

# 第三期训练营第二阶段 rust for linux 作业

## 一、环境配置

### 1、下载作业源码

git clone <https://github.com/cicvedu/cicv-r4l-3-fan5fan.git>

```
fan@Myco: /mnt/d/workspac X + v
fan@Myco: /mnt/d/workspace/rust/cicv-r4l-3-fan5fan$ ls
README.md busybox-1.36.1 linux r4l_experiment src_el000
fan@Myco: /mnt/d/workspace/rust/cicv-r4l-3-fan5fan$ tree
```

### 2、编译 BusyBox

#### 2.1、cd cicv-r4l-3-fan5fan/busybox-1.36.1/

```
fan@Myco: /mnt/d/workspac X + v
fan@Myco: /mnt/d/workspace/rust/cicv-r4l-3-fan5fan$ cd busybox-1.36.1/
fan@Myco: /mnt/d/workspace/rust/cicv-r4l-3-fan5fan/busybox-1.36.1$ |
```

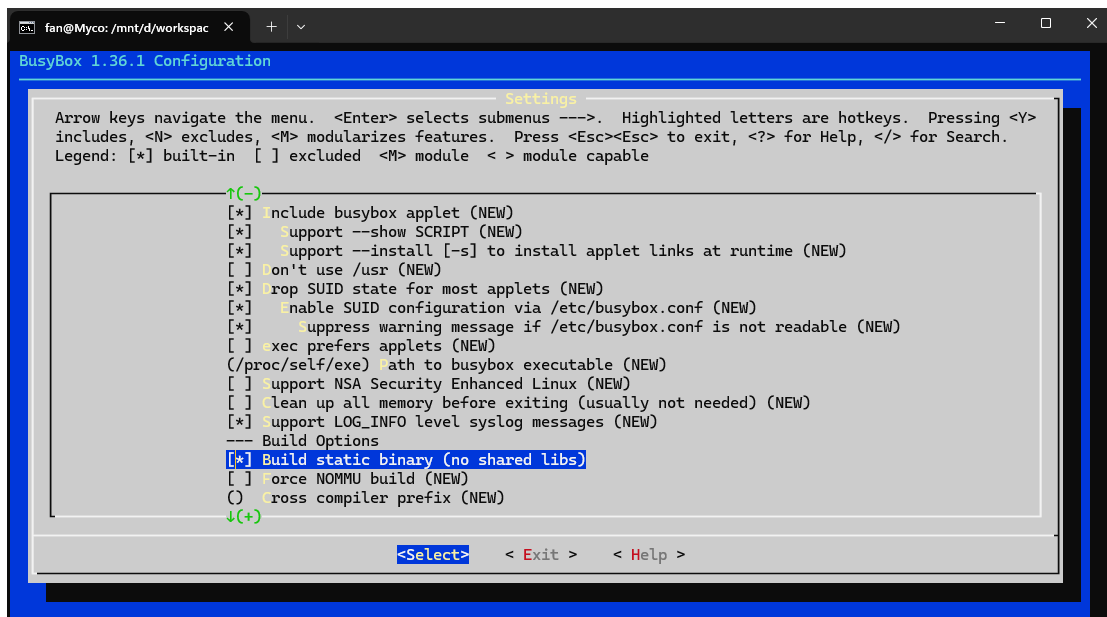
#### 2.2、make menuconfig

```
fan@Myco: /mnt/d/workspac X + v
BusyBox 1.36.1 Configuration

-- Busybox Configuration --
Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search.
Legend: [*] built-in [ ] excluded <M> module < > module capable

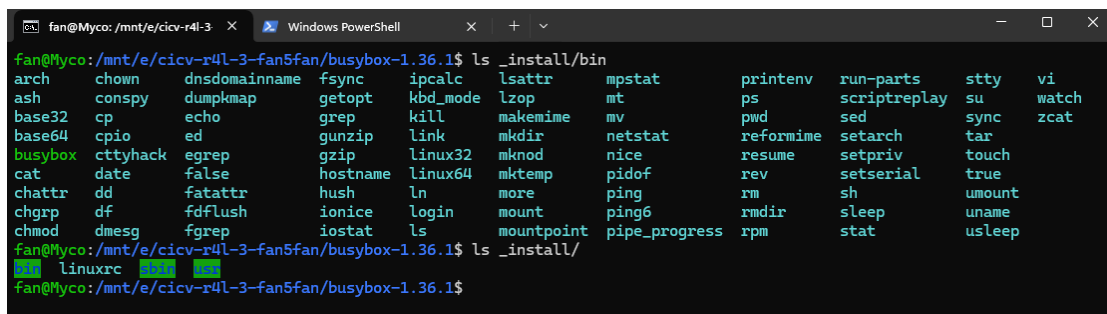
  Settings --->
--- Applets
    archival Utilities --->
    coreutils --->
    console Utilities --->
    debian Utilities --->
    libc-utils --->
    editors --->
    finding Utilities --->
    init Utilities --->
    login/Password Management Utilities --->
    linux Ext2 FS Progs --->
    linux Module Utilities --->
    linux System Utilities --->
    Miscellaneous Utilities --->
    Networking Utilities --->
    ↓(+)
```

#### 2.3、启动静态链接



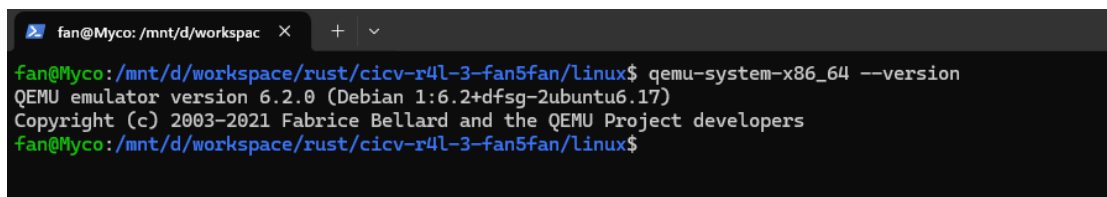
## 2.4、编译、安装

```
make install -j$(nproc)
```



## 3、安装 Qemu

```
sudo apt install qemu-system-x86
```



## 4、安装 rust

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

## 5、配置 linux 文件夹

```
fan@Myco: /mnt/d/workspac X + v
fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ ls
COPYING      Kbuild      MAINTAINERS  README.md    scripts
CREDITS      Kconfig     Makefile     rustc        rustfmt
Cargo.toml   rust-toolchain.toml  README      rustdoc      rustfmt.toml
fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ rustup override set $(scripts/min-tool-version.sh rustc)
info: syncing channel updates for '1.62.0-x86_64-unknown-linux-gnu'
info: latest update on 2022-06-30, rust version 1.62.0 (a8314ef7d 2022-06-27)
info: downloading component 'cargo'
info: downloading component 'clippy'
info: downloading component 'rust-docs'
18.3 MiB / 18.3 MiB (100 %) 7.8 MiB/s in 2s ETA: 0s
info: downloading component 'rust-std'
26.0 MiB / 26.0 MiB (100 %) 7.1 MiB/s in 4s ETA: 0s
info: downloading component 'rustc'
54.1 MiB / 54.1 MiB (100 %) 7.1 MiB/s in 7s ETA: 0s
info: downloading component 'rustfmt'
info: installing component 'cargo'
info: installing component 'clippy'
info: installing component 'rust-docs'
18.3 MiB / 18.3 MiB (100 %) 6.3 MiB/s in 2s ETA: 0s
info: installing component 'rust-std'
26.0 MiB / 26.0 MiB (100 %) 10.2 MiB/s in 2s ETA: 0s
info: installing component 'rustc'
54.1 MiB / 54.1 MiB (100 %) 15.4 MiB/s in 3s ETA: 0s
info: installing component 'rustfmt'

1.62.0-x86_64-unknown-linux-gnu installed - rustc 1.62.0 (a8314ef7d 2022-06-27)
info: override toolchain for '/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux' set to '1.62.0-x86_64-unknown-linux-gnu'

fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ rustup component add rust-src
info: downloading component 'rust-src'
info: installing component 'rust-src'

Installed package 'bindgen v0.56.0' (executable 'bindgen')
fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ rustup component add rustfmt
info: component 'rustfmt' for target 'x86_64-unknown-linux-gnu' is up to date
fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ rustup component add clippy
info: component 'clippy' for target 'x86_64-unknown-linux-gnu' is up to date

fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ make LLVM=1 rustavailable
Rust is available!
```

