2016 级计算机系统（1）大作业

作业概要：

使用 Verilog（其他 VHDL 语言亦可）实现一个 32 位 RISC-V 指令集的 CPU，并烧到 FPGA 上

使用 C++程序模拟内存读写，通过 USB-UART 在计算机和 FPGA 之间交换数据。

指令集：

本次大作业使用 RISC-V 开源指令集作为目标指令集，RISC-V 与 MIPS 指令集十分相似，但去掉了一些历史包袱（例如分支延迟槽、LO, HI 寄存器等）

RISC-V 的文档可其参见官方网站上的 User Level ISA Specification (https://riscv.org/specifications/)

需要实现的指令包括文档中 RV32I 2.1-2.7 提到的所有指令（部分指令如 fence 可能只有当实现了乱序执行时才会有实际作用）

内存：

受实际条件的限制，FPGA 开发板上没有专用的 RAM 芯片，因此所有的内存读写将被发送到计算机上进行模拟

FPGA 到计算机之间的通信将通过 UART-USB 协议进行，FPGA 发出的 UART 信号会被板上的 USB 芯片接收并以 USB 方式发送到计算机上，操作系统通常会将其以串口的形式暴露给其他程序。

UART 协议的细节你可以很容易得 Google 或百度得到，或者也可以参考助教自己的（并不优雅的）实现 (https://github.com/sxtyzhangzk/mips-cpu/blob/master/src/cpu/uart_comm.v)

关于与 FPGA 通信的程序你可以在 https://github.com/sxtyzhangzk/cpu-judge 处得到，你需要根据你自己的实现情况完成 adapter.cpp 的内容（注意该代码尚未完整测试，可能会有修改）

默认可用的物理内存空间的大小为 256MB (0x00000000-0x0fffffff)，其中最低的 4KB (0x00000000-0x00000fff) 请将其视为 IO 设备且**不应该被缓存**

字节序请使用小端序

另外 UART 协议的通信速率非常低，因此十分推荐有能力的同学在 FPGA 实现一个 Cache

CPU 结构：

你实现任意结构的 CPU（如多级流水，Tomasulo 等）

推荐基础相对薄弱的同学从标准的五级流水线开始

运行、测试和评分：

每位同学最终需要提交的是由 Vivado 生成的 Bitstream 文件和你自己修改过的 adapter.h 和 adapter.cpp 文件（提交方式待定）

运行时输入数据可以从地址 0x100 出读出，每次一个字节，输出数据应写入 0x104，每次一个字节

运行时间的计算方法为：从第一次内存读取开始计时，直到程序结束，程序结束的判定方式为向特定地址(0x108)处写入一个 0xff

根据最终完成的情况决定根据 Code Review 或由运行时间进行天梯排名给分

如果根据 Code Review 给分，你的设计架构将会作为主要评分依据，包括不同于 5 级的流水线、缓存、乱序执行，抑或是尝试设计全新的架构

如果根据天梯排名给分，你依然可以在 Code Review 中展现自己的创新来获得加分

测试数据将由助教提供一部分，其他数据由各位同学提供，原则上每位同学应当提供至少一份测试数据（汇编或 C 均可）

以下情形可酌情给予 Bonus:

1. 积极帮助其他同学并受到广泛好评
2. 提供多份高质量的测试数据
3. 实现 RISC-V 的特权指令并尝试在你的 CPU 上面运行 Linux（成功可获得大量 bonus，失败但能分享心得亦可获得 bonus）

Tips：

板子上的外部时钟频率是 100MHz，但是你可以自行添加一个锁相环（在 Vivado 中以 IP 核形式提供）来调节时钟频率，同时锁相环的锁定信号亦可作为上电复位信号使用

参考资料：

关于如何烧板子：https://wenku.baidu.com/view/c9486ce6a45177232f60a2f6.html

RISC-V 工具链：https://github.com/riscv/riscv-gnu-toolchain

助教自己的（十分不优雅的）MIPS CPU 实现：https://github.com/sxtyzhangzk/mips-cpu/