操作系统实验

|  |  |
| --- | --- |
| 编 制 人： | Lizy |
| 发布日期： | 2023.3.19 |
| 页 数： |  |

更改记录

|  |  |
| --- | --- |
| **发布日期** | **更改描述** |
| 20230303 | 第一版发布 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

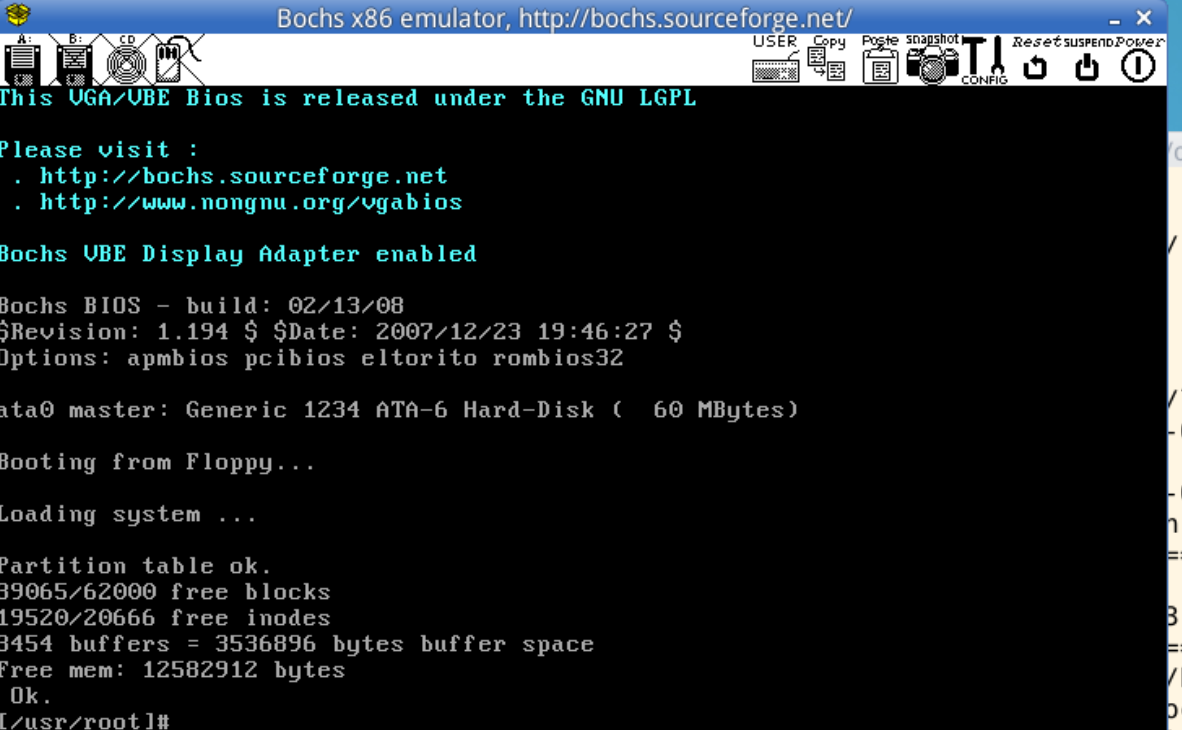
# 实验环境

本操作系统实验的硬件环境是 IA-32（x86）架构的 PC 机，主要软件环境是 Bochs + gcc + 你最喜欢的编辑器 / IDE + 你最喜欢的操作系统 + Linux 0.11 源代码。

## x86模拟器Bochs

Bochs是一个x86硬件平台的开源模拟器。它可以模拟各种硬件的配置。Bochs模拟的是整个PC平台，包括I/O设备、内存和BIOS。更为有趣的是，甚至可以不使用PC硬件来运行Bochs。事实上，它可以在任何编译运行Bochs的平台上模拟x86硬件。通过改变配置，可以指定使用的CPU(386、486或者586)，以及内存大小等。

虚拟机 VMware 或者 Microsoft Virtual PC也可以使用Windons系统。虽然 Bochs 的是模拟器，其原理决定了它的运行效率会低于虚拟机，但是Bochs 有虚拟机无可比拟的**调试操作系统**的能力。



