

# xv6-k210 环境配置手册

---

编写: SpriteGuo

## 目录:

### xv6-k210 环境配置手册

#### 前言

- 一、准备工作
- 二、交叉编译器的安装
- 三、qemu的安装
- 四、配置环境变量
- 五、下载并运行代码
  - 5.1 git下载
  - 5.2 本地下载传入
  - 5.3 修改makefile源文件
- 六、连接本地Vscode
- 附录A: Ubuntu扩容
- 附录B: 文件互传
- 附录C: 目录中英文切换
- 附录D: Ubuntu用vi编写代码的常用指令
  - 常用命令(vi下)
  - 常用快捷键(vi下)
- 附录E: Ubuntu基本设置及快捷键
- 附录F: Ubuntu系统汉化
- 附录G: Ubuntu系统网络连接

## 前言

---

本手册主要参考冰红茶大佬的复现工作，结合自己在复现过程中遇到的各种问题，仅供参考。

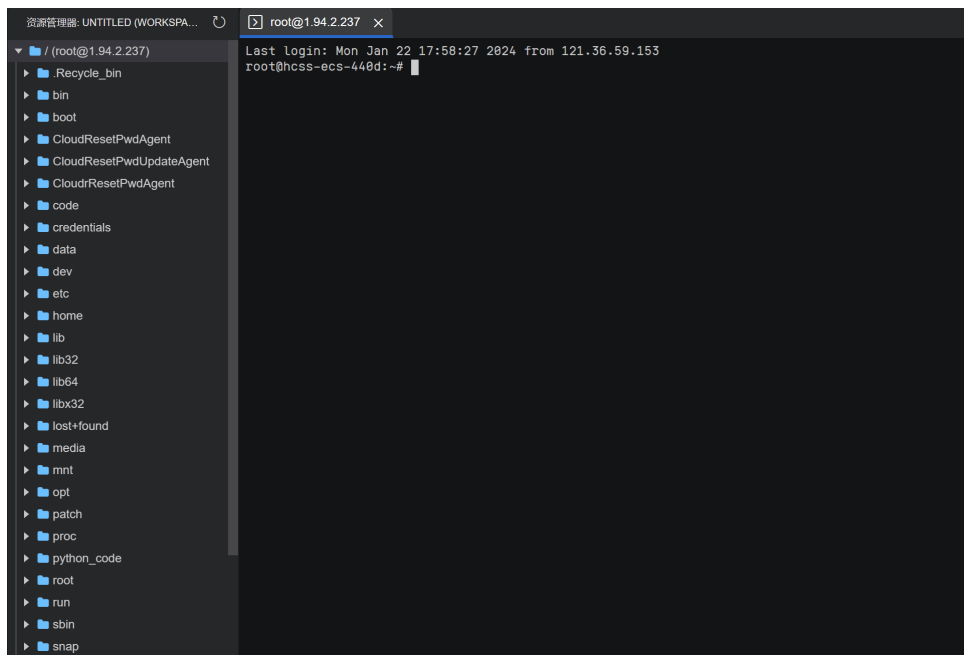
## 一、准备工作

### 1、Ubuntu

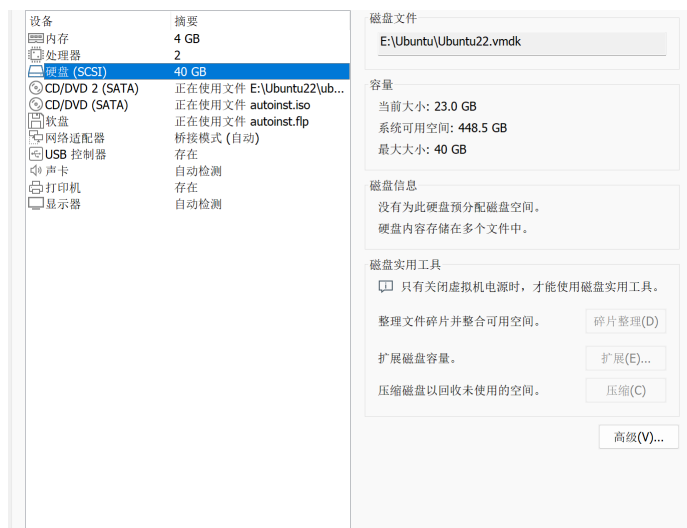
Ubuntu安装教程: [Ubuntu安装教程 \(非常详细\) 从零基础入门到精通, 看完这一篇就够了](#)  
[\\_deepstream5.1 ubuntu安装-CSDN博客](#)



或者云服务器也可以（例如我用的华为云）



2、新建虚拟机的话，内存空间推荐准备40G左右（当然内存空间也是可以增加的，如果之前建过了想扩容可以参考附录A：Ubuntu扩容）



## 二、交叉编译器的安装

- 1、安装包链接链接：<https://pan.baidu.com/s/1t2LxALSt0hePdg4avmYrog?pwd=p1jm>
- 2、下载完成后，将压缩包传输入虚拟机中，我这里通过的是共享文件夹比较简单（这里Windows与Ubuntu的文件传输方法参考附录B：文件互传）
- 3、注意，这里要确保我们的安装路径不含中文（如果Ubuntu已经汉化，最好目录名保持英文，如下图，如果已经修改了，也有可以修改的方法，参考附录C：目录中英文切换）



### 将标准文件夹更新到当前语言吗？

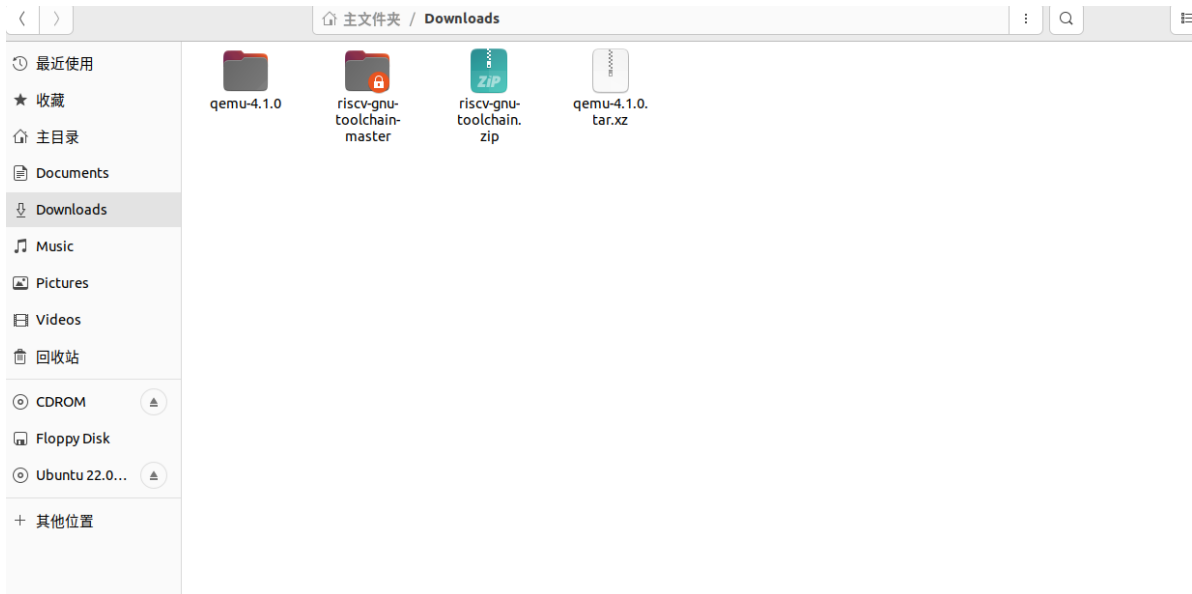
您已经以一种新语言登入。您可以将主文件夹下的某些标准文件夹名按照新语言进行自动更新。该更新将会更改以下文件夹：

当前文件夹名称	新的文件夹名称
/home/bingwu/Desktop	/home/bingwu/桌面
/home/bingwu/Downloads	/home/bingwu/下载
/home/bingwu/Templates	/home/bingwu/模板
/home/bingwu/Public	/home/bingwu/公共的
/home/bingwu/Documents	/home/bingwu/文档
/home/bingwu/Music	/home/bingwu/音乐
/home/bingwu/Pictures	/home/bingwu/图片
/home/bingwu/Videos	/home/bingwu/视频

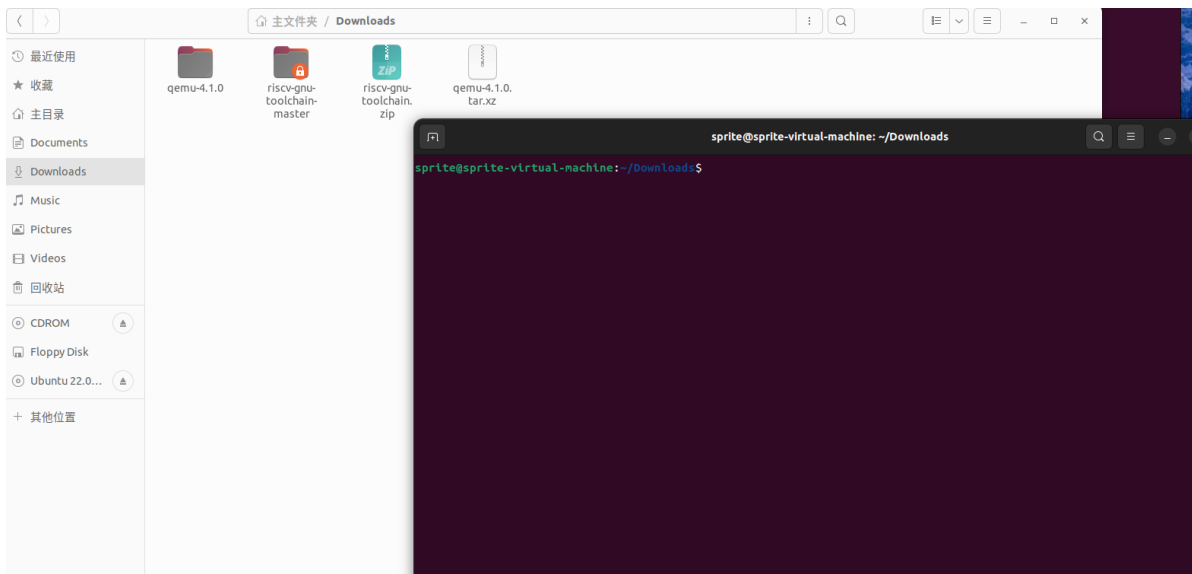
请注意，现有内容不会被移动。

☐ 不要再次询问我(D)

- 4、这里我是将压缩包放在主目录的Downloads目录下。



5、在当前目录下进入终端（Ctrl+Alt+T），如下图：

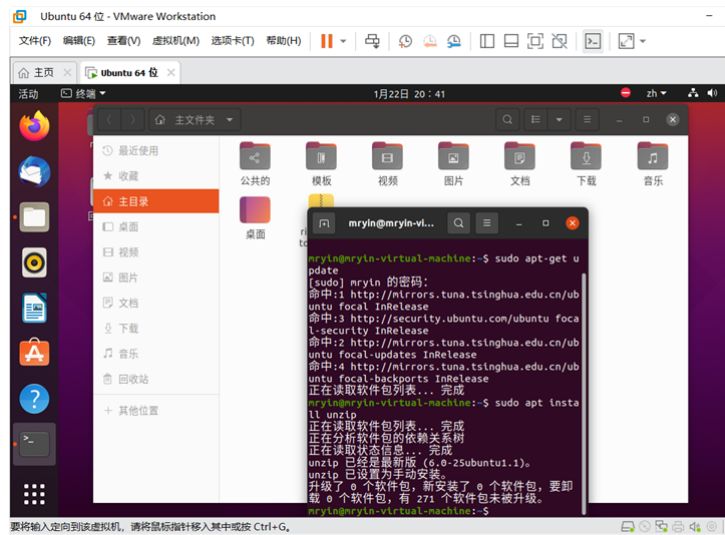


6、依次输入下面的命令进行安装：

```
sudo apt-get update #更新
```

```
sudo apt install unzip #解压
```





```
sudo apt-get install autoconf automake autotools-dev curl device-tree-compiler  
libmpc-dev libmpfr-dev libgmp-dev gawk build-essential bison flex texinfo gperf  
libtool patchutils bc zlib1g-dev git #安装依赖
```