## 二、作业一 编译 Linux 内核

```
fan@Myco: /mnt/d/workspac X + v
fan@Myco:/mnt/d/workspace/rust/cicv-r4l-3-fan5fan/linux$ make x86_64_defconfig
HOSTCC scripts/basic/fixdep
HOSTCC scripts/kconfig/confdata.o
HOSTCC scripts/kconfig/expr.o
HOSTCC scripts/kconfig/lexer.lex.o
HOSTCC scripts/kconfig/menu.o
HOSTCC scripts/kconfig/parser.tab.o
HOSTCC scripts/kconfig/preprocess.o
HOSTCC scripts/kconfig/symbol.o
HOSTCC scripts/kconfig/util.o
HOSTLD scripts/kconfig/conf
#
# No change to .config
#
```

```
fan@Myco: /mnt/d/workspac X + v
.config - Linux/x86 6.1.0-rc1 Kernel Configuration
> General setup

Arrow keys navigate the menu. <Enter> selects submenus ---- (or empty submenus ----). Highlighted letters
are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?>
for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module capable

^(-)
() Ininitramfs source file(s)
[*] Support initial ramdisk/ramfs compressed using gzip
[*] Support initial ramdisk/ramfs compressed using bzip2
[*] Support initial ramdisk/ramfs compressed using LZMA
[*] Support initial ramdisk/ramfs compressed using XZ
[*] Support initial ramdisk/ramfs compressed using LZ0
[*] Support initial ramdisk/ramfs compressed using LZ4
[*] Support initial ramdisk/ramfs compressed using ZSTD
[ ] Boot config support
[*] Preserve cpio archive mtimes in initramfs
Compiler optimization level (Optimize for performance (-O2)) ---->
[ ] Configure standard kernel features (expert users) ---->
[ ] Embedded system
Kernel Performance Events And Counters ---->
[*] Profiling support
[*] Rust support

<Select> < Exit > < Help > < Save > < Load >
```

```
fan@Myco: ~/cicv-r4l-3-fan5l X Windows PowerShell X + v
CC arch/x86/boot/compressed/cpuflags.o
CC arch/x86/boot/compressed/early_serial_console.o
CC arch/x86/boot/compressed/kaslr.o
CC arch/x86/boot/compressed/ident_map_64.o
CC arch/x86/boot/compressed/idt_64.o
AS arch/x86/boot/compressed/idt_handlers_64.o
AS arch/x86/boot/compressed/mem_encrypt.o
CC arch/x86/boot/compressed/pgtable_64.o
CC arch/x86/boot/compressed/acpi.o
AS arch/x86/boot/compressed/efi_thunk_64.o
CC arch/x86/boot/compressed/efi.o
CC arch/x86/boot/compressed/misc.o
GZIP arch/x86/boot/compressed/vmlinux.bin.gz
MKPIGGY arch/x86/boot/compressed/piggy.S
AS arch/x86/boot/compressed/piggy.o
LD arch/x86/boot/compressed/vmlinux
ZOFFSET arch/x86/boot/zoffset.h
OBJCOPY arch/x86/boot/vmlinux.bin
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)
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ ls
COPYING      LICENSES     README.md   certs        init         mm           rust         tools        vmlinux.o
CREDITS      MAINTAINERS System.map  crypto       io_uring     modules.builtin  samples     usr
Documentation Makefile     arch        drivers      ipc          modules.builtin.modinfo  scripts     virt
Kbuild       Module.symvers block       fs           kernel       modules.order   security    vmlinux
Kconfig      README       built-in.a include      lib          net           sound       vmlinux.a
fan@Myco:~/cicv-r4l-3-fan5fan/linux$
```

注：用 WSL 编译内核时，不能将文件放在 Windows 的文件系统内，必须放在 WSL 里面，不然会编译不过

### 三、作业二 对 Linux 内核进行一些配置

#### 1、操作步骤

##### 1.0、进入源码文件夹

```
cd cicv-r4l-3-fan5fan/src_e1000
```

##### 1.1、make

```
fan@Myco: ~/cicv-r4l-3-fan5l x Windows PowerShell
fan@Myco:~/cicv-r4l-3-fan5fan/src_e1000$ make LLVM=1
make -C ../linux M=$PWD
make[1]: Entering directory '/home/fan/cicv-r4l-3-fan5fan/linux'
RUSTC [M] /home/fan/cicv-r4l-3-fan5fan/src_e1000/r4l_e1000_demo.o
MODPOST /home/fan/cicv-r4l-3-fan5fan/src_e1000/Module.symvers
CC [M] /home/fan/cicv-r4l-3-fan5fan/src_e1000/r4l_e1000_demo.mod.o
LD [M] /home/fan/cicv-r4l-3-fan5fan/src_e1000/r4l_e1000_demo.ko
make[1]: Leaving directory '/home/fan/cicv-r4l-3-fan5fan/linux'
fan@Myco:~/cicv-r4l-3-fan5fan/src_e1000$ ls
Kbuild      Module.symvers  consts.rs       hw_defs.rs      r4l_e1000_demo.mod  r4l_e1000_demo.o
LICENSE     README.md       dump.dat        modules.order   r4l_e1000_demo.mod.c  r4l_e1000_demo.rs
Makefile    build_image.sh  e1000_ops.rs   r4l_e1000_demo.ko  r4l_e1000_demo.mod.o  ring_buf.rs
```

## 1.2、执行 ./build\_image.sh

```
fan@Myco: ~/cicv-r4l-3-fan5l x Windows PowerShell
[ 0.000000] Linux version 6.1.0-rc1 (fan@Myco) (Ubuntu clang version 14.0.0-1ubuntu1.1, Ubuntu LLD 14.0.0) #1 SMP PR4
[ 0.000000] Command line: root=/dev/ram rdinit=sbin/init ip=10.0.2.15::10.0.2.1:255.255.255.0 console=ttyS0 no_timerk
[ 0.000000] x86/fpu: x87 FPU will use FXSAVE
[ 0.000000] signal: max sigframe size: 1440
[ 0.000000] BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[ 0.000000] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000001fdfff] usable
[ 0.000000] BIOS-e820: [mem 0x00000000007fe000-0x00000000007fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000ffffc000-0x00000000ffffffffff] reserved
[ 0.000000] NX (Execute Disable) protection: active
[ 0.000000] SMBIOS 2.8 present.
[ 0.000000] DMI: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.15.0-1 04/01/2014
[ 0.000000] last_pfn = 0x7fe0 max_arch_pfn = 0x400000000
[ 0.000000] x86/PAT: Configuration [0-7]: WB WC UC- UC WB WP UC- WT
[ 0.000000] found SMP MP-table at [mem 0x000f5ba0-0x000f5baf]
[ 0.000000] RAMDISK: [mem 0x07d48000-0x07fdffff]
[ 0.000000] ACPI: Early table checksum verification disabled
[ 0.000000] ACPI: RSDP 0x0000000000f59e0 000014 (v00 BOCHS )
[ 0.000000] ACPI: RSDT 0x0000000007fe1905 000034 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: FACP 0x0000000007fe17b9 000074 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: DSDT 0x0000000007fe0040 001779 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: FACS 0x0000000007fe0000 000040
[ 0.000000] ACPI: APIC 0x0000000007fe182d 000078 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: HPET 0x0000000007fe18a5 000038 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: WAET 0x0000000007fe18dd 000028 (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: Reserving FACP table memory at [mem 0x7fe17b9-0x7fe182c]
[ 0.000000] ACPI: Reserving DSDT table memory at [mem 0x7fe0040-0x7fe17b8]
[ 0.000000] ACPI: Reserving FACS table memory at [mem 0x7fe0000-0x7fe003f]
```

```
fan@Myco: ~/cicv-r4l-3-fan5l x Windows PowerShell
[ 2.203201] printk: console [netcon0] enabled
[ 2.203851] netconsole: network logging started
[ 2.260234] tsc: Refined TSC clocksource calibration: 2112.005 MHz
[ 2.262084] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x1e717d1ec85, max_idle_ns: 440795234704 ns
[ 2.263177] clocksource: Switched to clocksource tsc
[ 2.757727] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/seriol/input/input3
[ 2.778030] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[ 2.795743] IP-Config: Complete:
[ 2.795927] device=eth0, hwaddr=52:54:00:12:34:56, ipaddr=10.0.2.15, mask=255.255.255.0, gw=10.0.2.1
[ 2.797357] host=10.0.2.15, domain=, nis-domain=(none)
[ 2.798270] bootserver=255.255.255.255, rootserver=255.255.255.255, rootpath=
[ 2.803728] cfg80211: Loading compiled-in X.509 certificates for regulatory database
[ 2.870234] modprobe (66) used greatest stack depth: 14272 bytes left
[ 2.887758] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 2.890568] platform regulatory.0: Direct firmware load for regulatory.db failed with error -2
[ 2.891856] cfg80211: failed to load regulatory.db
[ 2.893416] ALSA device list:
[ 2.893681] No soundcards found.
[ 2.951080] Freeing unused kernel image (initmem) memory: 1324K
[ 2.955058] Write protecting the kernel read-only data: 24576k
[ 2.959012] Freeing unused kernel image (text/rodata gap) memory: 2032K
[ 2.960730] Freeing unused kernel image (rodata/data gap) memory: 824K
[ 3.131633] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[ 3.132939] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[ 3.138192] Run sbin/init as init process
[ 3.195189] mount (71) used greatest stack depth: 14160 bytes left
[ 3.349714] mdev (73) used greatest stack depth: 13960 bytes left

Please press Enter to activate this console.
~ #
```

## 1.3、执行 ifconfig

```
fan@Mycro: ~/cicv-r4l-3-fanSI x Windows PowerShell
[ 2.959012] Freeing unused kernel image (text/rodata gap) memory: 2032K
[ 2.960730] Freeing unused kernel image (rodata/data gap) memory: 824K
[ 3.131633] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[ 3.132939] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[ 3.138192] Run/sbin/init as init process
[ 3.195189] mount (71) used greatest stack depth: 14160 bytes left
[ 3.349714] mdev (73) used greatest stack depth: 13960 bytes left

Please press Enter to activate this console.
~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 52:54:00:12:34:56
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fec0::5054:ff:fe12:3456/64 Scope:Site
          inet6 addr: fe80::5054:ff:fe12:3456/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:220 (220.0 B)  TX bytes:672 (672.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

~ # |
```

#### 1.4、执行 ping

```
fan@Mycro: ~/cicv-r4l-3-fanSI x Windows PowerShell
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
64 bytes from 10.0.2.2: seq=0 ttl=255 time=5.813 ms
64 bytes from 10.0.2.2: seq=1 ttl=255 time=0.667 ms
64 bytes from 10.0.2.2: seq=2 ttl=255 time=0.789 ms
64 bytes from 10.0.2.2: seq=3 ttl=255 time=0.783 ms
64 bytes from 10.0.2.2: seq=4 ttl=255 time=0.795 ms
64 bytes from 10.0.2.2: seq=5 ttl=255 time=1.014 ms
64 bytes from 10.0.2.2: seq=6 ttl=255 time=0.429 ms
64 bytes from 10.0.2.2: seq=7 ttl=255 time=0.451 ms
64 bytes from 10.0.2.2: seq=8 ttl=255 time=0.648 ms
64 bytes from 10.0.2.2: seq=9 ttl=255 time=0.768 ms
^C
--- 10.0.2.2 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 0.429/1.215/5.813 ms
~ # |
```

