

**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： 1619303

学号： 061920125

姓名： 郑 伟 林

## 一、文件清单：

|  |  |  |
| --- | --- | --- |
| 文件名称 | 文件内容 | 备注 |
| loader.asm | 程序加载器 |  |
| loader | 加载器转换的二进制文件 | 放在虚拟硬盘0号扇区 |
| hello.asm | Hello world程序 |  |
| hello | 在屏幕上显示Hello world | 放在虚拟硬盘10号扇区 |
| div\_err.asm | 除法出错中断处理程序 |  |
| div\_err | 除法出错程序二进制文件 | 放在虚拟硬盘30号扇区 |
| keyboard\_int.asm | 键盘中断处理程序 |  |
| keyboard\_int | 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 | 二进制文件 | 放在虚拟硬盘40号扇区 |
| clock.asm | 时钟程序 |  |
| clock | 显示时钟的二进制文件 | 放在虚拟硬盘50号扇区 |
| snake.asm | 贪吃蛇小游戏 |  |
| snake | 小游戏对应的二进制文件 | 放在虚拟硬盘100号扇区 |

## 二、本次完成的功能：

1. 简易加载器与HelloWorld程序：

一个简易的加载器，在没有操作系统的环境中，它能够加载运行符合指定格式的工作程序。

HelloWorld程序，用以验证加载器正常工作。

1. 中断系统程序：
   1. 键盘中断处理程序：识别处理回车键与键盘上“QWERTYUIOP”11个键的扫描码，并把10个字母解释成0~9十个数字键显示。
   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。实现在屏幕中央显示彩色背景的字符。
2. 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.加载器：

一个简易的加载器，在没有操作系统的环境下可以加载符合指定格式的工作程序。

其步骤如下：

（1）准备运行环境。

（2）确定工作程序

（3）读取指定工作程序的首个扇区

（4）验证签名信息

（5）获取工作程序的长度

（6）决定工作程序被加载到内存的起始位置。

（7）搬移首个扇区的内容

（8）读取指定工作程序的剩余扇区

（9）转移到工作程序执行

### 2.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，转换得到其对应的ASCII码，显示输出用户按下的字符，并在DOS界面中央输出相应的字符。

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，依次在屏幕中央输出带有底色的字符串

"No.90H handler is ready!"

### 3.实时时钟程序

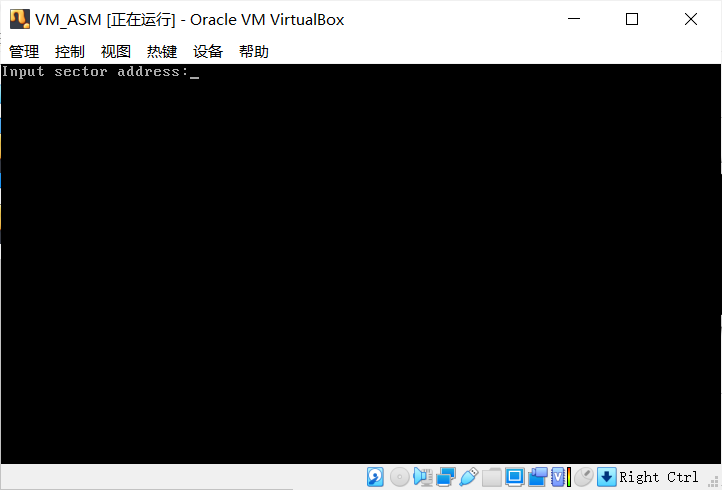
读取CMOS时钟模块信息，将系统时间实时显示在屏幕中央，并保持与系统时间同步，等待用户按下’#’键，退出该程序并返回加载器。

