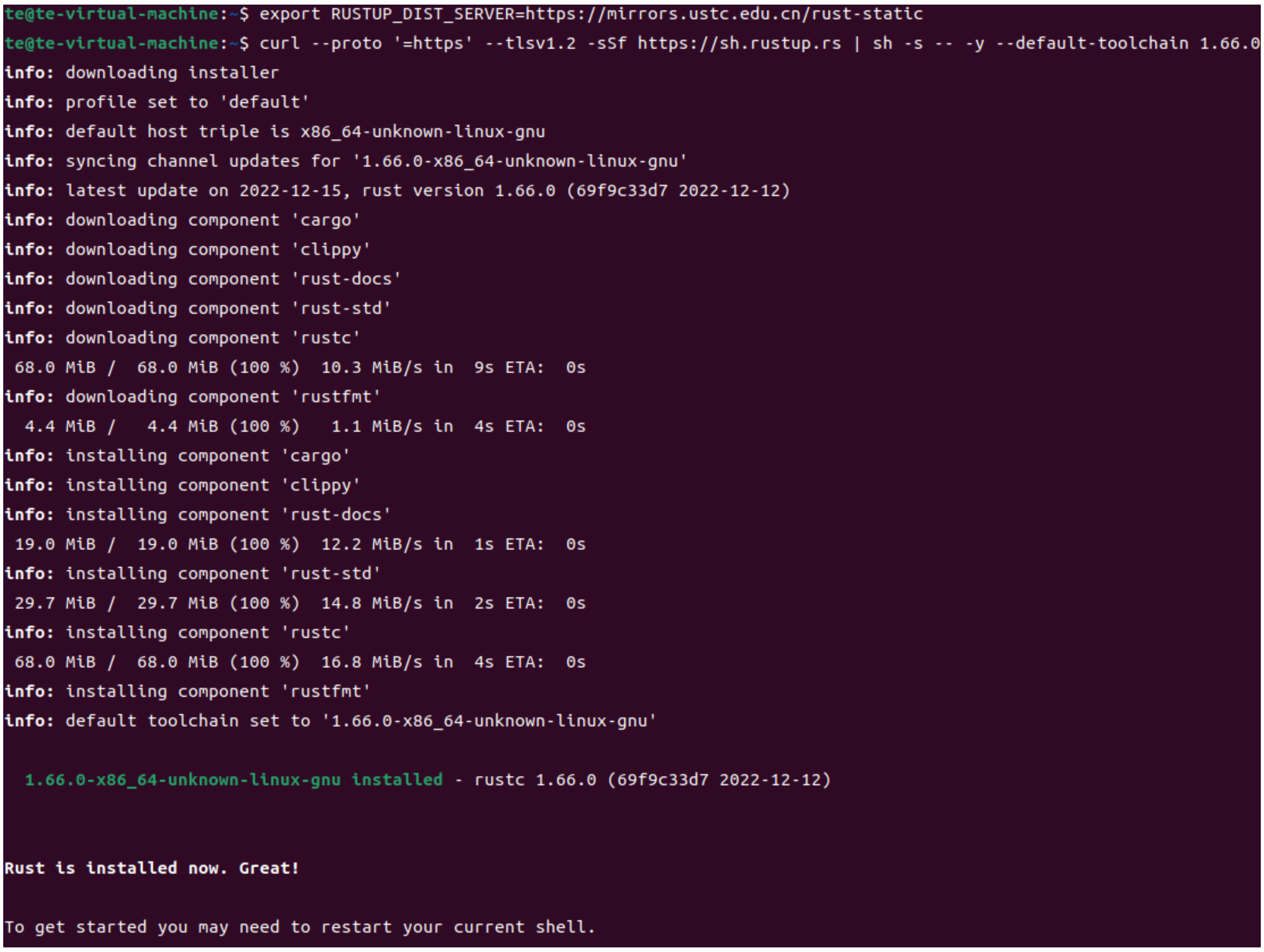
# 第二期作业

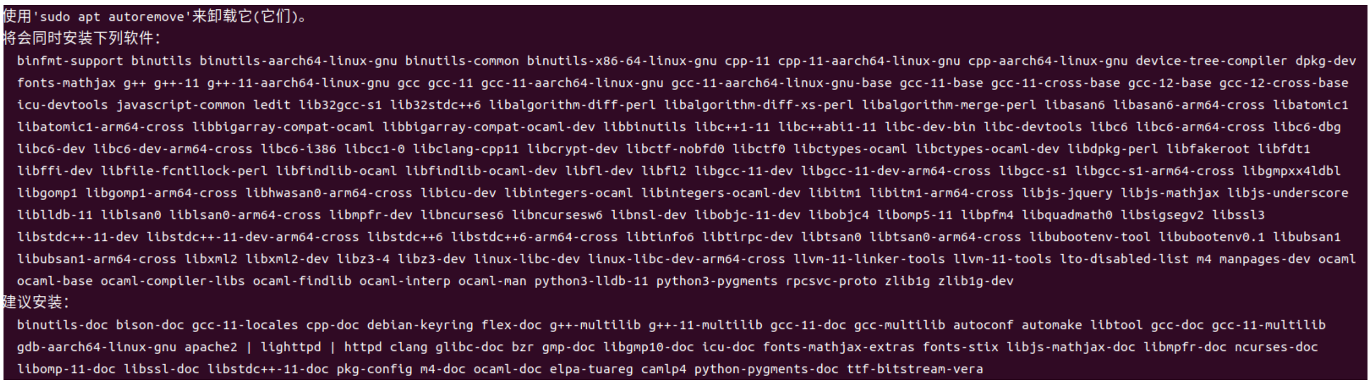
## 一、环境的搭建

### 1、安装Rust：



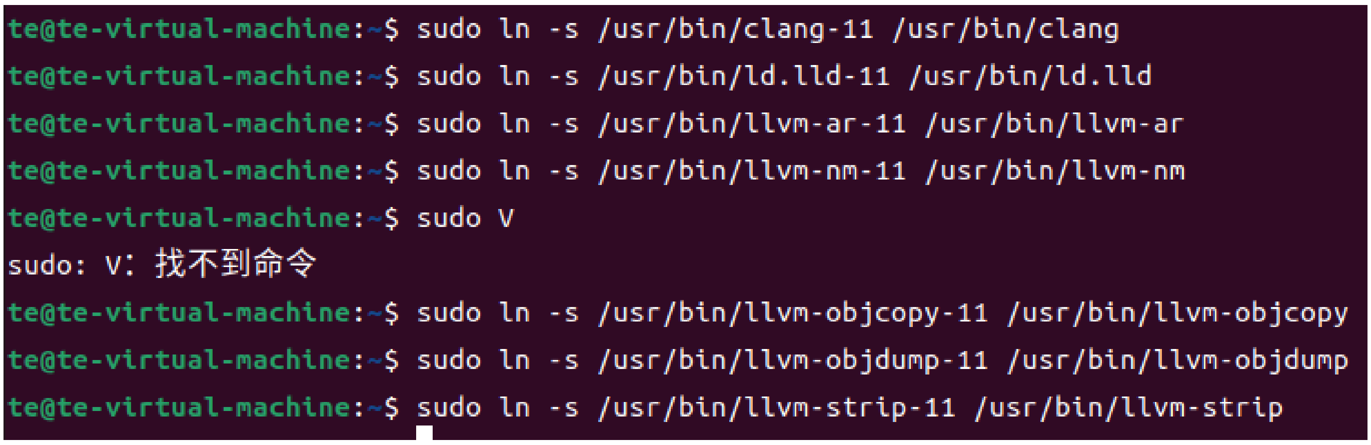
```
te@te-virtual-machine:~$ export RUSTUP_DIST_SERVER=https://mirrors.ustc.edu.cn/rust-static
te@te-virtual-machine:~$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh -s -- -y --default-toolchain 1.66.0
info: downloading installer
info: profile set to 'default'
info: default host triple is x86_64-unknown-linux-gnu
info: syncing channel updates for '1.66.0-x86_64-unknown-linux-gnu'
info: latest update on 2022-12-15, rust version 1.66.0 (69f9c33d7 2022-12-12)
info: downloading component 'cargo'
info: downloading component 'clippy'
info: downloading component 'rust-docs'
info: downloading component 'rust-std'
info: downloading component 'rustc'
 68.0 MiB /  68.0 MiB (100 %)  10.3 MiB/s in  9s ETA:  0s
info: downloading component 'rustfmt'
  4.4 MiB /   4.4 MiB (100 %)   1.1 MiB/s in  4s ETA:  0s
info: installing component 'cargo'
info: installing component 'clippy'
info: installing component 'rust-docs'
 19.0 MiB /  19.0 MiB (100 %)  12.2 MiB/s in  1s ETA:  0s
info: installing component 'rust-std'
 29.7 MiB /  29.7 MiB (100 %)  14.8 MiB/s in  2s ETA:  0s
info: installing component 'rustc'
 68.0 MiB /  68.0 MiB (100 %)  16.8 MiB/s in  4s ETA:  0s
info: installing component 'rustfmt'
info: default toolchain set to '1.66.0-x86_64-unknown-linux-gnu'

  1.66.0-x86_64-unknown-linux-gnu installed - rustc 1.66.0 (69f9c33d7 2022-12-12)


Rust is installed now. Great!

To get started you may need to restart your current shell.
```

