实验一：裸机控制与扇区引导程序设计

1 实验题目

格式化三张 1.44MB 的软驱，第一张留空，第二张在首扇区写满个人信息（学号、姓名拼音），第三张编写一个引导程序，显示学号姓名，并且从左上角以 45° 发射字母，碰到边界就反弹，保留运动轨迹。

2 实验目的

1. 了解原型操作系统设计实验教学方法与要求；
2. 了解计算机硬件系统开机引导方法与过程；
3. 掌握操作系统的引导程序设计方法与开发工具；
4. 学习 PC 字符显示方法、复习加强汇编语言程序设计能力。

3 实验要求

1. 掌握 PC 电脑利用 1.44MB 软驱的开机引导方法与过程的步骤；
2. 在自己的电脑上安装配置引导程序设计的开发工具与环境
3. 参考样版汇编程序，完成在 PC 虚拟机上设计一个 1.44MB 软驱的引导程序的完整工作。

4 实验方案

按照老师“由易到难，一点一点扩充”的思想，参考实验 ppt 的引导，我将按照如下顺序进行：

1. 学习 x86
2. 编写引导程序：从简单的“输出@”到复杂的“弹射字母”
3. 完成格式化软驱及其他任务

下文详述过程。

5 实验过程

5.1 学习 x86

由于计组课学习的是 mips，因此 x86 需要重新学习。我搜索了一篇博客[1]学习基础语法，里面只有一些基础的语句，但是完成实验已经够用了，余下的复杂指令以后用到再学。

与曾经学习的 mips 指令区别不大，大多只是些格式的差异。一些值得注意的差异包括：不同长度的寄存器及其混用、取址的方式与语法规定等。

5.2 配置虚拟机

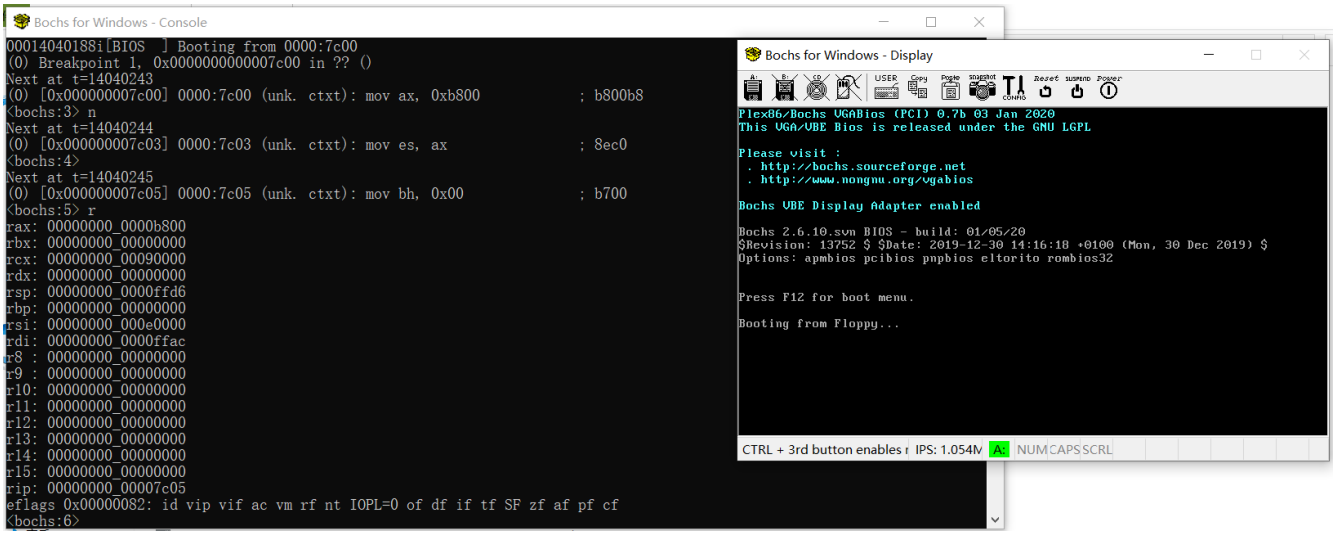
使用 bochs-2.6.11，它的优点在于可以用类似 gdb 的调试方法对代码进行调试，缺点是难上手。

