Lab A 实验报告

PB20000051 傅申

本实验通过 C++ 程序实现对 LC-3 汇编程序的编译. 基本框架是通过三次对汇编程序的逐行扫描来将其转化为字节码.

1 命令行参数

本程序共有 6 个命令行参数, 如下:

命令行参数	说明
`-h`	显示帮助信息
`-f [filepath]`	指定汇编程序文件路径
`-o [filepath]`	指定输出文件路径
`-d`	允许输出调试信息
`-e`	允许输出错误信息
`-s`	开启 16 进制模式

2 三次扫描

汇编器主类 `assembler` 的结构如下:

```
class assembler
{
private:
    label_map_tp label_map;
    std::string TranslateOprand(int current_address,
                                std::string str,
                                int opcode_length = 3);
public:
    int assemble(std::string input_filename,
                 std::string output_filename);
};
```

其中的 `label_map_tp` 可以看作是汇编文件中的 label 到值/地址的映射 (主要是地址). 在 `assembler::assemble()` 中对输入文件进行了三次逐行扫描, 将对应的字节码输出到文件

中. 在扫描过程中, 我们会对每行打上标记 (由枚举类型 `LineStatusType` 定义), 这些标记及其含义如下

标记	含义	值
`lPending`	待处理	-1
`lComment`	全行注释	0
`lOperation`	代码	1
`lPseudo`	伪代码 (如 `.ORIG`)	2

这些标记存储在函数的局部变量 `std::vector<LineStatusType> file_tag` 中, 类似这样的局部变量还有:

```
std::vector<std::string> origin_file;  // 原始行
std::vector<std::string> file_content; // 行中除去注释和空白字符的内容
std::vector<std::string> file_comment; // 行中的注释
std::vector<int> file_address;         // 每行对应在 LC-3 内存中的地址
```

这些变量帮助我们更好的扫描文件, 三次扫描的主要操作如下

2.1 Scan #0

- 除去每行开头和结尾的空白字符, 将剩余的文件内容存在 `origin_file` 中
- 将处理后的内容中所有的小写字符转为大写字符, 分离内容和注释后分别存在 `file_content` 和 `file_comment` 中
- 给每行打上标签 `lPending`, 如果整行都是注释, 则打上标签 `lComment`

2.2 Scan #1

- 判断文件是否格式正确 (以 `.ORIG` 开始, 伪代码是否合法)
- 处理伪代码和标签, 确定每行的地址
- 给每行打上对应的标记

2.3 Scan #2

- 翻译每行的代码并输出到文件

经过三次扫描, 即可实现对汇编文件的转化, 具体细节详见代码.