8086 汇编“打字游戏”小程序实验报告

数据科学与计算机学院 2017 级 软件工程 5 班 梁文杰 16303050

一、实验目的

1. 了解 x86 平台下的程序开发过程以及掌握 Emu8086 工具的使用；
2. 与 C/C++语言做对比，进一步加深对计算机底层知识的了解，深入体会程序优化问题。

二、设计思路与分析

设计思路：先从文件中读取文章信息 → 打印文章标题和文章内容 → 键盘输入后与文章当前字符进行匹配 → 匹配正确或错误时进行相关处理 → 游戏结束

后期添加游戏开始前的游戏说明，加入了分数项，处理加分扣分，提升用户体验。

程序的关键点：

1. 实现文件的读取；
2. 实现定点打印特定内容以及改变文字的颜色；
3. 实现数字转 ASCII 码并输出；
4. 实现键盘字符的输入与文章当前字符的匹配；
5. 实现程序的延迟（休眠）；

三、实现过程

一开始尝试用 C++语言实现最想法，希望根据 C++的程序结构在 x86 程序中还原，但在编写 x86 的过程中遇到较多困难，导致进展缓慢。

如寄存器数据的相互污染，导致得到了垃圾数据。在处理数字转 ASCII 码时，将数字除以 10，得到余数转 ASCII 并输出，但出现了永远除不完的问题，排查后发现，进行除法和输出的两段程序均使用了 AX 变量，导致数据出错。解决方法为输出前对 AX 寄存器的保护及输出后的恢复，另外一些情况则使用申请内存存放全局变量，用栈存放临时变量的方法避免寄存器数据的冲突。

又如在进行定点打印特定内容时，光标定位到特定位置时无法改变该位置文字的颜色，解决方法为程序中断指令实现光标定位，利用显存偏移地址更改文字颜色。

到后来，学会使用宏与子程序之后，将部分功能代码封装，调用这些宏与子程序进行编写程序，效率得到了翻倍，甚至体会到了编写高级语言的那种快感。但宏与子程序的调用时也时要特别注意寄存器数据的保护，以及调用返回时的恢复，否则容易产生 bug。

四、效果评价

简单总结一下，这次实验算是比较成功，达到实验目的。大致熟悉了 Emu8086 工具的使用及调试功能，同时 x86 汇编语言的编写也比较熟悉，学会了使用宏与子程序，能够灵活使用栈空间与内存空间。

CPU 对数据的打印是基于字符串的，所以一切其他类型的数据包括整数类型，浮点数类型都要先进行字符串的转换才能够输出，c++ 中的 cout 语句正是帮我们实现了这个步骤。输入输出的缓存区确实能够提高程序的执行效率，减少单独打印的次数，能够减少指令数，减少机器寻址的过程。

与 C++ 相比，x86 汇编语言对于屏幕定点定位以及较为简单的逻辑操作，简单运算，能够得到精准的代码，有较大的优势。在编写代码时，能有更大的尺度去优化自己的程序。但对于较为复杂的运算，浮点运算那些，汇编便相形见绌了，编写效率上无法与高级语言进行比较。因此可以得出一个结论在 C++ 代码中适当内联汇编是最佳选择。