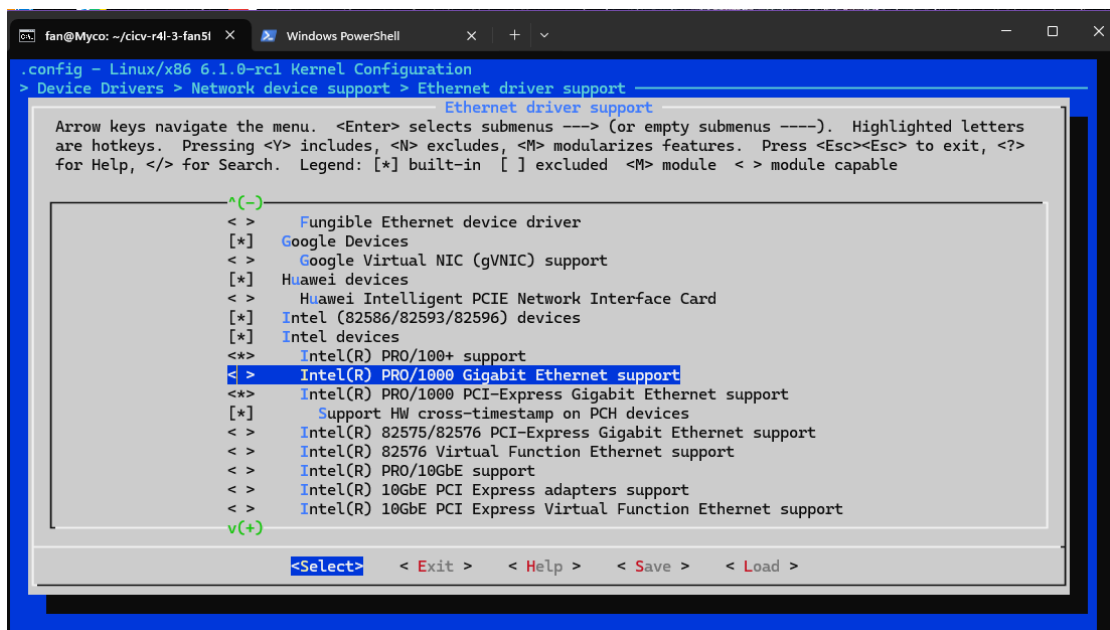
#### 1.5、安装 r4l\_e1000\_demo.ko

```
fan@Mycro: ~/cicv-r4l-3-fanSI x Windows PowerShell
~ # ls
bin          linuxrc      root         usr
dev          proc         sbin
etc          r4l_e1000_demo.ko  sys

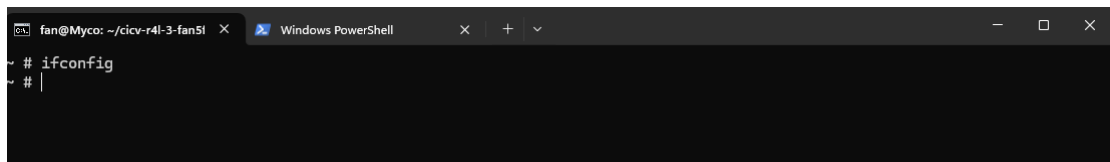
~ # insmod r4l_e1000_demo.ko
[ 369.831811] r4l_e1000_demo: loading out-of-tree module taints kernel.
[ 369.840387] r4l_e1000_demo: Rust for linux e1000 driver demo (init)
[ 369.842967] insmod (87) used greatest stack depth: 13336 bytes left
~ # |
```

#### 1.6、退出 Qemu

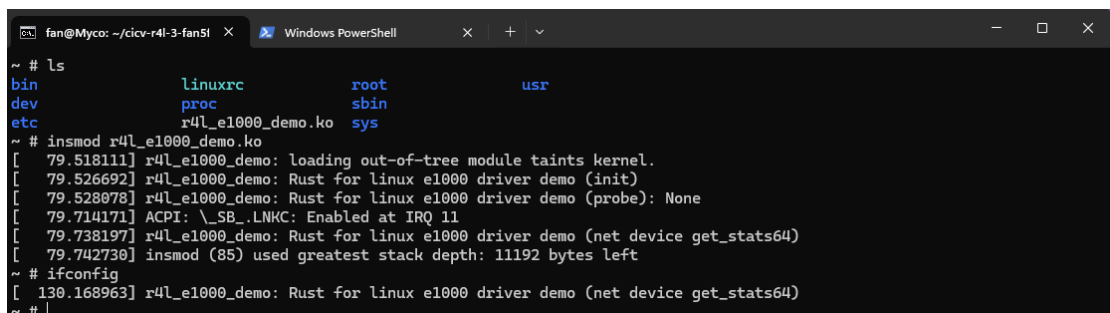
#### 1.7、进入 linux 文件夹，重新 make LLVM=1 menuconfig，重新编译 linux 内核



## 1.8、重新进入 Qemu，并执行 ifconfig



## 1.9、安装 r4l\_e1000\_demo.ko



## 1.10、设置

ip link set eth0 up

```
fan@Myco: ~/cicv-r4l-3-fanSl  x  Windows PowerShell  x  +  v
~ # ip link set eth0 up
[ 234.092130] r4l_e1000_demo: Rust for linux e1000 driver demo (net device open)
[ 234.096193] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 234.098806] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
~ # [ 234.106640] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 234.114033] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0, rdt=7, rdh=0
[ 234.115791] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 234.116416] r4l_e1000_demo: pending_irqs: 3
[ 234.117542] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 234.696005] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=7, rdh=0
[ 234.697117] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 234.697843] r4l_e1000_demo: pending_irqs: 3
[ 234.698197] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 234.887046] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=7, rdh=0
[ 234.887682] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 234.889110] r4l_e1000_demo: pending_irqs: 3
[ 234.889792] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 235.719924] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=7, rdh=0
[ 235.721561] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 235.721978] r4l_e1000_demo: pending_irqs: 3
[ 235.722919] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 235.725610] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=7, rdh=0
[ 235.726903] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 235.727133] r4l_e1000_demo: pending_irqs: 3
[ 235.728443] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 236.231693] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=7, rdh=0
[ 236.233094] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 236.233642] r4l_e1000_demo: pending_irqs: 3
[ 236.234146] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 239.687528] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=6, tdh=6, rdt=7, rdh=0
```

ip addr add broadcast 10.0.2.255 dev eth0

```
fan@Myco: ~/cicv-r4l-3-fanSl  x  Windows PowerShell  x  +  v
~ # ip addr add broadcast 10.0.2.255 dev eth0
[ 485.837050] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 485.839991] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
ip: RTNETLINK answers: Invalid argument
~ # [ 513.159856] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=7, rdh=0
[ 513.160936] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 513.161720] r4l_e1000_demo: pending_irqs: 3
[ 513.162306] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)

~ # |
```

ip addr add 10.0.2.15/255.255.255.0 dev eth0

```
fan@Myco: ~/cicv-r4l-3-fanSl  x  Windows PowerShell  x  +  v
~ # ip addr add 10.0.2.15/255.255.255.0 dev eth0
[ 600.906493] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 600.907423] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)

~ # |
```

ip route add default via 10.0.2.1

```
fan@Myco: ~/cicv-r4l-3-fanSl  x  Windows PowerShell  x  +  v
~ # ip route add default via 10.0.2.1
~ # |
```

ping 10.0.2.2



```
fan@Myco: ~/cicv-r4l-3-fan5l x Windows PowerShell
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
[ 780.387635] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=4, rdh=5
[ 780.390692] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 780.392335] r4l_e1000_demo: pending_irqs: 131
[ 780.392987] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=0 ttl=255 time=7.606 ms
[ 781.396255] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=5, rdh=6
[ 781.397992] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 781.398872] r4l_e1000_demo: pending_irqs: 131
[ 781.399775] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=1 ttl=255 time=5.849 ms
[ 782.403578] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=6, rdh=7
[ 782.404257] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 782.405074] r4l_e1000_demo: pending_irqs: 131
[ 782.405708] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=2 ttl=255 time=3.493 ms
[ 783.408120] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=7, rdh=8
[ 783.409910] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 783.411216] r4l_e1000_demo: pending_irqs: 131
[ 783.412306] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=3 ttl=255 time=6.042 ms
[ 784.415030] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=8, rdh=1
[ 784.416040] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 784.417031] r4l_e1000_demo: pending_irqs: 131
[ 784.417482] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=4 ttl=255 time=4.453 ms
^C
--- 10.0.2.2 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 3.493/5.488/7.606 ms
```

## 2、问题

2.1、编译成内核模块，是在哪个文件中以哪条语句定义的？

Kbuild 文件中的 `obj-m := r4l_e1000_demo.o` 这里定义的

2.2、该模块位于独立的文件夹内，却能编译成 Linux 内核模块，这叫做 out-of-tree module，请分析它是如何与内核代码产生联系的？

加载一个 out-of-tree 模块时，它通常有以下几种方式与内核代码发生关联：

1. 直接内核符号引用：模块可能会直接引用内核导出的符号，如函数和变量。
2. 内核服务：模块可能会使用内核提供的服务，如消息传递机制。
3. 设备文件节点：模块可能会创建设备文件节点，通过 `/dev` 目录与用户空间通信。
4. 模块参数：模块可以定义参数，在加载模块时传入这些参数。

## 四、作业三

1、在 `samples/rust` 里添加 `rust_helloworld.rs`

```

fan@Myco: ~/cicv-r4l-3-fan5l  Windows PowerShell
fan@Myco:~/cicv-r4l-3-fan5fan/linux/samples/rust$ cat rust_helloworld.rs
// SPDX-License-Identifier: GPL-2.0
// ! Rust minimal sample.

use kernel::prelude::*;

module! {
    type: RustHelloWorld,
    name: "rust_helloworld",
    author: "whocare",
    description: "hello world module in rust",
    license: "GPL",
}

struct RustHelloWorld {}

impl kernel::Module for RustHelloWorld {
    fn init(_name: &'static CStr, _module: &'static ThisModule) -> Result<Self> {
        pr_info!("Hello World from Rust module");
        Ok(RustHelloWorld {})
    }
}
fan@Myco:~/cicv-r4l-3-fan5fan/linux/samples/rust$

