

**汇编实验报告**

学院： 计算机科学与技术学院

专业： 软件工程

班级： 1619304

学号： 161930131

姓名： 马 睿

# 一、文件清单：

|  |  |  |
| --- | --- | --- |
| 文件名称 | 文件内容 | 备注 |
| loader.asm | 程序加载器 |  |
| Loader.com | 加载器转换的二进制文件 | 放在虚拟硬盘0号扇区 |
| Div\_Err.asm | 除法出错中断处理程序 |  |
| Div\_Err.com | 除法出错程序二进制文件 | 放在虚拟硬盘10号扇区 |
| KeyBoard\_Int.asm | 键盘中断处理程序 |  |
| KeyBoard\_Int.com | 键盘中断二进制文件 | 放在虚拟硬盘20号扇区 |
| Extend\_IO.asm | 扩展IO中断处理程序 |  |
| Extend\_IO.com | 二进制文件 | 放在虚拟硬盘100号扇区 |
| Program.asm | 第八章时钟程序 |  |
| Program.com | 二进制文件 | 放在虚拟硬盘200号扇区 |
| Snake.asm | “贪吃蛇”小游戏 |  |
| Snake.com | 小游戏对应的二进制文件 | 放在虚拟硬盘50号扇区 |

# 二、完成的功能：

1. 程序加载器

主要读取制定工作程序的首个扇区，加载相应程序

1. 中断系统程序：
   1. 键盘中断处理程序：当按下”M”，”A”，”R”，”U”，”I”中的任意字符，会使DOS界面输出1，按下其余字符后不做任何输出。即只会识别”M”，”A”，”R”，”U”，”I”这五个字符（不区分大小写）
   2. 除法中断处理程序：显示除法溢出的字符串，并返回到溢出之后的下一条指令继续执行。
   3. 扩展显示I/O程序：由自己手动设置更改90H号中断向量。当按下”R”，”G”，”B”（不区分大小写）后，之后所有按下的按键都会展示相应的颜色（红色R、绿色G、蓝色B）。默认颜色是蓝色。
2. 加载第八章时钟程序：

