

Task2

Step1 - 编译模块

Bash

```
1 make LLVM=1
```

```
cicv-r4l-Loliy/src_e1000 on master via v1.73.0
> make LLVM=1
make -C ../linux M=$PWD
make[1]: Entering directory '/home/loliy/Workspace/rust/cicv-r4l-Loliy/linux'
RUSTC [M] /home/loliy/Workspace/rust/cicv-r4l-Loliy/src_e1000/r4l_e1000_demo.o
MODPOST /home/loliy/Workspace/rust/cicv-r4l-Loliy/src_e1000/Module.symvers
CC [M] /home/loliy/Workspace/rust/cicv-r4l-Loliy/src_e1000/r4l_e1000_demo.mod.o
LD [M] /home/loliy/Workspace/rust/cicv-r4l-Loliy/src_e1000/r4l_e1000_demo.ko
make[1]: Leaving directory '/home/loliy/Workspace/rust/cicv-r4l-Loliy/linux'
```

Step2 - 启动 Linux 系统

Bash

```
1 ./build_image.sh
```

```
~ # 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)
```

Step3 - 禁用 E1000 网卡驱动

Bash

```
1 make LLVM=1 menuconfig
```

```
.config - Linux/x86 6.1.0-rc1 Kernel Configuration
> Device Drivers > Network device support > Ethernet driver support
    Ethernet driver support
    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

    ^(-)
    < >   Fujitsu FMV-J18x PCMCIA support
    [*]   Fungible devices
    < >   Fungible Ethernet device driver
    [*]   Google Devices
    < >   Google Virtual NIC (gVNIC) support
    [*]   Huawei devices
    < >   Huawei Intelligent PCIE Network Interface Card
    [*]   Intel (82586/82593/82596) devices
    [*]   Intel devices
    <*>   Intel(R) PRO/100+ support
    < >   Intel(R) PRO/1000 Gigabit Ethernet support
    <*>   Intel(R) PRO/1000 PCI-Express Gigabit Ethernet support
    [*]   Support HW cross-timestamp on PCH devices
    < >   Intel(R) 82575/82576 PCI-Express Gigabit Ethernet support
    < >   Intel(R) 82576 Virtual Function Ethernet support
    < >   Intel(R) PRO/10GbE support
    v(+)

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

```
cicv-r41-Loliyo/linux on master [!]
```

```
> make LLVM=1 menuconfig
```

```
configuration written to .config
```


```
*** End of the configuration.
```

```
*** Execute 'make' to start the build or try 'make help'.
```

Step4 - 重新编译 Linux 内核

Bash

```
1 make LLVM=1 -j$(nproc)
```

```
node
node.o
cicv-r4l-Loliy/linux on  master [!] took 2m38s
> make LLVM=1 -j$(nproc)
  SYNC      include/config/auto.conf.cmd
  DESCEND   objtool
  CALL      scripts/checksyscalls.sh
```

Step5 - 加载网络驱动模块

Bash

```
1 insmod r4l_e1000_demo.ko
```

```
~ # insmod r4l_e1000_demo.ko
[ 17.418087] r4l_e1000_demo: loading out-of-tree module taints kernel.
[ 17.424095] r4l_e1000_demo: Rust for linux e1000 driver demo (init)
[ 17.424558] r4l_e1000_demo: Rust for linux e1000 driver demo (probe): None
[ 17.603700] ACPI: \_SB_.LNKC: Enabled at IRQ 11
[ 17.625692] r4l_e1000_demo: Rust for linux e1000 driver demo (net device get_stats64)
[ 17.627902] insmod (80) used greatest stack depth: 11192 bytes left
~ # |
```

Step6 - 网络配置

Bash

```
1 ip link set eth0 up
2 ip addr add broadcast 10.0.2.255 dev eth0
3 ip addr add 10.0.2.15/255.255.255.0 dev eth0
4 ip route add default via 10.0.2.1
5 ping 10.0.2.2
```

```
[ 93.536389] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=3, tdh=0
[ 93.536940] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 93.537272] r4l_e1000_demo: pending_irqs: 3
[ 93.537571] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 93.538397] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=4, tdh=0
[ 93.538741] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 93.539244] r4l_e1000_demo: pending_irqs: 3
[ 93.539931] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 94.559767] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=5, tdh=0
[ 94.560524] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 94.560992] r4l_e1000_demo: pending_irqs: 3
[ 94.561342] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 97.439156] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=6, tdh=0
[ 97.439848] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 97.440234] r4l_e1000_demo: pending_irqs: 3
[ 97.440693] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 105.567051] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=7, tdh=0
[ 105.567541] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 105.569116] r4l_e1000_demo: pending_irqs: 3
[ 105.569332] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 121.438965] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=0
[ 121.439713] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 121.441625] r4l_e1000_demo: pending_irqs: 3
[ 121.441904] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
[ 156.767161] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=0
[ 156.767517] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 156.768009] r4l_e1000_demo: pending_irqs: 3
[ 156.768175] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
```

~ #

```

~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
[ 551.360319] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=7, tdh=3
[ 551.360952] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 551.361168] r4l_e1000_demo: pending_irqs: 131
[ 551.361353] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=0 ttl=255 time=2.206 ms
[ 552.362816] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=0, tdh=4
[ 552.363127] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 552.363885] r4l_e1000_demo: pending_irqs: 131
[ 552.364156] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=1 ttl=255 time=2.024 ms
[ 553.365390] r4l_e1000_demo: Rust for linux e1000 driver demo (net device start_xmit) tdt=1, tdh=5
[ 553.365849] r4l_e1000_demo: Rust for linux e1000 driver demo (handle_irq)
[ 553.366413] r4l_e1000_demo: pending_irqs: 131
[ 553.366551] r4l_e1000_demo: Rust for linux e1000 driver demo (napi poll)
64 bytes from 10.0.2.2: seq=2 ttl=255 time=2.237 ms
^C
--- 10.0.2.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 2.024/2.155/2.237 ms

```

Q & A

Q1: 编译成内核模块，是在哪个文件中以哪条语句定义的？

在 `src_e1000/Kbuild` 文件中，通过 `obj-m := r4l_e1000_demo.o` 语句定义的。

Kbuild 文件是用于构建内核模块的专用构建文件。它定义了编译模块的规则和指令，以及模块的依赖关系和安装路径等信息。每个内核模块都应该有自己的 Kbuild 文件，用于描述模块的构建和安装过程。

Q2: 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？

1. 在 `src_e1000/Makefile` 文件中，指定了内核源代码树的路径： `$(MAKE) -C $(KDIR)` `M=$$PWD`，其中 `$(KDIR)` 指向了上层目录下的 `linux` 目录
2. `make` 命令会根据 `Kbuild` 文件中的规则编译模块，并生成目标文件和模块文件
3. 进入 Linux 系统后，加载模块 `insmod r4l_e1000_demo.ko`