



# 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 (超算)

学生姓名: 黄玟瑜

学生学号: 19335074

实验地点: 中山大学广州校区东校园

实验成绩:

报告时间: 2021 年 3 月 11 日

# 1. 实验要求










熟悉现有 Linux 内核的编译过程和启动过程, 并在自行编译内核的基础上构建简单应用并启动; 利用精简的 Busybox 工具集构建简单的 OS,熟悉现代操作系统的构建过程。 此外,熟悉编译环境、相关工具集, 并能够实现内核远程调试;

- 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. 实验内容

## 环境配置

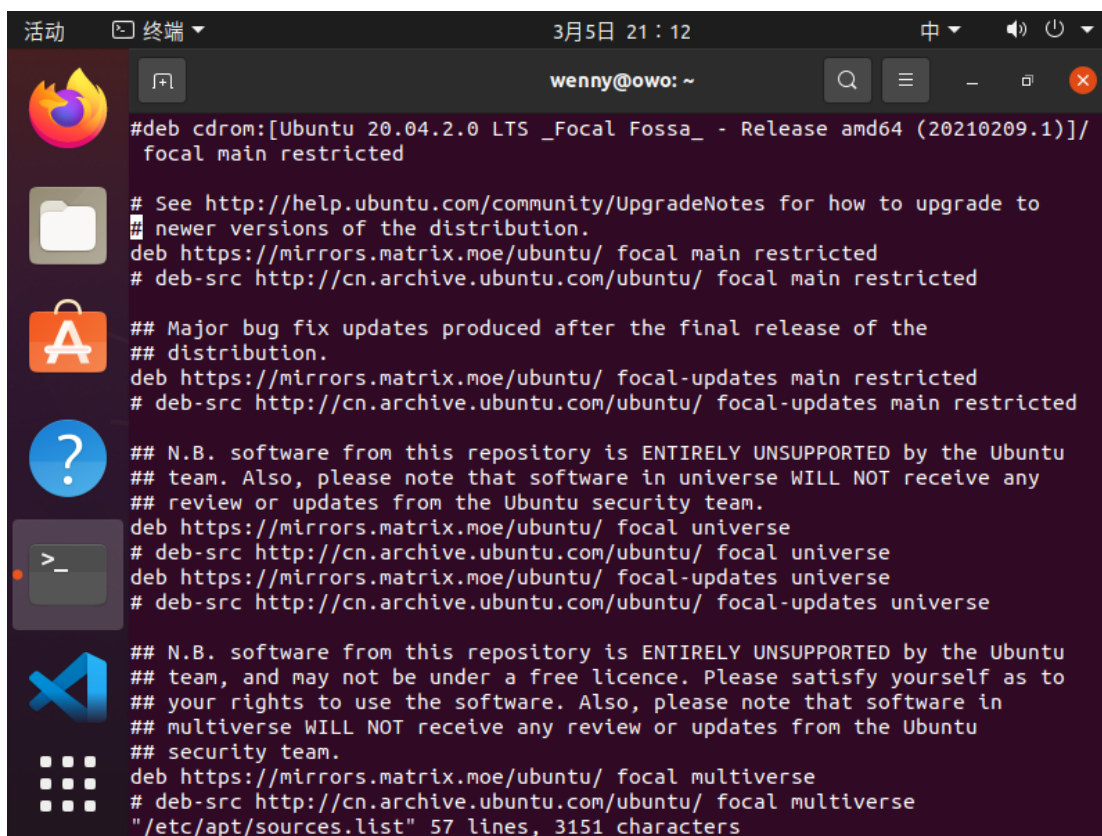
配置虚拟机, 参数如下:

设备	摘要
 内存	2 GB
 处理器	2
 硬盘 (SCSI)	30 GB
 CD/DVD (SATA)	正在使用文件 D:\ubuntu-20.0...
 网络适配器	NAT
 USB 控制器	存在
 声卡	自动检测
 打印机	存在
 显示器	自动检测