读取CMOS时钟模块信息，将系统时间实时显示在屏幕中央，并保持与系统时间同步，等待用户按下”#”键，退出该程序并返回加载器。

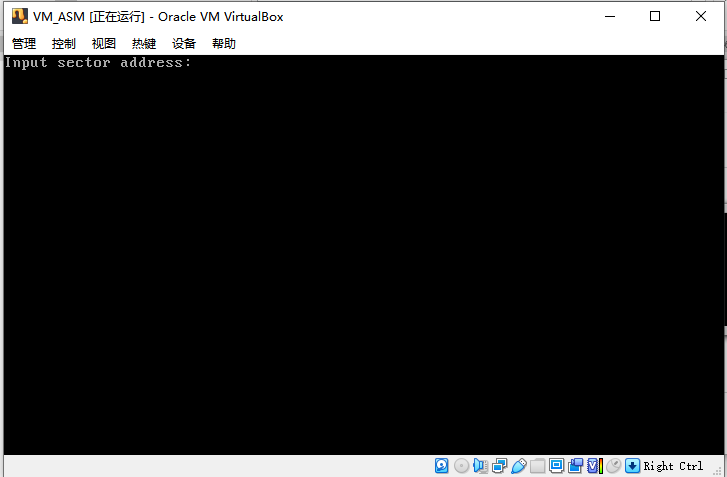
1. 用汇编语言编写的“贪吃蛇”小游戏

# 三、功能解释：

## 1.界面含义：

程序开始时，界面上会提示输入程序被写入的扇区号，从而跳转到对应的程序进行执行。

若输入的扇区号不合法，也会做出提示。



## 2.加载第八章程序

读取CMOS时钟模块信息，将系统时间实时显示在屏幕中央，并保持与系统时间同步，等待用户按下’#’键，退出该程序并返回加载器。

## 3.中断处理程序

在初始界面输入扇区号，并输入回车键，即可到达对应的中断处理程序界面。

1. 键盘中断处理程序

首先设置9号中断向量指向新的由自己编写的键盘中断处理程序，然后接受用户按键，直到用户按下回车键为止。

新的键盘中断处理程序：通过判断按键的扫描码，判断是否是”M”，”A”，”R”，”U”，”I”中的某一个字符，若是，则输出1；否则不做出任何响应。

1. 除法出错中断处理程序

首先开中断，接着保护通用寄存器和段寄存器DS，在除法中断发生后，显示提示信息；最后恢复被保护的寄存器，利用中断返回指令IRET实现中断返回。

1. 扩展显示I/O程序

修改90H号中断处理程序，其功能是以TTY方式显示带属性的字符。

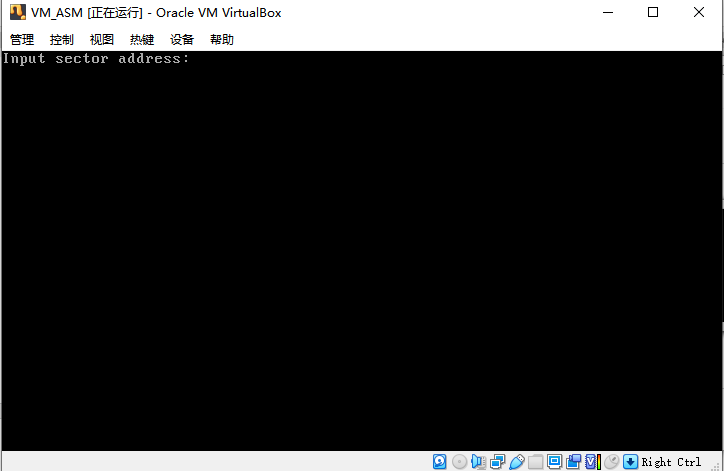
演示过程：首先工作程序设置90H中断向量，使其指向扩展的显示I/O程序，然后为了体现演示效果，当输入当按下”R”，”G”，”B”（不区分大小写）后，之后所有按下的按键都会展示相应的颜色（红色R、绿色G、蓝色B）。默认颜色是蓝色。

## 4.贪吃蛇小游戏

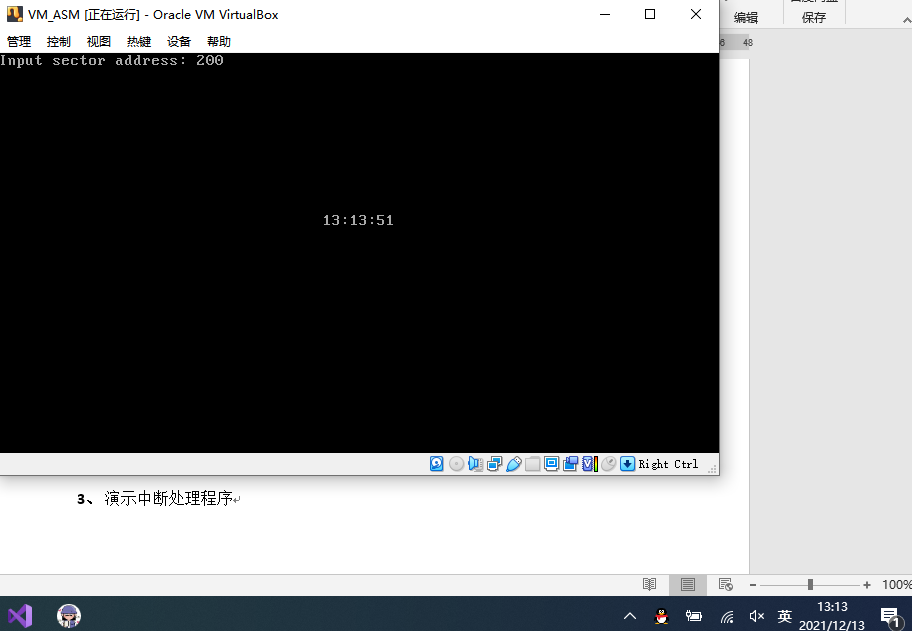
1. 按下回车键开始游戏
2. 按键盘上的W、A、S、D分别控制蛇的上左下右移动
3. 右上角展示当前的分数
4. 游戏结束后，按ESC键退出游戏，返回加载器