## GCC编译器

GCC 是和 Linux 一起成长起来的编译器。Linux 最初的版本就是由 GCC 编译的。现在 GCC 也是在自由软件领域应用最广泛的编译器。

GCC是一款Linux Gun计划当中的一款编译工具，用于编译C语言文件。

其显著特点就是可以**交叉编译**。

说道交叉编译，首先因为汇编语言的特点是，能够直接访问硬件，并且**每款CPU都有自己的汇编语言**（不同指令集对应不同汇编）。

由于嵌入式系统需要进行操作系统的移植，并且在主机上开发的软件程序需要跑到特点的嵌入式系统中，不同的嵌入式系统所用到的处理器也是不同的

如PC机CPU采用的是X86架构，即对应x86版汇编

手机大多数采用ARM体系架构 即对应 ARM汇编。

由此需要在当前CUP平台去编译出能在其它CPU平台所运行的程序时，就叫交叉编译。

此时就要安装交叉编译器（ARM-LINUX-GG等）。

## GDB调试器

GDB 调试器是 GCC 编译器的兄弟。做为自由软件领域几乎是唯一的**调试器**，它秉承了Unix类操作系统的一贯风格，采用**纯命令行**操作，有点儿类似 dos下的 debug。

[GDB 简明教程\_C - 蓝桥云课 (lanqiao.cn)](https://www.lanqiao.cn/courses/496)

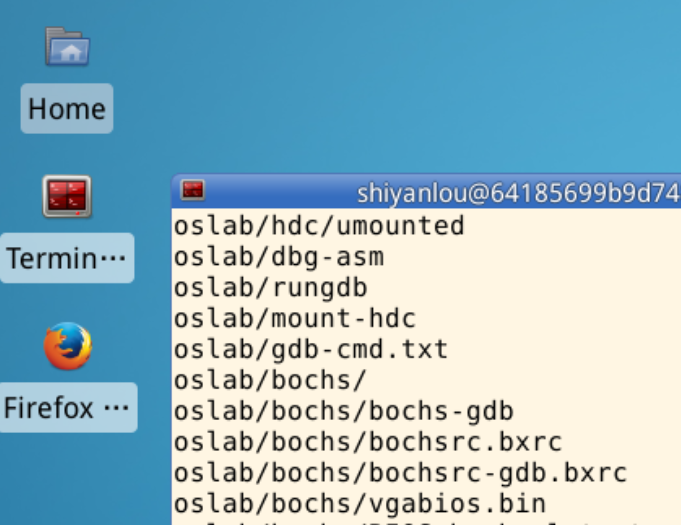
## Ubuntu (GNU/Linux)

Ubuntu是一个以**桌面应用**为主的Linux操作系统。

Ubuntu 也许不是目前最好用的 Linux 桌面发行版，但它一定是最流行的。主要特点是易用，非常的易用。现在，已经有越来越多的人开始用 Ubuntu 完全代替 Windows，享受更加自由、安全、守法的感觉。

Ubuntu 的主页是 http://www.ubuntu.com/ ，这里不仅可以免费下载到 iso 文件，甚至能免费申领 Ubuntu 的安装光盘。





## 实验环境的工作模式

1. 准备环境

hit-oslab 实验环境简称oslab，是一个压缩文件（hit-oslab-linux-20110823.tar.gz），这个文件已经下载到了/home/teacher目录和 /home/shiyanlou/oslab（大家一进入实验环境，就是点击左边的 terminal 打开终端以后，所在的目录就是/home/shiyanlou，这是大家的主目录）下，大家可以使用下面的命令解压展开压缩包即可工作。

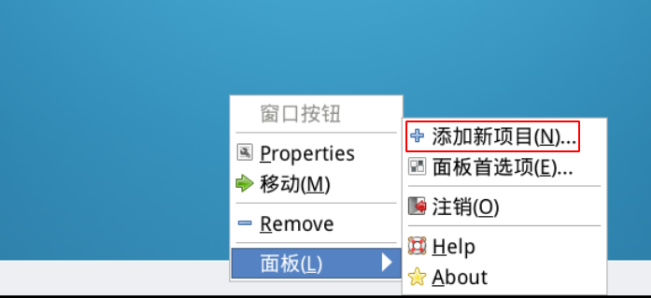
推荐大家使用如下的命令解压到 /home/shiyanlou/oslab/ 中。