```

## 2、修改 Kconfig

```

fan@Myco: ~/cicv-r4l-3-fan5l  fan@Myco: ~/cicv-r4l-3-fan5fr
# SPDX-License-Identifier: GPL-2.0

menuconfig SAMPLES_RUST
    bool "Rust samples"
    depends on RUST
    help
        You can build sample Rust kernel code here.

        If unsure, say N.

if SAMPLES_RUST
    config PRINT_HELLOWORLD
        tristate "Print Helloworld in Rust"
        help
            This option builds the Rust helloworld module sample.

            To compile this as a module, choose M here:
            the module will be called rust_helloworld.

            If unsure, say N.
    config SAMPLE_RUST_MINIMAL
        tristate "Minimal"

```

## 3、修改 Makefile

```

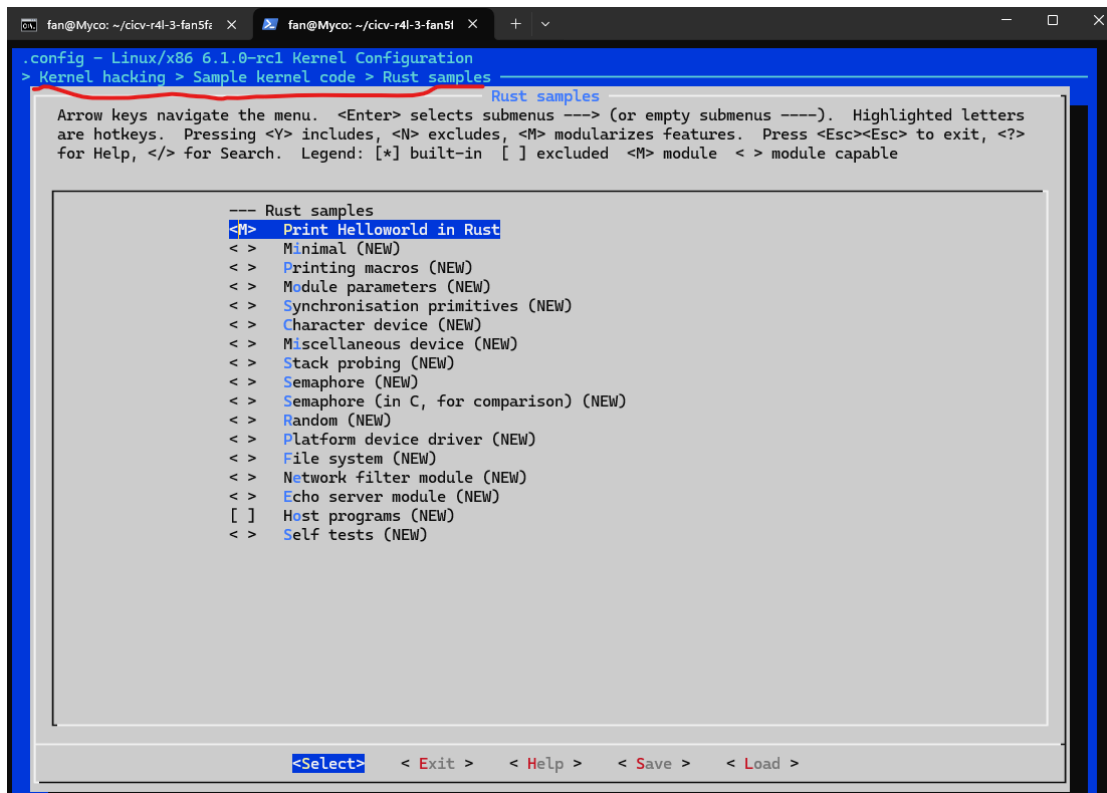
fan@Myco: ~/cicv-r4l-3-fan5l  fan@Myco: ~/cicv-r4l-3-fan5fr
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ cat samples/rust/Makefile
# SPDX-License-Identifier: GPL-2.0

obj-$(CONFIG_SAMPLE_RUST_MINIMAL) += rust_minimal.o
obj-$(CONFIG_SAMPLE_RUST_PRINT) += rust_print.o
obj-$(CONFIG_SAMPLE_RUST_MODULE_PARAMETERS) += rust_module_parameters.o
obj-$(CONFIG_SAMPLE_RUST_SYNC) += rust_sync.o
obj-$(CONFIG_SAMPLE_RUST_CHRDEV) += rust_chrdev.o
obj-$(CONFIG_SAMPLE_RUST_MISCDEV) += rust_miscdev.o
obj-$(CONFIG_SAMPLE_RUST_STACK_PROBING) += rust_stack_probing.o
obj-$(CONFIG_SAMPLE_RUST_SEMAPHORE) += rust_semaphore.o
obj-$(CONFIG_SAMPLE_RUST_SEMAPHORE_C) += rust_semaphore_c.o
obj-$(CONFIG_SAMPLE_RUST_RANDOM) += rust_random.o
obj-$(CONFIG_SAMPLE_RUST_PLATFORM) += rust_platform.o
obj-$(CONFIG_SAMPLE_RUST_NETFILTER) += rust_netfilter.o
obj-$(CONFIG_SAMPLE_RUST_ECHO_SERVER) += rust_echo_server.o
obj-$(CONFIG_SAMPLE_RUST_FS) += rust_fs.o
obj-$(CONFIG_SAMPLE_RUST_SELFTESTS) += rust_selftests.o

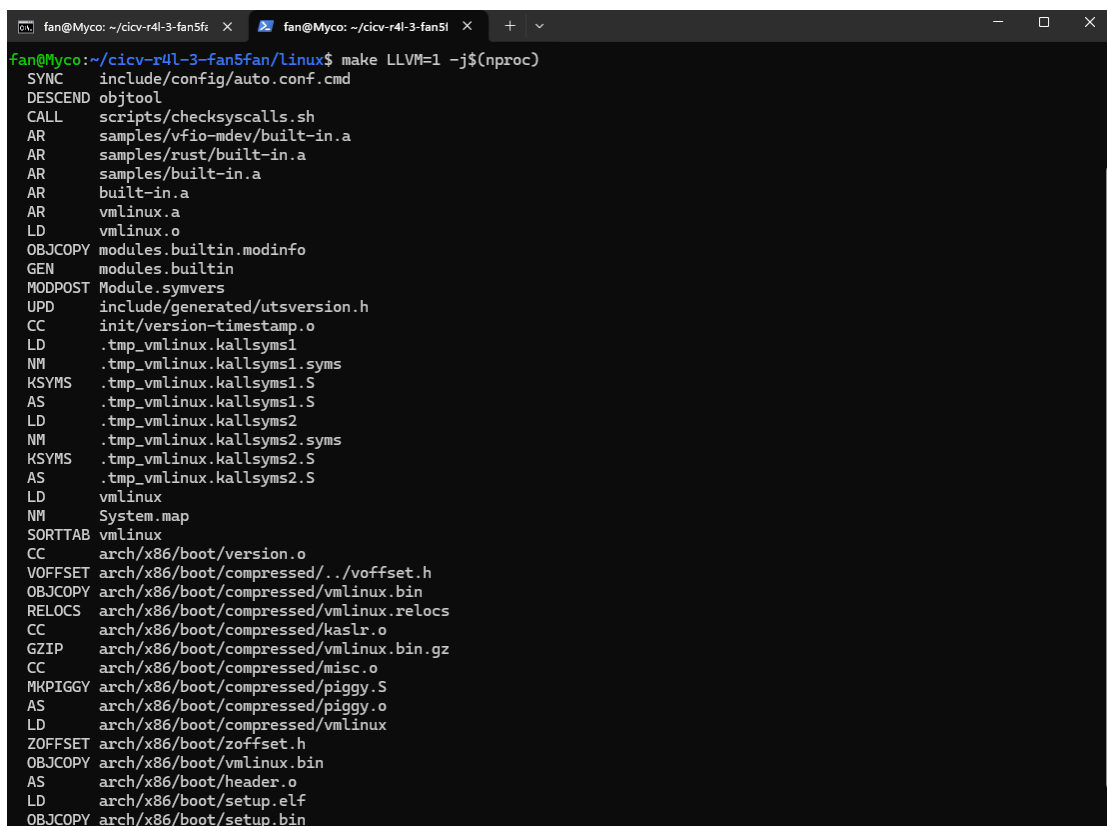
obj-$(CONFIG_PRINT_HELLOWORLD) += rust_helloworld.o
subdir-$(CONFIG_SAMPLE_RUST_HOSTPROGS) += hostprogs
fan@Myco:~/cicv-r4l-3-fan5fan/linux$

```

## 4、make LLVM=1 menuconfig



5、make LLVM=1 -j\$(nproc)



6、复制 rust\_helloworld.ko 文件到 src\_e1000/rootfs/目录下

```
fan@Myco: ~/cicv-r4l-3-fan5l  x  fan@Myco: ~/cicv-r4l-3-fan5f  x  +  v
~ # ls
bin          proc          sbin
dev          r4l_e1000_demo.ko  sys
etc          root          usr
linuxrc     rust_helloworld.ko
~ #
```

7、执行 [build\\_image.sh](#)

