象

```
2117 ns] Error!!!
reference: PC = 0x1c000710, wb_rf_wnum = 0x03, wb_rf_wdata = 0x00000000
mycpu    : PC = 0x1c000388, wb_rf_wnum = 0x04, wb_rf_wdata = 0xbfaff000
```

图 9 返回数据错误

（2）分析定位过程

这个错误比较难，在实际调试的过程中发现其应当是一个跳转指令，但 nextpc 并没有进行相应的调整。

（3）错误原因

错误原因为执行了紧随着跳转指令后一拍的指令（即 PC+4），而其本应取消该指令，而紧接着去执行目标指令。

（4）修正效果

```
assign fs_to_ds_valid =  fs_valid && fs_ready_go && !br_taken;
```

6、错误 6：移位错误

（1）错误现象

```
        [  22000 ns] Test is running, debug_wb_pc = 0x1c0626a4
--------------------------------------------------------------
[  30077 ns] Error!!!
    reference: PC = 0x1c063344, wb_rf_wnum = 0x0d, wb_rf_wdata = 0x01000000
    mycpu    : PC = 0x1c063344, wb_rf_wnum = 0x0d, wb_rf_wdata = 0x00000070
--------------------------------------------------------------
$finish called at time : 30117 ns : File "D:/2021-2022course/Computer Architecture/lab3/lab3/CPU_CDE/mycpu
```

图 10 返回数据错误

（2）分析定位过程

可以在 alu 中看出移位移反了。

（3）错误原因

Alu 计算错误，移位移反了。

（4）修正效果

```
assign sll_result = alu_src1 << alu_src2[4:0];   //*bug6 rj << i5
```

```
    // SRL, SRA result
assign sr64_result = {{32{op_sra & alu_src1[31]}}, alu_src1[31:0]} >> alu_src2[4:0];
```

7、**错误 7：**组合逻辑环问题

（1）**错误现象**

```
assign or_result  = alu_src1 | alu_src2 | alu_result;
```

图 11 错误代码

（2）**分析定位过程**

这个错误是在检查 alu 的时候一眼看出的。

（3）**错误原因**

Or 指令结果定义错误。

（4）**修正效果**

assign or_result = alu_src1 | alu_src2 ;

四、实验总结（可选）