实验 1-2 报告

学号：2016K8009929060
姓名：王晨赳

一、实验任务（10%）

实验任务是将上个学期的 CPU 移植到本次实验平台上来，为此需要修改相关接口。CPU 复位从 0xbfc00000 开始，仿存和取指分为两个 SRAM。需要实现 19 条指令，并将每条指令的 PC 带到写回级，以供验证之需。从 CPU 向外连出几个 debug 信号，方便验证。要求 CPU 采用实地址映射，采用延迟槽技术，并只有一个操作模式即核心态。验证分为 vivado 行为仿真验证和上板验证。在前者通过的情况下进行后者。

二、实验设计（30%）

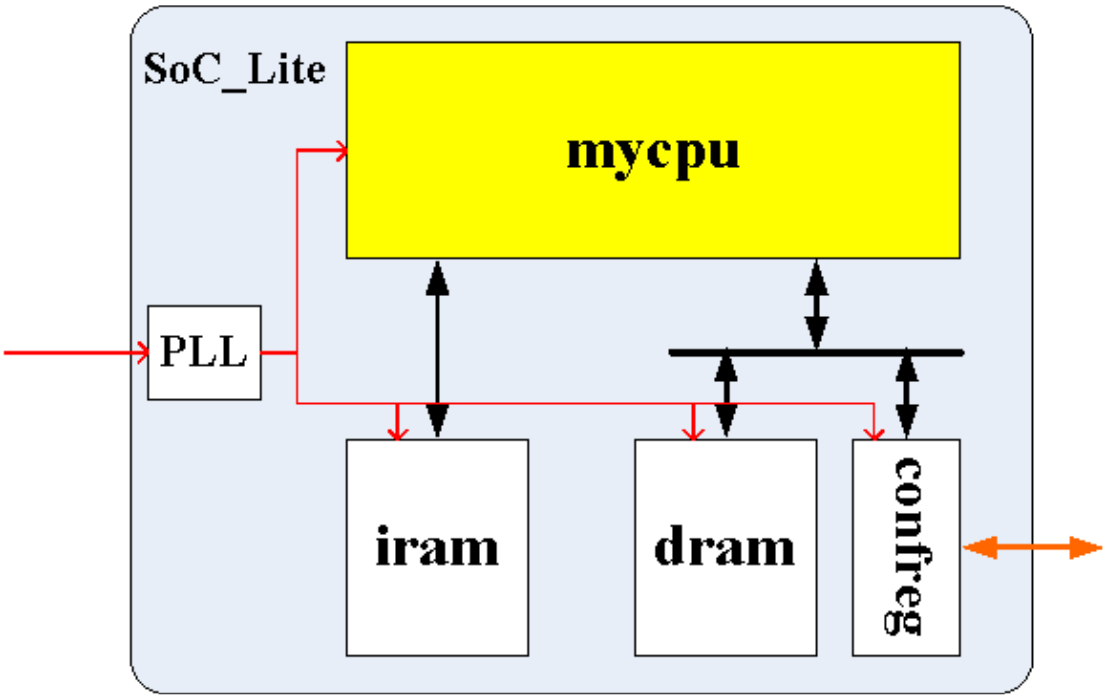

图 1：SOC_Lite 架构

首先理解 SoC_Lite、my_cpu、PLL、SRAM、confreg 之间的关系，各部分主要负责什么工作。实验的顶层是 SoC_Lite，PLL 是时钟，confreg 是配置寄存器。实验需要实现的是 my_cpu，用该 CPU 运行验证程序，CPU 需要访问 IRAM 获取指令，访问 DRAM 进行数据交互，最终使验证程序正确运行。上学期已经实现了多周期 CPU，故数据通路部分不需要做太大改动。实验主要是修改相关接口，并修改状态机，使 CPU 适应当前实验平台，并完成正确的状态跳转。CPU 取指和仿存用的是两个分开的 SRAM，这与之前的一样，需要注意 SRAM 是同步读同步写的，故要删去原来的应答信号。需要实现的 19 条指令上个学期就已经完成了，不需要做太大改动。为了方便验证需要连出几个 debug 信号，只需反映写回级的信息。每条指令都把 PC 带到写回级，是 PC 在复位时从 0xbfc00000 开始。至于实现核心态，延迟槽，实地址映射本次实验并不需要特地改动。

my_cpu 分为三个模块，分别为 CPU 核、ALU、寄存器堆。CPU 核工作时会调用 ALU 和寄存器堆模块。ALU 负责算术逻辑运算，寄存器堆是 CPU 内部的通用寄存器。

（一）模块 1：CPU 核设计

1、接口定义

表 1：CPU 核接口定义

名称	方向	位宽	功能描述
resetn	IN	1	复位信号，低电平有效
clk	IN	1	时钟信号
inst_sram_en	OUT	1	指令 SRAM 使能信号，高电平有效
inst_sram_wen	OUT	4	指令 SRAM 字节写使能信号，高电平有效
inst_sram_addr	OUT	32	指令 SRAM 读写地址
inst_sram_wdata	OUT	32	指令 SRAM 写数据
inst_sram_rdata	IN	32	指令 SRAM 读数据
data_sram_en	OUT	1	数据 SRAM 使能信号，高电平有效
data_sram_wen	OUT	4	数据 SRAM 字节写使能信号，高电平有效
data_sram_addr	OUT	32	数据 SRAM 读写地址
data_sram_wdata	OUT	32	数据 SRAM 写数据
data_sram_rdata	IN	32	数据 SRAM 读数据
debug_wb_pc	OUT	32	多周期写回级的 PC
debug_wb_rf_wen	OUT	4	多周期写回级寄存器堆的字节写使能信号
debug_wb_rf_wnum	OUT	32	多周期写回级寄存器堆的写地址
debug_wb_rf_wdata	OUT	32	多周期写回级寄存器堆的写数据