# 四、程序演示：

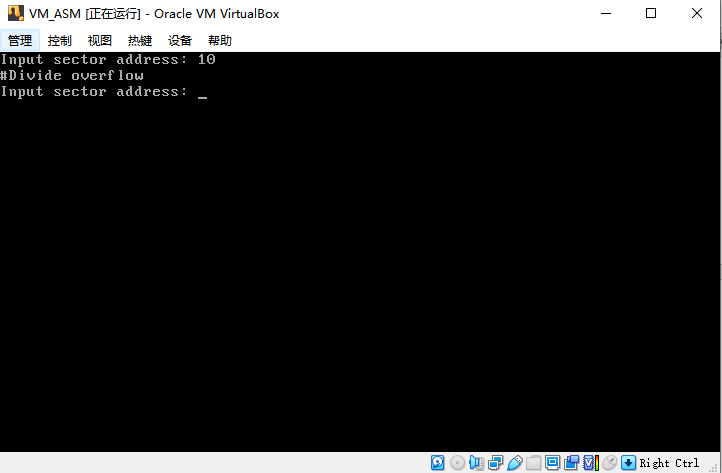
首先是初始界面：



1. 加载第八章程序，读取CMOS时钟模块信息，将系统时间实时显示在屏幕中央

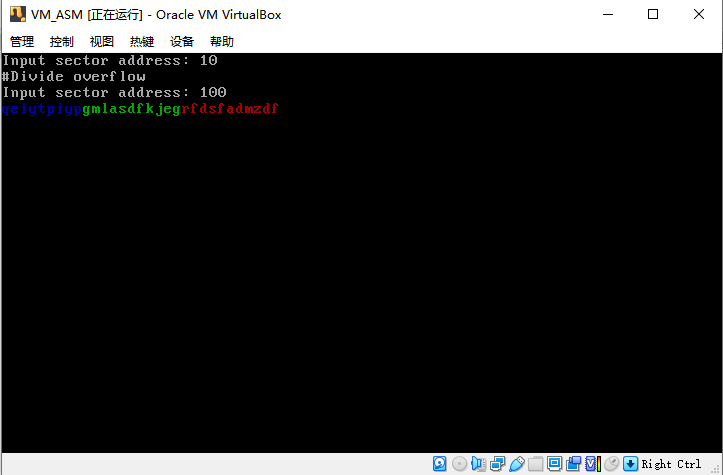


1. 演示中断处理程序
   1. 通过输入10，演示除法中断程序



屏幕上输出 “Divide overflow”，并返回加载器

* 1. 接下来输入100，演示扩展IO中断程序



在屏幕的中央显示输入对应颜色的字符。根据按键扫描码确定输出何种颜色的字符串。

