**计算机组织与体系结构实习Lab 2**

**1、RISC-V工具链环境准备**

下载并安装UCBerkeley提供的toolchain，Github：<https://github.com/riscv/riscv-tools>

注意：按照网站指示下载，并进行交叉编译。如果下载过慢，也可以网盘下载：[https://pan.baidu.com/s/1eR80gMY](https://pan.baidu.com/s/1eR80gMY" \t "_blank)

网盘下载后，执行下述命令。

$ tar xzvf freedom.tar.gz

$ cd freedom

$ git submodule update --init --recursive

由于安装过程需要安装sbt，如网速过慢，解决方法可参考：[https://cnrv.io/。](https://cnrv.io/%E3%80%82) 交叉编译时间比较长，请耐心等待。本lab需要使用的是RISC-V的64位工具链riscv64-unknown-elf-gcc。

**2、RISC-V功能级模拟器**

完成RISC-V rv64I（64位基本指令系统）的模拟，包括：

1. 需要实现的RISC-V rv64I的部分指令，可参考其中的 RISCV-simple-greencard.pdf。如需要查阅其他指令编码，可参考riscv-ISA-list.pdf。
2. 利用RISC-V编译器riscv64-unknown-elf-readelf完成用户程序的生成。
3. 模拟器应可以加载ELF格式的用户程序，支持程序的指令级运行。
4. ELF文件格式的解析部分，可采用BFD库函数，或使用相关开源代码。
5. 建议预留好多周期、流水线的相关扩展接口。
6. 建议为性能评测，例如跳转预测和存储层次预留相关接口。
7. 提供了一个简易的指令模拟器模板，可根据模板修改，也可自行编写代码。

**3、RISC-V性能模拟器**

完成RISC-V rv64I（64位基本指令系统）的性能模拟支持，包括：

1. 基于功能级模拟器的实现（如果有）进行扩展，使其支持性能模拟。
2. 自行划分流水线线阶段，建议参考教材设置为5级。
3. 对指令执行周期的设置，简易实现为可配置化的宏或者配置文件，主要是乘法、除法、取余等。默认设置应合理（例如64位乘法速度不应快于32位乘法）。
4. 应能够处理数据相关和控制相关。
5. 可考虑配置实现支持不同的跳转策略。
6. 对存储的访问延时配置，建议预留接口，以支持后继实验的实现。

**4、 检查要求**

1. **功能模拟和性能模拟，可以分别完成，也可以合并完成。**
2. 使用RISC-V提供的工具链交叉编译lab1中所写的程序（至少包含快速排序、矩阵乘法、求Ackermann函数，可以增加且数目不限），使用qemu模拟器检查交叉编译程序的正确性。
3. 使用自己实现的模拟器，运行自己所编译生成的RISCV可执行程序，并统计其动态指令数、周期数等性能计数信息。模拟器应可查看寄存器值和内存信息。
4. 模拟器应支持单步模式，并支持单步执行后的寄存器和内存查看。
5. 最终提交：lab报告（格式不限，可参阅模板）、模拟器源码、执行脚本或简要说明。
6. 如果实现带有图形界面的，建议提供命令行接口。
7. 如果实现完整的ISA，或实现其他扩展，**可加分**。
8. 如果更多测试程序并执行正确，**可酌情加分**。

**注意1**：编译C语言代码时，编译参数可参考： riscv64-unkonwing-elf-gcc -Wa,-march=rv64i -o xxx xxx.c ，以保证生成的用户程序代码中尽量使用rv64I中的指令。可以根据需要改变支持的指令。

**注意2**：可使用RISCV的qemu模拟器进行对照检查，课程网站上提供了一个源代码软件包可供安装。