这里要注意，官方是两行代码，我试了，第二行不会下载，导致后面会出错（报错情况如下图），如果上述代码复制后下载出错，可以在CSDN找一行格式的代码，比如：[蜂鸟v2 快速上手之Ubuntu环境配置 sudo apt-get install autoconf automake autotools-d-CSDN博客](#)。

```
python3 -d libtool  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 222 not upgraded.  
curl: (6) Could not resolve host: libmpc-dev  
curl: (6) Could not resolve host: libmpfr-dev  
curl: (6) Could not resolve host: libgmp-dev  
curl: (6) Could not resolve host: gawk  
curl: (3) URL using bad/illegal format or missing URL  
curl: (6) Could not resolve host: bison  
curl: (6) Could not resolve host: flex  
curl: (6) Could not resolve host: texinfo  
curl: (6) Could not resolve host: gperf  
curl: (6) Could not resolve host: libtool  
curl: (6) Could not resolve host: patchutils  
File zlib1g-dev is unavailable.  
root@hcss-ecs-440d:~# sudo apt-get install libnewlib-dev  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done
```

正常情况如下：

```
终端 1月22日 21:31
sprite@sprite-virtual-machine: ~
级。
需要下载 490 kB 的归档。
解压缩后会消耗 2,128 kB 的额外空间。
您希望继续执行吗? [Y/n] y
获取:1 http://us.archive.ubuntu.com/ubuntu jammy/main amd64 gparted-common all
.3.1-1ubuntu1 [71.9 kB]
获取:2 http://us.archive.ubuntu.com/ubuntu jammy/main amd64 gparted amd64 1.3.1
1ubuntu1 [418 kB]
已下载 490 kB, 耗时 6秒 (81.8 kB/s)^A
正在选中未选择的软件包 gparted-common。
(正在读取数据库 ... 系统当前共安装有 178377 个文件和目录。)
准备解压 .../gparted-common_1.3.1-1ubuntu1_all.deb ...
正在解压 gparted-common (1.3.1-1ubuntu1) ...
正在选中未选择的软件包 gparted。
准备解压 .../gparted_1.3.1-1ubuntu1_amd64.deb ...
正在解压 gparted (1.3.1-1ubuntu1) ...
正在设置 gparted-common (1.3.1-1ubuntu1) ...
正在设置 gparted (1.3.1-1ubuntu1) ...
正在处理用于 mailcap (3.70+nmu1ubuntu1) 的触发器 ...
正在处理用于 desktop-file-utils (0.26-1ubuntu3) 的触发器 ...
正在处理用于 hicolor-icon-theme (0.17-2) 的触发器 ...
正在处理用于 gnome-menus (3.36.0-1ubuntu3) 的触发器 ...
正在处理用于 man-db (2.10.2-1) 的触发器 ...
```

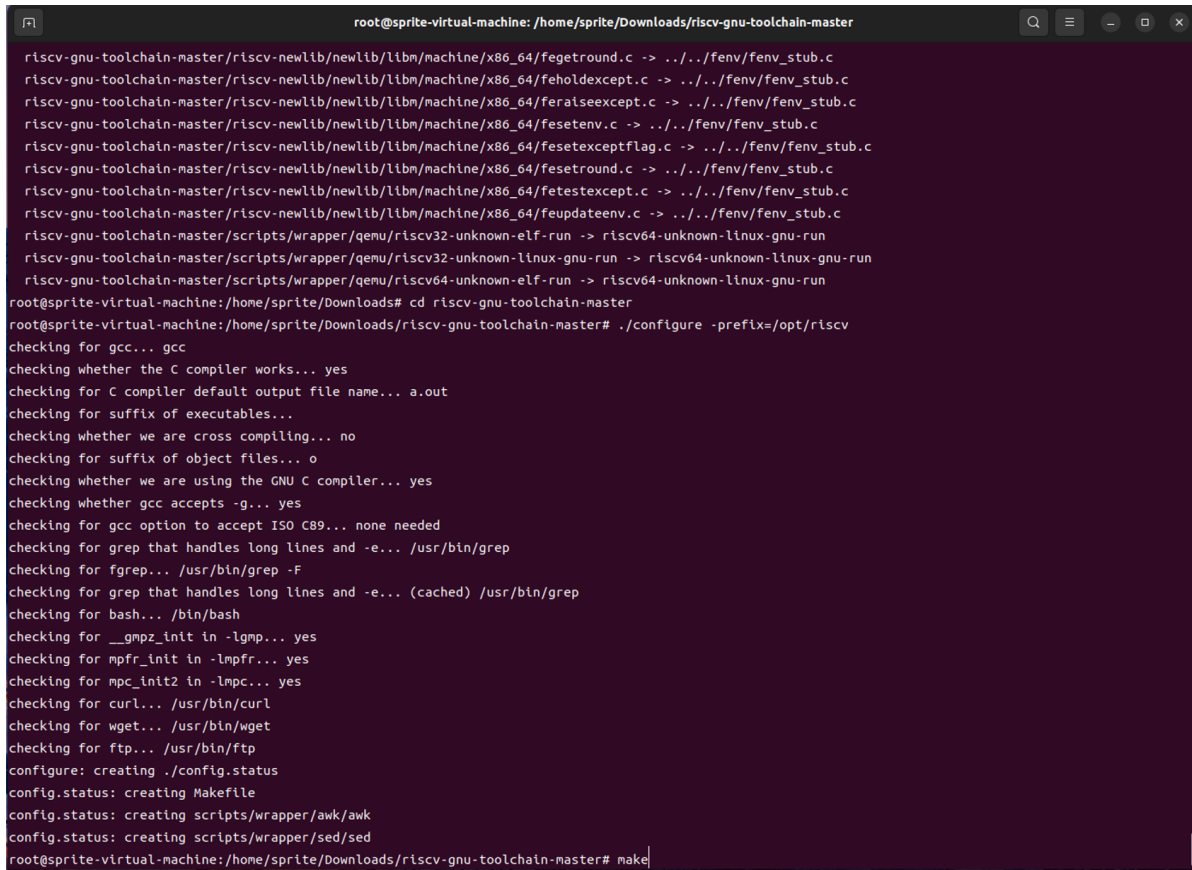
```
sudo apt-get install libnewlib-dev #安装依赖
```

```
unzip riscv-gnu-toolchain #解压
```

```
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fegetround.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feholdexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feraiseexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetenv.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetexceptflag.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetround.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fetestexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feupdateenv.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv32-unknown-elf-run -> riscv64-unknown-linux-gnu-run
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv32-unknown-linux-gnu-run -> riscv64-unknown-linux-gnu-run
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv64-unknown-elf-run -> riscv64-unknown-linux-gnu-run
mryin@mryin-virtual-machine:~$
```

解压完成后运行一下命令:

```
cd riscv-gnu-toolchain-master #切目录
./configure --prefix=/opt/riscv
make
```



```
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fegetround.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feholdexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feraiseexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetenv.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetexceptflag.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fesetround.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/fetestexcept.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/riscv-newlib/newlib/libm/machine/x86_64/feupdateenv.c -> ../../fenv/fenv_stub.c
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv32-unknown-elf-run -> riscv64-unknown-linux-gnu-run
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv32-unknown-linux-gnu-run -> riscv64-unknown-linux-gnu-run
riscv-gnu-toolchain-master/scripts/wrapper/qemu/riscv64-unknown-elf-run -> riscv64-unknown-linux-gnu-run
root@sprite-virtual-machine: /home/sprite/Downloads# cd riscv-gnu-toolchain-master
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master# ./configure -prefix=/opt/riscv
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for grep that handles long lines and -e... /usr/bin/grep
checking for fgrep... /usr/bin/grep -F
checking for grep that handles long lines and -e... (cached) /usr/bin/grep
checking for bash... /bin/bash
checking for __gmpz_init in -lgmp... yes
checking for mpfr_init in -lmpfr... yes
checking for mpc_init2 in -lmpc... yes
checking for curl... /usr/bin/curl
checking for wget... /usr/bin/wget
checking for ftp... /usr/bin/ftp
configure: creating ./config.status
config.status: creating Makefile
config.status: creating scripts/wrapper/awk/awk
config.status: creating scripts/wrapper/sed/sed
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master# make
```

此处需要等待时间较长，约半个多小时（这个期间，可以去安装qemu）

在这里我编译完成后，出现：[ERROR] configure: error: expat is missing or unusable的错误（如下图，这里冰红茶大佬没有出现这个问题）。

这边主要是缺失了expat库：libexpat1-dev，运行下列命令安装即可：

```
sudo apt-get install libexpat1-dev
```

```
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master

checking for etext... yes
checking for gawk... /usr/bin/gawk
checking for a BSD-compatible install... /usr/bin/install -c
checking whether ln -s works... yes
checking for x86_64-pc-linux-gnu-ranlib... ranlib
checking for bison... bison -y
checking for x86_64-pc-linux-gnu-ar... ar
checking for x86_64-pc-linux-gnu-dlltool... dlltool
checking for x86_64-pc-linux-gnu-windres... windres
checking for main in -lm... yes
checking for library containing gethostbyname... none required
checking for library containing socketpair... none required
checking for library containing kinfo_getvmmap... no
checking for library containing kinfo_getfile... no
checking for ld used by GCC... ld
checking if the linker (ld) is GNU ld... yes
checking for shared library run path origin... done
checking for iconv... yes
checking for iconv declaration...
extern size_t iconv (iconv_t cd, char * *inbuf, size_t *inbytesleft, char * *outbuf, size_t *outbytesleft);
checking for library containing waddstr... no
configure: WARNING: no enhanced curses library found; disabling TUI
checking for library containing tgetent... no
checking size of unsigned long long... 8
checking size of unsigned long... 8
checking size of unsigned __int128... 16
checking for library containing dlopen... none required
checking whether to use expat... yes
checking for libexpat... no
configure: error: expat is missing or unusable
make[2]: *** [Makefile:9532: configure-gdb] 错误 1
make[2]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib"
make[1]: *** [Makefile:851: all] 错误 2
make[1]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib"
make: *** [Makefile:418: stamps/build-gdb-newlib] 错误 2
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master# sudo apt-get install libexpat1-dev
正在读取软件包列表... 完成
正在分析软件包的依赖关系树... 完成
```