下面进行换源，输入

```
sudo vim /etc/apt/sources.list
```

将下载源替换为 Matrix 源，保存退出并查看



```
活动 终端 3月5日 21:12 中 wenny@owo: ~
#deb cdrom:[Ubuntu 20.04.2.0 LTS _Focal Fossa_ - Release amd64 (20210209.1)]/
focal main restricted

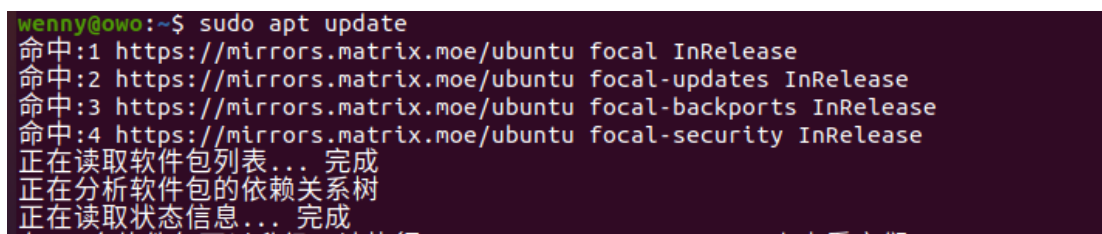
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb https://mirrors.matrix.moe/ubuntu/ focal main restricted
# deb-src http://cn.archive.ubuntu.com/ubuntu/ focal main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb https://mirrors.matrix.moe/ubuntu/ focal-updates main restricted
# deb-src http://cn.archive.ubuntu.com/ubuntu/ focal-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team. Also, please note that software in universe WILL NOT receive any
## review or updates from the Ubuntu security team.
deb https://mirrors.matrix.moe/ubuntu/ focal universe
# deb-src http://cn.archive.ubuntu.com/ubuntu/ focal universe
deb https://mirrors.matrix.moe/ubuntu/ focal-updates universe
# deb-src http://cn.archive.ubuntu.com/ubuntu/ focal-updates universe

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## multiverse WILL NOT receive any review or updates from the Ubuntu
## security team.
deb https://mirrors.matrix.moe/ubuntu/ focal multiverse
# deb-src http://cn.archive.ubuntu.com/ubuntu/ focal multiverse
"/etc/apt/sources.list" 57 lines, 3151 characters
```

更新 apt, 检查是否更换为 Matrix 的下载源, 由图可知换源完成。



```
wenny@owo:~$ sudo apt update
命中:1 https://mirrors.matrix.moe/ubuntu focal InRelease
命中:2 https://mirrors.matrix.moe/ubuntu focal-updates InRelease
命中:3 https://mirrors.matrix.moe/ubuntu focal-backports InRelease
命中:4 https://mirrors.matrix.moe/ubuntu focal-security InRelease
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
```

## 配置 C/C++ 环境，检查到 gcc 安装成功

```
wenny@owo:~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/9/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:hsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 9.3.0-17ubuntu1~20.04' --with-bugurl=file:///usr/share/doc/gcc-9/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++,gm2 --prefix=/usr --with-gcc-major-version-only --program-suffix=-9 --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-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-plugin --enable-default-pie --with-system-zlib --with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch --disable-werror --with-arch32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none=/build/gcc-9-HskZEa/gcc-9-9.3.0/debian/tmp-nvptx/usr,hsa --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 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
```

## 安装其他工具并检查到安装成功。

```
wenny@owo:~$ dpkg -s nasm
Package: nasm
Status: install ok installed
Priority: optional
Section: devel
Installed-Size: 3295
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Version: 2.14.02-1
```

```
wenny@owo:~$ dpkg -s qemu
Package: qemu
Status: install ok installed
Priority: optional
Section: oldlibs
Installed-Size: 120
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: foreign
Version: 1:4.2-3ubuntu6.14
```

```
wenny@owo:~$ dpkg -s cmake
Package: cmake
Status: install ok installed
Priority: optional
Section: devel
Installed-Size: 18758
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: foreign
Version: 3.16.3-1ubuntu1
```

```
wenny@owo:~$ dpkg -s libncurses5-dev
Package: libncurses5-dev
Status: install ok installed
Priority: optional
Section: oldlibs
Installed-Size: 6
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: same
Source: ncurses
Version: 6.2-0ubuntu2
```