创建一台虚拟机的方法为编写一个配置文件，如：

```
megs: 1                                 # memory 1MB
floppya: 1_44=test.img, status=inserted     # start from a 1.44MB floppy test.img
boot: floppy                            # start from a floopy
```

意为内存 1MB、使用 1.44MB 软盘启动的虚拟机，保存为 `.bxrc` 文件即可双击使用。

使用该软件的 `bochsdbg.exe` 可对 `.img` 文件进行调试，调试方法基本同 gdb，如下图：



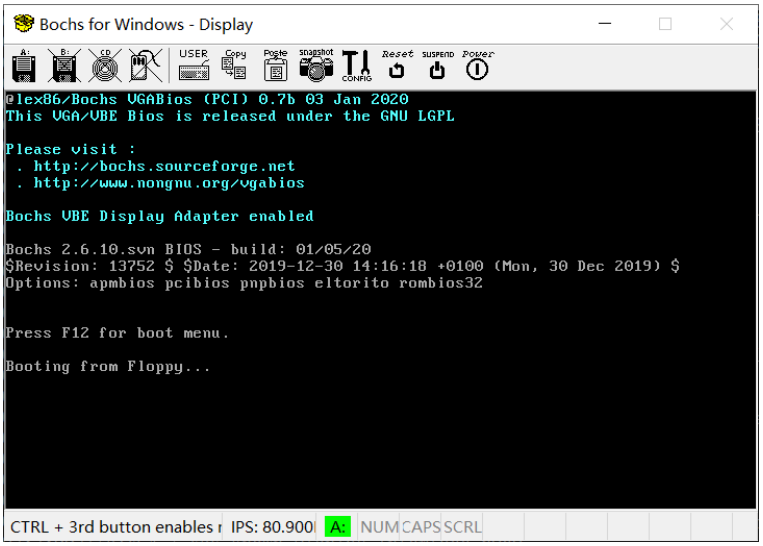
5.3 编写简单的引导程序

阅读 ppt 可知，引导程序本身不需要什么格式，最简单的“输出@”的功能只需 7 行：[2]

```
mov ax, 0xB800
mov es, ax              ; address
mov byte [es:0], '@'    ; print '@'
mov byte [es:1], 7      ; background & color
jmp $
times 510-($-$$) db 0   ; complete with 0
dw 0xaa55               ; end with boot symbol
```

其中 ppt 只给了前 5 行，还需要将首扇区的最后一个字节设为 `0xaa55` 才能被识别为引导代码。

使用 NASM 将该代码直接编译为 `test.img`，大小 512B。用配置好的虚拟机运行，结果如下：



bochs 软件本身会在窗口写入很多信息，但可以看到，第一个字符已被重新写为“@”，且颜色配置符合 `0x7`，说明代码被成功执行。

5.4 编写弹射字母的引导程序

有了上一步之后，编写“弹射字母”功能就成为简单的汇编练习了。

我设计的功能为，在 80×23 的屏幕上，大约中间的位置显示学号和姓名拼音缩写，然后从左上角 $(0,0)$ 开始向右下弹射一个字母，每前进一步将字母向前循环移位一位（即 $A \to B, B \to C, \cdots, Z \to A$），字母碰到边界后遵循物理反弹。单步延时约为 0.1s。

