项目报告

编程语言：Python　　前端：PyQt5

阶段一 CPU 的设计：

```
PS C:\Users\Lenovo\Desktop\ICS\ICSPJ> python test.py --bin "python src/cpu.py"
All correct!
```

新建 CPU 类，有内存，寄存器，条件码，状态码等成员变量。其中内存，寄存器用字典来存储，条件码和状态码用 int 来存储。此外，CPU 类中还用列表存有 yo 文件的所有行的内容。

CPU 类下有 trans_small_end，create_yaml,，Build_env，process_line ,run 五个函数。

trans_small_end 函数用来把小端转换为大端。实现的思路来源于 lab1 中 BitReverse，通过分治法，把 64 位数字的前 32 位和后 32 位进行交换，再把前 32 位的前 16 位和后 16 位，后 32 位的前 16 位和后 16 位进行交换，以此类推，直到转换为大端为止。

create_yaml 调用 Python ruamel_yaml 库函数，将信息以 yaml 文件的形式输出。

Build_env 函数根据 yo 文件中所有的字节来构建内存。

process_line 函数处理指令。方法是先读取前两个字节，确定指令的类型，再对应处理。

run 函数可以说是总函数，调用 run 函数后 CPU 先执行 Build_env 函数构建内存，然后执行 yo 文件的所有指令，并输出 yaml 文件。

阶段二

运用 PyQt5 库，以应用程序的方式展示 CPU 运行时的信息。

UIcpu.py 中 MyUi 类对应后端的实现，继承的 Ui_MainWindow 类在 UI.py 文件中，对应前端的实现。

后端的设计：

MyUi 类：

showReg 函数实现寄存器文本框的显示或隐藏。

showMem 函数实现内存文本框的显示或隐藏。

getFont 函数实现选择字体功能。

openMesg 函数实现打开文件的功能。

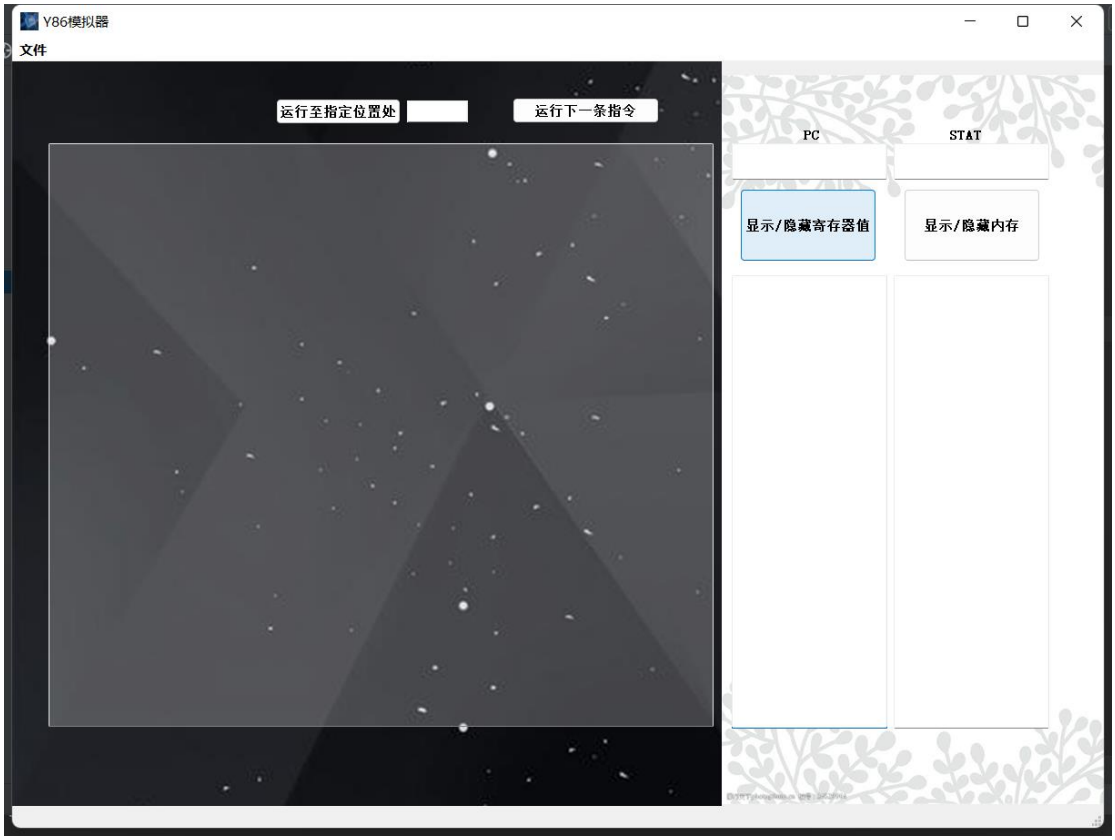
run_error 函数实现运行异常的报错。

showtext 函数实现 CPU 信息的实时更新。

runline 函数实现运行下一条指令功能。

runto 函数实现运行到指定位置的功能，类似于断点

前端的设计：

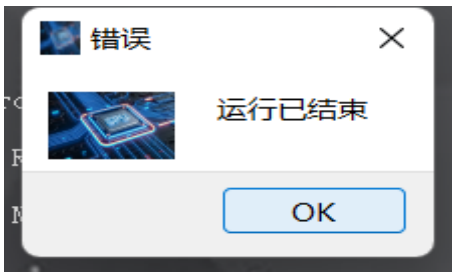


上图是应用程序的界面。

左上角“文件”菜单栏中有“打开”选项，用来打开 yo 文件，也可以通过直接拖拽的方式来打开 yo 文件。yo 文件内容展示在左半边的文本框中。此外，菜单栏中还有“字体”选项，用来选择字体。由于有些字体空格和文字大小不一致，所以我将 yo 文件对应的文本框字体设置为固定值，避免显示混乱

右侧显示寄存器的信息，可以选择显示或隐藏信息。点击“运行至指定位置处”按钮，cpu 会运行到按钮右侧输入的地址处（如 0xa），类似于 gdb 的断点功能。点击“运行下一条指令”，cpu 会运行下一条指令并更新信息。当所有指令运行结束或出现异常时，

会弹出对话框报错并终止运行。



代码运行方式

阶段一:在 Pycharm 中打开 src 文件夹,在终端中输入 python test.py --bin "python Stage1/cpu.py"

阶段二：在 Pycharm 中打开 src 文件夹，运行 Stage2 目录下 UIcpu.py 文件，或在终端中输入 python Stage2/UIcpu.py.