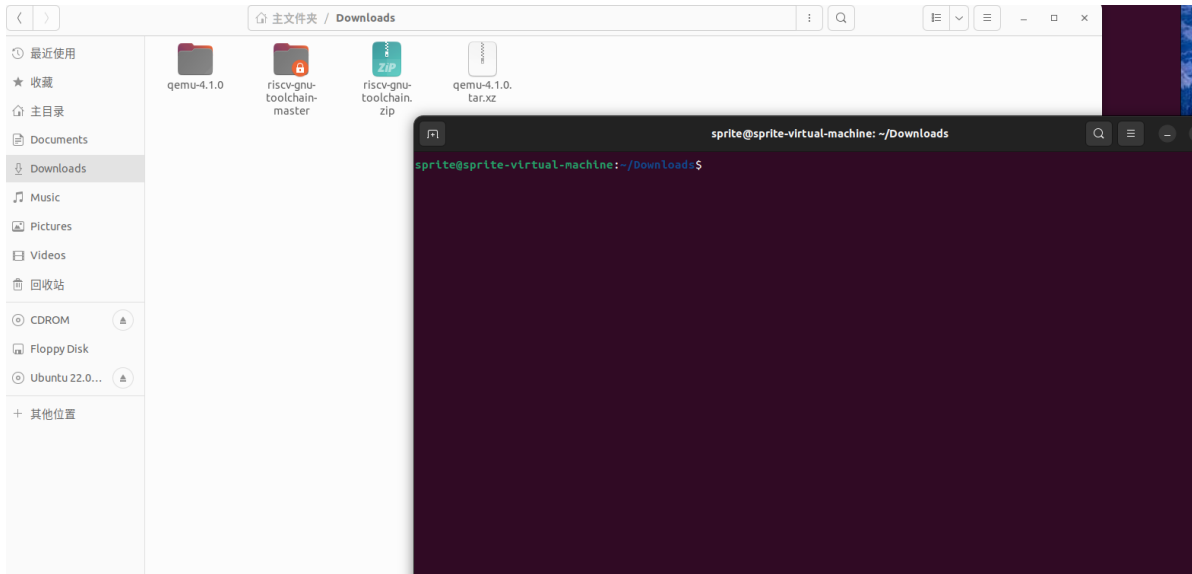
成功编译的结果如下:

```
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master

if test "x$files" != x ; then \
  for file in $files ; do \
    dir=`echo "$file" | sed 's,/[^/]*$,,'` ; \
    /bin/bash /home/sprite/Downloads/riscv-gnu-toolchain-master/riscv-gdb/gdb/data-directory/../../mkinstalldirs \
    /usr/bin/install -c -m 644 ./python/$file /opt/riscv/share/gdb/python/$dir ; \
  done ; \
fi
files=' ' ; \
if test "x$files" != x ; then \
  for file in $files ; do \
    dir=`echo "$file" | sed 's,/[^/]*$,,'` ; \
    /bin/bash /home/sprite/Downloads/riscv-gnu-toolchain-master/riscv-gdb/gdb/data-directory/../../mkinstalldirs \
    /usr/bin/install -c -m 644 ./guile/$file /opt/riscv/share/gdb/guile/$dir ; \
  done ; \
fi
/bin/bash /home/sprite/Downloads/riscv-gnu-toolchain-master/riscv-gdb/gdb/data-directory/../../mkinstalldirs /opt
mkdir -p -- /opt/riscv/share/gdb/system-gdbinit
files='elinos.py wrs-linux.py' ; \
for file in $files; do \
  f=/home/sprite/Downloads/riscv-gnu-toolchain-master/riscv-gdb/gdb/data-directory/../../system-gdbinit/$file ; \
  if test -f $f ; then \
    /usr/bin/install -c -m 644 $f /opt/riscv/share/gdb/system-gdbinit ; \
  fi ; \
done
make[7]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/gdb/data-directory"
make[6]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/gdb/data-directory"
make[5]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/gdb"
make[4]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/gdb"
make[3]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/gdb"
make[3]: 进入目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/libctf"
make[4]: 进入目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/libctf"
make[4]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/libctf"
make[3]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib/libctf"
make[2]: 对"install-target"无需做任何事。
make[2]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib"
make[1]: 离开目录"/home/sprite/Downloads/riscv-gnu-toolchain-master/build-gdb-newlib"
mkdir -p stamps/ && touch stamps/build-gdb-newlib
root@sprite-virtual-machine: /home/sprite/Downloads/riscv-gnu-toolchain-master#
```

### 三、qemu的安装

1、运行终端：这里我还是准备下载在Downloads内，因此依然在该目录运行终端。



2、运行下列命令：

```
sudo apt-get install build-essential pkg-config libboost-all-dev autoconf libtool  
libssl-dev flex bison ninja-build libglib2.0-dev libpixman-1-dev libslirp-dev  
libncurses5-dev libncursesw5-dev #安装依赖
```

下载：

```
wget https://download.qemu.org/qemu-4.1.0.tar.xz  
tar xvjf qemu-4.1.0.tar.xz  
cd qemu-4.1.0/
```

配置：

```
./configure --disable-kvm --disable-werror --prefix=/usr/local --target-  
list="riscv64-sofmmu "
```



```
selinux : YES 3.3

Subprojects
libvduse : YES
libvhost-user : YES

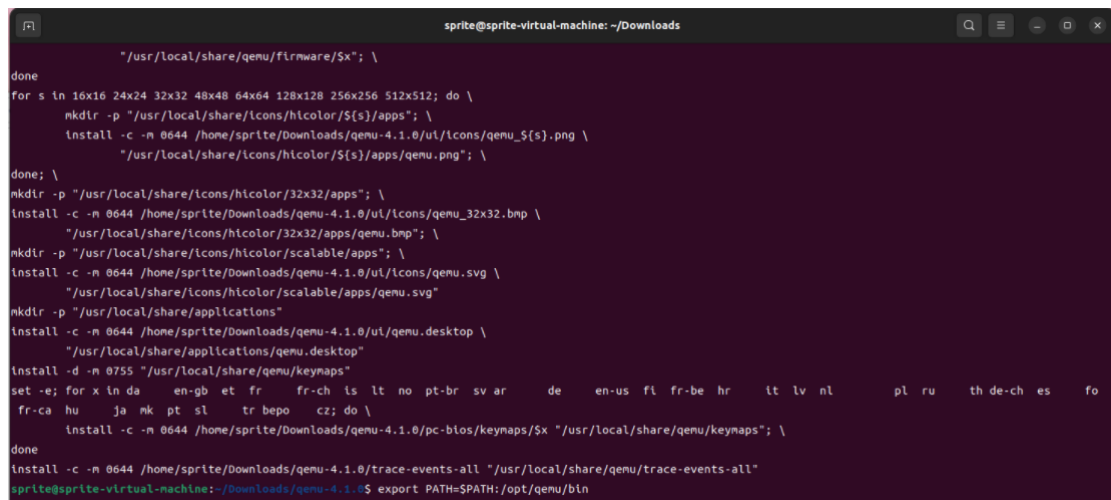
User defined options
Native files : config-meson.cross
prefix : /opt/qemu
slirp : enabled
vfio_user_server : disabled

Found ninja-1.10.1 at /usr/bin/ninja
Running postconf script '/media/a/280EF7DA0EF79F4E/tool/riscv64-linux/qemu-7.2.0/scripts/symlink-install-tree.py'
```

make -j\$(nproc) #注意编译时间较长

```
[2378/2387] Compiling C object tests/qtest/qos-test.p/virtio-blk-test.c.o
[2379/2387] Compiling C object tests/qtest/qos-test.p/vmxnet3-test.c.o
[2380/2387] Compiling C object tests/qtest/qos-test.p/virtio-scsi-test.c.o
[2381/2387] Compiling C object tests/qtest/qos-test.p/e1000e-test.c.o
[2382/2387] Compiling C object tests/qtest/readconfig-test.p/readconfig-test.c.o
[2383/2387] Compiling C object tests/qtest/qos-test.p/virtio-iommu-test.c.o
[2384/2387] Compiling C object tests/qtest/qos-test.p/vhost-user-blk-test.c.o
[2385/2387] Compiling C object tests/qtest/qos-test.p/vhost-user-test.c.o
[2386/2387] Linking target tests/qtest/readconfig-test
[2387/2387] Linking target tests/qtest/qos-test
make[1]: 离开目录"/media/a/280EF7DA0EF79F4E/tool/riscv64-linux/qemu-7.2.0/build"
changing dir to build for make ""...
make[1]: 进入目录"/media/a/280EF7DA0EF79F4E/tool/riscv64-linux/qemu-7.2.0/build"
[1/52] Generating tests/include/QAPI test (include) with a custom command
[2/20] Generating qemu-version.h with a custom command (wrapped by meson to capture output)
```

sudo make install



```
sprite@sprite-virtual-machine: ~/Downloads
"/usr/local/share/qemu/firmware/$x"; \
done
for s in 16x16 24x24 32x32 48x48 64x64 128x128 256x256 512x512; do \
    mkdir -p "/usr/local/share/icons/hicolor/$s/apps"; \
    install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/ui/icons/qemu_$s.png \
        "/usr/local/share/icons/hicolor/$s/apps/qemu.png"; \
done; \
mkdir -p "/usr/local/share/icons/hicolor/32x32/apps"; \
install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/ui/icons/qemu_32x32.bmp \
    "/usr/local/share/icons/hicolor/32x32/apps/qemu.bmp"; \
mkdir -p "/usr/local/share/icons/hicolor/scalable/apps"; \
install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/ui/icons/qemu.svg \
    "/usr/local/share/icons/hicolor/scalable/apps/qemu.svg"
mkdir -p "/usr/local/share/applications"
install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/ui/qemu.desktop \
    "/usr/local/share/applications/qemu.desktop"
install -d -m 0755 "/usr/local/share/qemu/keymaps"
set -e; for x in da en-gb et fr fr-ch is lt no pt-br sv ar de en-us fl fr-be hr it lv nl pl ru th de-ch es fo
fr-ca hu ja nk pt sl tr bepo cz; do \
    install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/pc-bios/keymaps/$x "/usr/local/share/qemu/keymaps"; \
done
install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/trace-events-all "/usr/local/share/qemu/trace-events-all"
sprite@sprite-virtual-machine: ~/Downloads/qemu-4.1.0$ export PATH=$PATH:/opt/qemu/bin
```

查看结果:

```
export PATH=$PATH:/opt/qemu/bin
qemu-system-riscv64 --version
```

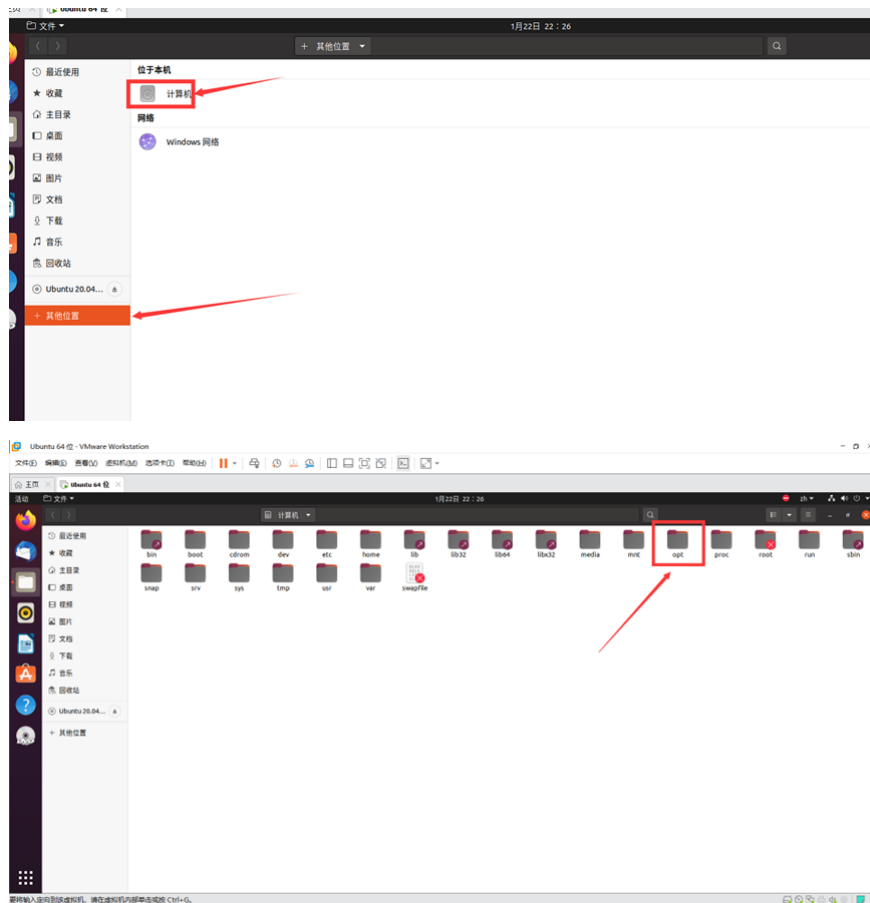
```
cd .. 返回上级目录，再看一次版本，检查是否环境配置好了
qemu-system-riscv64 --version
```