2、功能描述

CPU 核有一个状态机，分为 IF（取指）、ID（译码）、EX（执行）、MEM（访存）、WB（寄存器写回）五个状态以实现多周期 CPU。在 IF 阶段，CPU 拉高 inst_sram_en 信号，IRAM 会根据 inst_sram_addr 返回一条指令即为 inst_sram_rdata。CPU 得到指令后来到 ID 阶段，CPU 译码后将一些寄存器置位，来到 EX 阶段。此时根据 ID 阶段的控制信号进行操作，选择是进入 IF、WB、ST（向内存写）或是 LD（向内存读）状态。若是 ST 状态则要拉高 data_sram_en 信号，并准备好 data_sram_addr、data_sram_wen 和 data_sram_wdata，DRAM 根据它们将数据写入内存。若是 LD 状态则要拉高 data_sram_en 信号，并准备好 data_sram_addr，置 data_sram_wen 为 4’0000，DRAM 读出数据即为 data_sram_rdata。在 WB 阶段调用寄存器堆模块，将数据写回，若无写回则空等一拍。

（二）模块 2：ALU

1、接口定义

表 2：ALU 接口定义

名称	方向	位宽	功能描述
A	IN	32	输入的操作数
B	IN	32	输入的操作数
ALUci	IN	4	ALU 的控制信号，对应不同的运算
Overflow	OUT	1	有符号数运算溢出信号
CarryOut	OUT	1	无符号数运算进位信号
Zero	OUT	1	运算结果为零则拉高
Result	OUT	32	运算结果

2、功能描述

ALU 根据 ALUci 信号以及两个操作数进行运算，如加法、减法、与或非、移位运算。得到结果后输出，并将相应的标志寄存器置位。

（三）模块 3：寄存器堆

1、接口定义

表 3：寄存器堆接口定义

名称	方向	位宽	功能描述
clk	IN	1	时钟信号
resetn	IN	1	复位信号，低电平有效
waddr	IN	5	寄存器堆写地址
raddr1	IN	5	寄存器堆读地址 1
raddr2	IN	5	寄存器堆读地址 2
wen	IN	1	寄存器堆使能信号，高电平有效
wreg_strb	IN	3	寄存器堆字节写使能信号
wdata	IN	32	寄存器堆写数据
rdata1	OUT	32	寄存器堆读数据 1
rdata2	OUT	32	寄存器堆读数据 2

2、功能描述

CPU 内部的 32 个 32 位通用寄存器，有两个读端口和一个写端口，支持同步读、异步写。wen 拉高时，寄存器堆可进行写操作，根据字节写使能信号将 wdata 写入 waddr 所表示的寄存器内。寄存器堆根据 raddr1 和 raddr2 读出数据分别为 rdata1 和 rdata2。

三、实验过程（60%）

（一）实验流水账

9 月 8 日 18:00~22:00，开始实验，完成 rtl 设计，开始仿真，未通过。

9 月 9 日 14:00~17:30，仿真调试，找到错误尝试解决

9 月 9 日 19:00~23:00，解决 bug，仿真通过，上板通过

9 月 10 日 20:00~0:30，撰写 1-1 报告

9 月 16 日 19:00~20:00,修改代码风格，整理代码

9 月 18 日，撰写实验报告 1-2

（二）错误记录

1、错误 1

（1）错误现象

发现找不到仿真模块。

（2）分析定位过程

看控制台报错，明明各个模块都有了，却仍报错。最后对比模块名。

（3）错误原因

CPU 模块命名错误。

（4）修正效果

将模块名改为 mycpu_top，解决了错误。

（5）归纳总结（可选）

错误很小却让人难受，今后要仔细了解实验的各种细节。

2、错误 2

（1）错误现象

发现取出的指令总是慢一拍。

（2）分析定位过程

观察整个出指令的过程，从使能信号拉高，提供地址，到返回数据。

（3）错误原因

未认识到 SRAM 在使能信号拉高后的下一个时钟周期才会返回数据

（4）修正效果

将使能信号提前一拍拉高，错误解决。

（5）归纳总结（可选）

时序错误，应注意时序问题。

3、错误 3

（1）错误现象

控制台报错，PC 对不上。

（2）分析定位过程

找到错误发生位置，对照汇编代码定位错误。

（3）错误原因

jal 指令执行有误。

（4）修正效果

修改 jal 指令，使其经过写回级，在 WB 阶段写回返回地址。问题解决。

（5）归纳总结（可选）

逻辑错误，应认真思考每条指令的执行过程。

4、错误 4

（1）错误现象

```
[2451317 ns] Error!!!
    reference: PC = 0xbfc35e64, wb_rf_wnum = 0x02, wb_rf_wdata = 0xc822c7e8
    mycpu    : PC = 0xbfc35e64, wb_rf_wnum = 0x02, wb_rf_wdata = 0x00000000
```

图 2：TCL Console 报错信息

（2）分析定位过程

观察 PC 为 0xbfc35e64 附近的波形，一直找到之前的 sw 和 lw 指令。发现 sw 往内存存了一个数，但 lw 从相同地址取出的数据却不对。

（3）错误原因

data_sram_en 是读写使能信号，在其拉高的情况下，DRAM 既可读也可写。因此在需要 DRAM 进行读数据时应该将 data_sram_wen 置为 4’b0000，以免此时将 data_sram_wdata 写入 DRAM。

（4）修正效果

将 data_sram_wen 设为在 LD 阶段接 0。问题解决。

（5）归纳总结（可选）

对信号理解不够，以后应多加注意。

四、实验总结（可选）

写文档的时间不比 debug 的时间少。不过也可能是本次实验难度还不大。