1. **大实验概述**
   1. 实验目标
2. 设计并实现5段流水线CPU，支持MIPS16E指令，包含25条基本指令和5条扩展指令
3. 支持Term终端程序的串口通信
4. 支持Flash自启动
5. 支持扩展指令实现VGA显示
   1. 实验硬件环境

1. FPGA 芯片：Spartan-3E系列的 XC3S1200EFGG320芯片，320个管脚，120万门，FBGA封装。
2. CPLD芯片：XC9500系列的XC95144XL-7TQ100芯片，100个管脚，144个宏单元，采用TQFP封装
3. SRAM：型号为IS61LV25616-10TI，电压需求为3.3V，每片存储容量为256K\*16b
4. Flash：MT28F640J3，其数据线为16位，地址线为23位。
5. 带VGA输入的显示器
   1. 实验软件环境
6. FPGA开发工具软件Xilinx ISE 14.7
7. FlashAndRam.exe 支持对flash ram1 ram2的读写
8. Term.exe支持与监控程序的串口通信
9. Assembler.exe将改写后的汇编版本监控程序转为二进制机器码
10. **各模块设计**
    1. RegFile模块

2.1.1模块功能概述

1. 寄存器组包括8个通用寄存器和增加的T、IH、SP、RA寄存器
2. 根据控制信号和时钟将数据写入寄存器或从寄存器读数据
3. 将寄存器的值通过VGA在显示器上显示

2.1.2 模块数据接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 名称 | 长度 | 来源或去处 | 注释 |
| In | clk | 1 | 板子 |  |
|  | clk25 | 1 | 板子 |  |
|  | rst | 1 | 板子 |  |
|  | reg\_readno1 | 4 | ID | 读第一个寄存器的编号 |
|  | reg\_readno2 | 4 | ID | 读第二个寄存器的编号 |
|  | reg\_writeno | 4 | WB | 写寄存器的编号 |
|  | reg\_writedata | 16 | WB | 写入寄存器的内容 |
|  | reg\_we | 1 | WB | 是否写入 |
| out | id\_readdata1 | 16 | ID | 读出的第一个寄存器的内容 |
|  | id\_readdata2 | 16 | ID | 读出的第二个寄存器的内容 |
|  | id\_T | 16 | ID | T的内容 |
|  | id\_SP | 16 | ID | SP的内容 |
|  | id\_IH | 16 | ID | IH的内容 |
|  | id\_RA | 16 | ID | RA的内容 |
|  | vga\_r0 | 16 | VGA | 往VGA输出的寄存器内容（下同） |
|  | vga\_r1 | 16 | VGA |  |
|  | vga\_r2 | 16 | VGA |  |
|  | vga\_r3 | 16 | VGA |  |
|  | vga\_r4 | 16 | VGA |  |
|  | vga\_r5 | 16 | VGA |  |
|  | vga\_r6 | 16 | VGA |  |
|  | vga\_r7 | 16 | VGA |  |

* 1. EXE模块
     1. 模块功能概述

接收ID段的操作数，根据控制信号执行制定的运算，结果写入寄存器中。

* + 1. ALU操作号

|  |  |
| --- | --- |
| 操作号 | 操作 |
| 0000 | ADD |
| 0001 | SUB |
| 0010 | AND |
| 0011 | OR |
| 0110 | CMP |
| 1001 | NOT |
| 0100 | SLL |
| 0101 | SRA |
| 0111 | SLTUI |

