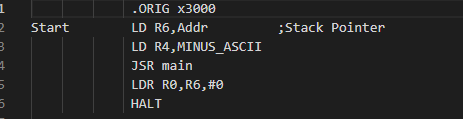
Lab3 实验报告

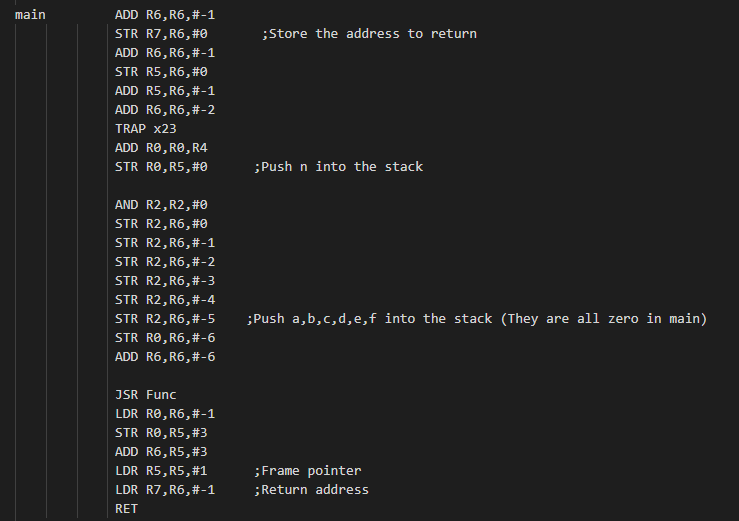
王嵘晟

PB17111614

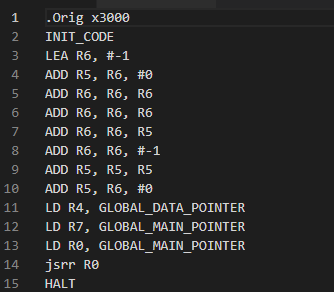
1. 程序区别
   1. 函数

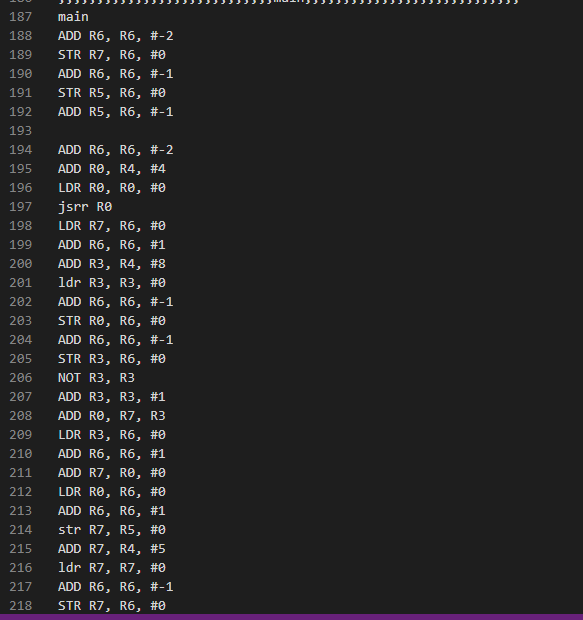
Lab2中提交的我自己写的LC3程序是通过提前把寄存器R6和R5分别存储栈指针和帧指针，main函数中除了实现C语言中main函数的本来功能：读入n的值并传参调用func函数以外，还将n的值以及a,b,c,d,e,f的值给压入栈中，并移动栈指针。但压栈过程是压入元素结束后一起移动栈指针，栈指针直接减去压入元素个数（这么写逻辑有点乱）而func函数通过栈的存储模式实现递归调用





而compiler出来的LC3程序同样把R6作为栈指针，但没有用到帧指针。Main函数初始化了栈的存储，每压入一个元素，栈指针-1，逻辑清晰。Func函数的实现逻辑和手写函数基本一致。





其他区别：手写函数跳转使用的是JSR和BR，而编译器生成函数多用JSRR和JMP。

1.2 prolog 和 epilog（“保存现场”和“恢复现场”）

两个程序都是通过把栈指针和帧指针的值存入栈里，以实现“保存”

而恢复过程就是把栈指针和帧指针重新从栈里读取出来并存入R6 R5，完成“恢复”，并通过对寄存器内容的逻辑运算将栈指针指向存入最终结果的位置，把它存入R0

* 1. 栈操作

Lab2中的手写程序制定了栈底的位置在xFFD0,直接LD得到

而compiler出来的LC3程序通过“巧妙”的逻辑运算实现栈底位置在xEFFF。

Lab2程序的栈的结构为最底层为栈指针R6，紧接着是帧指针R5，然后是n的值，然后是最终结果的值，然后是a,b,c,d,e,f的值。

而compiler出来的LC3程序最底层为最终结果，然后是栈指针R6，然后是n的值然后是a,b,c,d,e,f的值。

1. Pros and cons（优势与劣势）

Lab2中的手写程序代码行数更少，但是逻辑不够清晰。

编译器编译出来的程序代码代码行数多，但是逻辑清晰。编译器编译出来的程序对寄存器的使用个数更少。

1. 整数调用机制是怎么样的？

整数调用机制基于八个寄存器R0~R7, 前两个寄存器也多用于返回数值。最多XLEN位宽的标量在单个参数寄存器中传递，如果没有，则在值堆栈中传递。 当传入寄存器时，比XLEN位窄的标量根据其类型的符号加宽，最多为32位，然后符号扩展为XLEN位。宽度为2✕XLEN的标量在一对参数寄存器中传递，如果没有，则在值堆栈中传递。 如果只有一个寄存器可用，则低位XLEN位在寄存器中传递，高位XLEN位在堆栈上传递。

在基本整数调用约定中，可变参数以与命名参数相同的方式传递，但有一个例外。 具有2✕XLEN位对齐和大小最多2✕XLEN位的变量参数在对齐的寄存器对中传递（即，该对中的第一个寄存器是偶数编号的），或者如果没有可用则在值堆栈上传递。 在堆栈上传递了可变参数之后，所有将来的参数也将在堆栈上传递（即，由于对齐的寄存器对规则，最后一个参数寄存器可能未被使用）。