



# 本科生实验报告

实验课程:操作系统原理实验

---

实验名称:编译内核/利用已有内核构建 OS

---

专业名称:计算机科学与技术

---

学生姓名:邓栩瀛

---

学生学号:21307035

---

实验地点:实验中心 D402

---

实验成绩:

---

报告时间:2023. 3. 3

# 「lab1 编译内核/利用已有内核构建OS」

## 1、实验要求

1. 搭建OS内核开发环境包括：代码编辑环境、编译环境、运行环境、调试环境等。
2. 下载并编译i386（32位）内核，并利用qemu启动内核。
3. 熟悉制作initramfs的方法。
4. 编写简单应用程序随内核启动运行。
5. 编译i386版本的Busybox，随内核启动，构建简单的OS。
6. 开启远程调试功能，进行调试跟踪代码运行。
7. 撰写实验报告

## 2、实验过程

### step1:环境配置

新建虚拟机：安装Virtualbox和Ubuntu 18.04桌面版



配置C/C++环境

```
1 | sudo apt install binutils
2 | sudo apt install gcc
3 | gcc -v
```

活动 终端 星期六 20:49 zh

ikonia@ikonia-VirtualUbuntu: ~

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

```
正在设置 lib32gcc-7-dev (7.5.0-3ubuntu1~18.04) ...
正在设置 gcc-7-multilib (7.5.0-3ubuntu1~18.04) ...
正在设置 gcc-multilib (4:7.4.0-1ubuntu2.3) ...
正在处理用于 libc-bin (2.27-3ubuntu1.2) 的触发器 ...
ikonia@ikonia-VirtualUbuntu:~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/7/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 7.5.0-3ubuntu1~18.04' --with-bugurl=file:///usr/share/doc/gcc-7/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++ --prefix=/usr --with-gcc-major-version-only --program-suffix=-7 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --enable-bootstrap --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-libmplex --enable-plugin --enable-default-pie --with-system-zlib --with-target-system-zlib --enable-objc-gc=auto --enable-multiarch --disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~18.04)
ikonia@ikonia-VirtualUbuntu:~$
```

#### 安装相关工具

```
1 sudo apt install qemu
2 sudo apt install cmake
3 sudo apt install libncurses5-dev
4 sudo apt install bison
5 sudo apt install flex
6 sudo apt install libssl-dev
7 sudo apt install libc6-dev-i386
8 sudo apt install gcc-multilib
9 sudo apt install g++-multilib
```

```
活动 终端 星期六 20:48 zh
ikonia@ikonia-VirtualUbuntu: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
解压缩后会消耗 7,865 kB 的额外空间。
您希望继续执行吗? [Y/n] y
获取:1 http://security.ubuntu.com/ubuntu bionic-security/main amd64 libssl-dev
amd64 1.1.1-1ubuntu2.1~18.04.21 [1,569 kB]
已下载 1,569 kB, 耗时 3秒 (485 kB/s)
正在预设软件包 ...
(正在读取数据库 ... 系统当前共安装有 136353 个文件和目录。)
正准备解包 .../libssl1.1_1.1.1-1ubuntu2.1~18.04.21_amd64.deb ...
正在将 libssl1.1:amd64 (1.1.1-1ubuntu2.1~18.04.21) 解包到 (1.1.1-1ubuntu2.1~18.04.21) 上 ...
正在选中未选择的软件包 libssl-dev:amd64。
正准备解包 .../libssl-dev_1.1.1-1ubuntu2.1~18.04.21_amd64.deb ...
正在解包 libssl-dev:amd64 (1.1.1-1ubuntu2.1~18.04.21) ...
正在设置 libssl1.1:amd64 (1.1.1-1ubuntu2.1~18.04.21) ...
正在设置 libssl-dev:amd64 (1.1.1-1ubuntu2.1~18.04.21) ...
正在处理用于 libc-bin (2.27-3ubuntu1.2) 的触发器 ...
ikonia@ikonia-VirtualUbuntu:~$ sudo apt install libc6-dev-i386
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会同时安装下列软件:
gcc-7-multilib gcc-multilib lib32asan4 lib32atomic1 lib32cilkrtss
lib32gcc-7-dev lib32gcc1 lib32gomp1 lib32itm1 lib32mpx2 lib32quadmath0
lib32stdc++6 lib32ubsan0 libc6-dev-x32 libc6-i386 libc6-x32 libx32asan4
libx32atomic1 libx32cilkrtss5 libx32gcc-7-dev libx32gcc1 libx32gomp1
libx32itm1 libx32quadmath0 libx32stdc++6 libx32ubsan0
下列【新】软件包将被安装:
gcc-7-multilib gcc-multilib lib32asan4 lib32atomic1 lib32cilkrtss
lib32gcc-7-dev lib32gcc1 lib32gomp1 lib32itm1 lib32mpx2 lib32quadmath0
```