### 2、安装依赖



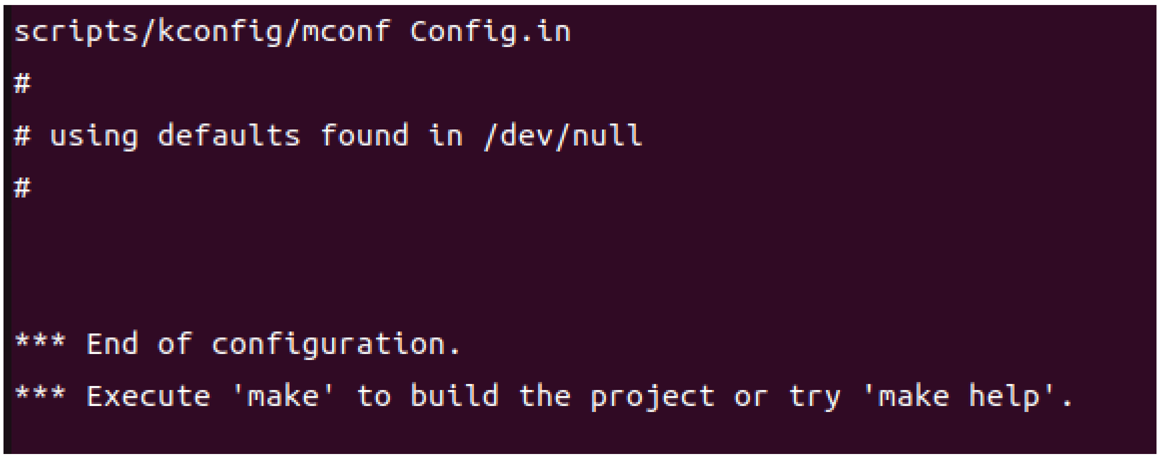
```
使用'sudo apt autoremove'来卸载它(它们)。
将会同时安装下列软件：
  binfmt-support binutils binutils-aarch64-linux-gnu binutils-common binutils-x86-64-linux-gnu cpp-11 cpp-11-aarch64-linux-gnu cpp-aarch64-linux-gnu device-tree-compiler dpkg-dev
  fonts-mathjax g++ g++-11 g++-11-aarch64-linux-gnu gcc gcc-11 gcc-11-aarch64-linux-gnu gcc-11-aarch64-linux-gnu-base gcc-11-base gcc-11-cross-base gcc-12-base gcc-12-cross-base
  icu-devtools javascript-common ledit lib32gcc-s1 lib32stdc++6 libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libasan6-arm64-cross libatomic1
  libatomic1-arm64-cross libbigarray-compat-ocaml libbigarray-compat-ocaml-dev libbinutils libc++1-11 libc++abi1-11 libc-dev-bin libc-devtools libc6 libc6-arm64-cross libc6-dbg
  libc6-dev libc6-dev-arm64-cross libc6-i386 libcc1-0 libclang-cpp11 libcrypt-dev libctf-nobfd0 libctf0 libctypes-ocaml libctypes-ocaml-dev libdpkg-perl libfakeroot libfdt1
  libffi-dev libfile-fcntllock-perl libfindlib-ocaml libfindlib-ocaml-dev libfl-dev libfl2 libgcc-11-dev libgcc-11-dev-arm64-cross libgcc-s1 libgcc-s1-arm64-cross libgmpxx4ldbl
  libgomp1 libgomp1-arm64-cross libhwasan0-arm64-cross libicu-dev libintegers-ocaml libintegers-ocaml-dev libitm1 libitm1-arm64-cross libjs-jquery libjs-mathjax libjs-underscore
  liblldb-11 liblsan0 liblsan0-arm64-cross libmpfr-dev libncurses6 libncursesw6 libnsl-dev libobjc-11-dev libobjc4 libomp5-11 libpfm4 libquadmath0 libsigsegv2 libssl3
  libstdc++-11-dev libstdc++-11-dev-arm64-cross libstdc++6 libstdc++6-arm64-cross libtinfo6 libtirpc-dev libtsan0 libtsan0-arm64-cross libubootenv-tool libubootenv0.1 libubsan1
  libubsan1-arm64-cross libxml2 libxml2-dev libz3-4 libz3-dev linux-libc-dev linux-libc-dev-arm64-cross llvm-11-linker-tools llvm-11-tools lto-disabled-list m4 manpages-dev ocaml
  ocaml-base ocaml-compiler-libs ocaml-findlib ocaml-interp ocaml-man python3-lldb-11 python3-pygments rpcsvc-proto zlib1g zlib1g-dev
建议安装：
  binutils-doc bison-doc gcc-11-locales cpp-doc debian-keyring flex-doc g++-multilib g++-11-multilib gcc-11-doc gcc-multilib autoconf automake libtool gcc-doc gcc-11-multilib
  gdb-aarch64-linux-gnu apache2 | lighttpd | httpd clang glibc-doc bzr gmp-doc libgmp10-doc icu-doc fonts-mathjax-extras fonts-stix libjs-mathjax-doc libmpfr-doc ncurses-doc
  libomp-11-doc libssl-doc libstdc++-11-doc pkg-config m4-doc ocaml-doc elpa-tuareg camlp4 python-pygments-doc ttf-bitstream-vera
```