### 4.贪吃蛇小游戏

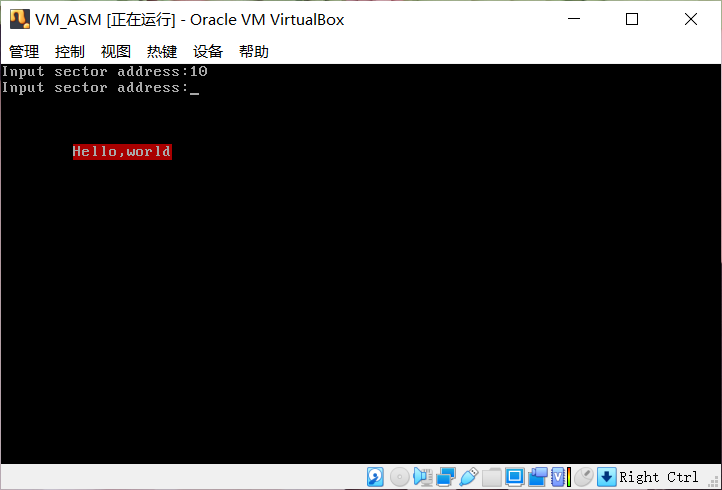
一个简易的贪吃蛇游戏。进入后直接按回车开始游戏，“W”“S”“A”“D”控制蛇上下左右移动，碰墙或碰到自己则游戏结束，吃到屏幕上的“@”则长度加一，分数加一，速度也稍微提升一些，右上角记录玩家得到的分数。游戏结束后按Esc键返回加载器。