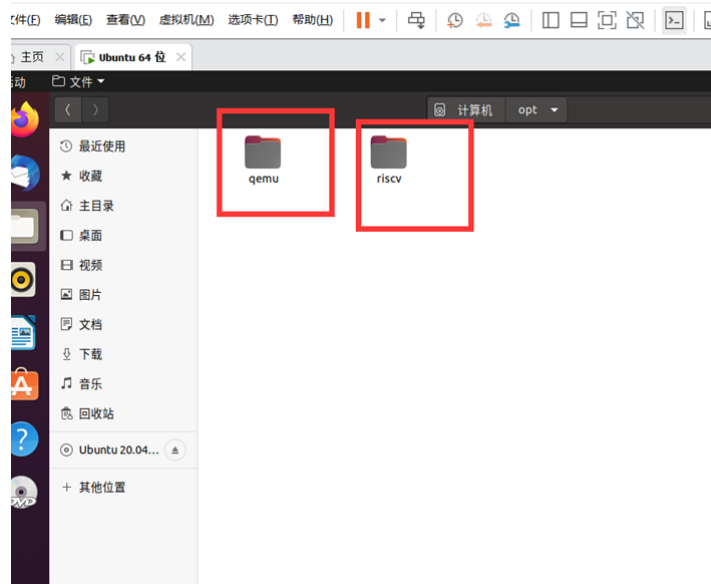
```
install -c -m 0644 /home/sprite/Downloads/qemu-4.1.0/trace-events-all "/usr/local/share/qemu/trace-events-all"
sprite@sprite-virtual-machine:~/Downloads/qemu-4.1.0$ export PATH=$PATH:/opt/qemu/bin
sprite@sprite-virtual-machine:~/Downloads/qemu-4.1.0$ qemu-system-riscv64 --version
qemu-system-riscv64: -version: Could not open '-version': No such file or directory
sprite@sprite-virtual-machine:~/Downloads/qemu-4.1.0$ qemu-system-riscv64 --version
QEMU emulator version 4.1.0
Copyright (c) 2003-2019 Fabrice Bellard and the QEMU Project developers
sprite@sprite-virtual-machine:~/Downloads/qemu-4.1.0$ cd..
cd.: 未找到命令
sprite@sprite-virtual-machine:~/Downloads/qemu-4.1.0$ cd ..
sprite@sprite-virtual-machine:~/Downloads$ qemu-system-riscv64 --version
QEMU emulator version 4.1.0
Copyright (c) 2003-2019 Fabrice Bellard and the QEMU Project developers
sprite@sprite-virtual-machine:~/Downloads$
```

## 四、配置环境变量

安装完成后需要进行环境变量的配置：

在这个之前，先去看看qemu和riscv是否下载好。





如果有的话，开始配置环境变量：

第一步：关闭所有终端，出去在主目录中，重新打开一个终端。

输入：

```
sudo vi ~/.bashrc #打开文件后在最末尾写入路径
export PATH="$PATH:/opt/riscv/bin" #编译器的路径
export PATH="$PATH:/opt/qemu/bin" # qemu的路径    这俩是添加到文件末尾的两行。
source ~/.bashrc    运行该命令使配置的路径生效：
```

注：在vi中，建议使用下面命令（更多vi常见命令如附录D：Ubuntu用vi编写代码的常用指令）

```
step1 (Shift+g) [跳到文件最后一行]
step2 (Shift+4) [跳到行最后一个字符]
```

编写情况：

```
fi
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile.d/
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
export PATH="$PATH:/opt/riscv/bin"
export PATH="$PATH:/opt/qemu/bin"
```

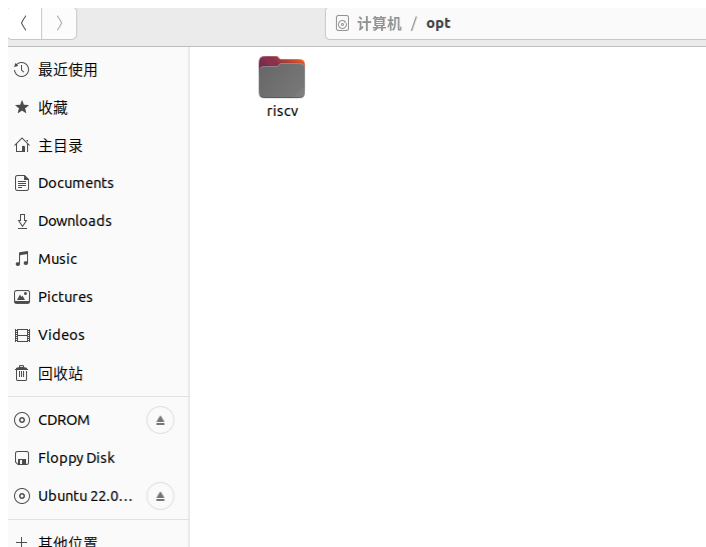
配置后你在任何目录下输入下面都会有对应输出：

```
qemu-system-riscv64 --version
riscv64-unknown-elf-gcc --version
```

```
mryin@mryin-virtual-machine:/opt/riscv$ source ~/.bashrc
mryin@mryin-virtual-machine:/opt/riscv$ qemu-system-riscv64 --version
QEMU emulator version 7.2.0
Copyright (c) 2003-2022 Fabrice Bellard and the QEMU Project developers
mryin@mryin-virtual-machine:/opt/riscv$ riscv64-unknown-elf-gcc --version
riscv64-unknown-elf-gcc (GCC) 10.1.0
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```



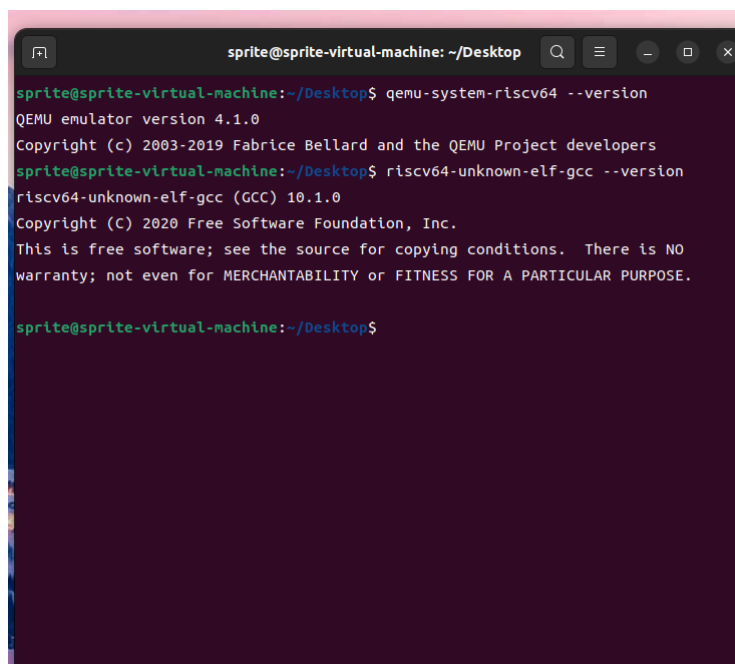
当然上面是冰红茶大佬的情况，实际上我在安装完成后opt文件夹内只有riscv文件。（可能是下载路径的原因吧，具体原因也没有去细查，其实在上述第三节最后我们就发现安装完成后环境变量已经给我们自动配置好了，究竟是不是这样我也不确定，没有去仔细查阅资料）



如果你的情况和我一样，那么只需要配置riscv的环境变量即可，步骤差不多，只要文件末尾加入riscv的位置即可：

```
export PATH="$PATH:/opt/riscv/bin" #编译器的路径
```

最后实现的效果是一样的：



## 五、下载并运行代码

1、下载源代码仓库：[HUST-OS/xv6-k210: Port XV6 to K210 board! \(github.com\)](https://github.com/HUST-OS/xv6-k210).

## 5.1 git下载

#下载git (可以找CSDN命令是什么)

#配置git的代理, 如果是clash:

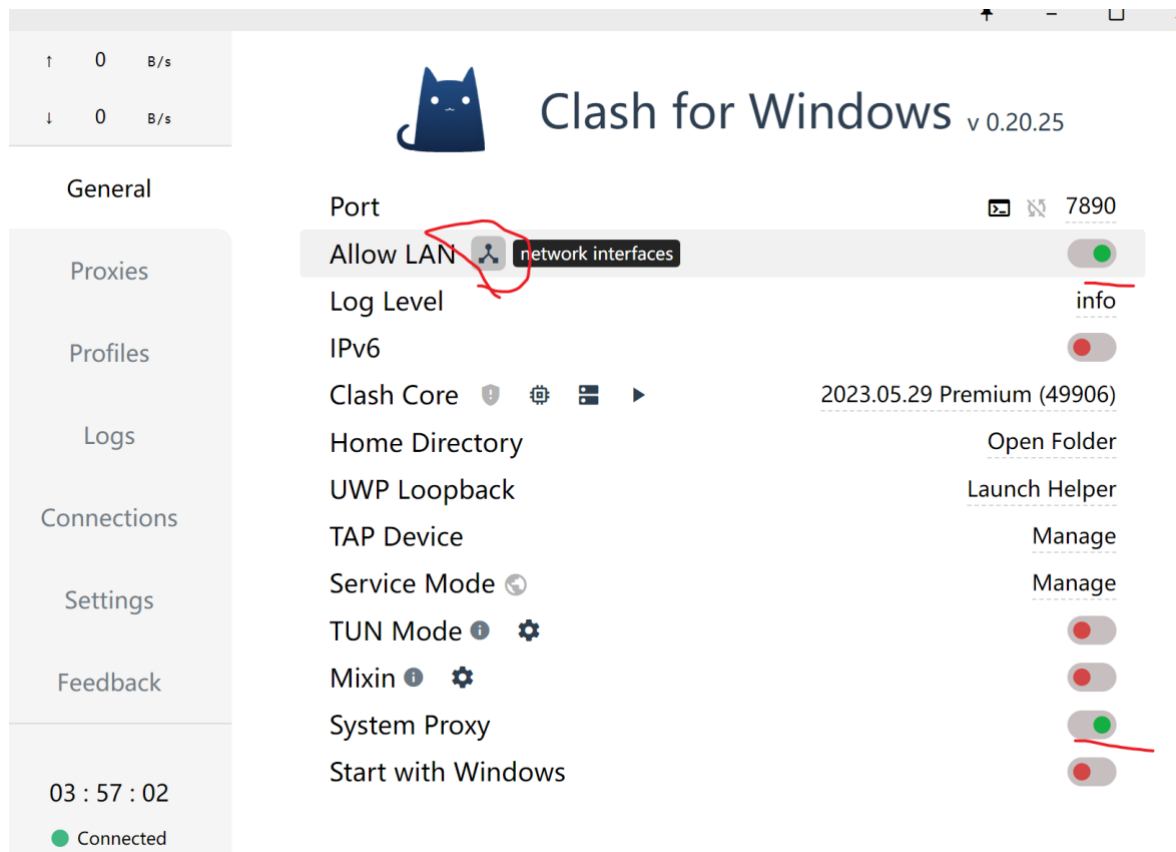
```
git config -global http.https: /github.com.proxy socks5: /192.168.129.137:7890
#设置代理
```

#如果要取消