## 3、配置环境，修改软链接



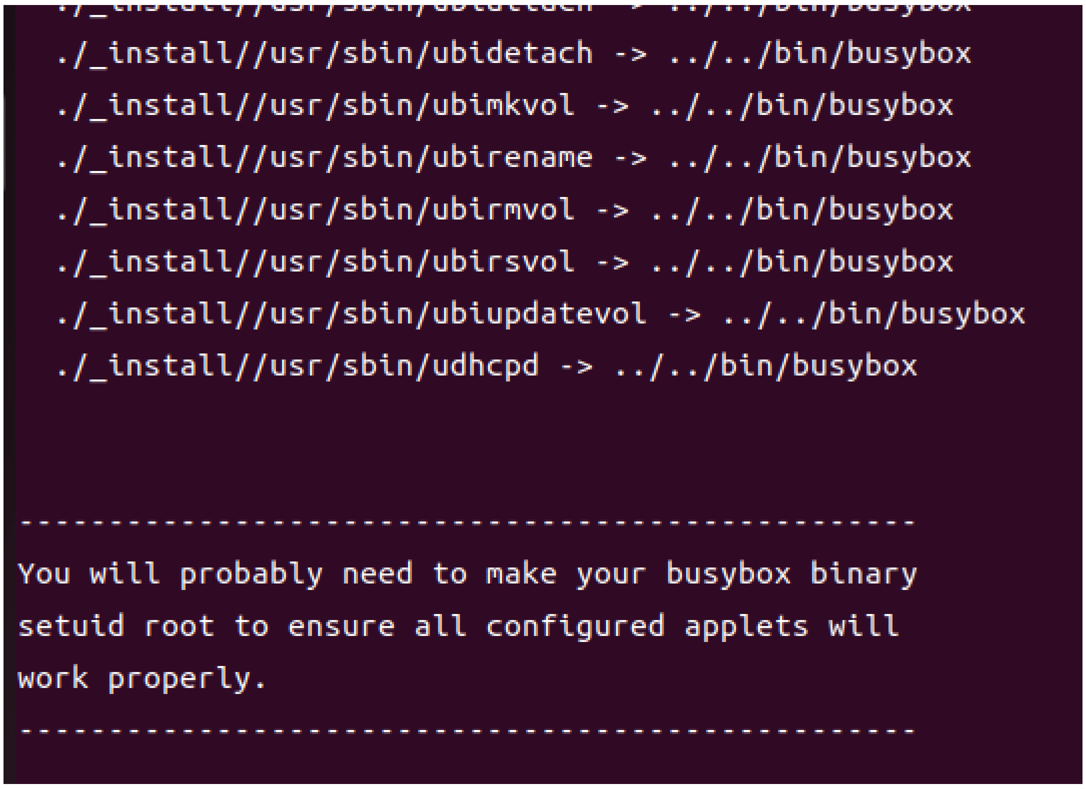
```
te@te-virtual-machine:~$ sudo ln -s /usr/bin/clang-11 /usr/bin/clang
te@te-virtual-machine:~$ sudo ln -s /usr/bin/ld.lld-11 /usr/bin/ld.lld
te@te-virtual-machine:~$ sudo ln -s /usr/bin/llvm-ar-11 /usr/bin/llvm-ar
te@te-virtual-machine:~$ sudo ln -s /usr/bin/llvm-nm-11 /usr/bin/llvm-nm
te@te-virtual-machine:~$ sudo V
sudo: V: 找不到命令
te@te-virtual-machine:~$ sudo ln -s /usr/bin/llvm-objcopy-11 /usr/bin/llvm-objcopy
te@te-virtual-machine:~$ sudo ln -s /usr/bin/llvm-objdump-11 /usr/bin/llvm-objdump
te@te-virtual-machine:~$ sudo ln -s /usr/bin/llvm-strip-11 /usr/bin/llvm-strip
```

## 4、编译BusyBox

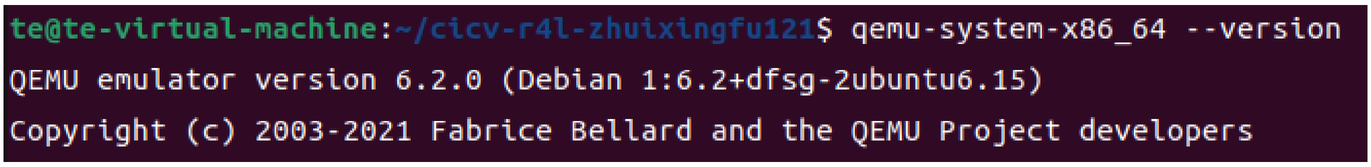


```
scripts/kconfig/mconf Config.in
#
# using defaults found in /dev/null
#


*** End of configuration.
*** Execute 'make' to build the project or try 'make help'.
```

## 5、安装Qemu

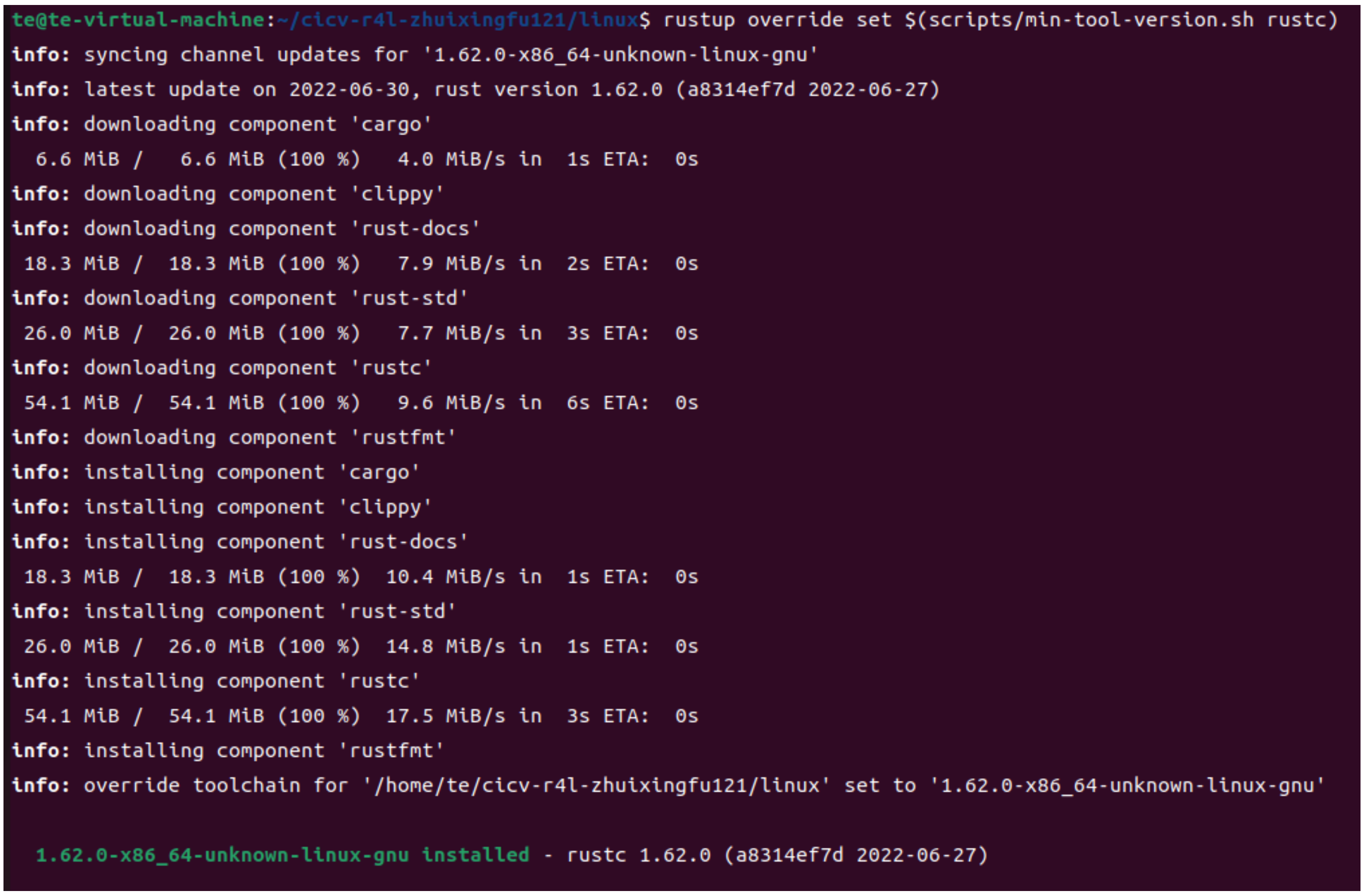


```
  ./_install//usr/sbin/ubidetach -> ../../bin/busybox
  ./_install//usr/sbin/ubimkvol -> ../../bin/busybox
  ./_install//usr/sbin/ubirename -> ../../bin/busybox
  ./_install//usr/sbin/ubirmvol -> ../../bin/busybox
  ./_install//usr/sbin/ubirsvol -> ../../bin/busybox
  ./_install//usr/sbin/ubiupdatevol -> ../../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.
--------------------------------------------------
```