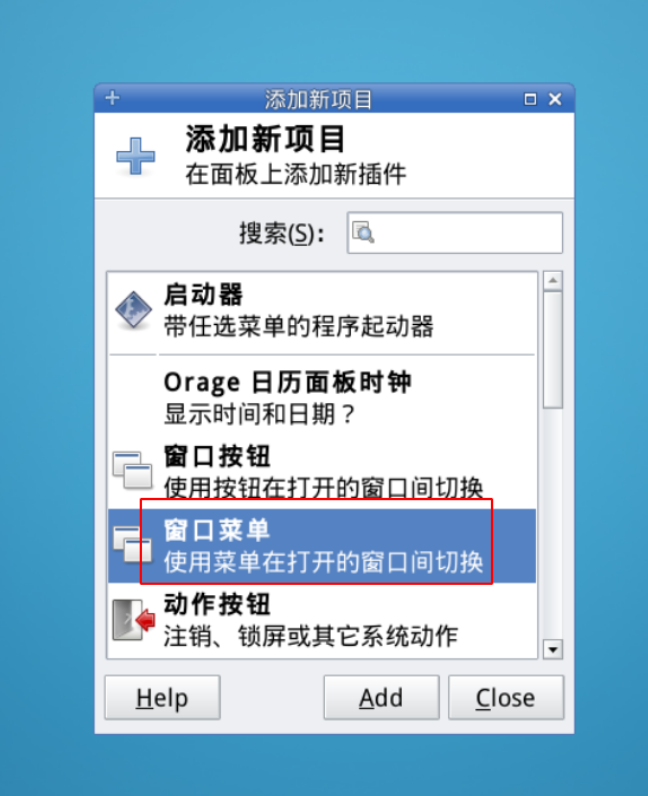


$tar -zxvf xxxfile.tar.gz

>>将xxxfile解压到oslab文件夹中，oslab存放着xxxfile.tar.gz压缩包，再将其解压到oslab->oslab中；

如果终端窗口最小化后无法找到，可以在任务栏右键，面板 -> 添加新项目 -> 窗口菜单 -> Add 来打开显示。



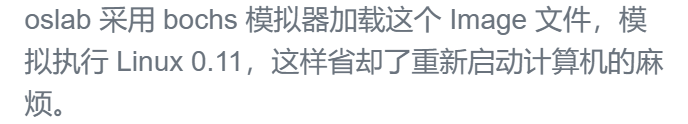


1. 文件结构

* Image 文件

oslab 工作在一个宿主操作系统之上，我们使用的 Linux，在宿主操作系统之上完成对 Linux 0.11 的开发、修改和编译之后，在 linux-0.11 目录下会生产一个名为 Image 的文件，它就是编译之后的目标文件。

该文件内已经包含引导和所有内核的二进制代码。如果拿来一张软盘，从它的 0 扇区开始，逐字节写入 Image 文件的内容，就可以用这张软盘启动一台真正的计算机，并进入 Linux 0.11 内核。



* bochs 目录

bochs 目录下是与 bochs 相关的执行文件、数据文件和配置文件。

* run 脚本

run 是运行 bochs 的脚本命令。

运行后 bochs 会自动在它的虚拟软驱 A 和虚拟硬盘上各挂载一个镜像文件，软驱上挂载是 linux-0.11/Image，硬盘上挂载的是 hdc-0.11.img。

因为 bochs 配置文件中的设置是从软驱 A 启动，所以 Linux 0.11 会被自动加载。

而 Linux 0.11 会驱动硬盘，并mount 硬盘上的文件系统，也就是将 hdc-0.11.img 内镜像的文件系统挂载到 0.11 系统内的根目录 —— /。在 0.11 下访问文件系统，访问的就是 hdc-0.11.img 文件内虚拟的文件系统。

* hdc-0.11.img 文件

hdc-0.11.img 文件的格式是 Minix 文件系统的镜像。

Linux 所有版本都支持这种格式的文件系统，所以可以直接在宿主 Linux 上通过 mount 命令访问此文件内的文件，达到宿主系统和 bochs 内运行的 Linux 0.11 之间交换文件的效果。

Windows 下目前没有（或者是还没发现）直接访问 Minix 文件系统的办法，所以要借助于 fdb.img，这是一个 1.44M 软盘的镜像文件，内部是 FAT12 文件系统。将它挂载到 bochs 的软驱 B，就可以在 0.11 中访问它。而通过 filedisk 或者 WinImage，可以在 Windows 下访问它内部的文件。

hdc-0.11.img 内包含有：

* Bash shell；
* 一些基本的 Linux 命令、工具，比如 cp、rm、mv、tar；
* vi 编辑器；
* gcc 1.4 编译器，可用来编译标准 C 程序；
* as86 和 ld86；
* Linux 0.11 的源代码，可在 0.11 下编译，然后覆盖现有的二进制内核。