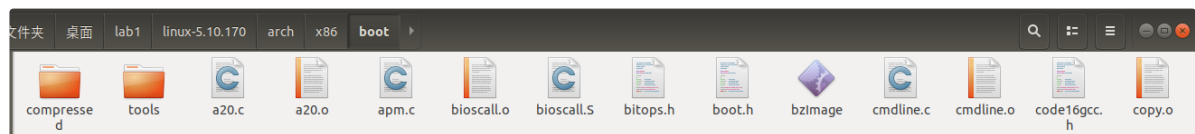
## step2:编译Linux内核

下载: <https://www.kernel.org/> 下载内核5.10.170

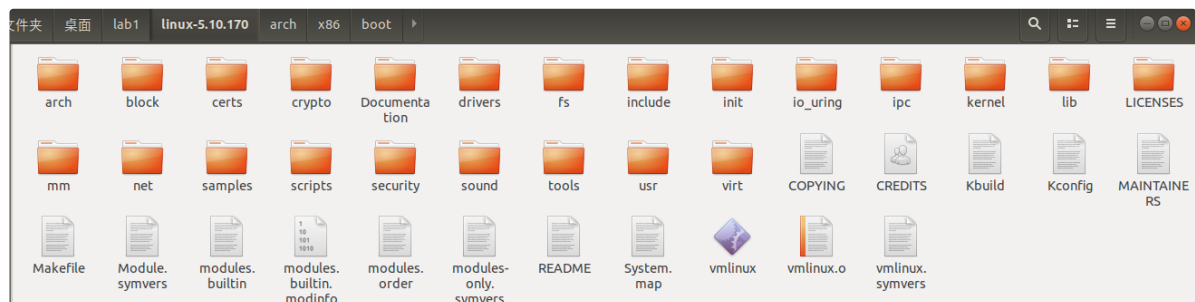
编译:

```
1 make i386_defconfig
2 make menuconfig
3 //选择kernel hacking、Compile-time checks and compiler options、[ ] Compile
  the kernel with debug info
4 make -j8//编译内核
```