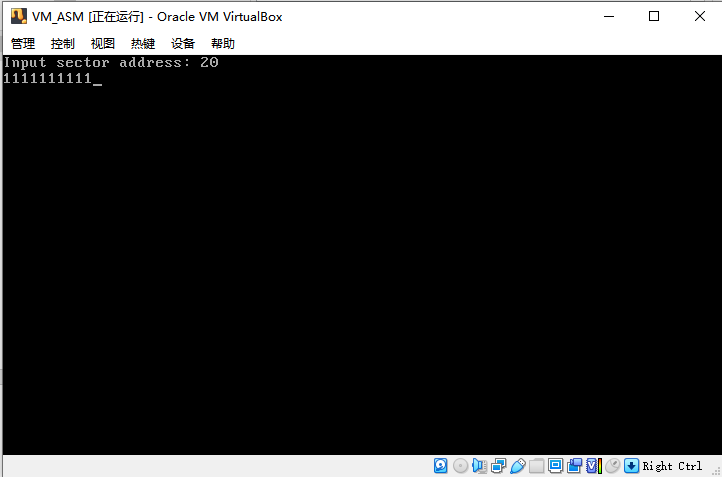
按下‘R’ 或‘r’键，之后按下的字符以黑底红字的形式输出显示。

按下‘G’或‘g’键，之后按下的字符以黑底绿字的形式输出显示。

按下‘B’ 或‘b’键，之后按下的字符以黑底蓝字的形式输出显示。

初始会以黑底蓝色进行显示

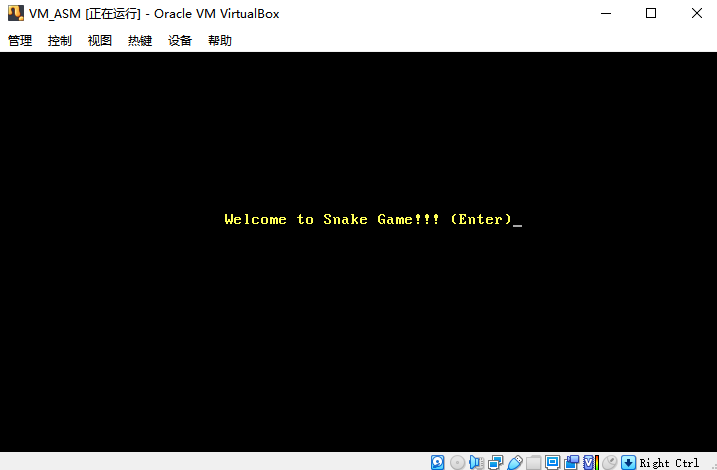
* 1. 接下来通过输入20，演示键盘中断程序



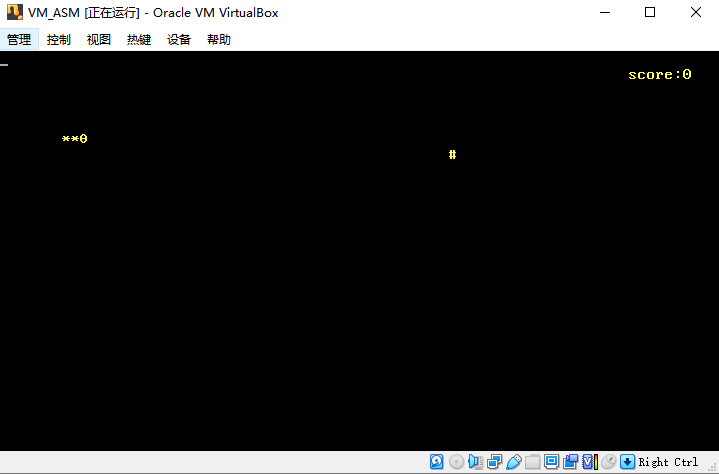
按下”MARUImarui123polqwer”，会输出10个1，因为匹配到了10个相关字符。