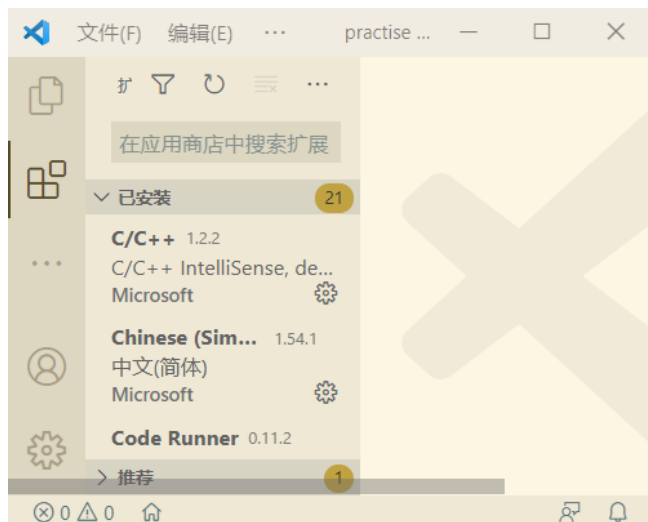
```
wenny@owo:~$ dpkg -s bison
Package: bison
Status: install ok installed
Priority: optional
Section: devel
Installed-Size: 1980
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: foreign
Version: 2:3.5.1+dfsg-1
```

```
wenny@owo:~$ dpkg -s flex
Package: flex
Status: install ok installed
Priority: optional
Section: devel
Installed-Size: 952
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: foreign
Version: 2.6.4-6.2
```

```
wenny@owo:~$ dpkg -s libssl-dev
Package: libssl-dev
Status: install ok installed
Priority: optional
Section: libdevel
Installed-Size: 7818
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: same
Source: openssl
Version: 1.1.1f-1ubuntu2.2
```

```
wenny@owo:~$ dpkg -s libc6-dev-i386
Package: libc6-dev-i386
Status: install ok installed
Priority: optional
Section: libdevel
Installed-Size: 11803
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Source: glibc
Version: 2.31-0ubuntu9.2
```