8、执行 insmod 加载 rust\_helloworld.ko

```
fan@Myco: ~/cicv-r4l-3-fan5f  x  fan@Myco: ~/cicv-r4l-3-fan5l  x  +  v
[ 2.013903] registered taskstats version 1
[ 2.014279] Loading compiled-in X.509 certificates
[ 2.023019] cryptomgr_test (44) used greatest stack depth: 15584 bytes left
[ 2.032167] PM: Magic number: 0:431:939
[ 2.034056] printk: console [netcon0] enabled
[ 2.034540] netconsole: network logging started
[ 2.066467] ata2: found unknown device (class 0)
[ 2.076862] ata2.00: ATAPI: QEMU DVD-ROM, 2.5+, max UDMA/100
[ 2.089787] scsi 1:0:0:0: CD-ROM            QEMU      QEMU DVD-ROM    2.5+ PQ: 0 ANSI: 5
[ 2.127766] sr 1:0:0:0: [sr0] scsi3-mmc drive: 4x/4x cd/rw xa/form2 tray
[ 2.128496] cdrom: Uniform CD-ROM driver Revision: 3.20
[ 2.149392] sr 1:0:0:0: Attached scsi generic sg0 type 5
[ 2.587686] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/seriol/input/input3
[ 2.601826] tsc: Refined TSC clocksource calibration: 2111.972 MHz
[ 2.603762] clocksource: tsc: mask: 0xffffffffffffff max_cycles: 0x1e715e1f384, max_idle_ns: 440795306599 ns
[ 2.606298] clocksource: Switched to clocksource tsc
[ 14.890052] cfg80211: Loading compiled-in X.509 certificates for regulatory database
[ 14.960222] modprobe (67) used greatest stack depth: 14272 bytes left
[ 14.975220] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 14.977810] platform regulatory.0: Direct firmware load for regulatory.db failed with error -2
[ 14.981247] ALSA device list:
[ 14.982172]   No soundcards found.
[ 14.983223] cfg80211: failed to load regulatory.db
[ 15.038179] Freeing unused kernel image (initmem) memory: 1324K
[ 15.039830] Write protecting the kernel read-only data: 24576k
[ 15.045956] Freeing unused kernel image (text/rodata gap) memory: 2032K
[ 15.047575] Freeing unused kernel image (rodata/data gap) memory: 840K
[ 15.218891] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[ 15.219596] Run sbin/init as init process
[ 15.269815] mount (72) used greatest stack depth: 14160 bytes left
[ 15.426089] mdev (74) used greatest stack depth: 13944 bytes left

Please press Enter to activate this console.
~ # ls
bin          proc          sbin
dev          r4l_e1000_demo.ko  sys
etc          root          usr
linuxrc     rust_helloworld.ko
~ # insmod rust_helloworld.ko
[ 27.210838] rust_helloworld: Hello World from Rust module
~ #
```

Makefile 文件添加一行：

```
obj-$(CONFIG_PRINT_HELLOWORLD) += rust_helloworld.o
```

Kconfig 文件添加：

```
config PRINT_HELLOWORLD
    tristate "Print Helloworld in Rust"
    help
```

This option builds the Rust helloworld module sample.

To compile this as a module, choose M here:

the module will be called rust\_helloworld.

If unsure, say N.

## 五、作业四

### 1、修改 r4l\_e1000\_demo.rs

```
fn device_remove(&self) {  
    pr_info!("Rust for linux e1000 driver demo (device_remove)\n");  
  
    let netdev = self._netdev_reg.dev_get();  
    netdev.netif_carrier_off();  
    netdev.netif_stop_queue();  
}  
  
fn remove(dev: &mut pci::Device, data: &Self::Data) {  
    pr_info!("Rust for linux e1000 driver demo (remove)\n");  
  
    let bars = dev.selectBars((bindings::IORESOURCE_MEM |  
bindings::IORESOURCE_IO) as u64);  
    dev.release_selected_regions(bars);  
    dev.disable_device();  
    drop(data);  
}
```

### 2、修改 linux/rust/kernel/pci.rs

```
pub trait Driver {  
    fn remove(dev: &mut Device, _data: &Self::Data);  
}  
  
extern "C" fn remove_callback(pdev: *mut bindings::pci_dev) {
```

```

// SAFETY: `pdev` is guaranteed to be a valid, non-null pointer.

let ptr = unsafe { bindings::pci_get_drvdata(pdev) };

// SAFETY:

// - we allocated this pointer using `T::Data::into_pointer`,

//   so it is safe to turn back into a `T::Data`.

// - the allocation happened in `probe`, no-one freed the memory,

//   `remove` is the canonical kernel location to free driver data. so OK

// to convert the pointer back to a Rust structure here.

let data = unsafe { T::Data::from_pointer(ptr) };

let mut dev = unsafe { Device::from_ptr(pdev) };

T::remove(&mut dev, &data);

<T::Data as driver::DeviceRemoval>::device_remove(&data);
}

```

并添加两个函数

```

impl Device {

    pub fn release_selected_regions(&mut self, bars: i32) {

        unsafe { bindings::pci_release_selected_regions(self.ptr, bars) };

    }

    pub fn disable_device(&mut self) {

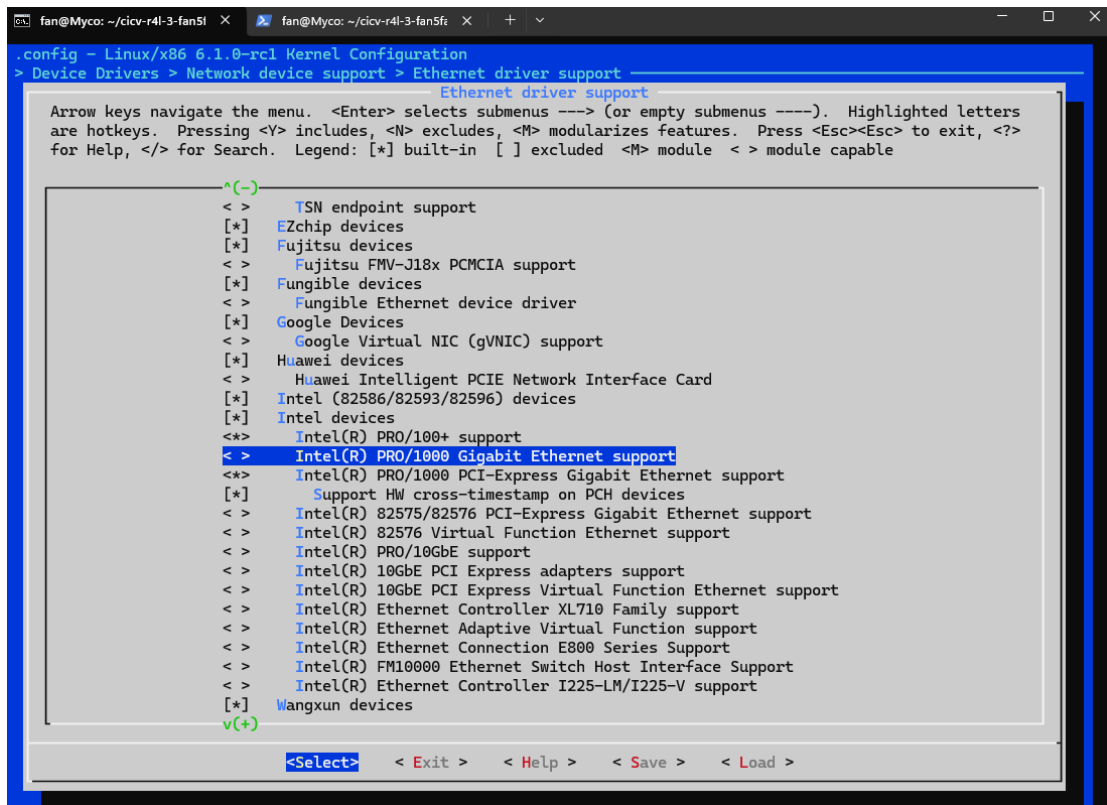
        unsafe { bindings::pci_disable_device(self.ptr) };

    }

}

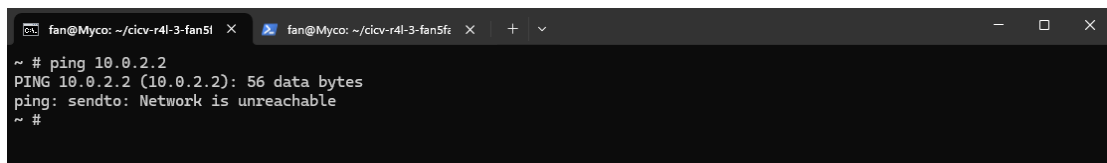
```

3、取消选中内置 e1000 网卡驱动，重新编译内核

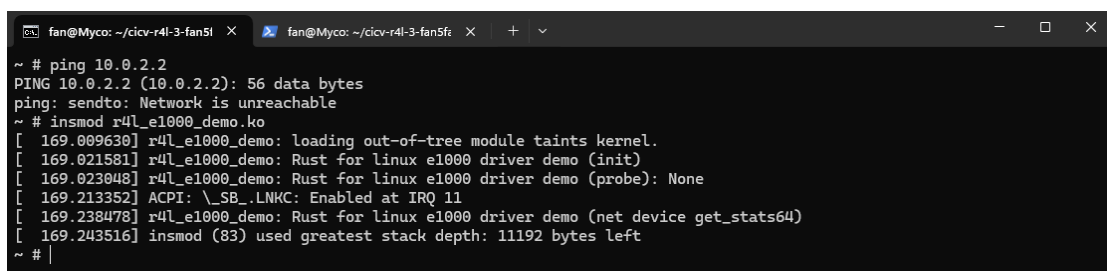


4、运行 src\_e1000/build\_image.sh

5、进入 qemu，并测试网络



6、加载 r4l\_e1000\_demo.ko



7、添加网络设置

```

fan@Myco: ~/cicv-r4l-3-fan5l  X  fan@Myco: ~/cicv-r4l-3-fan5f  X  +  v
~ # ip link set eth0 up
[ 245.137947] r4l_e1000_demo: Rust for linux e1000 driver demo (net device open)
[ 245.144627] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 245.147087] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
~ # [ 245.154046] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 245.162832] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0, rdt=7, rdh=0
[ 245.165307] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 245.165846] r4l_e1000_demo: pending_irqs: 3
[ 245.166781] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 245.727774] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=7, rdh=0
[ 245.729844] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 245.730549] r4l_e1000_demo: pending_irqs: 3
[ 245.731213] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 245.791519] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=7, rdh=0
[ 245.792146] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 245.793699] r4l_e1000_demo: pending_irqs: 3
[ 245.794146] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 246.751796] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=7, rdh=0
[ 246.753212] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 246.754065] r4l_e1000_demo: pending_irqs: 3
[ 246.754778] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 246.756654] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=7, rdh=0
[ 246.758540] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 246.759362] r4l_e1000_demo: pending_irqs: 3
[ 246.760293] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 247.455586] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=7, rdh=0
[ 247.456915] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 247.457686] r4l_e1000_demo: pending_irqs: 3
[ 247.458269] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 250.718596] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=6, tdh=6, rdt=7, rdh=0
[ 250.719668] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 250.721580] r4l_e1000_demo: pending_irqs: 3
[ 250.722631] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)

~ # [ 258.718983] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=7, tdh=7, rdt=7, rdh=0
[ 258.720065] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 258.720663] r4l_e1000_demo: pending_irqs: 3
[ 258.720995] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)