1. 汇编语言编写的贪吃蛇小游戏

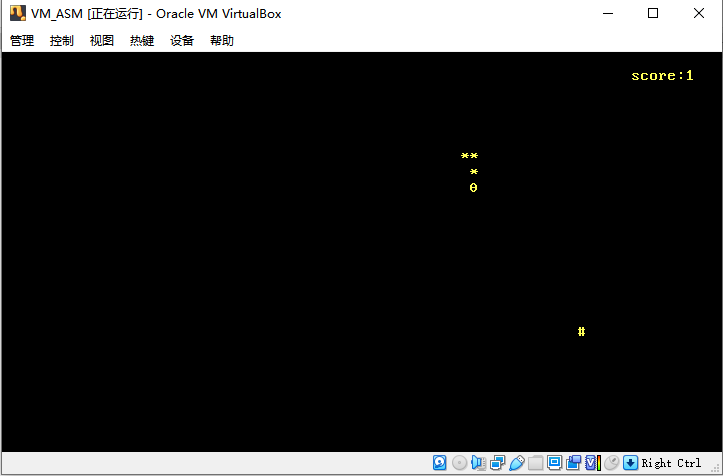
按下 ’Enter’ 开始游戏



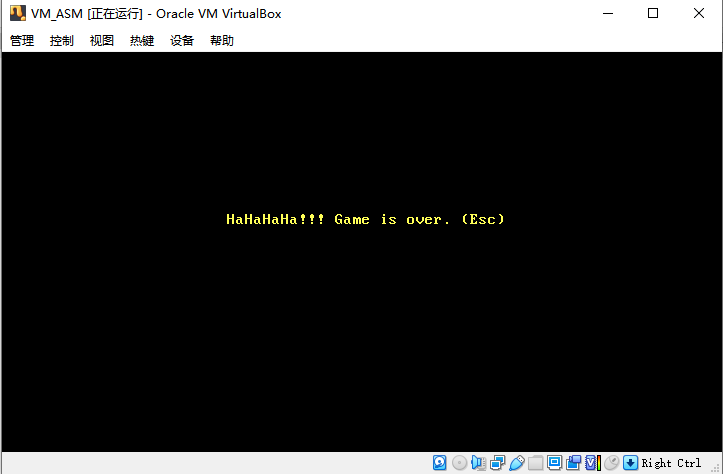
分别按下’W’, ’A’ ‘S’, ‘D’,蛇的上左下右移动



吃到 ‘#’ 得到1分



按下 ’ESC’ 键退出游戏，返回加载器



# 五、遇到的问题：

## 整体架构的理解

因为先前写汇编作业的时候，都是在.c或.cpp文件内编写的，只需要完成相应的功能即可，不需要考虑如何加载程序、程序入口、程序长度等细节。而此次实验需要考虑到这些方面，不是简简单单的实现功能就可以了。另外，原先的输入、输出可以使用scanf、printf等输出函数，而现在需要用到中断指令，如何处理好输入输出以及数据的准备就尤为重要。

## 签名不匹配

在修改过加载器代码中的签名为”MR”后，开始编写除法出错中断程序，将签名改为了”MR”,然后利用fixvhdwr将程序写入了对应的扇区。

在执行除法出错中断程序时，发现总是输出Invalid code，起初以为是程序实现有问题，检查发现并没有，然后查看输出该错误信息的可能情况，发现应该是和签名有关。仔细分析发现，原先”YANG”的长度是4，而”MR”的长度是2，在获取程序的签名时是截取4个字节的数据，所以导致签名不匹配，报了这个错误。最后将签名改成了“MARY”

## 无法将程序写入磁盘

在写好程序加载器后，准备用fixvhdwr将程序写入磁盘。然后选择磁盘后，总是报错：无法打开虚拟硬盘。搜了好久，以为是虚拟化的问题，询问室友，室友觉得是因为没有开Hpyer-v服务的原因，下载该服务并开启后，发现并没有起作用。网上也找不到答案，又怀疑是磁盘的问题，将我的磁盘用室友的电脑打开时并没有报错。

最后想到，我的VirtualBox文件夹操作一般都是需要管理员权限的，而我在打开fixvhdwr时不是用管理员身份打开的，可能是因为这个导致没有权限修改磁盘文件。最后用管理员身份打开fixvhdwr，发现可以将程序写入磁盘了。这一个问题就花费了我一个晚上的时间！

## 纯汇编语言开发的困难

在上学期完成计组的实验和课程设计，接触到了许多计算机底层的知识：在完成计组实验时，阅读了大量的汇编程序；在完成课设时，解除并用c语言实现了一定量的理论课学到的分页、中断等知识。我觉得这两个过程为我这学期的汇编语言理论课和编写汇编程序打下了坚实的基础，入手实验也比较得心应手。