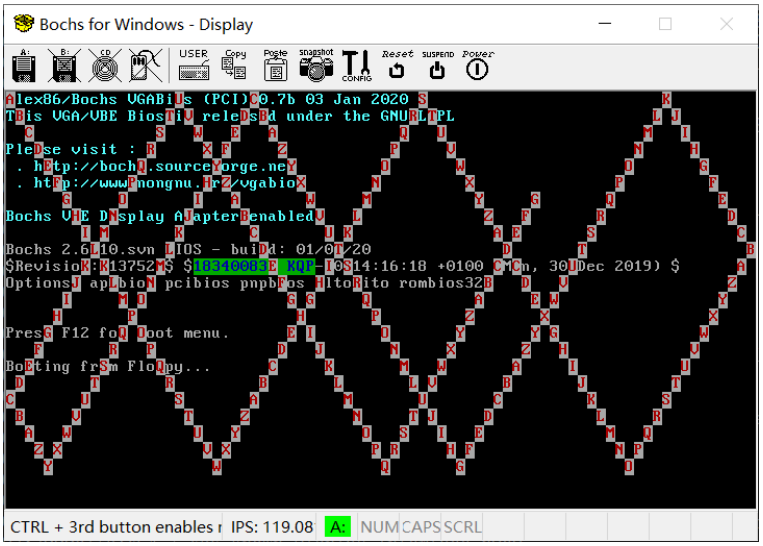
显示学号及姓名拼音缩写直接采用上一步的输出方式。

弹射代码编写为 `foo` 函数。使用寄存器变量 bh, bl 作为当前字母的坐标，dh, dl 为字母运动方向向量（左上为 $(-1,-1)$，左下为 $(1,-1)$，右上为 $(-1,1)$，右下为 $(1,1)$），ch 表示当前字母，初值为 A。每次循环，输出 ch 至显存坐标 (bh, bl)，然后更新变量：$bh \leftarrow bh + dh$, $bl \leftarrow bl + dl$, $ch \leftarrow ch + 1$，最后判断若到达了屏幕边界则更改方向向量、字母超出 Z 则重置为 A。

单步延迟代码编写为 `delay` 函数。执行一个空的循环体 $10^7 + 7$ 次即可。

代码见子目录 `src`。

使用 NASM 编译为 `shoot.img`，大小 512B。用配置好的虚拟机运行，结果如下：



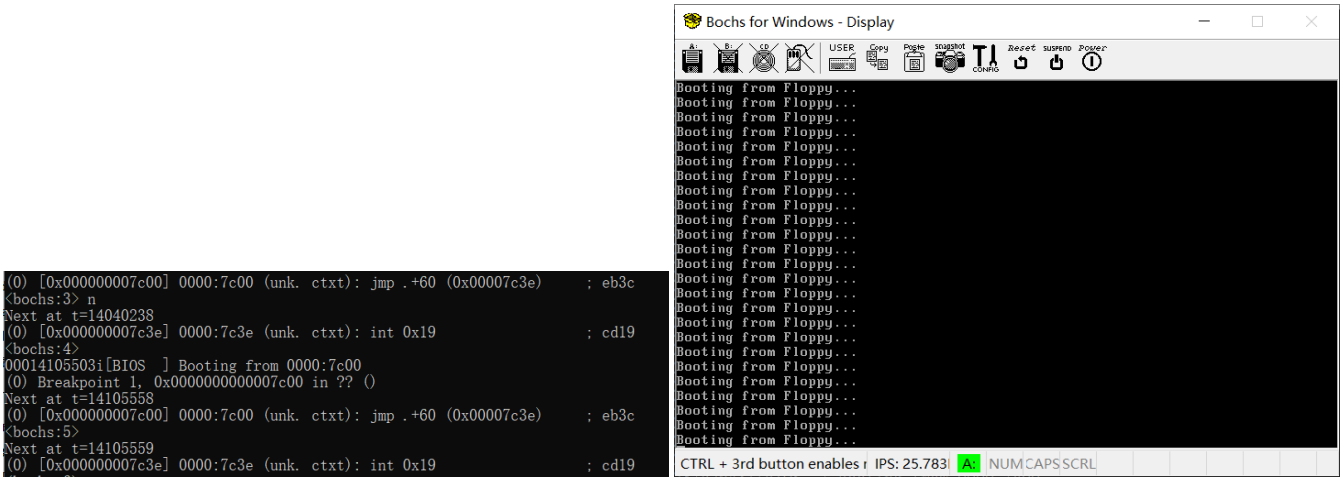
可以看到，学号及姓名拼音缩写显示在中间，字母弹射成功，循环滚动成功（重叠位置显示最后一次经过时的字母）。助教可运行虚拟机查看动态效果。