## oslab命令窗（bochs模拟器）

命令：$ pwd >>确认路径；

命令：$ ls -l >>查看该目录的内容；

命令：$ ls -al >>查看该目录的内容及其路径；

命令： $ ./run >>执行当前目录下的run脚本，目录的起始都是./

命令：$ cd ~/oslab/ >>转移到 oslab目录下

命令： $ cd .. >>返回上一级目录

命令：$cp ./Image ../Image >>复制当前目录下的Image到上一层目录中

## 调试

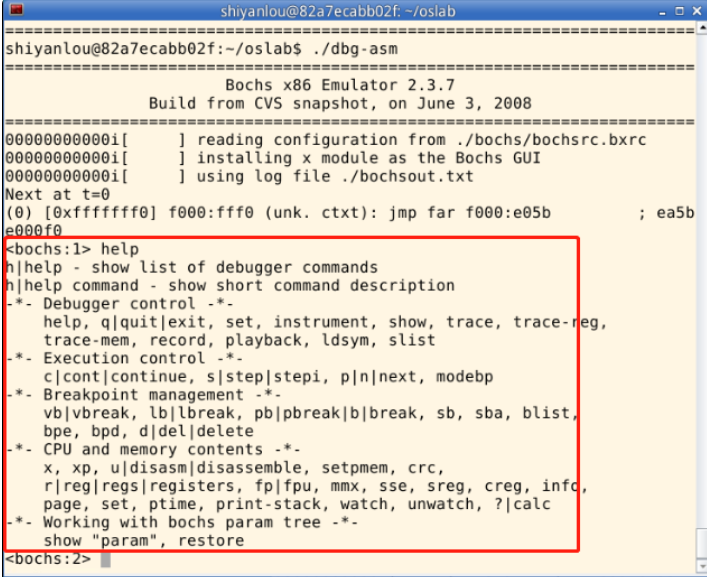
内核调试分为两种模式：汇编级调试和 C 语言级调试。

1. 汇编级调试

汇编级调试需要执行命令：

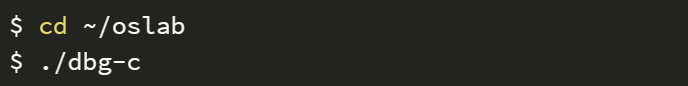


可以用命令 help 来查看调试系统用的基本命令。更详细的信息请查阅 Bochs 使用手册。



1. C 语言级调试