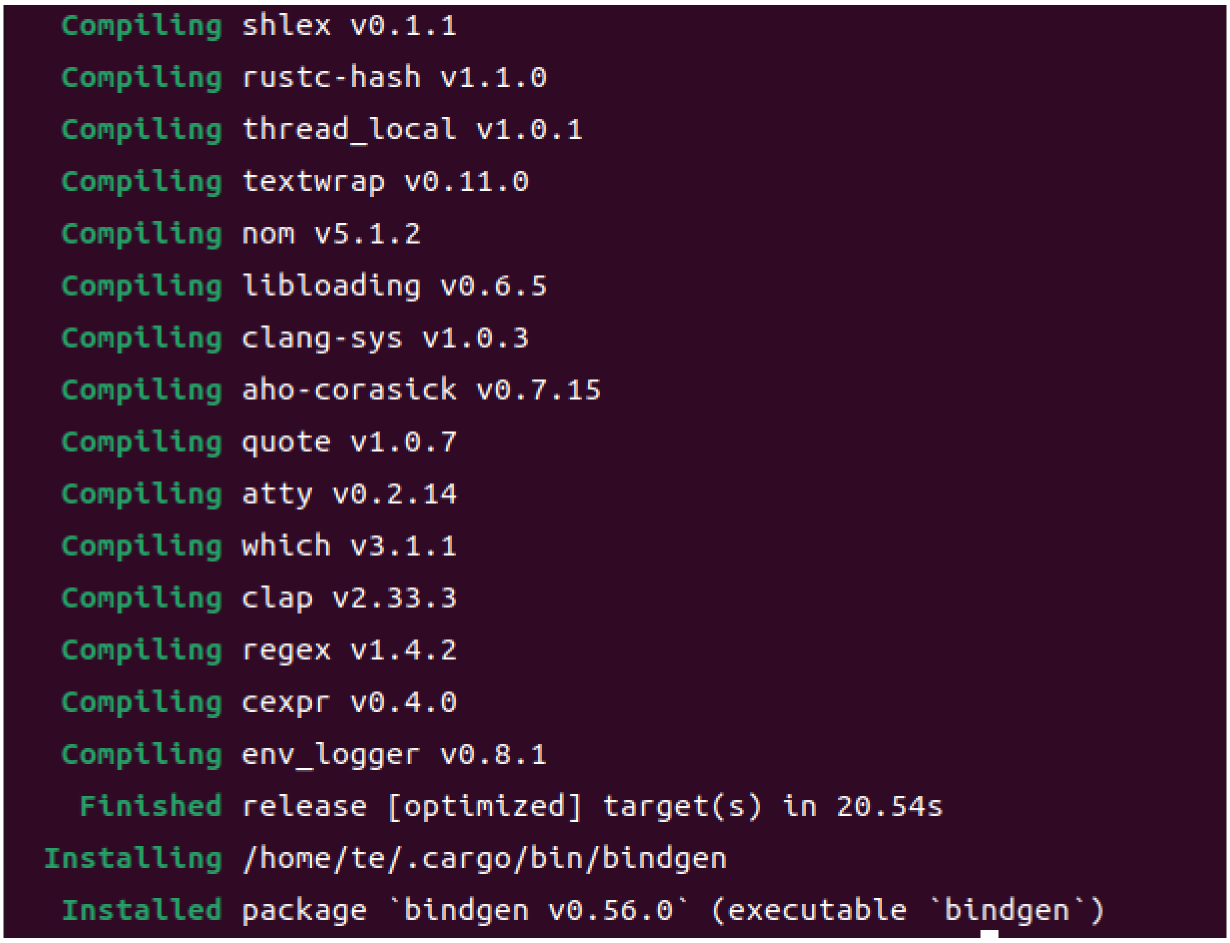


```
te@te-virtual-machine:~/cicv-r4l-zhuixingfu121$ qemu-system-x86_64 --version
QEMU emulator version 6.2.0 (Debian 1:6.2+dfsg-2ubuntu6.15)
Copyright (c) 2003-2021 Fabrice Bellard and the QEMU Project developers
```

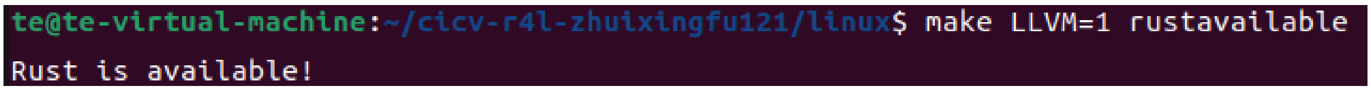
## 6、进入源码，使能Rust



```
te@te-virtual-machine:~/cicv-r4l-zhuixingfu121/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'
  6.6 MiB /   6.6 MiB (100 %)   4.0 MiB/s in  1s ETA:  0s
info: downloading component 'clippy'
info: downloading component 'rust-docs'
 18.3 MiB /  18.3 MiB (100 %)   7.9 MiB/s in  2s ETA:  0s
info: downloading component 'rust-std'
 26.0 MiB /  26.0 MiB (100 %)   7.7 MiB/s in  3s ETA:  0s
info: downloading component 'rustc'
 54.1 MiB /  54.1 MiB (100 %)   9.6 MiB/s in  6s 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 %)  10.4 MiB/s in  1s ETA:  0s
info: installing component 'rust-std'
 26.0 MiB /  26.0 MiB (100 %)  14.8 MiB/s in  1s ETA:  0s
info: installing component 'rustc'
 54.1 MiB /  54.1 MiB (100 %)  17.5 MiB/s in  3s ETA:  0s
info: installing component 'rustfmt'
info: override toolchain for '/home/te/cicv-r4l-zhuixingfu121/linux' set to '1.62.0-x86_64-unknown-linux-gnu'

  1.62.0-x86_64-unknown-linux-gnu installed - rustc 1.62.0 (a8314ef7d 2022-06-27)
```