```
git config -global -unset http.proxy git config global -unset https.proxy
```

#然后clone项目到虚拟机合适的地方

```
git clone https: /github.com/HUST-OS/xv6-k210
```



## vEthernet (Default Switch)

Address: 192.168.114.177  
Netmask: 255.255.255.240 (28)  
MAC: a8:15:dd:b9:62:63

## VirtualBox Host-Only Network

Address: 192.168.56.1  
Netmask: 255.255.255.0 (24)  
MAC: 0a:00:27:00:00:0f

## VMware Network Adapter VMnet1

Address: 192.168.30.1  
Netmask: 255.255.255.0 (24)  
MAC: 00:50:56:c0:00:01

## VMware Network Adapter VMnet8

Address: 192.168.31.1  
Netmask: 255.255.255.0 (24)  
MAC: 00:50:56:c0:00:08

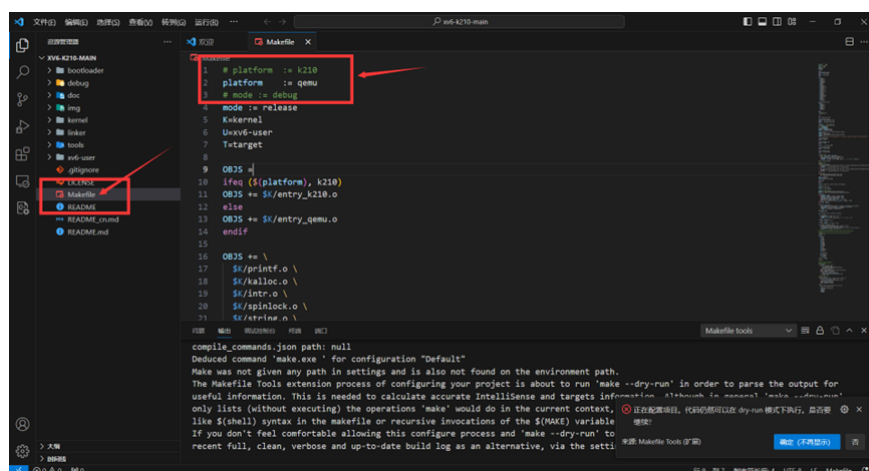
## WLAN

Address: 192.168.129.137  
Netmask: 255.255.255.0 (24)  
MAC: 14:85:7f:ff:84:ee

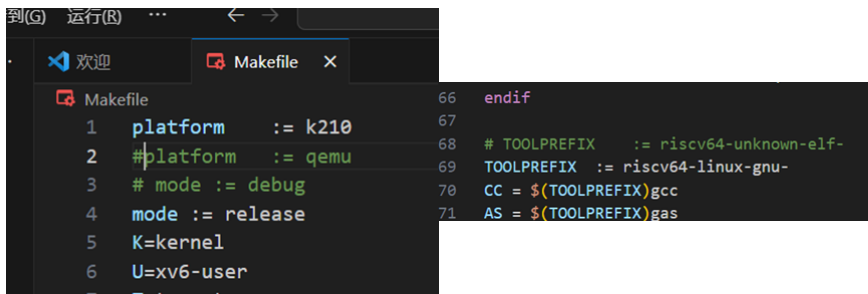
## 5.2 本地下载传入

在本地下载好压缩包通过共享文件夹传入到虚拟机中

## 5.3 修改makefile源文件

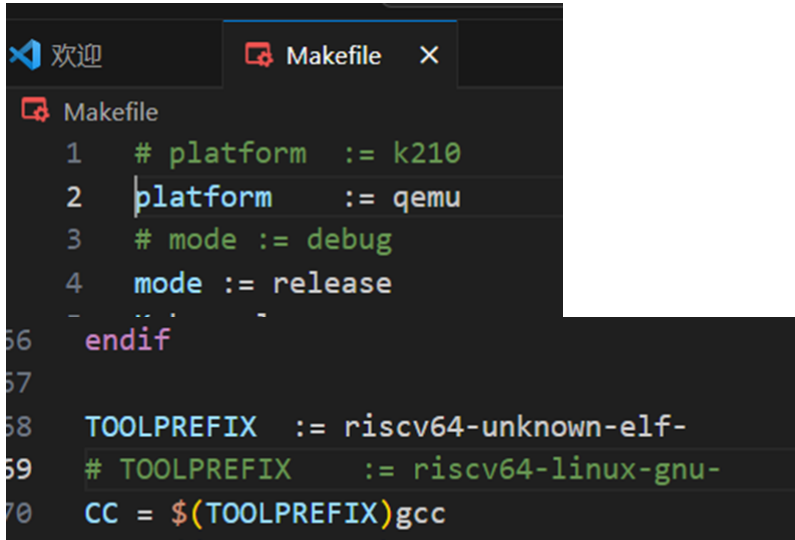


修改两处：



```
1 platform := k210
2 #platform := qemu
3 # mode := debug
4 mode := release
5 K=kernel
6 U=xv6-user
66 endif
67
68 # TOOLPREFIX := riscv64-unknown-elf-
69 TOOLPREFIX := riscv64-linux-gnu-
70 CC = $(TOOLPREFIX)gcc
71 AS = $(TOOLPREFIX)gas
```

改为:

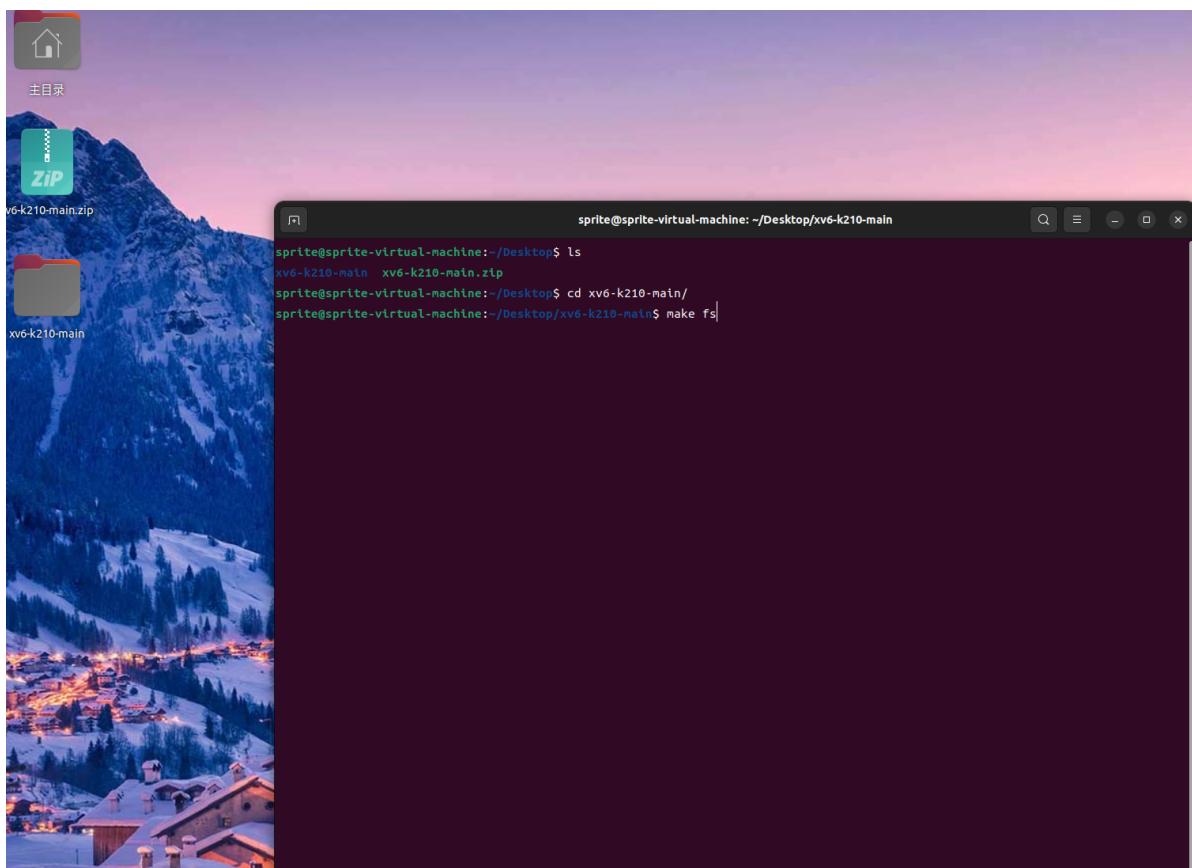


```
1 # platform := k210
2 platform := qemu
3 # mode := debug
4 mode := release
56 endif
57
58 TOOLPREFIX := riscv64-unknown-elf-
59 # TOOLPREFIX := riscv64-linux-gnu-
70 CC = $(TOOLPREFIX)gcc
```

接着就可以在虚拟机中进行运行。

从文件所在目录进入终端（比如我在desktop）并输入以下命令即可得到运行结果：

```
make fs
make run platform=qemu
```



```
sprite@sprite-virtual-machine: ~/Desktop/xv6-k210-main
268435450字节 (268 MB, 256 MiB) 已复制, 2.42183 s, 111 MB/s
mkfs.fat 4.2 (2021-01-31)
[sudo] sprite 的密码:
sprite@sprite-virtual-machine:~/Desktop/xv6-k210-main$ make run platform=qemu
[rustsbi] RustSBI version 0.1.1

[rustsbi] Platform: QEMU (Version 0.1.0)
[rustsbi] misa: RV64ACDFIMSU
[rustsbi] mdeleg: 0x222
[rustsbi] medeleg: 0xb1ab
[rustsbi-dtb] Hart count: cluster0 with 2 cores
[rustsbi] Kernel entry: 0x80200000

hart 0 init done
hart 1 init done
init: starting sh
=> / $
=> / $
=> / $
```

## 六、连接本地Vscode

1、安装[openssh](#)-server，首先进入终端，输入：

```
sudo apt-get install openssh-server
```

(如果在桌面打开中断安装可能失败，需要cd一下)