* + 1. 模块数据接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 名称 | 长度 | 来源或去处 | 注释 |
| In | exe\_alu1\_operand1 | 16 | ID | ALU第一个操作数 |
|  | exe\_alu1\_operand2 | 16 | ID | ALU第二个操作数 |
|  | exe\_alu1\_opkind | 4 | ID | ALU运算类型 |
|  | exe\_regwrite | 1 | ID | 是否写入寄存器 |
|  | exe\_regdst | 4 | ID | 写入哪个寄存器 |
|  | exe\_memwrite | 1 | ID | 是否SW |
|  | exe\_memwritedata | 16 | ID | 写入内存的内容 |
|  | exe\_memtoreg | 1 | ID | 写回的内容是来自ALU的输出还是访存得到的数据 |
| out | mem\_regwrite | 1 | MEM | 是否需要写入寄存器 |
|  | mem\_regdst | 4 | MEM | 写入哪个寄存器 |
|  | mem\_wboraddr | 16 | MEM | ALU1算出的结果 |
|  | mem\_memwrite | 1 | MEM | 是否SW |
|  | mem\_memtoreg | 1 | MEM | 写回的内容是来自ALU的输出还是访存得到的数据 |
|  | pang\_mem\_memwrite | 1 | MEM | 是否SW传给旁路 |
|  | pang\_mem\_wboraddr | 16 | MEM | ALU1算出的结果 |
|  | pang\_mem\_memwritedata | 16 | MEM | 写入内存的内容 |
|  | pang\_mem\_memtoreg | 1 | MEM | 写回的内容是来自ALU的输出还是访存得到的数据 |
|  | if\_wboraddr | 16 | IF | ALU1算出的结果，直接传给IF段 |
|  | if\_memwrite | 1 | IF | 是否SW传给IF段 |
|  | if\_memwritedata | 16 | IF | 写入内容的内容传给IF段 |
|  | if\_flush\_from\_exe | 1 | IF | PC是否不能+1（此时在按MEM的要求读写指令地址区） |
|  | if\_id\_flush\_from\_exe | 1 | IF | PC是否不能+1 |
|  | id\_regwrite\_from\_exe | 1 | ID | 是否需要写入寄存器 |
|  | id\_regwritedata\_from\_exe | 16 | ID | 写入内存的内容 |
|  | id\_regdst\_from\_exe | 4 | ID | 写入哪个寄存器 |
|  | id\_isLW\_from\_exe | 1 | ID | 是否读取 |

1. **支持Flash自启动**
   1. 功能概述

由于RAM断电后无法保存数据，因此需要每次上电后，利用FlashAndRam.exe 将监控程序写入RAM2。而将监控程序保存在flash中，每次上电后，由CPU 程序控制将Flash中的监控程序写入Ram2后，自动开始执行监控程序，可以避免这一麻烦的操作。

* 1. 模块数据接口

|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 名称 | 长度 | 注释 |
| in | clk | 1 |  |
|  | rst | 1 |  |
|  | start | 1 | 0表示自启动阶段结束，本模块停止工作 |
| inout | flashData | 16 | Flash16位数据线 |
| inout | flashAddr | 23 | Flash23位地址线 |
| out | flashByte | 1 | 操作模式 |
|  | flashVpen | 1 | 写保护 常置1 |
|  | flashCE | 1 | 使能 |
|  | flashOE | 1 | 读 |
|  | flashWE | 1 | 写 |
|  | flashRP | 1 | 0重置 1工作 |
|  | ram2\_en | 1 | ram2的使能 |
|  | ram2\_oe | 1 | Ram2读使能 |
|  | ram2\_we | 1 | Ram2写使能 |
|  | ram2addr | 18 | Ram2 18位地址线 |
| inout | ram2data | 16 | Ram2 16位数据线 |

* 1. 工作原理

首先利用 FlashAndRam.exe 将监控程序的机器码写入 Flash。在 Top 模块下新加入一个模块 FlashLoader。这个模块主要维护一个全局的控制信号 start。每次按下 rst 后，将 start 置 为 1，在 start 为 1 的情况下，流水线上各个模块都不会工作，只有 FlashLoader 这个模块会工作。该模块内部运行一个状态机，每个状态机的循环周期内，先从 Flash 指定地址读出一 条指令，然后将这条指令写入 RAM2，然后地址加 1。直到把整个监控程序都写入 RAM2 后， 该模块会把 start 改为 0。当 start 为 0 后，流水线各个模块开始工作，而 FlashLoader 模块停止工作。

感想：

从刚开始设计数据通路不知从何下手，到最后造出可以正常工作带扩展功能的CPU，过程可谓一波三折。最初设计数据通路的时候，考虑的过于简单，忽略了冲突等问题，小班教学时助教严厉地指出了我们的问题，通过后来的学习不断查阅资料，终于设计出了完整的数据通路。由数据通路开始了我们造计算机的第一步。

然而每一步都不是那么容易，写完流水线，冲突未处理；处理了冲突，显示不出OK······每一步都会出现新的bug，随着设计的深入，又会发现之前一个又一个的漏洞。当然代码的问题都是可以解决的，还有一些硬件的问题很是头痛。有时，电脑会识别不了串口，而且是小组内三个同学电脑都识别不了，过了一会又莫名其妙的好了，这些玄学问题伴随着实验的全过程，好在板子还是给力，最终有惊无险的完成了实验。

当然，这漫长而又短暂的三个星期里，学到的东西还是很有价值的。不单单是专业知识的提升，更是对自己的一种磨练，造计算机每天工作十几个小时，连续熬夜一个礼拜，磨练了自己的意志品质，累了的时候队友之间互相鼓励，完成了实验，也增加了友谊。希望所有奋斗过三星期的同学能有所收获，最后感谢刘卫东老师，李山山老师的理论指导，感谢助教的细心帮助。