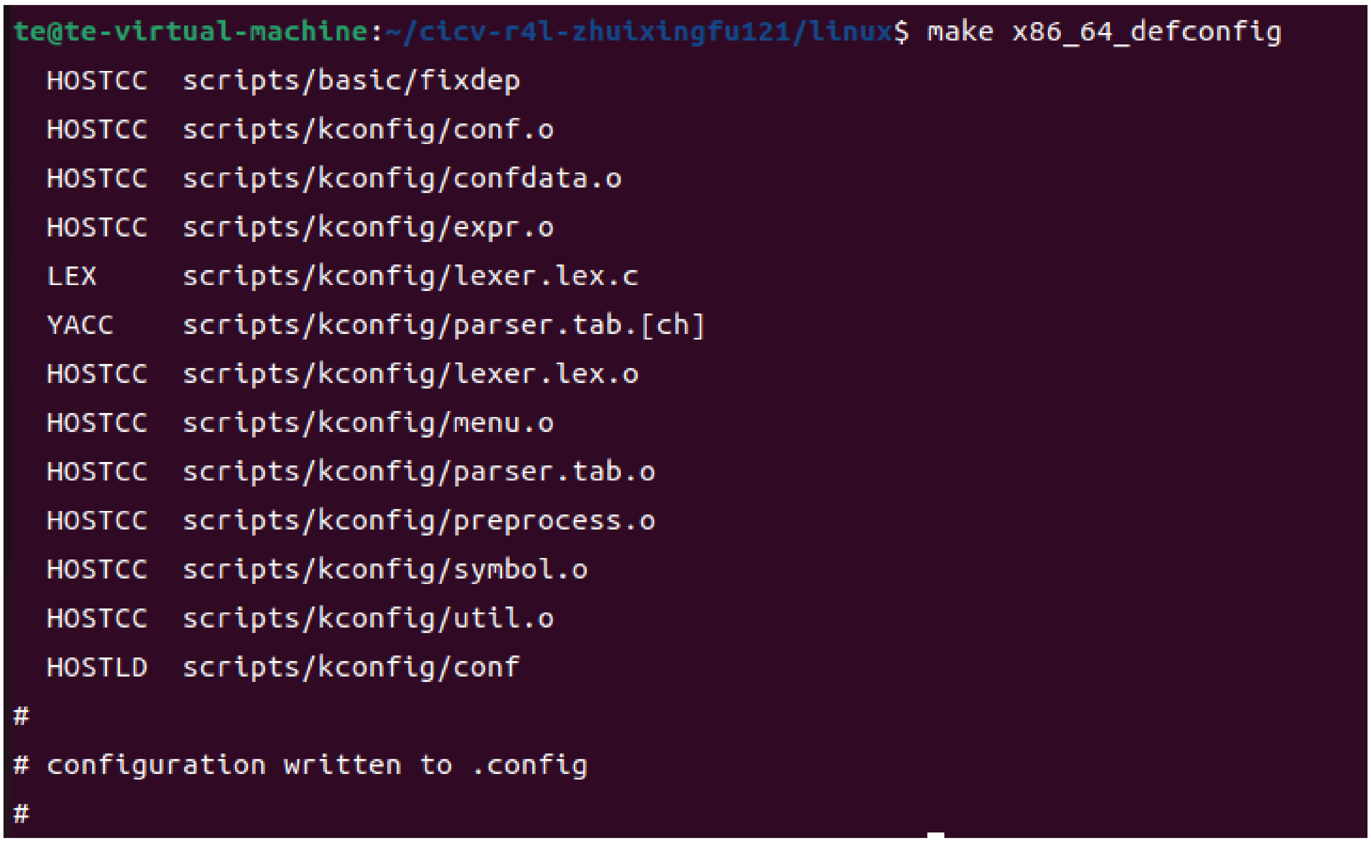


```
   Compiling shlex v0.1.1
   Compiling rustc-hash v1.1.0
   Compiling thread_local v1.0.1
   Compiling textwrap v0.11.0
   Compiling nom v5.1.2
   Compiling libloading v0.6.5
   Compiling clang-sys v1.0.3
   Compiling aho-corasick v0.7.15
   Compiling quote v1.0.7
   Compiling atty v0.2.14
   Compiling which v3.1.1
   Compiling clap v2.33.3
   Compiling regex v1.4.2
   Compiling cexpr v0.4.0
   Compiling env_logger v0.8.1
    Finished release [optimized] target(s) in 20.54s
  Installing /home/te/.cargo/bin/bindgen
   Installed package `bindgen v0.56.0` (executable `bindgen`)
```

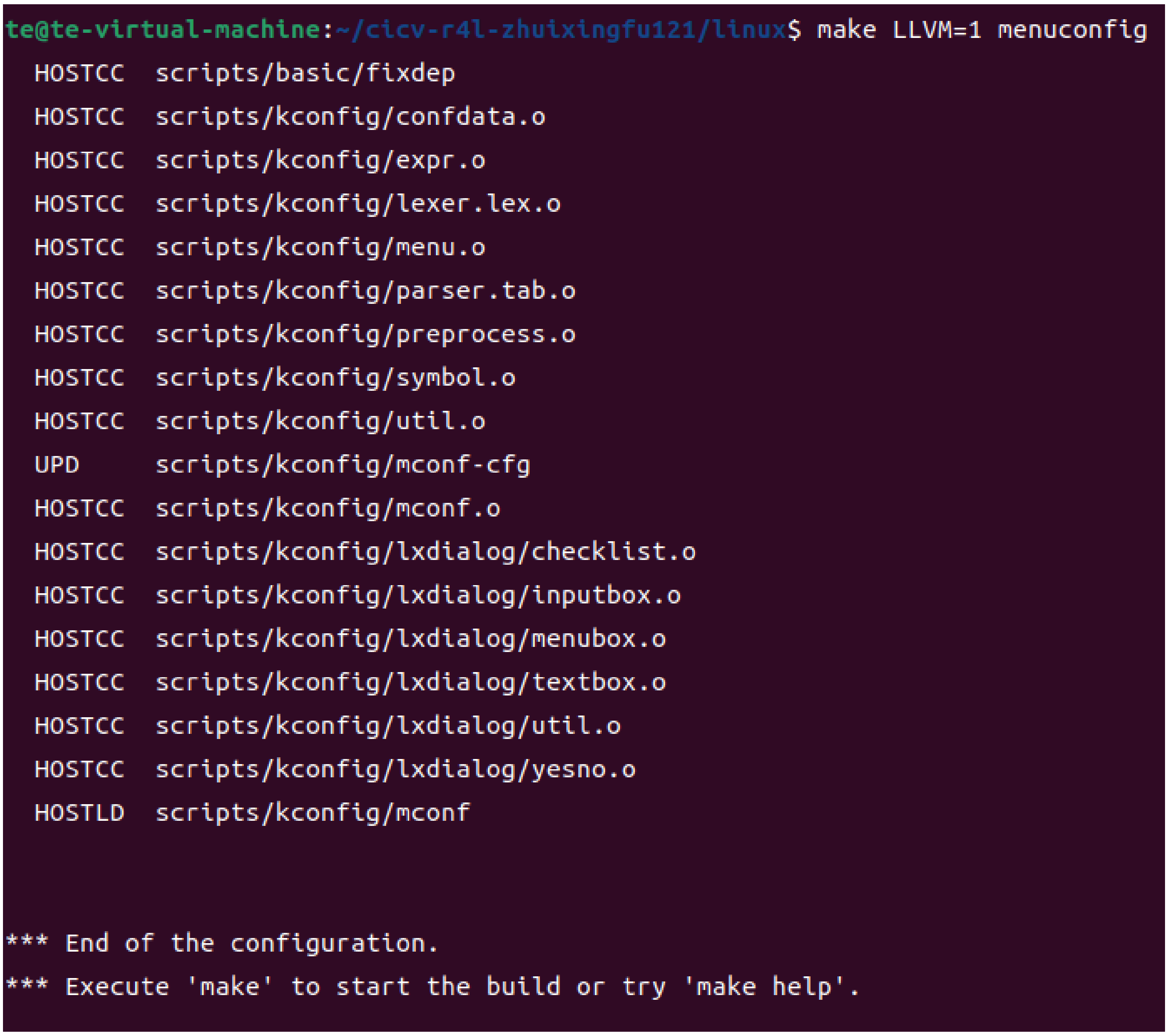


```
te@te-virtual-machine:~/cicv-r4l-zhuixingfu121/linux$ make LLVM=1 rustavailable
Rust is available!
```