## 四、程序演示：

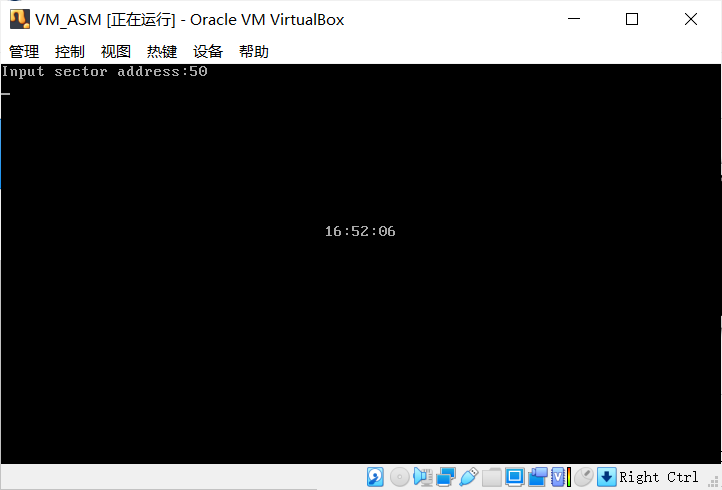
1. 首先是bootloader界面：



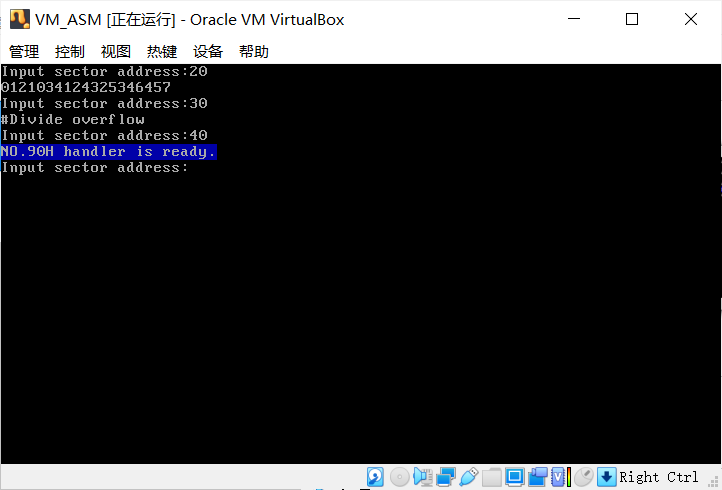
Hello\_world测试程序：



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



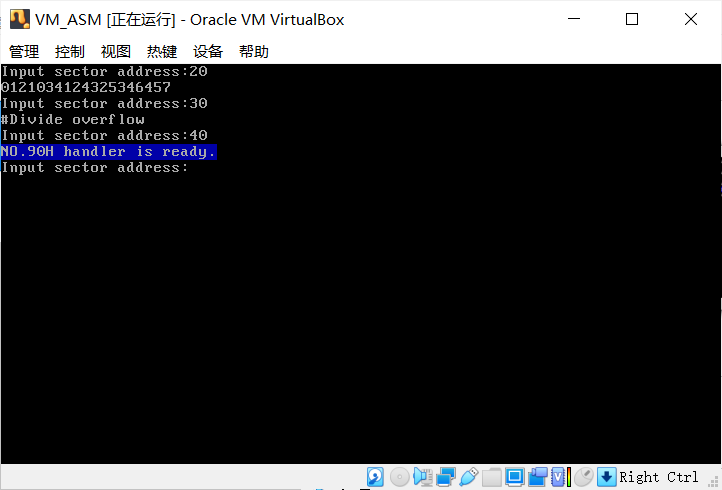
1. 演示中断处理程序
   1. 通过输入30，演示除法中断程序#



屏幕上输出 “Divide overflow”，并返回加载器

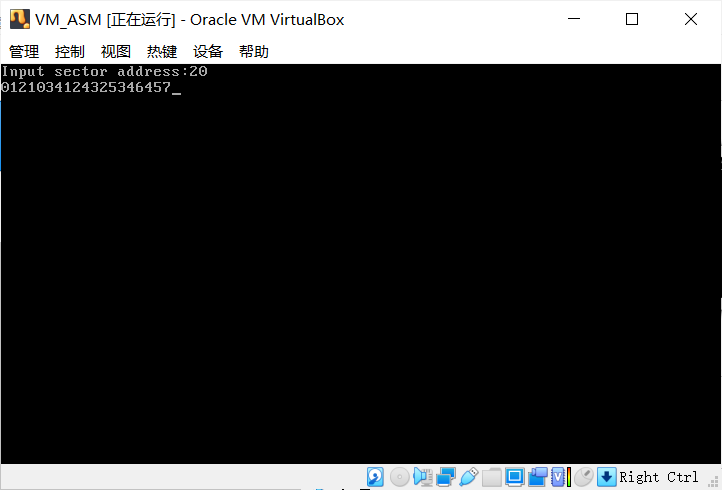
* 1. 接下来输入40，演示扩展IO中断程序

在屏幕的中央分别显示带有底色的字符串



* 1. 接下来通过输入20，演示键盘中断程序

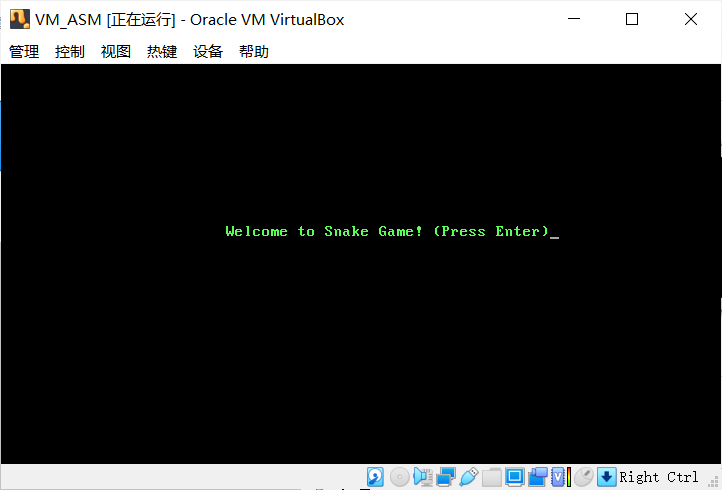
输入键盘上的Q~P键，回映射到0~9并显示在屏幕上。



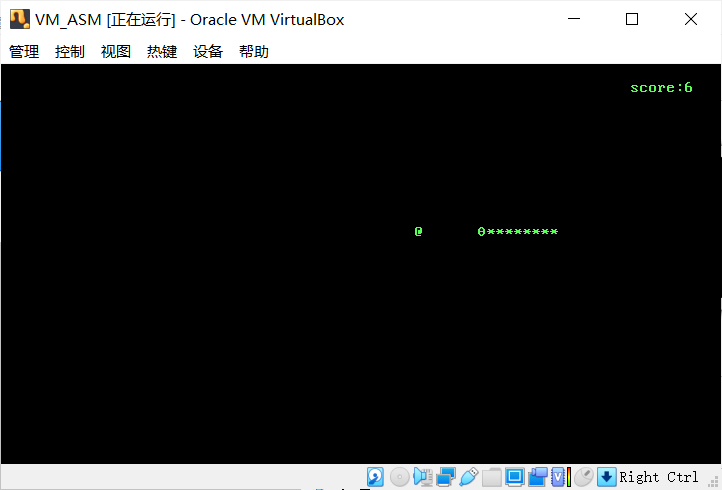
然后按回车键，可以返回加载器

1. 汇编语言编写的贪吃蛇小游戏

输入100进入贪吃蛇程序

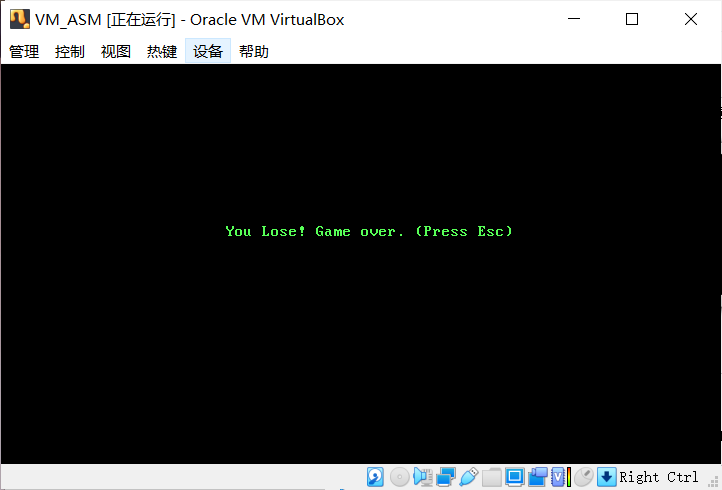


按下’Enter’进入游戏



