ERISC程序可视化——大作业报告

1. 小组成员

* 软93，刘怡豪，2018010560
* 软93，周航，
* 软92，翁兆天

1. 程序设计思路

本程序采用面向对象程序设计思路，将程序拆解为不同对象的组合：

1. Computer（计算机）类，完成ERISC指令的运行、流程控制与位图文件的输出；
2. Register（寄存器）类，完成ERISC指令集中与寄存器有关的指令；
3. Memory（内存）类，完成ERISC指令集中与内存有关的指令；
4. Stack（栈）类，完成ERISC指令集中于栈有关的指令；
5. Input（输入）类，完成输入文件的解析与ERISC指令行的获取。

在程序运行时，首先将输入文件（\*.ERISC）送到Input类中解析，针对每一行都生成一条相应的ERISC指令，并将其放入向量存储；然后Computer类不断读入Input类中存储的ERISC指令并运行，换句话说，即根据不同的指令调用Register类、Memory类或Stack类相应的成员函数，并控制Input类行序号的跳转，如此之后，便同时完成了指令运行与流程控制。

得益于类良好的封装特性，本程序模块之间弱耦合，在总体程序框架设计结束之后，便可以容易地进行分工合作。同时本程序使用诸如重载运算符、异常传播等技巧，大大降低了程序代码的冗余度，结合详细的注释，使程序可读性极大提高。本程序另一方面尤其注重输入的鲁棒性与提示信息的完备性，以此给予使用者更友好的体验。

1. 基本功能完成情况

本程序完成了所有的基本功能，包括：

1. 内存访问指令（Memory类）：load, store
2. 栈操作指令（Stack类）：push, pop
3. 寄存器计算指令（Register类）：mov, add, sub, mul, div, rem, and, or
4. 控制指令：行标识、无条件跳转（jal）、条件跳转（beq, bne, blt, bge）、函数调用（cal）、函数返回（ret）
5. 指令操作可视化：.bmp文件输出（draw）、.txt文件输出（end）
6. 必做任务：task1.ERISC, task2.ERISC, task3.ERISC, task4.ERISC
7. 扩展功能说明

本程序扩展了以下功能，包括：

1. 寄存器计算指令（Register类）：xor（异或）、not（位反）
2. 输入文件（.ERISC）鲁棒性：

只要输入程序符合语法，本程序可以处理任意数量多余的空格。

1. 输入文件（.ERISC）注释：

本程序支持双斜杠“//”单行注释与井号“#”单行注释。

1. 输入文件（.ERISC）错误位置追踪：

得益于异常向上传播的机制，本程序能够追踪输入文件中哪一行出现了哪一类型的错误，并及时输出在显示器上。

1. 程序运行时间计时：

本程序能够计算指令运行的时间，并输出在显示器上。

1. 关于时间效率

本程序经过详细统计计算后得出，执行一条指令大约需要0.0115ms的时间，或1s大约能执行指令86956条，因此指令集的运行效率不是很高，运行任务四大约需要240s的时间，约的时间复杂度。不同的硬件设备测量得到数值可能不同，因此仅供参考。另外本程序输出“.txt”文件大约需要10~20s的时间。

1. 小组分工情况

* 刘怡豪：程序框架设计、指令集实现
* 周航：位图文件输出
* 翁兆天：ERISC文件输入