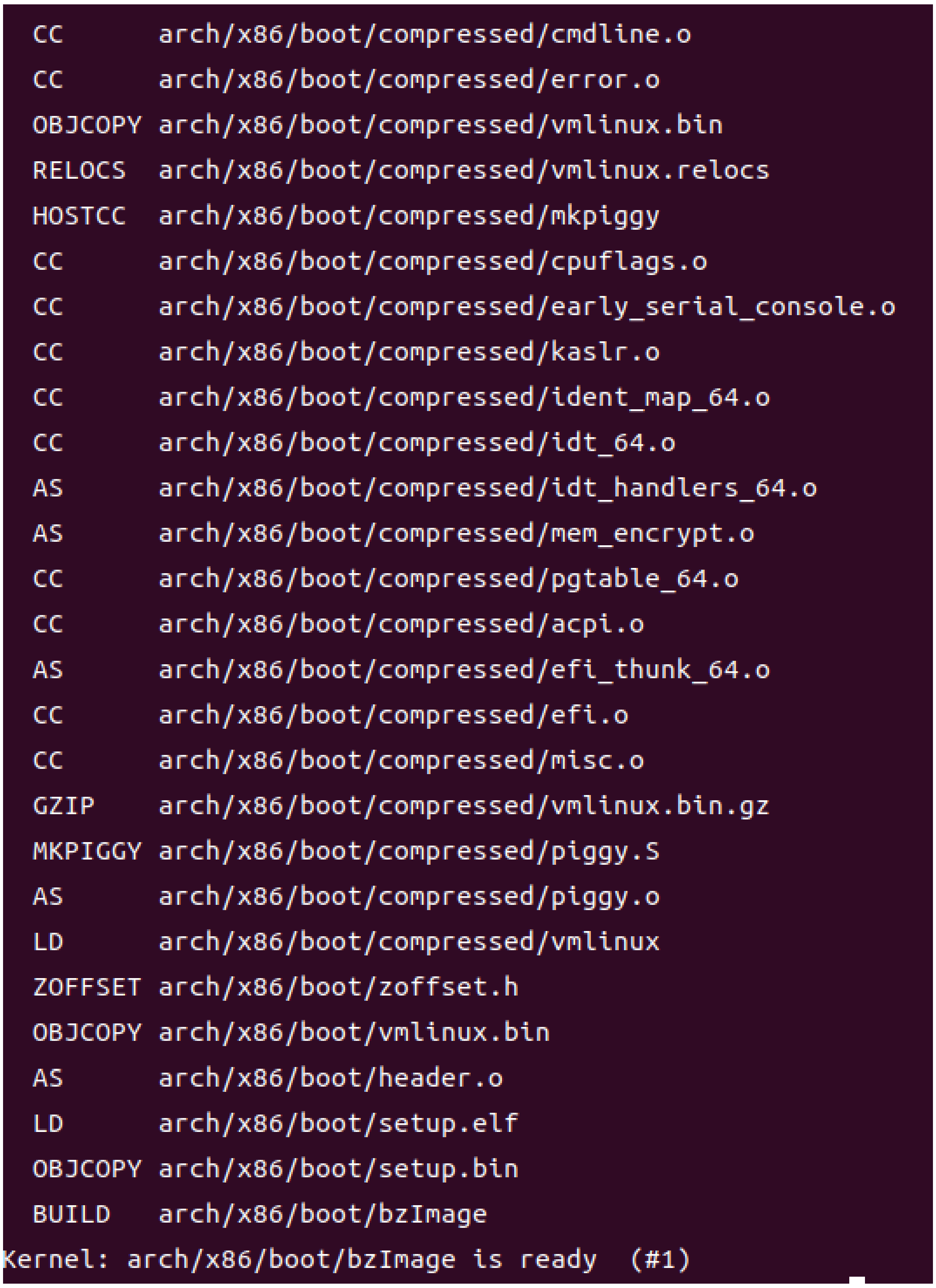
## 二、作业1：编译内核



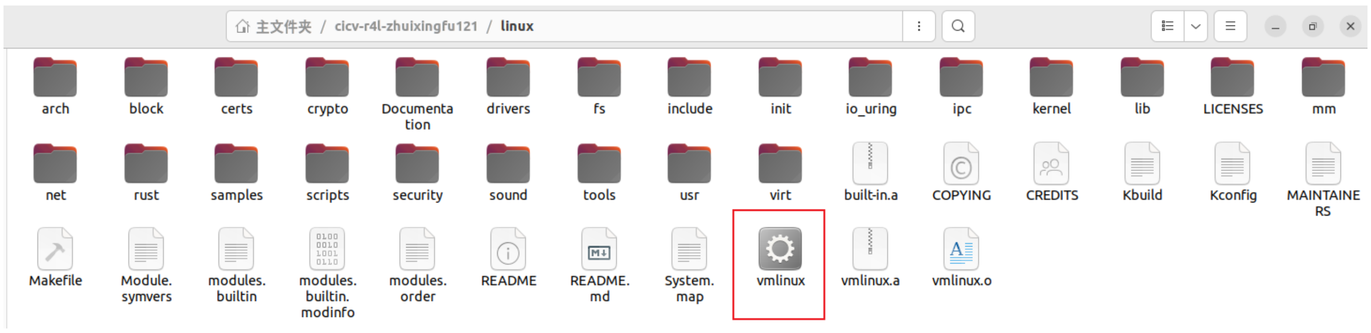
```
te@te-virtual-machine:~/cicv-r4l-zhuixingfu121/linux$ make x86_64_defconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
  HOSTCC  scripts/kconfig/confdata.o
  HOSTCC  scripts/kconfig/expr.o
  LEX     scripts/kconfig/lexer.lex.c
  YACC    scripts/kconfig/parser.tab.[ch]
  HOSTCC  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
#
# configuration written to .config
#
```



```
te@te-virtual-machine:~/cicv-r4l-zhuixingfu121/linux$ make LLVM=1 menuconfig
  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
  UPD     scripts/kconfig/mconf-cfg
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTLD  scripts/kconfig/mconf


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



```
  CC      arch/x86/boot/compressed/cmdline.o
  CC      arch/x86/boot/compressed/error.o
  OBJCOPY arch/x86/boot/compressed/vmlinux.bin
  RELOCS  arch/x86/boot/compressed/vmlinux.relocs
  HOSTCC  arch/x86/boot/compressed/mkpiggy
  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)
```



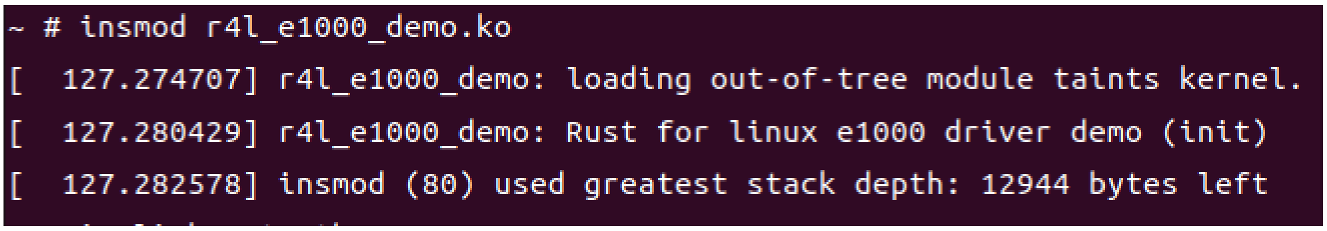
## 三、作业2：编译网卡驱动

1、编译成内核模块，是在哪个文件中以哪条语句定义的？

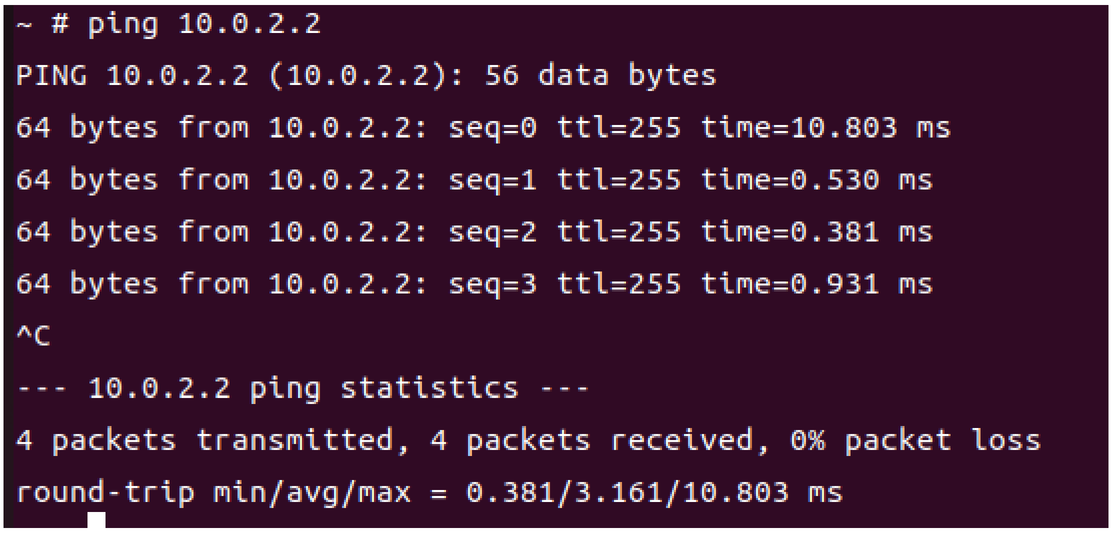
```
Kbuild中的obj-m := r4l_e1000_demo.o
```

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？