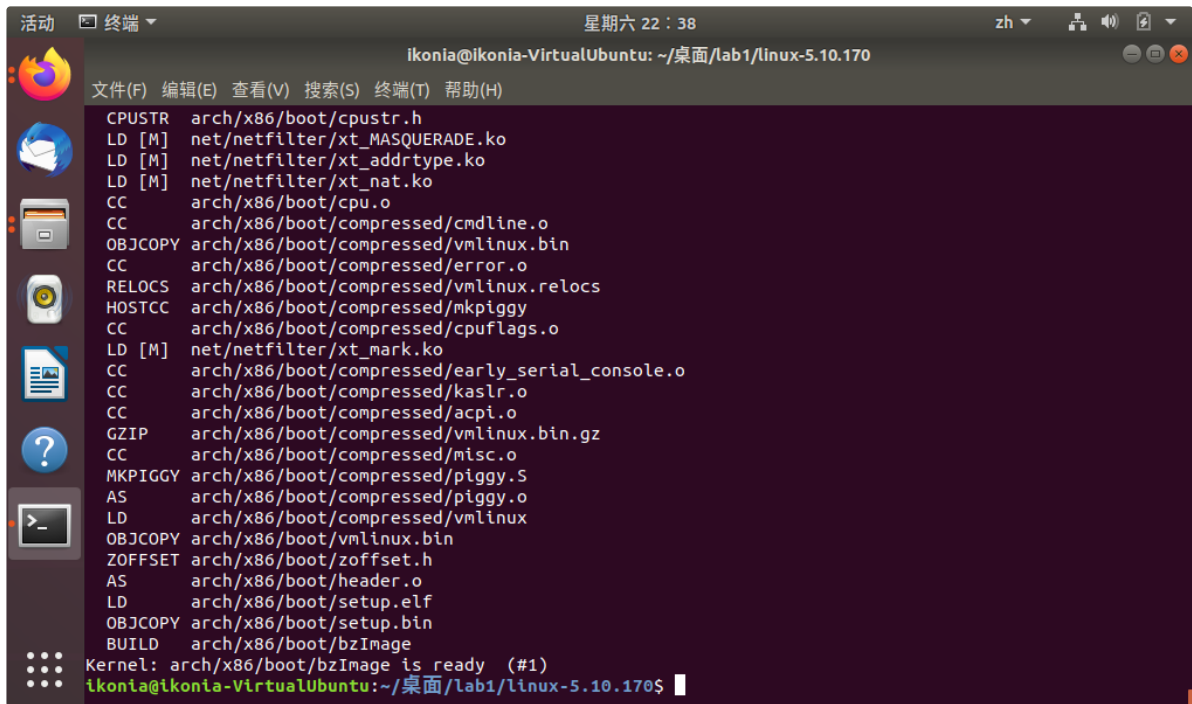
Linux压缩镜像 `linux-5.10.170/arch/x86/boot/bzImage`



符号表 `linux-5.10.170/vmlinux` 已生成



编译完成



```
ikonია@ikonია-VirtualUbuntu: ~/桌面/lab1/linux-5.10.170
CPUSTR arch/x86/boot/cpustr.h
LD [M] net/netfilter/xt_MASQUERADE.ko
LD [M] net/netfilter/xt_addrtype.ko
LD [M] net/netfilter/xt_nat.ko
CC arch/x86/boot/cpu.o
CC arch/x86/boot/compressed/cmdline.o
OBJCOPY arch/x86/boot/compressed/vmlinux.bin
CC arch/x86/boot/compressed/error.o
RELOCS arch/x86/boot/compressed/vmlinux.relocs
HOSTCC arch/x86/boot/compressed/mkpiggy
CC arch/x86/boot/compressed/cpuflags.o
LD [M] net/netfilter/xt_mark.ko
CC arch/x86/boot/compressed/early_serial_console.o
CC arch/x86/boot/compressed/kaslr.o
CC arch/x86/boot/compressed/acpi.o
GZIP arch/x86/boot/compressed/vmlinux.bin.gz
CC arch/x86/boot/compressed/misc.o
MKPIGGY arch/x86/boot/compressed/piggy.S
AS arch/x86/boot/compressed/piggy.o
LD arch/x86/boot/compressed/vmlinux
OBJCOPY arch/x86/boot/vmlinux.bin
ZOFFSET arch/x86/boot/zoffset.h
AS arch/x86/boot/header.o
LD arch/x86/boot/setup.elf
OBJCOPY arch/x86/boot/setup.bin
BUILD arch/x86/boot/bzImage
Kernel: arch/x86/boot/bzImage is ready (#1)
ikonია@ikonია-VirtualUbuntu:~/桌面/lab1/linux-5.10.170$
```

### step3:启动内核并调试

使用 `qemu` 启动内核并开启远程调试

```
1 qemu-system-i386 -kernel linux-5.10.169/arch/x86/boot/bzImage -s -S -append  
"console=ttyS0" -nographic
```