```
sprite@sprite-virtual-machine: ~/Desktop
sprite@sprite-virtual-machine:~/Desktop$ sudo apt-get install openssh-server
[sudo] sprite 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树... 完成
正在读取状态信息... 完成
将会同时安装下列软件：
  ncurses-term openssh-sftp-server ssh-import-id
建议安装：
  molly-guard monkeysphere ssh-askpass
下列【新】软件包将被安装：
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
升级了 0 个软件包，新安装了 4 个软件包，要卸载 0 个软件包，有 64 个软件包未被升级。
需要下载 752 kB 的归档。
解压后会消耗 6,050 kB 的额外空间。
您希望继续执行吗？ [Y/n] y
获取:1 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openssh-sftp-server amd64 1:8.9p1-3ubuntu0.6 [38.7 kB]
获取:2 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openssh-server amd64 1:8.9p1-3ubuntu0.6 [435 kB]
获取:3 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ncurses-term all 6.3-2ubuntu0.1 [267 kB]
获取:4 http://us.archive.ubuntu.com/ubuntu jammy/main amd64 ssh-import-id all 5.11-0ubuntu1 [10.1 kB]
已下载 752 kB，耗时 3秒 (277 kB/s)
正在预设定软件包 ...
正在选中未选择的软件包 openssh-sftp-server。
(正在读取数据库 ... 系统当前共安装有 237510 个文件和目录。)
准备解压 .../openssh-sftp-server_1%3a8.9p1-3ubuntu0.6_amd64.deb ...
正在解压 openssh-sftp-server (1:8.9p1-3ubuntu0.6) ...
正在选中未选择的软件包 openssh-server。
准备解压 .../openssh-server_1%3a8.9p1-3ubuntu0.6_amd64.deb ...
正在解压 openssh-server (1:8.9p1-3ubuntu0.6) ...
正在选中未选择的软件包 ncurses-term。
准备解压 .../ncurses-term_6.3-2ubuntu0.1_all.deb ...
```

启动ssh服务：

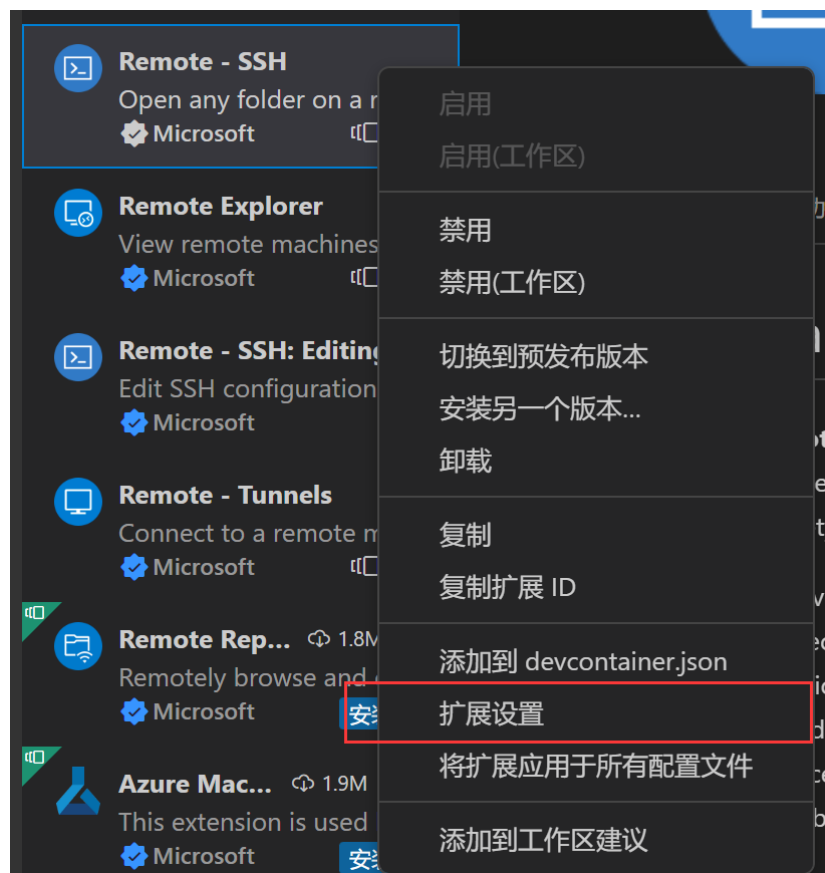
```
sudo systemctl start ssh #启动ssh服务
```

```
正在处理用于 man-db (2.10.2-1) 的触发器 ...
正在处理用于 ufw (0.36.1-4ubuntu0.1) 的触发器 ...
sprite@sprite-virtual-machine:~/Desktop$ sudo systemctl start ssh
sprite@sprite-virtual-machine:~/Desktop$
```

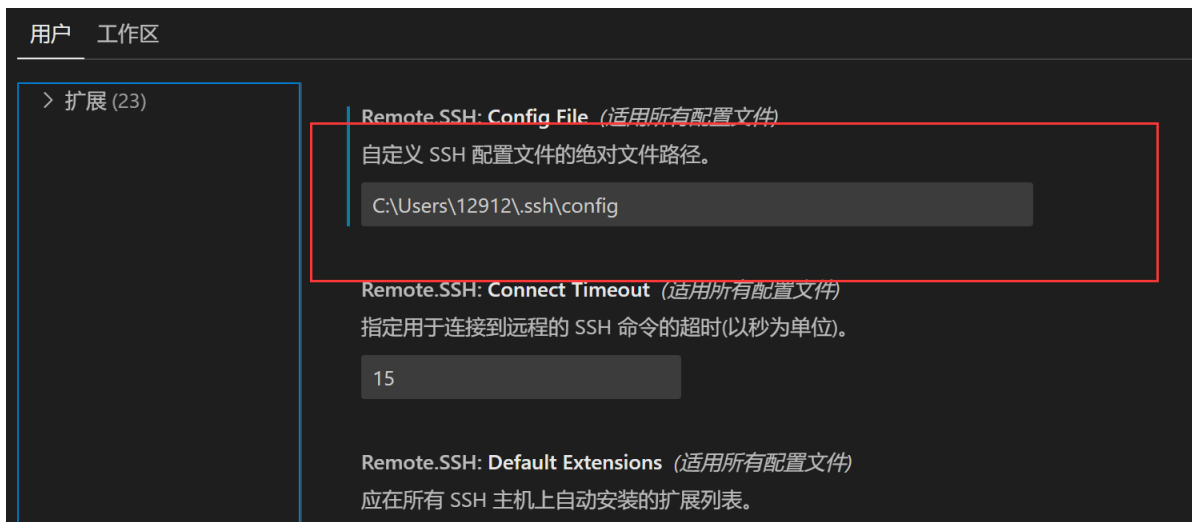
接着回到本地计算机，打开Vscode，下载插件：



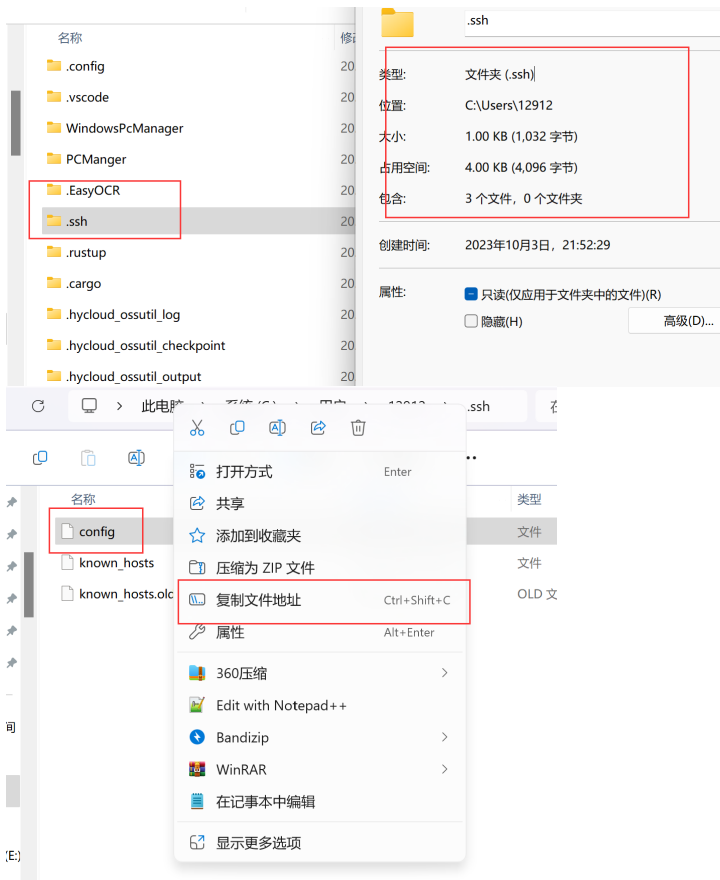
进行扩展设置：



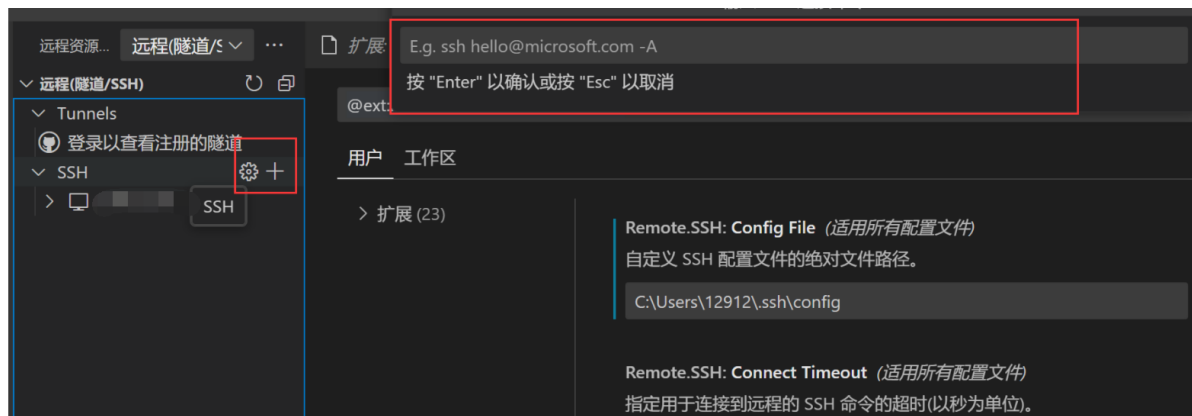
设置ssh配置文件路径：



关于路径地址：如下所示，找到相应文件夹复制文件地址即可。



接下来开始建立远程：

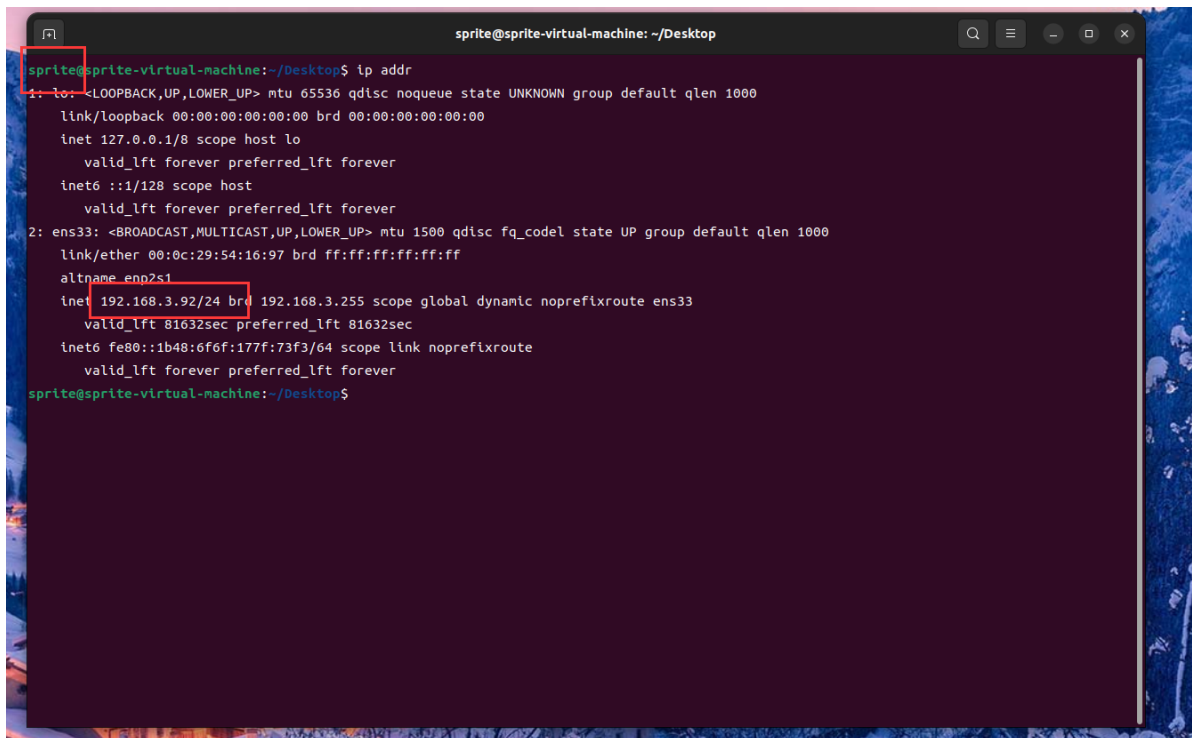


接下来输入ssh+用户名@ip地址

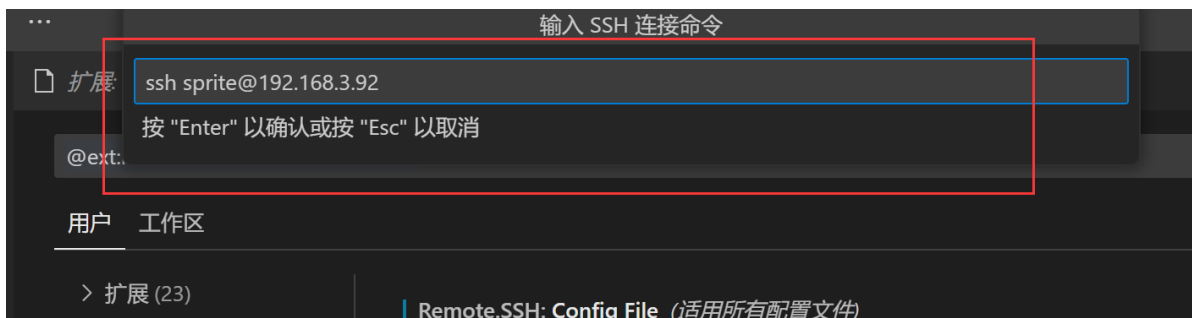
用户名为虚拟机终端@符号前面的。

可以在终端输入以下命令查询：

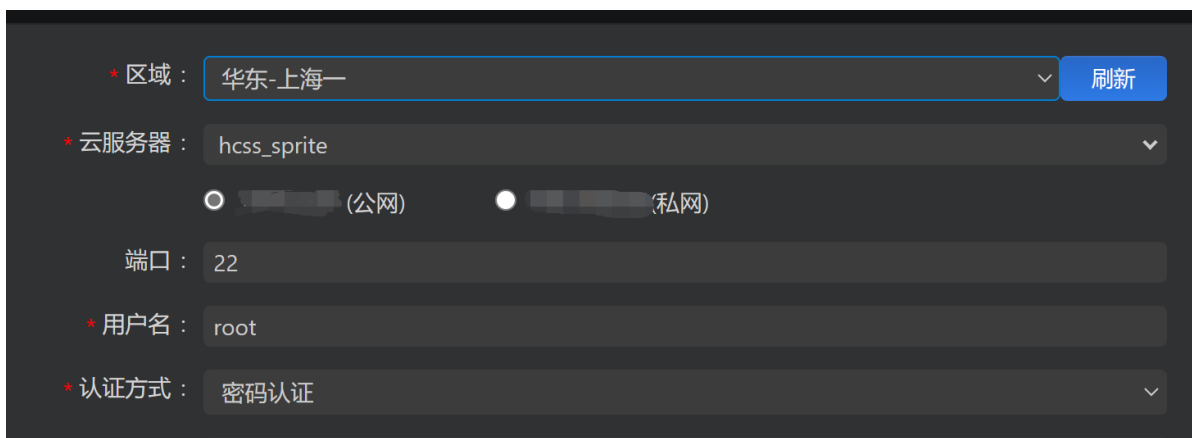
```
ip addr
```