但是这中间也发现了许多难点：

1. 遇到问题资料查找麻烦。很多问题在网上很难找到甚至找不到答案，虚拟机环境问题、工具使用问题等。而使用高级语言遇到的问题基本都可以获取到很多答案，基本总有一个可以解决自身出现的问题。
2. 调试不方便。原先在完成作业时，还可以利用VS中的断点，对嵌入汇编进行调试，查看程序出现的问题、在哪出现了问题。而此次实验要完全使用汇编语言实现，无法用先前的方法进行调试，只能用人脑模拟运行程序查错，再写入磁盘查看虚拟机的输出信息判断程序是否正确，过程繁琐且效率不高。
3. 工具少。支持高级语言的工具非常多，无论是测试工具还是编码工具都，而且提供的很多功能，如语句错误、变量重复定义等；而此次实验完全靠自己进行查错、“变量”检查等工作。

# 六、心得体会

我记得当时在上C语言的时候，老师总是会要求能用加法实现的就不要用乘法，能用循环实现的就不要用递归等等，当时根本不理解为什么，为什么能用那些“简单”方法实现的，还要做复杂的转换。学了汇编语言后，现在就可以完全理解了。

而且我觉得汇编语言和高级语言有一个共同点，那就是——无论是前者还是后者，都需要精化算法，在使用更少空间的情况下，让程序执行的更快，往往实现一个功能或一个算法是很简单的，但是如何“更好”地实现就变得不容易了。在写汇编作业时，我也努力地在往这个方向靠，觉得能更好、更妙地实现程序会非常有成就感。

本来我想完成一个飞机大战的小游戏，在网上也查询了相关的资料，但是始终无法用NASM进行编译，不知道如何获取自定义段的基地址，从而获取该段中的所有数据，因此改成了一个贪吃蛇的小游戏，因为大一用C++实现过一个简易的贪吃蛇，这次想用汇编语言再次实现一下。

同时，这次考试有点失利。首先是第一道题，我错误的将char数组类型看成了int，导致整个题都写错了，等我发现这个错误的时候离考试结束只有一分钟了，没来得及改，特别特别特别遗憾；最后一题原本写完了，然后老师讲解题意后发现我误解了题意，所以又重新写了一遍。

做完这次试验，汇编语言课程也就算真正告一段落了。如果仅仅只是完成了平时布置的作业，我觉得只是掌握了汇编的一半，通过这次的实验才算是真正理解了一次汇编语言（当然不是汇编语言的全部）——如何输入输出、如何重写中断、加载程序。在尝试编写飞机大战程序时，还接触了MASM的使用方法，其实我觉得MASM写起来要比NASM更容易一些，有些语法NASM实现起来可能比较麻烦，当然NASM的适用范围更广，我觉得更应该使用NASM。

我觉得汇编语言加上上学期的计算机组成原理，还有老师的“沉浸式教学”，让我对计算机底层的理论和实践都有了更深的理解，让我从计算机的表面走向了底层。尤其清晰的记得老师上课说的一句话：软件工程不应该只是简简单单的实现一个软件，应该走向更底层。我觉得越往底层走，对计算机体系了解更深入，更有利于向高层进行扩展，对高层的掌握也就更好，也可以更熟练的使用高层。老师上课还会讲x86和MIPS的区别和联系，拓展了视野和想法，让我不仅仅局限于x86体系，尝试用不同的思路实现底层功能。

非常感谢老师这一学期，富有激情的讲课风格，相比于上学期的计算机组成原理课程（很遗憾不是老师授课），我更有动力和激情学习汇编语言，这期间也让我收获了很多。同时受老师的影响，我尝试报名了“一生一芯”活动，想通过此次活动对计算机底层有一个更深的了解，并付诸实践。

最后祝老师工作顺利，开开心心！