`gdb` 调试

```
1 //新建一个Terminal  
2 gdb  
3 file linux-5.10.170/vmlinux //加载符号表  
4 target remote:1234 //连接已经启动的qemu进行调试  
5 break start_kernel //为start_kernel函数设置断点  
6 c //运行
```

运行以 `kernel panic` 结束

#### step4: 制作initramfs

创建 `Hello world initramfs`

```
1 //用gcc编译helloworld.c
2 gcc -o helloworld -m32 -static helloworld.c
```

加载 `initramfs`

```
1 //用cpio打包initramfs
2 echo helloworld | cpio -o --format=newc > hwinitramfs
3 //启动内核，并加载initramfs
4 qemu-system-i386 -kernel linux-5.10.170/arch/x86/boot/bzImage -initrd
hwinitramfs -s -S -append "console=ttyS0 rdinit=helloworld" -nographic
```

```
lab1: Hello World
[ 3.492624] tsc: Refined TSC clocksource calibration: 2497.249 MHz
[ 3.493483] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x23ff132d161, max_idle_ns: 44079
5293778 ns
[ 3.500286] clocksource: Switched to clocksource tsc
[ 3.703612] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input3
[ 30.270572] hrtimer: interrupt took 9110311 ns
```

`gdb` 中输出了 `lab1: Hello world\n`

#### step5: 编译并启动Busybox

编译 `Busybox`

```
1 make defconfig
2 make menuconfig
3 // settings->勾选Build BusyBox as a static binary(no shared libs)
4 // 设置(-m32 -march=i386) Additional CFLAGS和(-m32) Additional LDFLAGS
5 //编译
6 make -j8
7 make install
```

```
活动 终端
ikonia@ikonia-VirtualUbuntu: ~/桌面/lab1/busybox-1_33_0
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
./install/usr/sbin/nbd-client -> ../bin/busybox
./install/usr/sbin/nologin -> ../bin/busybox
./install/usr/sbin/ntpd -> ../bin/busybox
./install/usr/sbin/partprobe -> ../bin/busybox
./install/usr/sbin/popmaildir -> ../bin/busybox
./install/usr/sbin/powertop -> ../bin/busybox
./install/usr/sbin/rdate -> ../bin/busybox
./install/usr/sbin/rdev -> ../bin/busybox
./install/usr/sbin/readahead -> ../bin/busybox
./install/usr/sbin/readprofile -> ../bin/busybox
./install/usr/sbin/remove-shell -> ../bin/busybox
./install/usr/sbin/rtswake -> ../bin/busybox
./install/usr/sbin/sendmail -> ../bin/busybox
./install/usr/sbin/setfont -> ../bin/busybox
./install/usr/sbin/setlogcons -> ../bin/busybox
./install/usr/sbin/svlogd -> ../bin/busybox
./install/usr/sbin/telnetd -> ../bin/busybox
./install/usr/sbin/tftpd -> ../bin/busybox
./install/usr/sbin/ubiattach -> ../bin/busybox
./install/usr/sbin/ubidetach -> ../bin/busybox
./install/usr/sbin/ubikvol -> ../bin/busybox
./install/usr/sbin/ubirename -> ../bin/busybox
./install/usr/sbin/ubirmvol -> ../bin/busybox
./install/usr/sbin/ubirsvol -> ../bin/busybox
./install/usr/sbin/ubiupdatevol -> ../bin/busybox
./install/usr/sbin/udhcpd -> ../bin/busybox

-----
You will probably need to make your busybox binary
setuid root to ensure all configured applets will
work properly.
-----
ikonia@ikonia-VirtualUbuntu: ~/桌面/lab1/busybox-1_33_0$
```

## init 程序

```
1 #!/bin/sh
2 mount -t proc none /proc
3 mount -t sysfs none /sys
4 echo -e "\nBoot took $(cut -d' ' -f1 /proc/uptime) seconds\n"
5 exec /bin/s
```