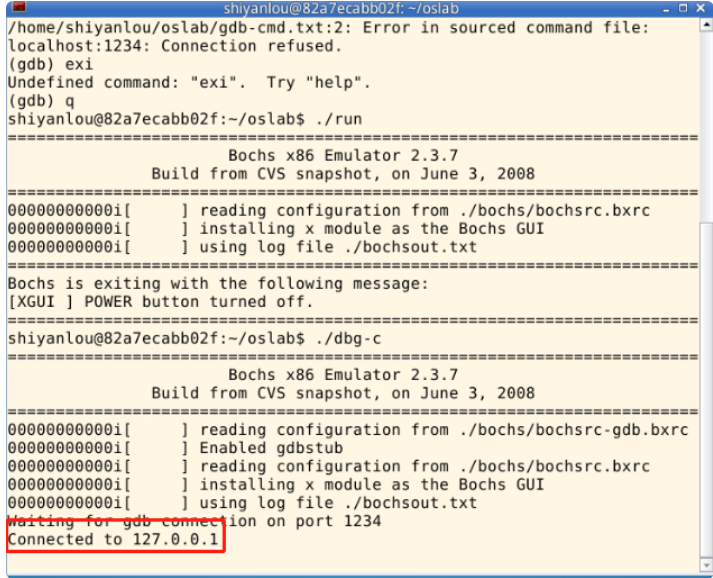
C 语言级调试稍微复杂一些。首先执行如下命令：



然后再打开一个终端窗口，执行：



出现下图所示的提示，才说明连接成功：

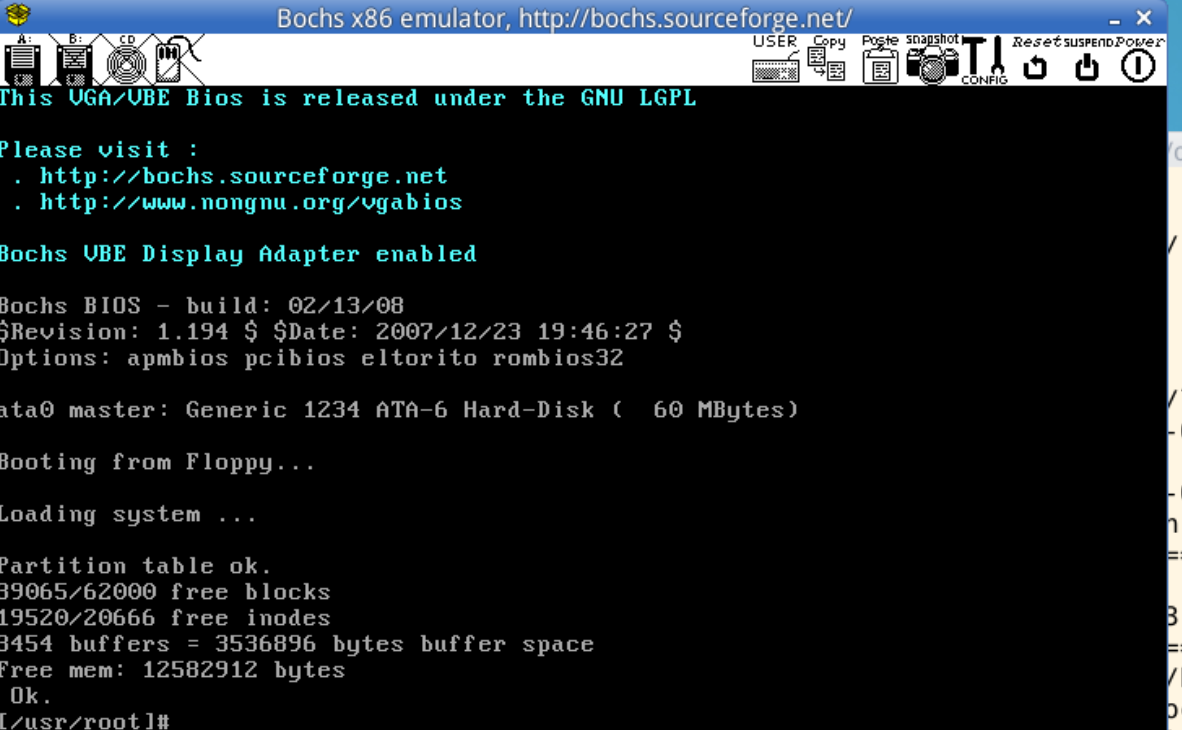


新终端窗口中运行的是 GDB 调试器。关于 gdb 调试器请查阅 GDB 使用手册。

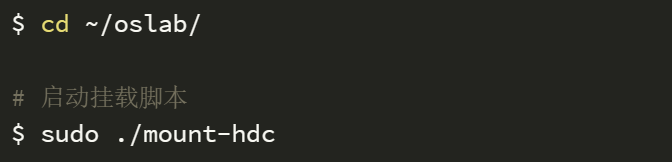
## 文件交换

接下来讲解一下 Ubuntu 和 Linux 0.11 之间的文件交换如何启动。

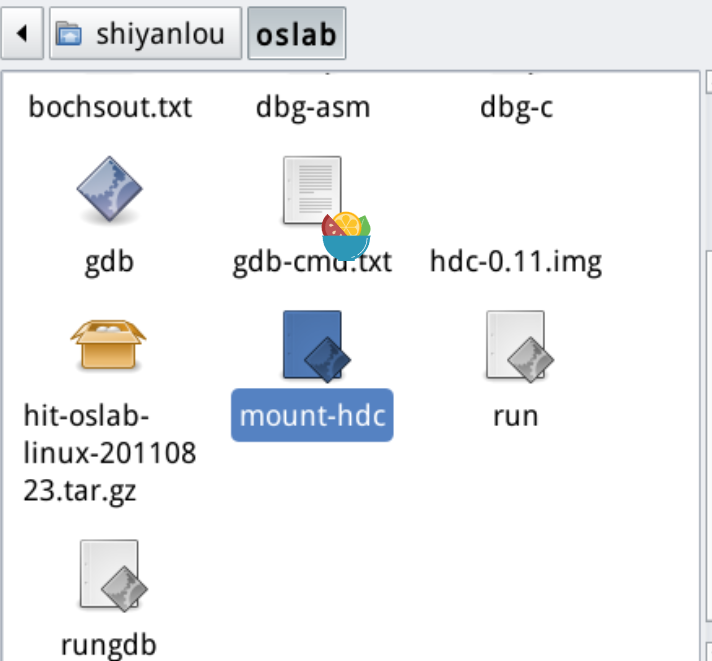
开始设置文件交换之前，务必关闭所有的 Bochs 进程。



oslab 下的 hdc-0.11-new.img 是 0.11 内核启动后的根文件系统镜像文件，相当于在 bochs 虚拟机里装载的硬盘。在 Ubuntu 上访问其内容的方法是：