```
Out-of-tree模块与内核代码产生联系的关键是内核提供了一些接口和工具，使得编写、编译和加载这些模块成为可能。
```

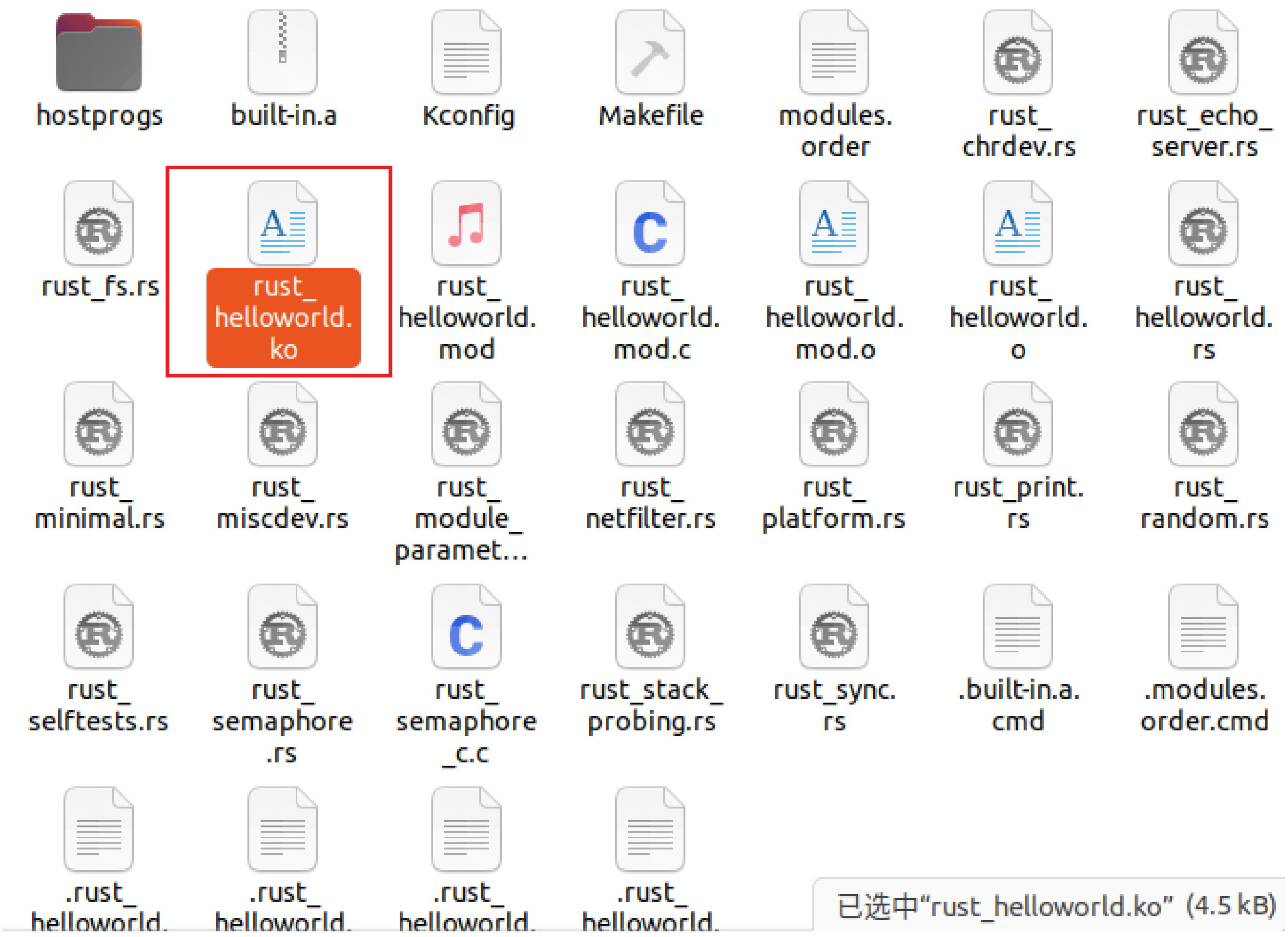


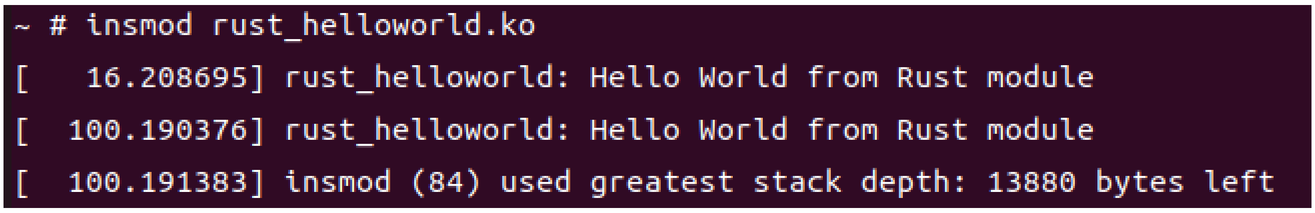
```
~ # insmod r4l_e1000_demo.ko
[  127.274707] r4l_e1000_demo: loading out-of-tree module taints kernel.
[  127.280429] r4l_e1000_demo: Rust for linux e1000 driver demo (init)
[  127.282578] insmod (80) used greatest stack depth: 12944 bytes left
```



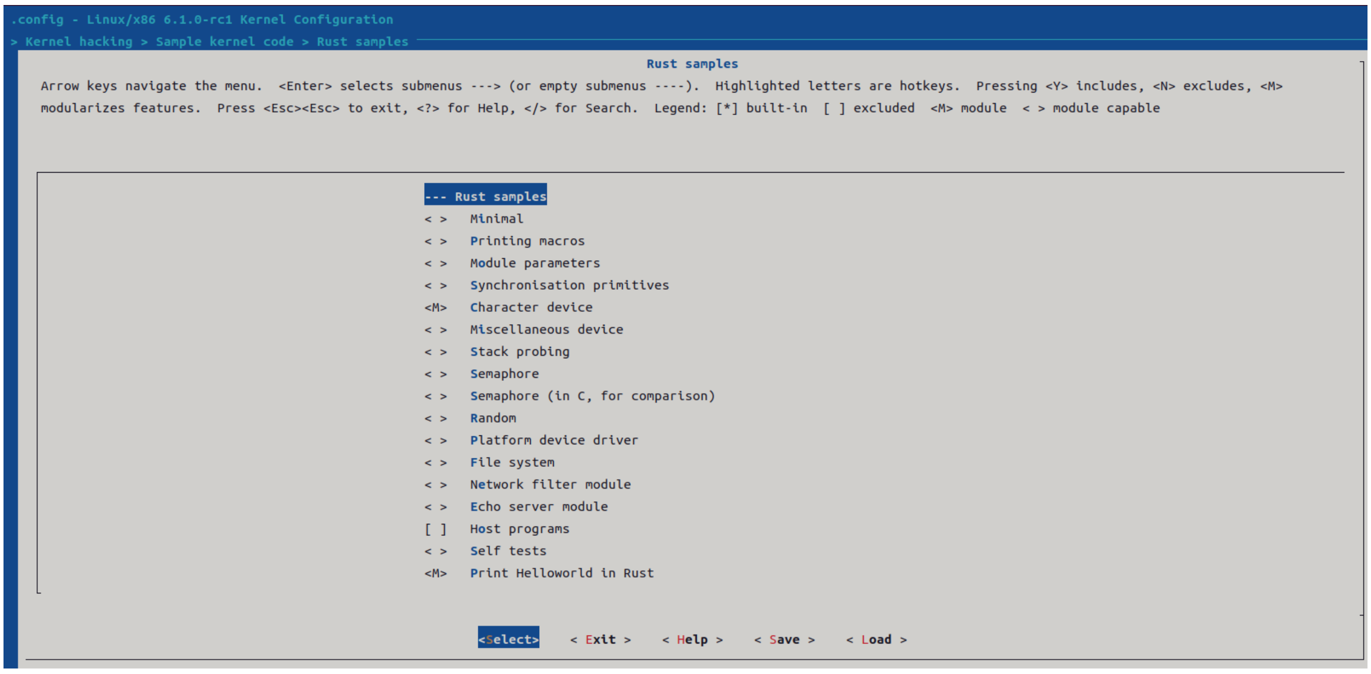
```
~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2): 56 data bytes
64 bytes from 10.0.2.2: seq=0 ttl=255 time=10.803 ms
64 bytes from 10.0.2.2: seq=1 ttl=255 time=0.530 ms
64 bytes from 10.0.2.2: seq=2 ttl=255 time=0.381 ms
64 bytes from 10.0.2.2: seq=3 ttl=255 time=0.931 ms
^C
--- 10.0.2.2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.381/3.161/10.803 ms
```

## 四、作业3：编译Helloworld

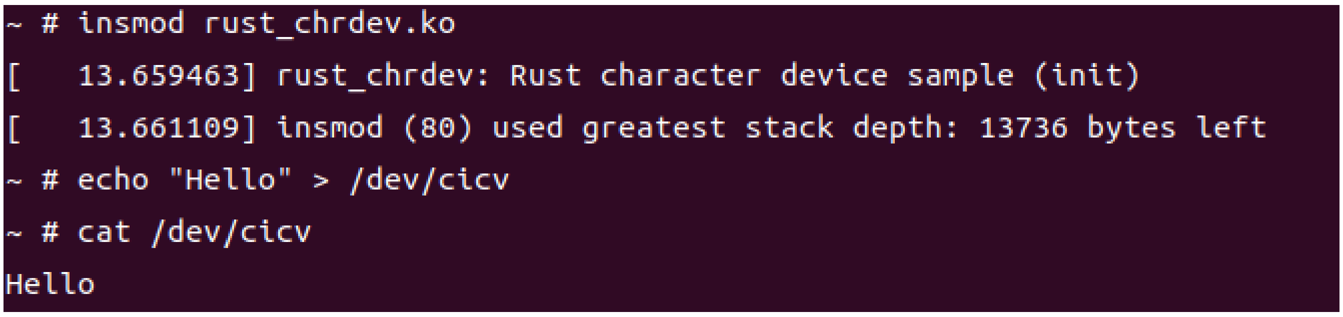