将 x86-busybox 下面的内容打包归档成 cpio 文件, 以供Linux内核做 initramfs 启动执行在 mybusybox 文件夹下运行

```
1 find . -print0 | cpio --null -ov --format=newc | gzip -9 > ~/桌面/lab1/initramfs-busybox-x86.cpio.gz
```

```
活动 终端
ikonia@ikonia-VirtualUbuntu: ~/桌面/lab1
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
(gedit:2247): WARNING **: 10:04:33.326: Set document metadata failed: 不支持设置属性 metadata:gedit-position
ikonia@ikonia-VirtualUbuntu:~/桌面/lab1/mybusybox$ sudo chmod u+x init
ikonia@ikonia-VirtualUbuntu:~/桌面/lab1/mybusybox$ find . -print0 | cpio --null -ov --format=newc | gzip -9 > ~/桌面/lab1/initramfs-busybox-x86.cpio.gz

./init
./sbin
./sbin/bootchartd
./sbin/remod
./sbin/ifenslave
./sbin/syslogd
./sbin/tunctl
./sbin/init
./sbin/start-stop-daemon
./sbin/logread
./sbin/fsck.mntx
./sbin/arp
./sbin/swapon
./sbin/pivot_root
./sbin/depmod
./sbin/freerandisk
./sbin/lfdowm
./sbin/poweroff
./sbin/devmem
./sbin/lproute
./sbin/lsmode
./sbin/lp
./sbin/namelf
./sbin/fstrim
./sbin/hwclock
./sbin/fbsplash
./sbin/hdparm
./sbin/setconsole
./sbin/makedevs
./sbin/findfs
```

## 加载 busybox

```
1 cd ~/lab1
2 qemu-system-i386 -kernel linux-5.10.170/arch/x86/boot/bzImage -initrd
  initramfs-busybox-x86.cpio.gz -nographic -append "console=ttyS0"
```



```
活动 终端
星期二 11:03
ikonla@ikonla-VirtualUbuntu: ~/桌面/lab1

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

[ 3.800586] NET: Registered protocol family 10
[ 3.800586] Segment Routing with IPv6
[ 3.810881] slt: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[ 3.864273] NET: Registered protocol family 17
[ 3.866201] Key type dns_resolver registered
[ 3.866653] mce: Unable To Init MCE device (rc: -5)
[ 3.866674] IPI shorthand broadcast: enabled
[ 3.866674] sched_clock: Marking stable (3723400041, 142274478)->(4889910994, -1024236475)
[ 3.876366] registered taskstats version 1
[ 3.876641] Loading compiled-in X.509 certificates
[ 3.880473] PM: Magic number: 11:350:8
[ 3.897890] PM: hash matches drivers/base/power/main.c:904
[ LibreOffice-Writer ] lntk: console [netcon0] enabled
[ 3.898903] netconsole: network logging started
[ 3.902808] cfg80211: Loading compiled-in X.509 certificates for regulatory database
[ 3.994639] modprobe (59) used greatest stack depth: 6972 bytes left
[ 4.027573] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 4.04324] ALSA device list:
[ 4.059468] No soundcards found.
[ 4.061392] platform regulatory.0: Direct firmware load for regulatory.db failed with error -2
[ 4.061738] cfg80211: failed to load regulatory.db
[ 4.107128] Freeing unused kernel image (initmem) memory: 684K
[ 4.129348] Write protecting kernel text and read-only data: 15256k
[ 4.130817] Run /init as init process
[ 4.168330] mount (63) used greatest stack depth: 6924 bytes left
[ 4.196355] cut (65) used greatest stack depth: 6860 bytes left

Boot took 4.04 seconds

/bin/sh: can't access tty; job control turned off
/ # [ 4.414766] input: InExPS/2 Generic Explorer Mouse as /devices/platform/18042/serio1/input/input3

ls
dev      etc      linuxrc  root     sys
init     proc    sbin    usr

/ #
```