安装配置 VSCode

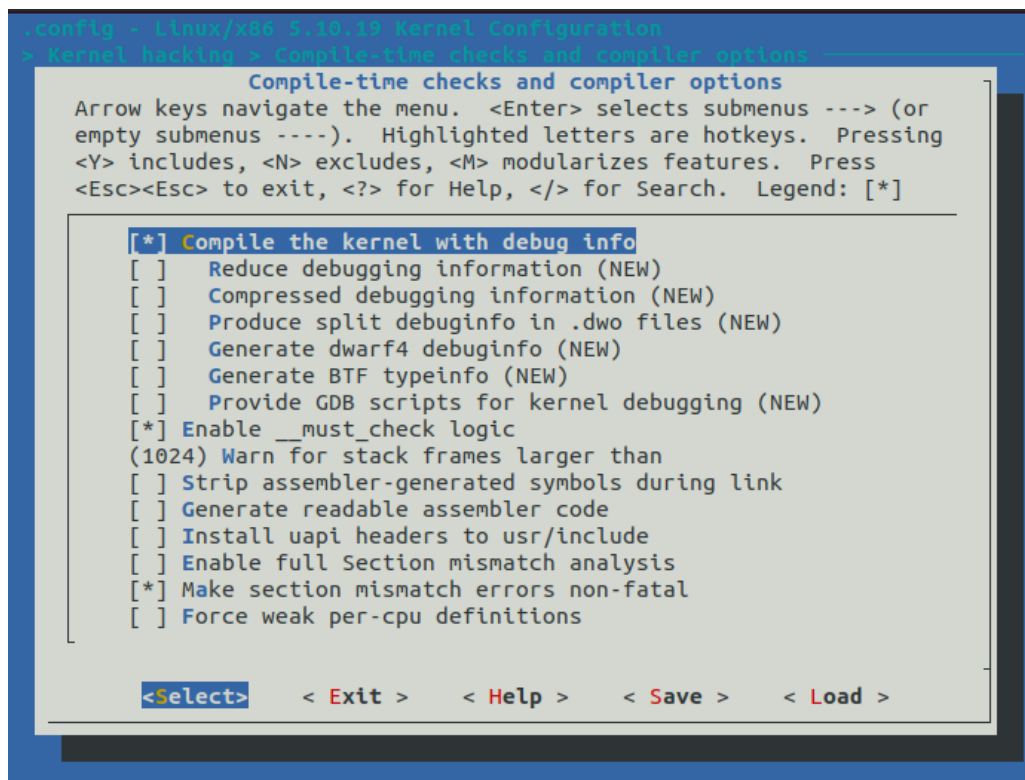


## 编译 Linux 内核

下载并解压 Linux 内核

```
wenny@owo:~$ cd lab1
wenny@owo:~/lab1$ ls
busybox-1_33_0  helloworld.c          linux-5.10.19
Busybox.tar.gz  hwinitramfs           linux-5.10.19.tar
helloworld      initramfs-busybox-x86.cpio.gz  mybusybox
```

打开图像界面依次选择相应选项



## 编译内核

```
wenny@owo:~/lab1/linux-5.10.19$ make -j8
SYNC    include/config/auto.conf.cmd
CALL    scripts/atomic/check-atomics.sh
CALL    scripts/checksyscalls.sh
CHK     include/generated/compile.h
Kernel: arch/x86/boot/bzImage is ready (#1)
wenny@owo:~/lab1/linux-5.10.19$ ls
```

## 检查到压缩镜像和符号表已生成

```
wenny@owo:~/lab1/linux-5.10.19/arch/x86/boot$ ls
a20.c          cpuflags.o      mtools.conf.in  version.o
a20.o          cpu.o           pm.c            vesa.h
apm.c          cpustr.h        pmjump.o        video-bios.c
bioscall.o     ctype.h         pmjump.S        video-bios.o
bioscall.S     early_serial_console.c  pm.o            video.c
bitops.h       early_serial_console.o  printf.c        video.h
boot.h         edd.c           printf.o        video-mode.c
bzImage        edd.o           regs.c          video-mode.o
cmdline.c      genimage.sh     regs.o          video.o
cmdline.o      header.o        setup.bin       video-vesa.c
code16gcc.h    header.S        setup.elf       video-vesa.o
compressed     install.sh      setup.ld        video-vga.c
copy.o         main.c          string.c        video-vga.o
copy.S         main.o          string.h        vmlinux.bin
cpu.c          Makefile        string.o        voffset.h
cpucheck.c     memory.c        tools           zoffset.h
cpucheck.o     memory.o        tty.c           version.c
cpuflags.c     mkcpustr        tty.o
cpuflags.h     mkcpustr.c      version.c
```

```
wenny@owo:~/lab1/linux-5.10.19$ ls
arch      fs          LICENSES      net          usr
block     include    MAINTAINERS   README       vmlinux
certs     init       Makefile      samples      vmlinux.o
COPYING   ipc        mm            security     vmlinux.symvers
CREDITS   Kbuild     modules.builtin  sound
crypto    Kconfig    modules.builtin.modinfo  System.map
Documentation  kernel     modules.order   tools
drivers   lib        Module.symvers
```

使用 qemu 启动内核并开启远程调试，发现 qemu 并未输出任何信息

```
wenny@owo:~/lab1$ qemu-system-i386 -kernel linux-5.10.19/arch/x86/boot/bzImage -s -S -append "console==ttyS0" -nographic
```