```

```

fan@Myco: ~/cicv-r4l-3-fan5l  X  fan@Myco: ~/cicv-r4l-3-fan5f  X  +  v
~ # ip addr add broadcast 10.0.2.255 dev eth0
[ 308.584967] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 308.587541] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
ip: RTNETLINK answers: Invalid argument
~ # |

```

```

fan@Myco: ~/cicv-r4l-3-fan5l  X  fan@Myco: ~/cicv-r4l-3-fan5f  X  +  v
~ # ip addr add 10.0.2.15/255.255.255.0 dev eth0
[ 347.129750] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 347.130824] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
~ # |

```

```

fan@Myco: ~/cicv-r4l-3-fan5l  X  fan@Myco: ~/cicv-r4l-3-fan5f  X  +  v
~ # ip route add default via 10.0.2.1
~ # |

```



```
fan@Mycro: ~/cicv-r4l-3-fan5l x fan@Mycro: ~/cicv-r4l-3-fan5f x + v
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
[ 426.765906] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=7, rdh=0
[ 426.768034] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 426.768625] r4l_e1000_demo: pending_irqs: 131
[ 426.769599] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 426.773869] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=0, rdh=1
[ 426.775770] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 426.776759] r4l_e1000_demo: pending_irqs: 131
[ 426.778002] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=0 ttl=255 time=23.559 ms
[ 427.783574] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=1, rdh=2
[ 427.785226] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 427.785556] r4l_e1000_demo: pending_irqs: 131
[ 427.785939] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=1 ttl=255 time=4.406 ms
[ 428.788459] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=6, tdh=6, rdt=2, rdh=3
[ 428.789217] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 428.790095] r4l_e1000_demo: pending_irqs: 131
[ 428.790597] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=2 ttl=255 time=3.460 ms
[ 429.792719] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=7, tdh=7, rdt=3, rdh=4
[ 429.794647] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 429.795801] r4l_e1000_demo: pending_irqs: 131
[ 429.796403] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=3 ttl=255 time=4.508 ms
[ 430.797734] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0, rdt=4, rdh=5
[ 430.798908] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 430.799627] r4l_e1000_demo: pending_irqs: 131
[ 430.800631] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=4 ttl=255 time=4.965 ms
^C
--- 10.0.2.2 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 3.460/8.179/23.559 ms
```

## 8、卸载 r4l\_e1000\_demo.ko，重新加载 r4l\_e1000\_demo.ko

```
fan@Mycro: ~/cicv-r4l-3-fan5l x fan@Mycro: ~/cicv-r4l-3-fan5f x + v
~ # rmmod r4l_e1000_demo.ko
[ 122.664677] r4l_e1000_demo: Rust for linux e1000 driver demo (exit)
[ 122.665718] r4l_e1000_demo: Rust for linux e1000 driver demo (remove)
[ 122.857874] r4l_e1000_demo: Rust for linux e1000 driver demo (device_remove)
[ 122.864706] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
ping: sendto: Network is unreachable
~ # insmod r4l_e1000_demo.ko
[ 184.061430] r4l_e1000_demo: Rust for linux e1000 driver demo (init)
[ 184.062363] r4l_e1000_demo: Rust for linux e1000 driver demo (probe): None
[ 184.269919] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
~ #
~ # ip link set eth0 up
[ 198.549417] r4l_e1000_demo: Rust for linux e1000 driver demo (net device open)
[ 198.553170] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 198.555864] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
~ # [ 198.563237] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 198.571711] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0, rdt=7, rdh=0
[ 198.574495] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 198.574814] r4l_e1000_demo: pending_irqs: 3
[ 198.576142] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 199.180274] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=7, rdh=0
[ 199.180835] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 199.181598] r4l_e1000_demo: pending_irqs: 3
[ 199.182184] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 199.184235] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=7, rdh=0
[ 199.184921] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 199.185996] r4l_e1000_demo: pending_irqs: 3
[ 199.186623] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 200.204820] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=7, rdh=0
[ 200.206124] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 200.206962] r4l_e1000_demo: pending_irqs: 3
[ 200.207734] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 200.209804] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=7, rdh=0
[ 200.211681] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 200.213390] r4l_e1000_demo: pending_irqs: 3
[ 200.214617] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 200.908715] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=7, rdh=0
[ 200.909963] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 200.910554] r4l_e1000_demo: pending_irqs: 3
```

## 9、设置网络并 ping 通

```

~ #
~ # ip addr add broadcast 10.0.2.255 dev eth0
[ 209.784164] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 209.787194] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
ip: RTNETLINK answers: Invalid argument
~ # [ 212.107893] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=7, tdh=7, rdt=7, rdh=0
[ 212.109093] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 212.109482] r4l_e1000_demo: pending_irqs: 3
[ 212.110688] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
~ #
~ # ip addr add 10.0.2.15/255.255.255.0 dev eth0
[ 220.402417] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 220.403711] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
~ #
~ # [ 227.468501] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0, rdt=7, rdh=0
[ 227.469610] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 227.470726] r4l_e1000_demo: pending_irqs: 3
[ 227.471252] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
ip route add default via 10.0.2.1
~ #
~ # ip route add default via 10.0.2.1
ip: RTNETLINK answers: File exists
~ #
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
[ 249.112578] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=7, rdh=0
[ 249.114687] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 249.115000] r4l_e1000_demo: pending_irqs: 131
[ 249.115576] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 249.119562] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=0, rdh=1
[ 249.120837] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 249.121619] r4l_e1000_demo: pending_irqs: 131
[ 249.122722] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=0 ttl=255 time=20.283 ms
[ 250.128650] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=1, rdh=2
[ 250.129707] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 250.131441] r4l_e1000_demo: pending_irqs: 131
[ 250.132209] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=1 ttl=255 time=5.556 ms

```

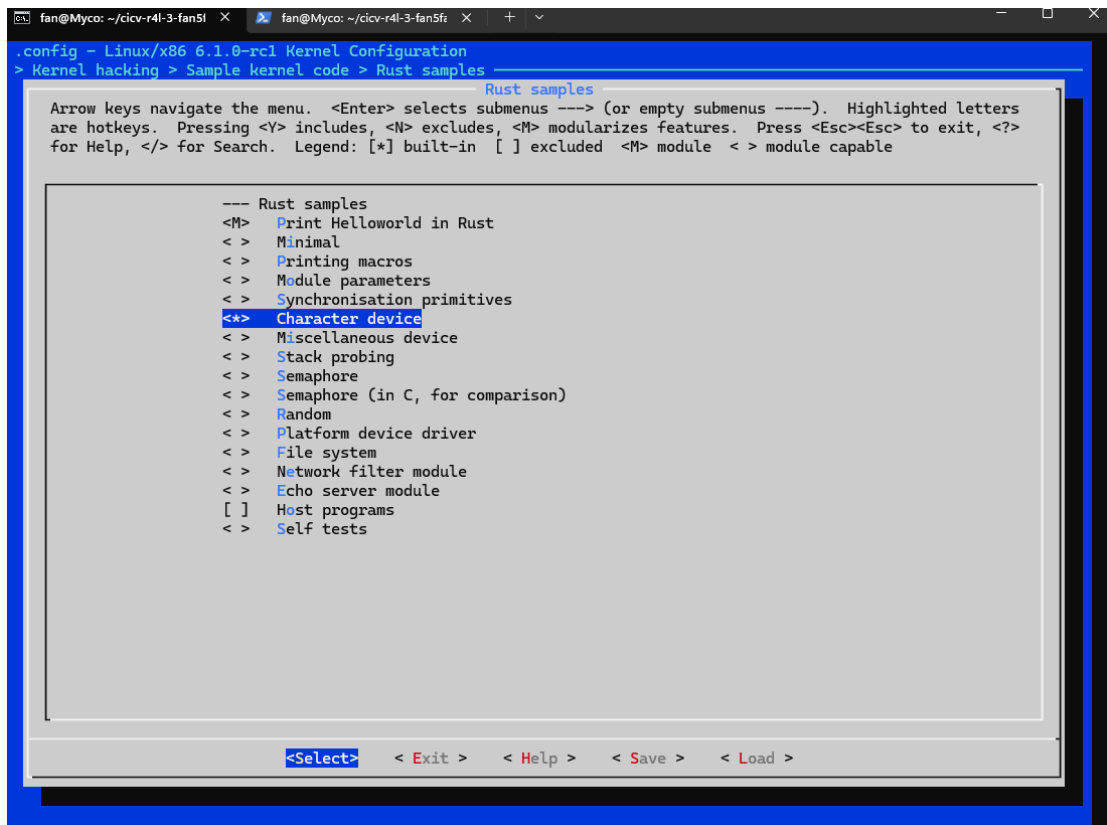
```

~ #
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
[ 249.112578] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=1, rdt=7, rdh=0
[ 249.114687] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 249.115000] r4l_e1000_demo: pending_irqs: 131
[ 249.115576] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 249.119562] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=2, tdh=2, rdt=0, rdh=1
[ 249.120837] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 249.121619] r4l_e1000_demo: pending_irqs: 131
[ 249.122722] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=0 ttl=255 time=20.283 ms
[ 250.128650] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=3, rdt=1, rdh=2
[ 250.129707] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 250.131441] r4l_e1000_demo: pending_irqs: 131
[ 250.132209] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=1 ttl=255 time=5.556 ms
[ 251.135298] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=4, rdt=2, rdh=3
[ 251.135949] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 251.137090] r4l_e1000_demo: pending_irqs: 131
[ 251.137972] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=2 ttl=255 time=4.652 ms
[ 252.140892] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=5, rdt=3, rdh=4
[ 252.142150] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 252.142661] r4l_e1000_demo: pending_irqs: 131
[ 252.143592] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=3 ttl=255 time=4.496 ms
^C
--- 10.0.2.2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 4.496/8.746/20.283 ms