## step6:完成Linux 0.11内核的编译、启动和调试

- 编译32位版本的linux 0.11内核
- 使用 `qemu-system-i386` 加载、启动内核

```
1 | qemu-system-i386 -m 16 -boot a -fda Image -hda hdc-0.11.img -s -S
```

- 进入Linux 0.11操作系统，熟悉该操作系统的命令：比如查看目录结构，运行简单的shell命令如 `ls`、`ping` 等
- 利用gdb进行调试远程调试
- 利用gdb进行调试远程调试
- 加载linux 0.11的符号表（位于 `tools/system`）

```
1 | file tools/system
```

- `target remote:1234` 远程连接qemu调试
- 设置源码目录: `directory` linux 0.11的源码路径

```
活动 qemu-system-i386
星期二 17:22
ikonla@ikonla-VirtualUbuntu: ~/桌面/lab1/Linux-0.11-master

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
(gdb) file Linux-0.11-master/tools/system
Linux-0.11-master/tools/system: 没有那个文件或目录.
(gdb) file tools/system
Reading symbols from tools/system...done.
(gdb) target remote:1234
Remote debugging using 1234
do_execve (elfp=0x0 <startup_32>, tmp=0, filename=0x0 <startup_32>,
do_argv=0x0 <startup_32>, envp=0x0 <startup_32>) at exec.c:192
192      int sh_bang = 0;
(gdb) b main
Breakpoint 1 at 0x6773: file init/main.c, line 107.
(gdb) c
Continuing.

Breakpoint 1, main () at init/main.c:107
107 {
/* The startup routine assumes (well, ...) this
*/
(gdb)

ScalBIOS (version 1.10.2-1ubuntu1)
iPXE (http://ipxe.org) 00:03:0 C980 PC12.10 PnP PMM+00F8DD0+00ECD00 C980

Booting from Floppy...
IceCityOS is booting ...

Now we are in setup ...

Cursor POS:0E00
Memory SIZE:3B80KB

HD Info
Cylinders:0079
Heads:0010
Sectors:003F

prmt explicitly to remove the restrictions.
specified for 'hdc-0.11.img' and probing guessed r
Automatically detecting the format is dangerous for raw images, write o
perations on block 0 will be restricted.
Specify the 'raw' format explicitly to remove the restrictions.
```

- 设置汇编代码的形式: `set disassembly-flavor intel`
- 在关键位置设置断点如在地址0x7c00、内核入口函数（main）等
- 观察 0x7DFE和0x7DFF地址的内容



- 熟悉linux 0.11操作系统与主机操作系统之间的文件交换
- 用 `fdisk` 命令查看磁盘的分区情况以及文件类型(minix): `fdisk hdc-0.11.img`
- 创建本地挂载目录 `mkdir hdc`

```

ikonia@ikonia-VirtualUbuntu: ~/桌面/lab1/Linux-0.11-master$ fdisk hdc-0.11.img
欢迎使用 fdisk (util-linux 2.31.1)。
更改将停留在内存中，直到您决定将更改写入磁盘。
使用写入命令前请三思。

命令(输入 m 获取帮助): mkdir hdc

帮助:
DOS (MBR)
a  开关 可启动 标志
b  编辑套套的 bsd 磁盘标签
c  开关 dos 兼容性标志

常规
d  删除分区
F  列出未分区的空闲区
l  列出已知分区类型
n  添加新分区
p  打印分区表
t  更改分区类型
T  检查分区表
t  打印某个分区的相关信息

杂项
m  打印此菜单
u  更改 显示/记录 单位
x  更多功能(仅限专业人员)

脚本
I  从 sfdisk 脚本文件加载磁盘布局
O  将磁盘布局存储为 sfdisk 脚本文件
  