5.5 制作三张软驱

最后一步为按要求制作三张软驱。借用别的同学的正版 VMware 得到三张 DOS 格式化的空软驱，如图：

```
00000000:  eb 3c 90 49 50 52 54 20 36 2e 32 00 02 01 01 00    ?恒PRT 6.2.....
00000010:  02 e0 00 40 0b f0 09 00 12 00 02 00 00 00 00 00    .?@.?.........
00000020:  00 00 00 00 00 00 29 7a 36 e3 f3 20 20 20 20 20    ......)z6泱
00000030:  20 20 20 20 20 20 46 41 54 31 32 20 20 20 cd 19          FAT12   ?
00000040:  cc cc 00 00 00 00 00 00 00 00 00 00 00 00 00 00    烫..............
00000050:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000060:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000070:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000080:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000090:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000a0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000b0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000c0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000d0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000e0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000000f0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000100:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000110:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000120:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000130:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000140:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000150:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000160:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000170:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000180:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000190:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001a0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001b0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001c0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001d0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001e0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001f0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 aa    ..............U?
```

可以观察到，内容符合 FAT12 文件系统的格式[3]。首扇区最后以 0xaa55 结束，第一句为跳转指令，跳转至 0x3e 开始引导代码。此空盘含有一份简单的引导代码，调试得到其功能为不停地“重启”以致死循环：



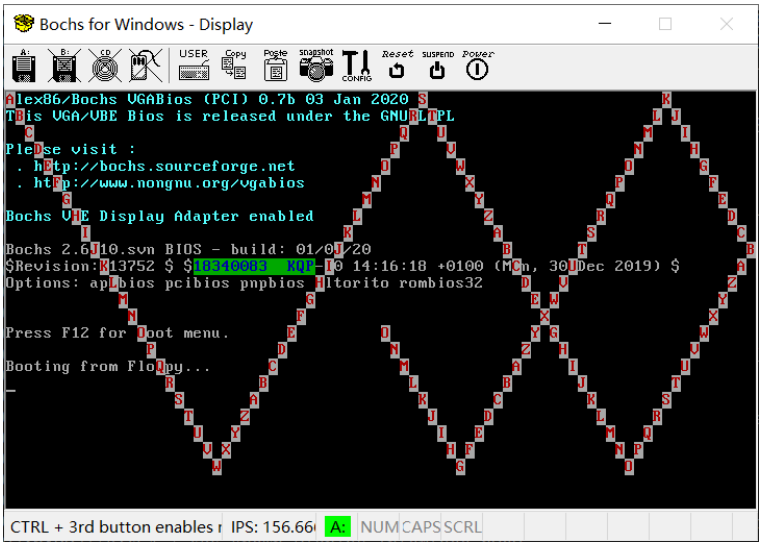
第二张软驱用 HexEditor 工具写满学号和姓名拼音如图：

```
00000000:  eb 3c 90 49 50 52 54 20 36 2e 32 00 02 01 01 00    .<.IPRT 6.2.....
00000010:  02 e0 00 40 0b f0 09 00 12 00 02 00 00 00 00 00    ...@............
00000020:  00 00 00 00 00 00 29 7a 36 e3 f3 20 20 20 20 20    ......)z6..
00000030:  20 20 20 20 20 20 46 41 54 31 32 20 20 20 cd 19          FAT12   ..
00000040:  cc cc 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000050:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000060:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000070:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000080:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000090:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000000a0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000000b0:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000000c0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000000d0:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000000e0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000000f0:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000100:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000110:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000120:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000130:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000140:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000150:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000160:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000170:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
00000180:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
00000190:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000001a0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000001b0:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000001c0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000001d0:  00 00 00 00 00 4b 75 61 6e 67 51 69 50 65 6e 67    .....KuangQiPeng
000001e0:  31 38 33 34 30 30 38 33 00 00 00 00 00 00 00 00    18340083........
000001f0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 aa    ..............U.
```