```
~ # insmod rust_helloworld.ko
[   16.208695] rust_helloworld: Hello World from Rust module
[  100.190376] rust_helloworld: Hello World from Rust module
[  100.191383] insmod (84) used greatest stack depth: 13880 bytes left
```

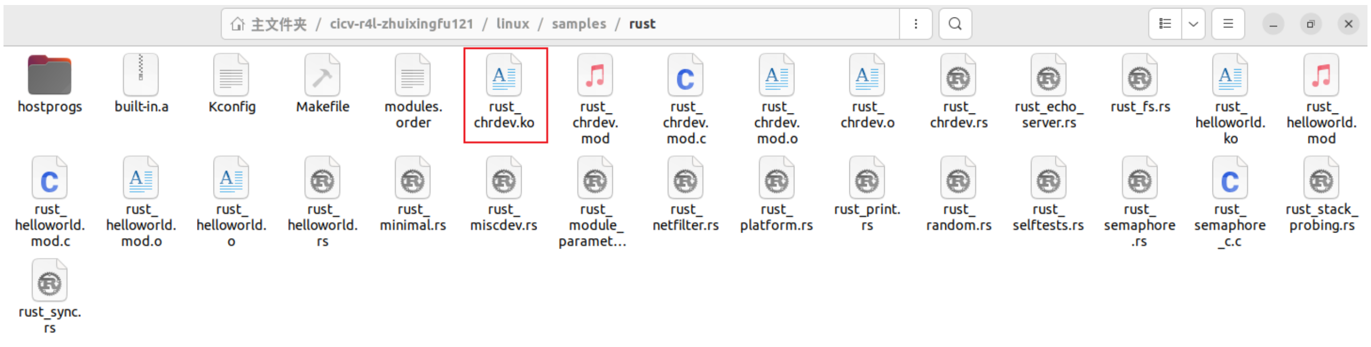
## 六、作业5：Rust编写字符设备驱动



```
.config - Linux/x86 6.1.0-rc1 Kernel Configuration
> Kernel hacking > Sample kernel code > Rust samples
                                   Rust samples
  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

                         --- Rust samples
                         < >   Minimal
                         < >   Printing macros
                         < >   Module parameters
                         < >   Synchronisation primitives
                         <M>   Character device
                         < >   Miscellaneous device
                         < >   Stack probing
                         < >   Semaphore
                         < >   Semaphore (in C, for comparison)
                         < >   Random
                         < >   Platform device driver
                         < >   File system
                         < >   Network filter module
                         < >   Echo server module
                         [ ]   Host programs
                         < >   Self tests
                         <M>   Print Helloworld in Rust

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



```
~ # insmod rust_chrdev.ko
[   13.659463] rust_chrdev: Rust character device sample (init)
[   13.661109] insmod (80) used greatest stack depth: 13736 bytes left
~ # echo "Hello" > /dev/cicv
~ # cat /dev/cicv
Hello
```


主文件夹 / cicv-r4l-zhuixingfu121 / linux / samples / rust
hostprogs
built-in.a
Kconfig
Makefile
modules.order
rust_chrdev.ko
rust_chrdev.mod
rust_chrdev.mod.c
rust_chrdev.mod.o
rust_chrdev.o
rust_chrdev.rs
rust_echo_server.rs
rust_fs.rs
rust_helloworld.ko
rust_helloworld.mod
rust_helloworld.mod.c
rust_helloworld.mod.o
rust_helloworld.o
rust_helloworld.rs
rust_minimal.rs
rust_miscdev.rs
rust_module_paramet...
rust_netfilter.rs
rust_platform.rs
rust_print.rs
rust_random.rs
rust_selftests.rs
rust_semaphore.rs
rust_semaphore_c.c
rust_stack_probing.rs
rust_sync.rs