qemu 在等待 gdb 输入的指令后才能继续执行，接下来我们启动 gdb

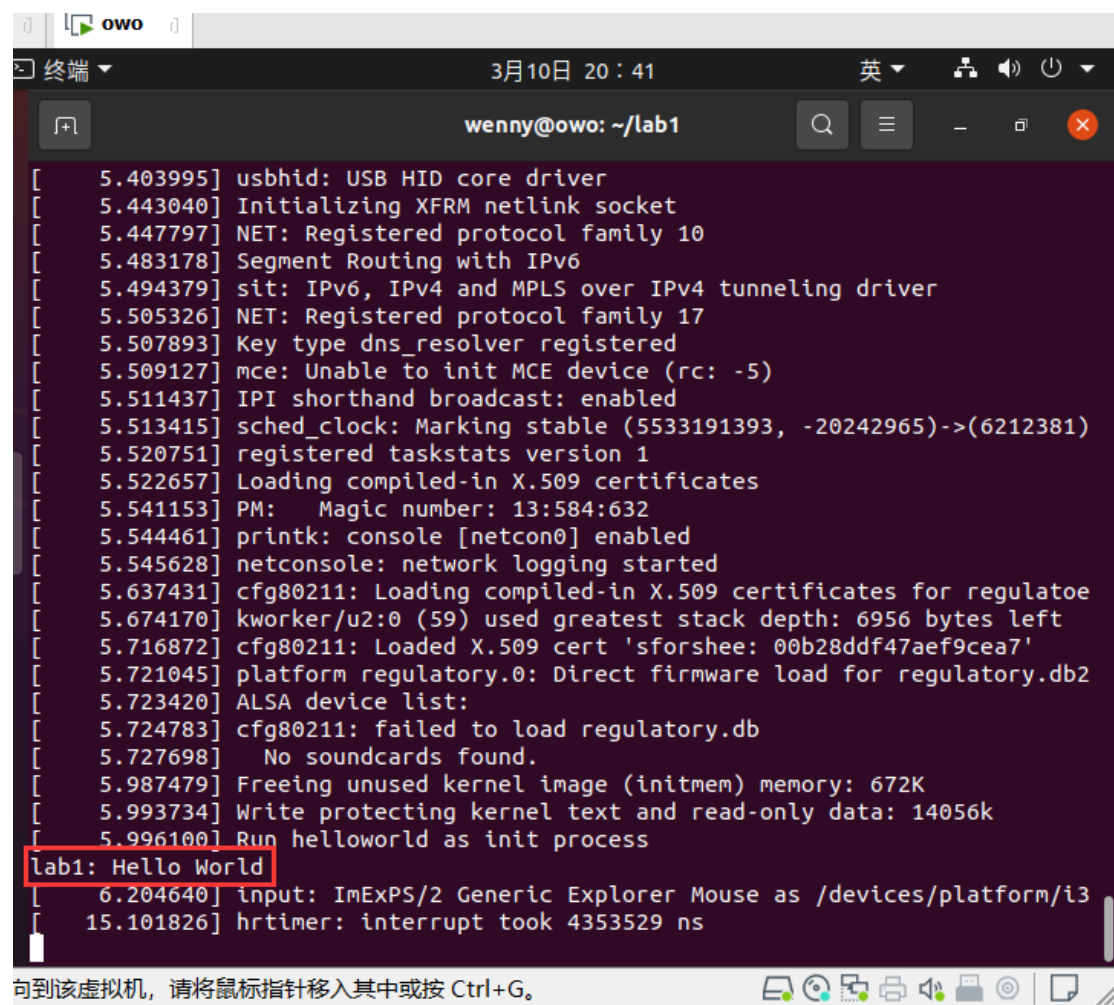


## gdb 调试

```
(gdb) file linux-5.10.19/vmlinux
Reading symbols from linux-5.10.19/vmlinux...
(gdb) target remote:1234
Remote debugging using :1234
0x00000fff0 in ?? ()
(gdb) break start_kernel
Breakpoint 1 at 0xc1faf7de: file init/main.c, line 849.
(gdb) c
Continuing.
```

加载 initramfs 后再进行 gdb 调试，可以看到输出了

lab1:Hello World



```
[ 5.403995] usbhid: USB HID core driver
[ 5.443040] Initializing XFRM netlink socket
[ 5.447797] NET: Registered protocol family 10
[ 5.483178] Segment Routing with IPv6
[ 5.494379] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[ 5.505326] NET: Registered protocol family 17
[ 5.507893] Key type dns_resolver registered
[ 5.509127] mce: Unable to init MCE device (rc: -5)
[ 5.511437] IPI shorthand broadcast: enabled
[ 5.513415] sched_clock: Marking stable (5533191393, -20242965)->(6212381)
[ 5.520751] registered taskstats version 1
[ 5.522657] Loading compiled-in X.509 certificates
[ 5.541153] PM: Magic number: 13:584:632
[ 5.544461] printk: console [netcon0] enabled
[ 5.545628] netconsole: network logging started
[ 5.637431] cfg80211: Loading compiled-in X.509 certificates for regulatoe
[ 5.674170] kworker/u2:0 (59) used greatest stack depth: 6956 bytes left
[ 5.716872] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 5.721045] platform regulatory.0: Direct firmware load for regulatory.db2
[ 5.723420] ALSA device list:
[ 5.724783] cfg80211: failed to load regulatory.db
[ 5.727698] No soundcards found.
[ 5.987479] Freeing unused kernel image (initmem) memory: 672K
[ 5.993734] Write protecting kernel text and read-only data: 14056k
[ 5.996100] Run helloworld as init process
lab1: Hello World
[ 6.204640] input: ImEXPS/2 Generic Explorer Mouse as /devices/platform/i3
[ 15.101826] hrtimer: interrupt took 4353529 ns
```