第三张软驱，将 `shoot.img` 的代码粘贴到引导代码开始处 `0x3e`，如下图：

```
00000000:  eb 3c 90 49 50 52 54 20 36 2e 32 00 02 01 01 00    ?<?IPRT 6.2.....
00000010:  02 e0 00 40 0b f0 09 00 12 00 02 00 00 00 00 00    .?@.?.........
00000020:  00 00 00 00 00 00 29 7a 36 e3 f3 20 20 20 20 20    ......)z6?    
00000030:  20 20 20 20 20 20 46 41 54 31 32 20 20 20 b8 00          FAT12   ?.
00000040:  b8 8e c0 b7 00 b3 00 b6 01 b2 01 b5 41 b1 5a 26    ?????????.???A?Z&
00000050:  c6 06 68 06 31 26 c6 06 6a 06 38 26 c6 06 6c 06    ?h.1&?j.8&?l.
00000060:  33 26 c6 06 6e 06 34 26 c6 06 70 06 30 26 c6 06    3&?n.4&?p.0&?.
00000070:  72 06 30 26 c6 06 74 06 38 26 c6 06 76 06 33 26    r.0&?t.8&?v.3&
00000080:  c6 06 78 06 20 26 c6 06 7a 06 20 26 c6 06 7c 06    ?x. &?z. &?|.
00000090:  4b 26 c6 06 7e 06 51 26 c6 06 80 06 50 b8 14 00    K&?~.Q&?.P?.
000000a0:  89 c7 81 c7 20 03 6b ff 02 26 c6 45 01 a1 40 83    ??? .k  .&?E.  ?
000000b0:  f8 20 7e ec e8 0b 00 66 bf 87 96 98 00 66 4f 75    ? ~??..f???.fOu
000000c0:  fc c3 e8 f2 ff 0f b6 ff 6b ff 50 0f b6 c3 01 c7      ??  .?k  P.??.?
000000d0:  6b ff 02 26 88 2d 26 c6 45 01 74 00 f7 00 d3 80    k  .&?&?.t.?????
000000e0:  ff 16 75 02 b6 ff 80 ff 00 75 02 b6 01 80 fb 4f      .u.??  .u.?????
000000f0:  75 02 b2 ff 80 fb 00 75 02 b2 01 80 fd 5a 75 03    u.??u.?????u.
00000100:  80 ed 1a 80 c5 01 eb ba 00 00 00 00 00 00 00 00    ????????........
00000110:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000120:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000130:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000140:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000150:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000160:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000170:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000180:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00000190:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001a0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001b0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001c0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001d0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001e0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
000001f0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 aa    ..............U?
```

运行虚拟机如下：



可以看到，引导代码被成功执行。

6 实验总结

第一次操统实验，对于我来说是完全陌生的环境，因此有很多挑战。实验报告是以流畅的方式叙述实验过程的，但实际操作过程远没有这么流畅。

第一个挑战是得到实验任务后无从下手。解决方案是看 ppt，里面给出了从零开始的方法，即从最简单的输出“@”开始。一步一步调通，每一步都是自己写的，这种感觉胜过直接修改 `stoneN.asm`。

第二个挑战是汇编语言 x86。我们的计组课学习 mips 语言，几乎没有使用过 x86，因此需要重新学习。所幸它们大部分内容是相通的，学会一个后学另一个并不困难。

第三个挑战是虚拟机配置。我选择了直接用 bochs，这款软件新手不友好，一开始点进去总是直接 kill

掉。后来查阅百度，学会了编写配置文件，并逐条注释以保留有用的功能，最终得到了最精简的三行配置文件。我很高兴平时习惯使用 gdb，面对新的调试工具和调试任务并不感到陌生。

第四个挑战是如何完整地运行一份代码。实验 ppt 上说输出“@”只需这五行，但实际上只有这五行的 `.img` 根本无法执行。后来经过同学相助，得知我漏了 `0xaa55` 因此机器未找到引导扇区。加上去之后，代码运行成功了，这时候离完成实验也就不远了。

第五个挑战是编写功能更多的引导程序。尽管算法很容易设计，但由于是用汇编写的，写下来有各种各样的语法问题，包括不同长度的寄存器不能直接运算、`es` 的偏移必须使用指定寄存器、寄存器差点不够用等等。

最后是将引导程序粘贴到 FAT12 格式的软驱上，第一次复制过去后运行虚拟机依然死循环，通过 debug 找到跳转指令跳转到的位置，才发现是粘贴的位置错了。

跌跌撞撞也终于完成了实验。虽说遇到很多挑战，但其实都是些很普通的小事，多数源于对操作环境的不熟悉。也因为第一次接触这些软件，操作效率也比较低。但是迈出第一步之后，后面应该就能熟练起来了。

参考文献

[1] x86汇编快速入门, `https://www.cnblogs.com/YukiJohnson/archive/2012/10/27/2741836.html`

[2] 凌应标,00实验课,第20页

[3] 凌应标,网课2,第13页