通过’W’ ‘S’ ‘A’ ‘D’ 来控制蛇上下左右移动

当碰到自身或碰到边缘时，游戏结束



按下 ’ESC’ 键退出游戏，返回加载器

## 五、遇到的问题：

在有学长的提醒之下，本次实验避免遇到学长所遇到的问题如签名长度问题、键盘码与ASCII码问题等。

我尝试了操作系统的移植，找到一个Mike OS的操作系统，在分析其文件后，大概了解其结构：包括一个bootloader来引导操作系统，bootloader将引导操作系统内核加载，然后再加载其它程序，因此移植思路是用我们所写的bootloader来加载内核。直接改写内核为我们所期望的工作程序格式后，用bootloader加载时，只能加载出一个框架，其文字信息内容等都无法显示，推测是一些支持库没写入，但问题是要如何写入。该操作系统构建是将其程序都烧录到一个虚拟软盘中生产系统镜像，但我们直接写入虚拟硬盘后的位置该是怎样的暂未搞懂，因此移植未完成。

## 六、心得体会

通过本次汇编实验，我完成了再裸机上执行自己写的bootloader，然后引导执行一些程序。在这个过程中，我体会到了底层开发的不易，要先编写汇编代码，再用汇编器将代码编译成可执行文件，然后再存储到硬盘中，用加载器引导执行程序。但是我在这个实验中感受到了计算机的魅力，那就是用代码搭建起一个新的世界—程序的世界。希望在下学期学过操作系统后，我能理解如何用bootloader来引导操作系统的启动，希望能完成本次实验落下的遗憾，将mikeOS成功的启动。

最后，感谢本学期老师的教导。