```
sprite@sprite-virtual-machine: ~/Desktop$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:54:16:97 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.3.92/24 brd 192.168.3.255 scope global dynamic noprefixroute ens33
        valid_lft 81632sec preferred_lft 81632sec
    inet6 fe80::1b48:6f6f:177f:73f3/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
sprite@sprite-virtual-machine: ~/Desktop$
```



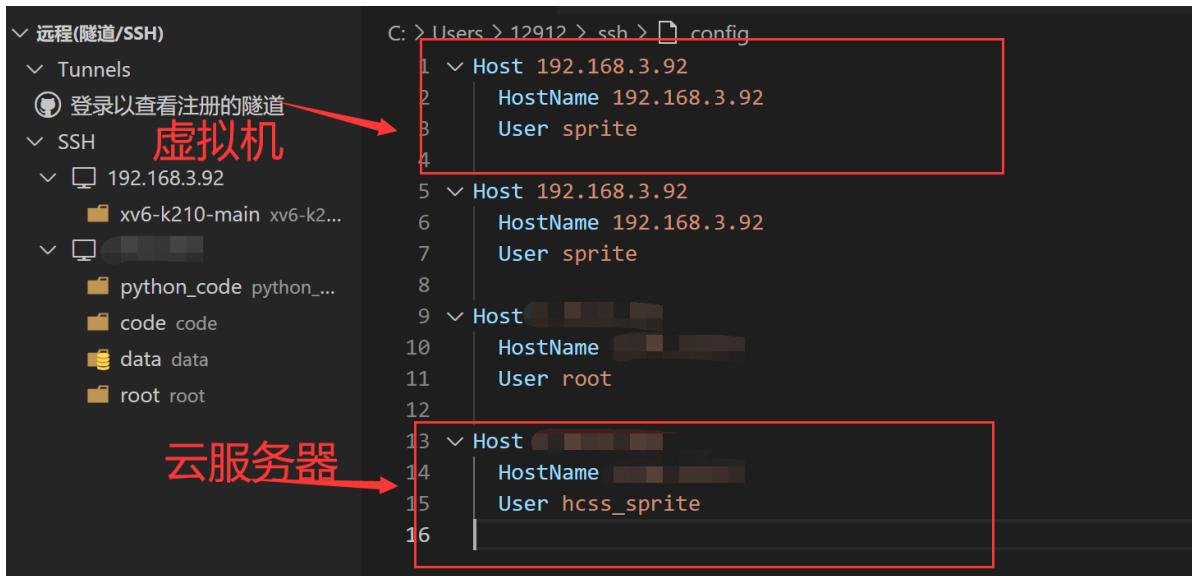
当然云服务器的连接和上面的流程一样，其用户名和ip地址可以进行查询（选用公网ip）：



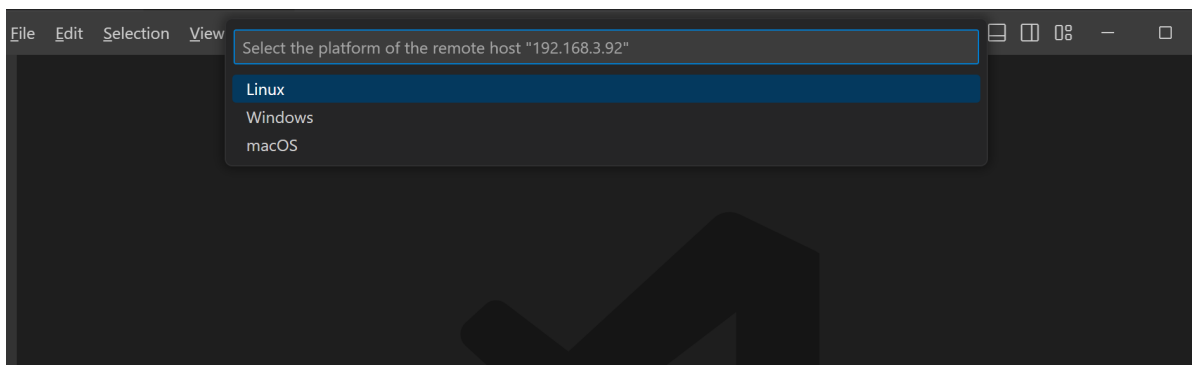
这里不再展示具体流程，还有问题自行查阅资料即可。

完成连接后：

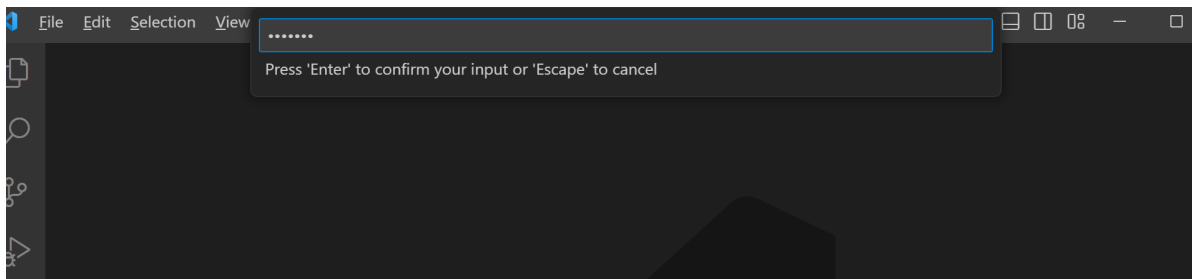




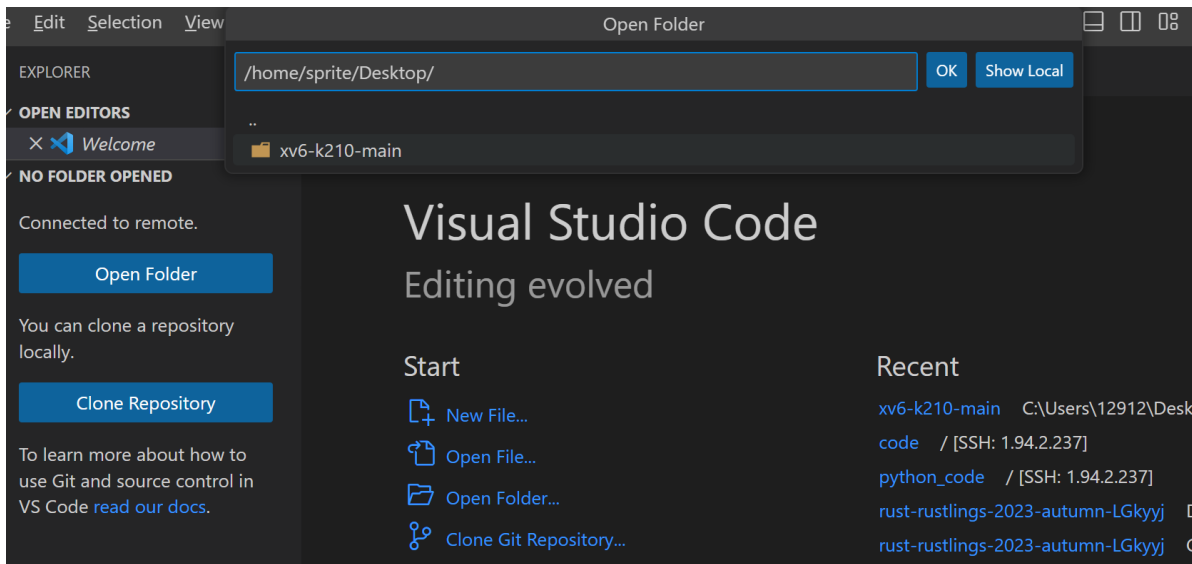
接下来进行配置，选择linux：



输入虚拟机用户密码：



等待下载完成后打开文件夹：



打开之后，快捷键ctrl+~（键盘esc按键下面的按键）打开终端，输入：

```
make run platform=qemu
```

运行成功:

```
sprite@sprite-virtual-machine:~/Desktop/xv6-k210-main$ make run platform=qemu
[rustsbi] RustSBI version 0.1.1