```

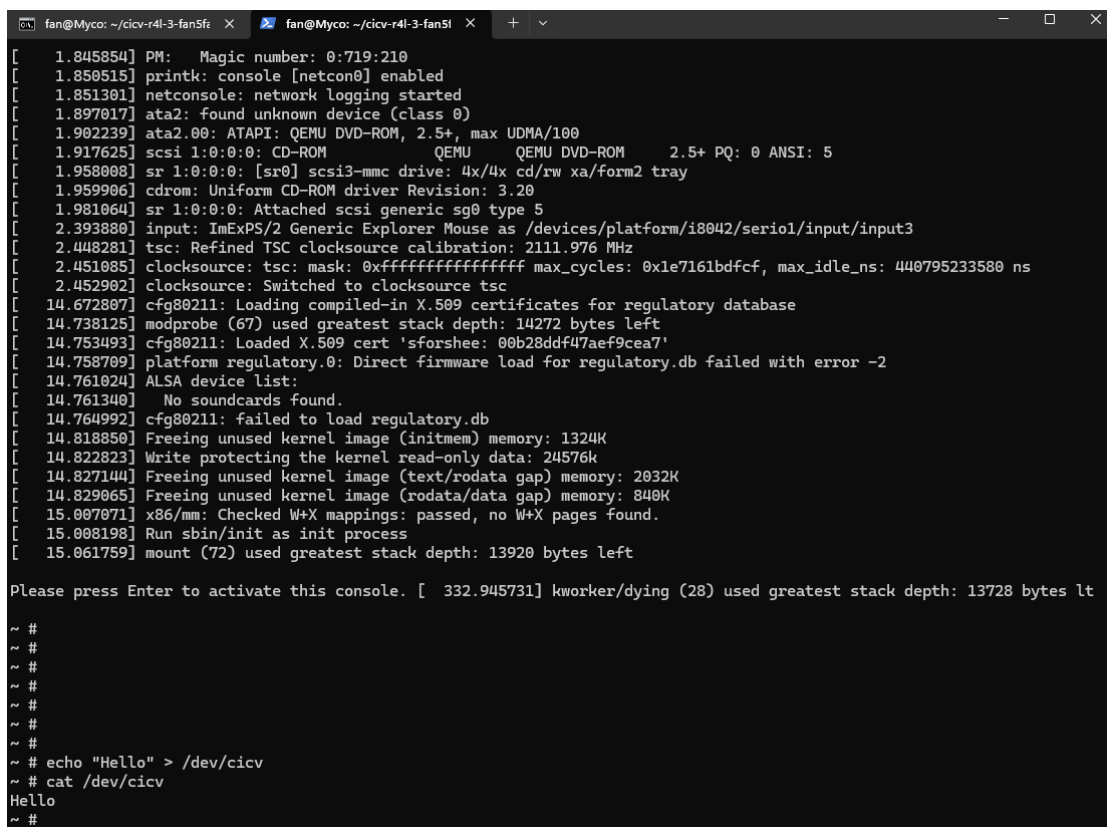
## 六、作业五

### 1、make LLVM=1 menuconfig



2、重新编译内核

3、执行操作



- 4、作业 5 中的字符设备/dev/cicv 是怎么创建的？它的设备号是多少？它是如何与我们写的字符设备驱动关联上的？

```
echo "mknod /dev/cicv c 248 0" >> etc/init.d/rcS
```

- 1、由系统启动时 init.d/rcS 文件的 mknod 指令创建
- 2、它的设备号是 248，次设备号是 0
- 3、kernel::file::Operations trait 是一个函数指针的集合. 这些操作大部分负责实现系统调用, 在驱动注册时, 实现了这个 trait 的结构与驱动关联.

```
let mut chrdev_reg = chrdev::Registration::new_pinned(name, 0, module)?;
```

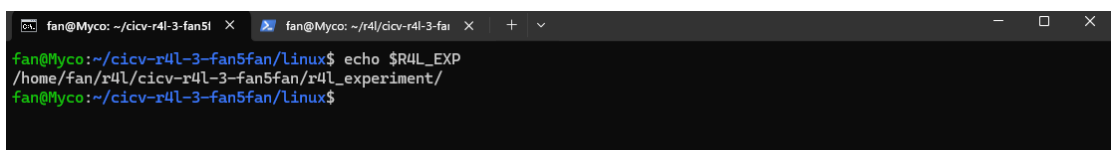
```
chrdev_reg.as_mut().register::(<RustFile>()?);
```

RustFile 这个结构实现了 kernel::file::Operations。

## 七、项目小试验

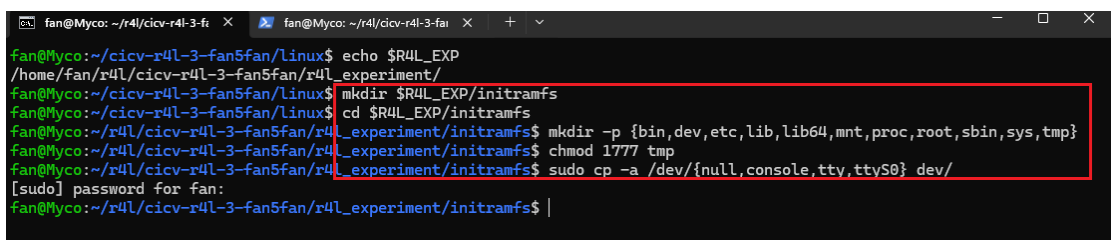
### 1、环境配置

#### 1.1、增加环境变量



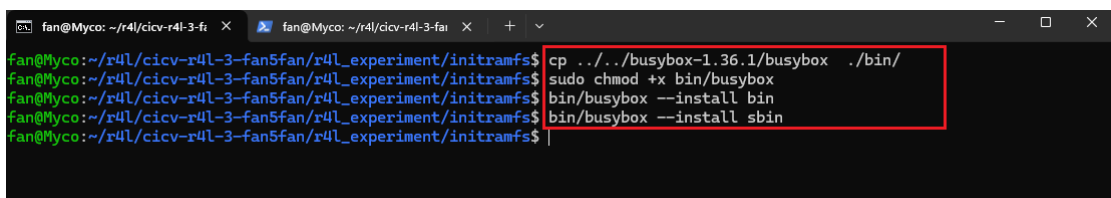
```
fan@Myco: ~/cicv-r4l-3-fan5fai x  fan@Myco: ~/r4l/cicv-r4l-3-fai x + v
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ echo $R4L_EXP
/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/
fan@Myco:~/cicv-r4l-3-fan5fan/linux$
```

#### 1.2、initramfs 镜像设置



```
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ echo $R4L_EXP
/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ mkdir $R4L_EXP/initramfs
fan@Myco:~/cicv-r4l-3-fan5fan/linux$ cd $R4L_EXP/initramfs
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ mkdir -p {bin,dev,etc,lib,lib64,mnt,proc,root,sbin,sys,tmp}
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ chmod 1777 tmp
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ sudo cp -a /dev/{null,console,ttv,ttvS0} dev/
[sudo] password for fan:
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ |
```

#### 1.3、将之前静态编译的 busybox 拷贝到 initramf/bin 下



```
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cp ../../busybox-1.36.1/busybox ./bin/
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ sudo chmod +x bin/busybox
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ bin/busybox --install bin
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ bin/busybox --install sbin
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ |
```

#### 1.4、编写 init 脚本

```
fan@Myco: ~/r4l/cicv-r4l-3-fai X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat init
#!/bin/busybox sh

# Mount the /proc and /sys filesystems.
mount -t proc none /proc
mount -t sysfs none /sys

# Boot real things.

# NIC up
ip link set eth0 up
ip addr add 10.0.2.15/24 dev eth0
ip link set lo up

# Wait for NIC ready
sleep 0.5

# Make the new shell as a login shell with -l option
# Only login shell read /etc/profile
setsid sh -c 'exec sh -l </dev/ttyS0 >/dev/ttyS0 2>&1'

fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$
```

## 1.5、更多设置

```
fan@Myco:~/r4l/cicv-r4l-3-fai X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat etc/hosts
127.0.0.1 localhost
10.0.2.2 host_machine
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat etc/profile
alias ll='ls -l'
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat etc/passwd
root:x:0:0:root:/root:/bin/bash
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat etc/group
root:x:0:
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$
```

## 1.6、构建 initramfs 镜像

```
fan@Myco:~/r4l/cicv-r4l-3-fai X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ find . -print0 | cpio --null -ov --format=newc | gzip -9 > .
./initramfs.cpio.gz
./sys
./lib64
./dev
./dev/null
./dev/ttyS0
./dev/console
./dev/tty
./init
./lib
./tmp
./sbin
./sbin/lpq
./sbin/ifplugd
./sbin/delgroup
./sbin/tc
./sbin/unix2dos
./sbin/findfs
./sbin/w
./sbin/eject
./sbin/ssl_client
./sbin/stty
./sbin/route
./sbin/swapoff
./sbin/uptime
./sbin/mkdosfs
./sbin/fdformat
./sbin/nohup
./sbin/mknod
./sbin/chroot
./sbin/readprofile
./sbin/diff
./sbin/lusb
./sbin/mim
./sbin/zcip
./sbin/envuidgid
./sbin/free
./sbin/mt
./sbin/adduser
```