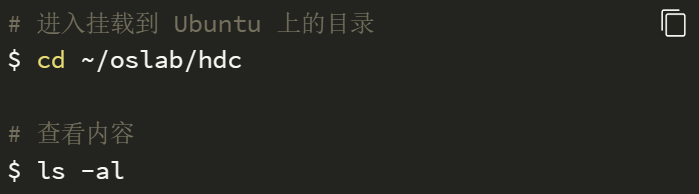


大家使用 sudo 时，password 是 shiyanlou，也有可能不会提示输入密码。



sudo是linux系统管理指令，是允许系统管理员让普通用户执行一些或者全部的root命令的一个工具，如halt，reboot，su等等。这样不仅减少了root用户的登录 和管理时间，同样也提高了安全性。sudo不是对shell的一个代替，它是面向每个命令的。

之后，hdc 目录下就是和 0.11 内核一模一样的文件系统了，可以读写任何文件（可能有些文件要用 sudo 才能访问）。



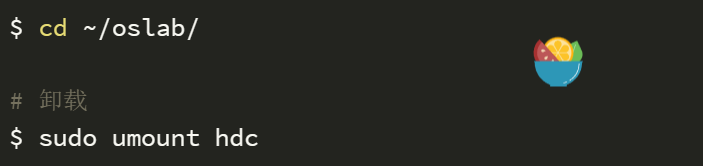
写操作：

然后在 Ubuntu 的 ~/oslab/hdc/usr/root 目录下创建一个 xxx.c 文件，



编辑内容之后记得保存。

读写操作结束后，不要忘了卸载这个文件系统：

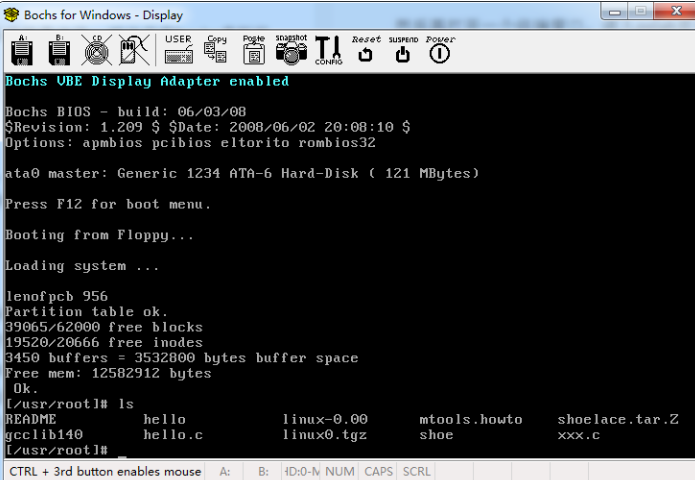


最后进入/oslab/文件夹下执行run操作:

$ cd ~/oslab/

$ ./run

这样就启动了bochs进入Linux0.11中，然后ls就会看到这个 xxx.c（即如下图所示）。



这样就避免了在 Linux 0.11 上进行编辑 xxx.c 的麻烦，因为 Linux 0.11 作为一个很小的操作系统，其上的编辑工具只有 vi，使用起来非常不便。

**注意：**在关闭 Bochs 之前，需要先在 0.11 的命令行运行 “sync”，确保所有缓存数据都存盘后，再关闭 Bochs。

## 编译链接

无论那种系统，都执行下面两个命令编译和链接 bootsect.s：

$ as86 -0 -a -o bootsect.o bootsect.s

$ ld86 -0 -s -o bootsect bootsect.o

其中 -0（注意：这是数字 0，不是字母 O）表示生成 8086 的 16 位目标程序，-a 表示生成与 GNU as 和 ld 部分兼容的代码，-s 告诉链接器 ld86 去除最后生成的可执行文件中的符号信息。

如果这两个命令没有任何输出，说明编译与链接都通过了。

其中 bootsect.o 是中间文件的二进制目标文件。bootsect 是编译、链接后的可执行文件。