向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。



## 编译并启动 Busybox

下载并解压 Busybox

```
wenny@owo:~$ cd lab1
wenny@owo:~/lab1$ ls
busybox-1_33_0 helloworld.c linux-5.10.19
Busybox.tar.gz hwinitramfs linux-5.10.19.tar
helloworld initramfs-busybox-x86.cpio.gz mybusybox
wenny@owo:~/lab1$
```

编译 Busybox, 完成后如下所示

```
./_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.
-----
```

## 制作 Initramfs

使用如下简单脚本作为 init, 并加上执行权限

```
init
/home/wenny/lab1/mybusybox

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/sh
```

通过 cpio 打包好文件

```
./bin/stty
./proc
4455 块
```

加载 Busybox, 再使用 ls 命令查看当前文件夹

```
wenny@owo: ~/lab1
[ 5.747835] PM: Magic number: 13:774:562
[ 5.751218] printk: console [netcon0] enabled
[ 5.752765] netconsole: network logging started
[ 5.761603] cfg80211: Loading compiled-in X.509 certificates for regulatoe
[ 5.919194] modprobe (59) used greatest stack depth: 6964 bytes left
[ 5.936970] modprobe (60) used greatest stack depth: 6940 bytes left
[ 5.978243] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 5.986321] ALSA device list:
[ 5.990593] platform regulatory.0: Direct firmware load for regulatory.db2
[ 5.993517] No soundcards found.
[ 5.995746] cfg80211: failed to load regulatory.db
[ 6.252872] Freeing unused kernel image (initmem) memory: 672K
[ 6.259716] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i3
[ 6.266149] Write protecting kernel text and read-only data: 14056k
[ 6.267873] Run /init as init process
[ 6.586106] mount (64) used greatest stack depth: 6676 bytes left

Boot took 6.65 seconds

/bin/sh: can't access tty; job control turned off
/ #
/ #
/ #
/ # why
/bin/sh: why: not found
/ # ls
bin      etc      linuxrc  root     sys
dev      init     proc    /sbin    usr
/ #
```

至此实验完成。

### 3. 总结

在这次实验中, 先配置了实验的环境, 再编译 Linux 内核, 用 qemu 启动该内核并开启远程调试, 最后还编译并启动了 Busybox, 在前面所述两个过程中还制作了 initramfs。

总的来说实验的过程是比较顺利的, 在实验前我已下载 VMware Workstation Pro 并安装了 Ubuntu20.04 系统, 不同于老师推荐的 Virtualbox 和 Unbuntu18.04, 也有同学告诉我 18 版本更稳定, 所以有点担心实验中出现版本不兼容的问题, 但目前为止还没遇到。

在 Windows 系统下配过 VSC 因此在 Linux 下配置也没遇到什么问

题，同时也发觉了 `sudo apt-get install` 的便利之处。

在这次实验中还遇到了许多有关概念的问题,比如 `qemu`、`Initramfs` 等等,通过资料大概了解到 `qemu` 是一个模拟处理器的软件,能够用来启动内核,`Initramfs` (`initram file system`) 在 Linux 系统启动时使用,可以在启动早期提供一个用户态环境,借助它可以完成一些内核在启动阶段不易完成的工作,在这次实验文件都是被打包成 `cpio` 格式的 `Initramfs`,而不是`.tar`、`.gz` 等格式,通过查阅资料可知, Linux 系统只支持 `cpio` 格式的 `Initramfs`。

此外,通过编译并启动 `Busybox` 的过程接触了解了 `Busybox` 这个工具,它是一个集成了三百多个最常用 Linux 命令和工具的软件,有些人将 `BusyBox` 称为 Linux 工具里的瑞士军刀,希望在以后的过程中自己能够更熟悉这个工具,学会它的使用方法,并且能够利用它来构建简单的 OS。