## 1.7、通过 boot.sh 脚本启动

## 1.8、在主机上设置 NFS 服务器

```
fan@Myco: ~/r4l/cicv-r4l-3-fi X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment$ sudo apt install nfs-kernel-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  keyutils libnfsidmap1 libwrap0 nfs-common rpcbind
Suggested packages:
  open-iscsi watchdog
The following NEW packages will be installed:
  keyutils libnfsidmap1 libwrap0 nfs-common nfs-kernel-server rpcbind
0 upgraded, 6 newly installed, 0 to remove and 1 not upgraded.
Need to get 569 kB of archives.

fan@Myco:~/r4l/cicv-r4l-3-fi X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment$ cat /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
# to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver 127.0.0.1(insecure,rw,sync,no_root_squash)
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment$ sudo /etc/init.d/rpcbind restart
Restarting rpcbind (via systemctl): rpcbind.service.
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment$ sudo /etc/init.d/nfs-kernel-server restart
Restarting nfs-kernel-server (via systemctl): nfs-kernel-server.service.
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment$ |
```

## 1.9、实现挂载, rebuild initramfs

```
fan@Myco: ~/r4l/cicv-r4l-3-fi X fan@Myco: ~/r4l/cicv-r4l-3-fai X + v
[ 2.674261] NET: Registered PF_PACKET protocol family
[ 2.676470] 9pnet: Installing 9P2000 support
[ 2.677357] Key type dns_resolver registered
[ 2.679662] IPI shorthand broadcast: enabled
[ 2.680751] sched_clock: Marking stable (2587854873, 92217779)->(2696136287, -16063635)
[ 2.684110] registered taskstats version 1
[ 2.684656] Loading compiled-in X.509 certificates
[ 2.697065] cryptomgr_test (44) used greatest stack depth: 15584 bytes left
[ 2.706238] PM: Magic number: 0:332:381
[ 2.708102] printk: console [netcon0] enabled
[ 2.708674] netconsole: network logging started
[ 2.713188] cfg80211: Loading compiled-in X.509 certificates for regulatory database
[ 2.791496] modprobe (66) used greatest stack depth: 14272 bytes left
[ 2.811018] cfg80211: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[ 2.815266] platform regulatory.0: Direct firmware load for regulatory.db failed with error -2
[ 2.818084] ALSA device list:
[ 2.819095] cfg80211: failed to load regulatory.db
[ 2.821137] No soundcards found.
[ 2.891753] Freeing unused kernel image (initmem) memory: 1324K
[ 2.895302] Write protecting the kernel read-only data: 24576k
[ 2.901248] Freeing unused kernel image (text/rodata gap) memory: 2032K
[ 2.903228] Freeing unused kernel image (rodata/data gap) memory: 820K
[ 3.122840] x86/mm: Checked W&X mappings: passed, no W&X pages found.
[ 3.127801] Run /init as init process
[ 3.160045] mount (70) used greatest stack depth: 14160 bytes left
[ 3.176352] mount (71) used greatest stack depth: 13920 bytes left
[ 3.198304] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[ 3.202132] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[ 3.245469] ip (73) used greatest stack depth: 12912 bytes left
[ 3.336184] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input3
~ #
~ #
~ #
~ # mount -t nfs -o nolock host_machine:/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_exp
eriment/driver /mnt
[ 54.903422] mount (76) used greatest stack depth: 12896 bytes left
~ #
~ #
~ # ls /mnt
001_hello_world 002_completion
~ #
```

## 1.10、设置 pts device node

```
fan@Myco: ~/r4l/cicv-r4l-3-fai x fan@Myco: ~/r4l/cicv-r4l-3-fai x + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat init
#!/bin/busybox sh

# Mount the /proc and /sys filesystems.
mount -t proc none /proc
mount -t sysfs none /sys
# Boot real things.

# NIC up
ip link set eth0 up
ip addr add 10.0.2.15/24 dev eth0
ip link set lo up

# Wait for NIC ready
sleep 2

# Make the new shell as a login shell with -l option
# Only login shell read /etc/profile
setsid sh -c 'exec sh -l </dev/ttyS0 >/dev/ttyS0 2>&1'

// 工作目录
mount -t nfs -o noexec host_machine:/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver /mnt

sleep 0.5

// pts device node
mount -t devpts devpts /dev/pts

// telnet service
telnetd -l /bin/sh
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$
```

### 1.11、在 boot.sh 中加入一下 qemu 启动参数

```
fan@Myco: ~/r4l/cicv-r4l-3-fai x fan@Myco: ~/r4l/cicv-r4l-3-fai x + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat ../boot.sh
#!/bin/sh
kernel_image="./linux/arch/x86/boot/bzImage"

qemu-system-x86_64 \
-kernel $kernel_image \
-append "console=ttyS0" \
-initrd ./initramfs.cpio.gz \
-nographic \
-device e1000,mac=52:54:00:12:34:50,netdev=host_net \
-netdev user,id=host_net,hostfwd=tcp::7023->:23 \
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ |
```

### 1.12、开启 telnet server

```
fan@Myco: ~/r4l/cicv-r4l-3-fai x fan@Myco: ~/r4l/cicv-r4l-3-fai x + v
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$ cat init
#!/bin/busybox sh

# Mount the /proc and /sys filesystems.
mount -t proc none /proc
mount -t sysfs none /sys
# Boot real things.

# NIC up
ip link set eth0 up
ip addr add 10.0.2.15/24 dev eth0
ip link set lo up

# Wait for NIC ready
sleep 2

# Make the new shell as a login shell with -l option
# Only login shell read /etc/profile
setsid sh -c 'exec sh -l </dev/ttyS0 >/dev/ttyS0 2>&1'

// 工作目录
mount -t nfs -o noexec host_machine:/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver /mnt

sleep 0.5

// pts device node
mount -t devpts devpts /dev/pts

// telnet service
telnetd -l /bin/sh
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/initramfs$
```

### 1.13、在本机通过 telnet server 连接 qemu 控制台

```
fan@Myco: ~/r4l/cicv-r4l-3-fan5fan X fan@Myco: ~/r4l/cicv-r4l-3-fan5fan X Telnet localhost X + v - □ X

/mnt/002_completion # ls
Makefile      Readme      completion.h  completion.mod  completion.mod.o  load_module.sh
Module.symvers completion.c  completion.ko  completion.mod.c completion.o        modules.order
/mnt/002_completion #
```

## 1.14、测试

```
fan@Myco:~/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion$ make
make LLVM=1 -C ../../../../linux M=/home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion modules
make[1]: Entering directory '/home/fan/r4l/cicv-r4l-3-fan5fan/linux'
CC [M] /home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion/completion.o
MODPOST /home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion/Module.symvers
CC [M] /home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion/completion.mod.o
LD [M] /home/fan/r4l/cicv-r4l-3-fan5fan/r4l_experiment/driver/002_completion/completion.ko
make[1]: Leaving directory '/home/fan/r4l/cicv-r4l-3-fan5fan/linux'

/mnt/002_completion # ./load_module.sh
[ 5705.340068] completion: loading out-of-tree module taints kernel.
[ 5705.349745] completion is loaded
[ 5705.399384] load_module.sh (89) used greatest stack depth: 12912 bytes left
/mnt/002_completion #

/mnt/002_completion # cat /dev/completion
[ 6150.874998] completion_open() is invoked
[ 6150.875361] completion_read() is invoked
[ 6150.875566] process 104(cat) is going to sleep

/mnt/002_completion # cat /dev/completion
[ 6150.874998] completion_open() is invoked
[ 6150.875361] completion_read() is invoked
[ 6150.875566] process 104(cat) is going to sleep
[ 6204.617991] completion_open() is invoked
[ 6204.619400] completion_write() is invoked
[ 6204.619667] process 99(sh) awakening the readers...
[ 6204.620923] awoken 104(cat)
/mnt/002_completion # |

Telnet localhost X + v - □ X

/mnt/002_completion # echo "something" > /dev/completion
/mnt/002_completion # |

Telnet localhost X + v - □ X

/mnt/002_completion # echo "something" > /dev/completion
/mnt/002_completion # |
```

## 2、实战要求

本次小试验主要完成 driver 目录下的 002\_completion 的 rust 代码重构，提交了相应 Linux C 代码，具体要求请查看 Linux C 代码的 Readme。

源码路径：

[https://github.com/cicvedu/cicv-r4l-3-fan5fan/tree/master/linux/samples/rust/rust\\_completion/](https://github.com/cicvedu/cicv-r4l-3-fan5fan/tree/master/linux/samples/rust/rust_completion/)

实验结果截图：



```
fan@Myco: ~/r4/cicv-r4l-3-fi x fan@Myco: ~/r4/cicv-r4l-3-fai x + v
/mnt # ls
001_hello_world 002_completion completion.ko load_module.sh
/mnt # ./load_module.sh load
[34374.149807] completion: completion is loaded (init)
/mnt #
/mnt #
/mnt # cat /dev/completion
[34385.086416] completion: completion_open() is invoked
[34385.087549] completion: completion_read() is invoked
[34385.088335] completion: process 168(cat) is going to sleep
[34399.213255] completion: completion_open() is invoked
[34399.214850] completion: completion_write() is invoked
[34399.218891] completion: process 80(sh) awakening the readers...
[34399.221196] completion: awoken 168(cat)
/mnt #
```

```
Telnet localhost x + v
~ # echo "something" > /dev/completion
~ #
```