```

- 显式磁盘空间 `df -h`
- 挂载linux 0.11硬盘镜像 `sudo mount -t minix -o loop,offset=512 hdc-0.11.img ~/home/ikonia/桌面/lab1/Linux-0.11-master/hdc`
- 挂载后的hdc目录结构

```

命令(输入 m 获取帮助): c
分区号 (1-3, 默认 3): 3

命令(输入 m 获取帮助): sudo mount -t minix -o loop,offset=512 hdc-0.11.img ~/home/ikonia/桌面/lab1/Linux-0.11-master/hdc

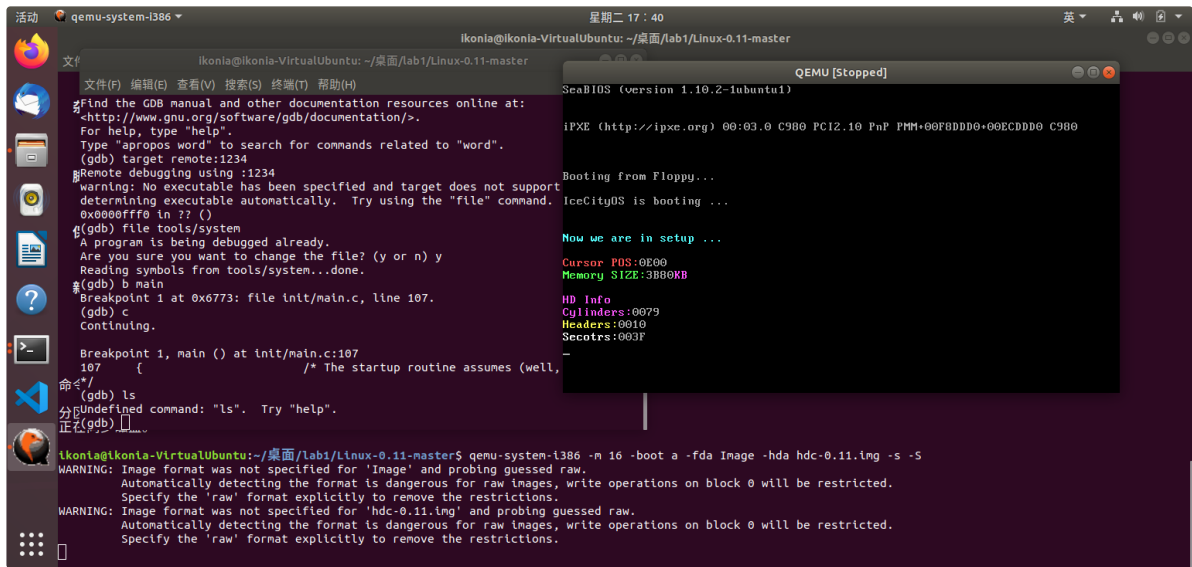
创建了一个新分区 1, 类型为"Linux native", 大小为 31.4 MiB。
创建了一个新分区 2, 类型为"Linux swap", 大小为 23.5 MiB。
创建了一个新分区 3, 类型为"Whole disk", 大小为 54.9 MiB。
创建新的 Sun 磁盘标签。

命令(输入 m 获取帮助): cd hdc
分区号 (1-3, 默认 3): 1

命令(输入 m 获取帮助): ls
 0 未分配      4 SunOS usr      8 SunOS home     82 Linux swap
 1 启动        5 整盘          9 SunOS alt 扇区 83 Linux native
 2 SunOS 根    6 SunOS stand   a SunOS cachefs 8e Linux LVM
 3 SunOS 交换  7 SunOS var     b SunOS 保留     fd Linux raid 自动

命令(输入 m 获取帮助):
  
```

- 在hdc中创建文件 `cd hdc/usr` `sudo touch hello.txt` `sudo vim hello.txt`
- 卸载文件系统hdc `sudo umount /dev/loop` (查看具体的loop设备)



### 3、总结

1. 本次实验对Linux操作系统有了初步的了解
2. 熟悉制作initramfs的方法
3. 熟悉gdb、qemu等工具的使用方法
4. 能够尝试编写简单应用程序并随内核启动运行
5. 能够编译i386版本的Busybox，随内核启动，从而构建简单的OS