[rustsbi] Platform: QEMU (Version 0.1.0)
[rustsbi] misa: RV64ACDFIMSU
[rustsbi] mideleg: 0x222
[rustsbi] medeleg: 0xb1ab
[rustsbi-dtb] Hart count: cluster0 with 2 cores
[rustsbi] Kernel entry: 0x80200000

hart 0 init done
hart 1 init done
init: starting sh
-> / $
```

注：如果运行过程中发生以下错误：

```
sprite@sprite-virtual-machine:~/Desktop/xv6-k210-main$ make run platform=qemu
qemu-system-riscv64: -device virtio-blk-device,drive=x0,bus=virtio-mmio-bus.0: Failed to get "write" lock
Is another process using the image [fs.img]?
make: *** [Makefile:152: run] 错误 1
```

首先检查虚拟机中终端是否在运行，如果在运行先关掉虚拟机中的终端再会Vscode运行；如果还是不行，重启虚拟机后重新连接再次调试。

## 附录A：Ubuntu扩容

关于Ubuntu的扩容可以参考下面这篇博客：[Ubuntu空间不足，如何扩容（超详细讲解） ubuntu扩容-CSDN博客](#)

可能有同学会发现自己的情况与上面这篇博客有所不同，扩容时发生报错。（比如我的就不是）那么接下来可以参考下面这篇博客：[Ubuntu扩容报错：Unable to resize read-only file system /dev/sda3 the file system can not be resized while it is mou-CSDN博客](#)

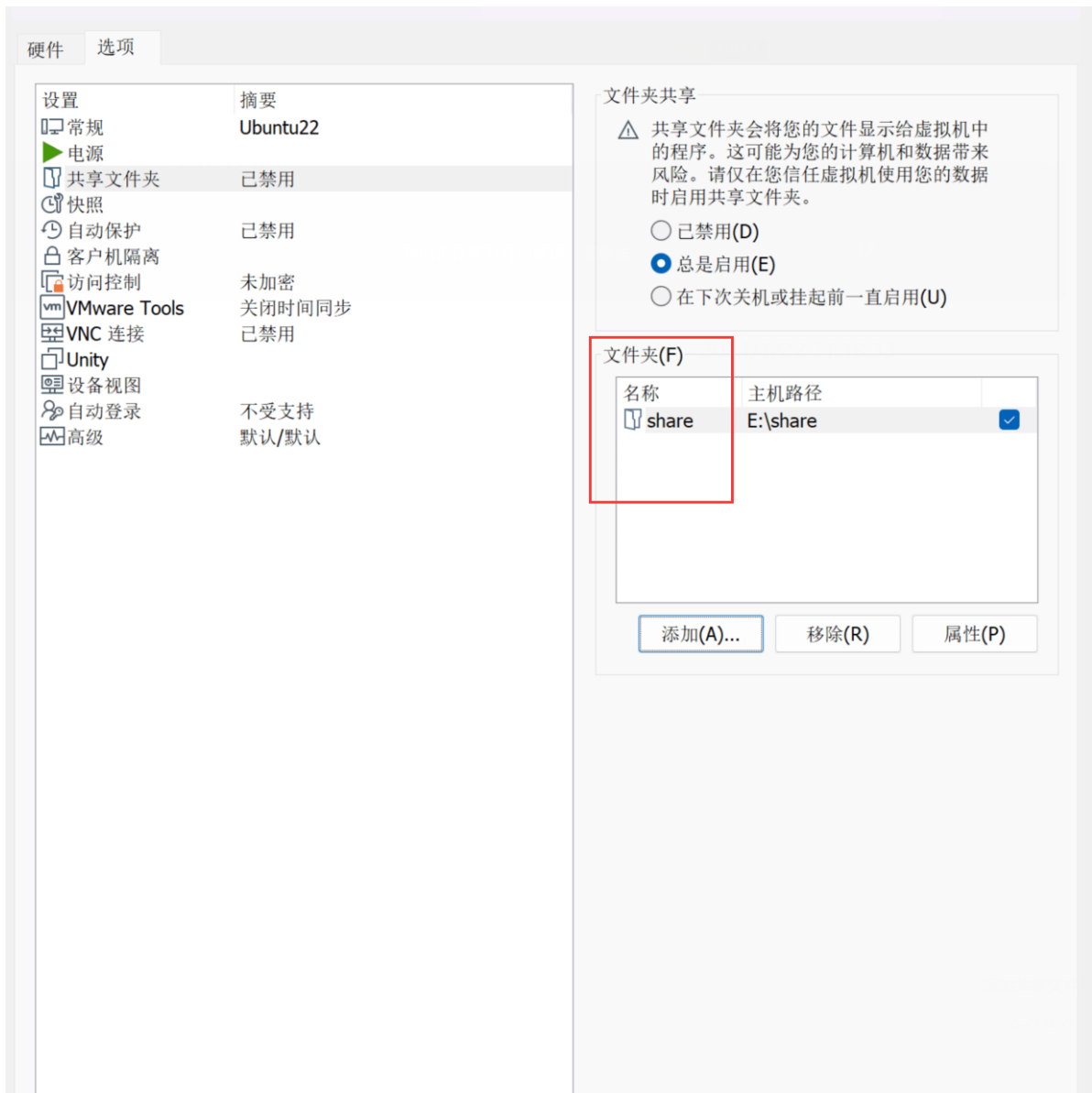
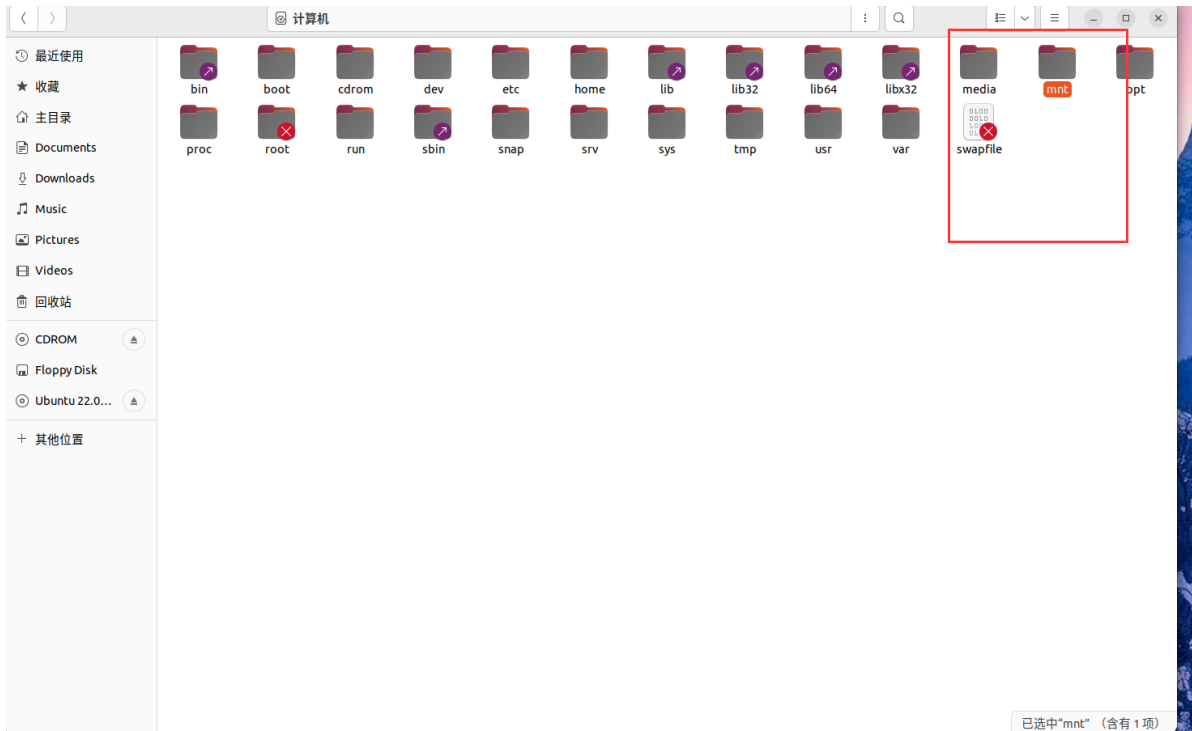
对于我来说，基本上面两篇博客就能解决我的需求，如果还有其他问题请自行查阅资料。

## 附录B：文件互传

windows主机和ubuntu互传文件主要有四种方法，，可以参考如下博客：[windows主机和ubuntu互传文件的4种方法 ubuntu和windows共享文件夹-CSDN博客](#)

在这里我主要用的是第一种方法：通过共享文件夹。

在过程中我也发现了一些问题：Ubuntu找不到共享目录文件夹，/mnt/hgfs下为空。（正常情况共享文件夹在/mnt/hgfs下，比如我的文件夹名为share，则虚拟机内共享文件夹为/mnt/hgfs/share。



确保建立共享目录并启用后，输入

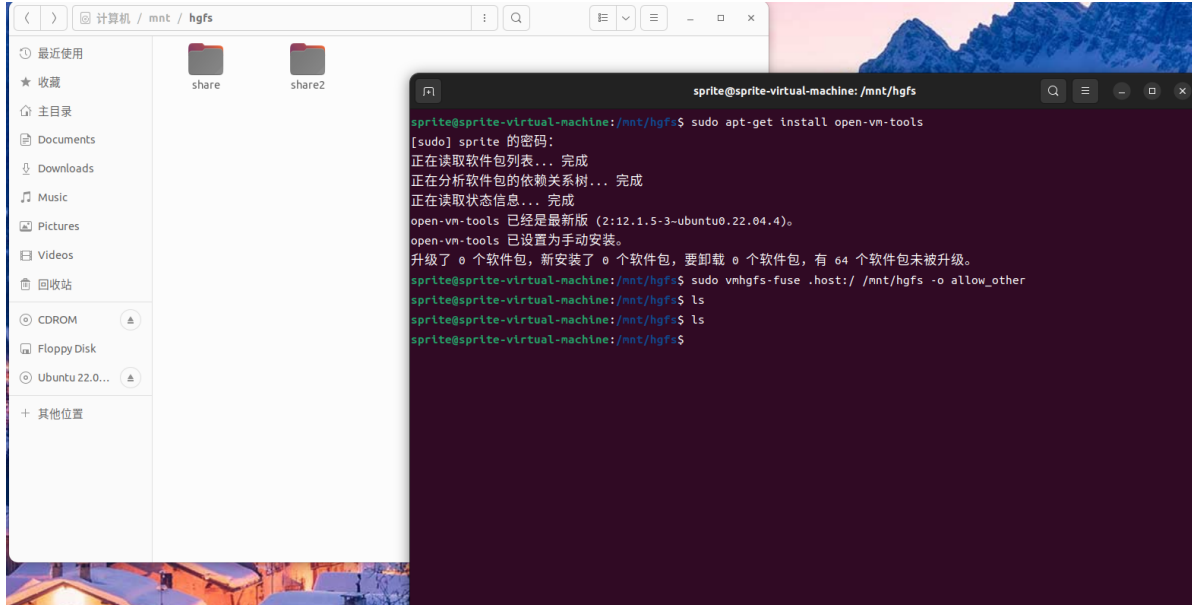
```
cd /mnt/hgfs
ls
```

发现找不到共享目录，原因可能是之前ubuntu重启了，输入

```
sudo apt-get install open-vm-tools
```

```
sudo vmhgfs-fuse .host:/ /mnt/hgfs -o allow_other
或者 sudo vmhgfs-fuse .host:/ /mnt/hgfs -o nonempty -o allow_other
```

一般第一个就行，就因为有人可能第二个会报'-o nonempty'错误。



## 附录C：目录中英文切换

关于Ubuntu16.04目录名称中英文切换方法如下：

Ctrl+Alt+T（默认终端快捷键）打开终端，依次输入以下命令：


1.首先更改系统语言环境为英文

```
export LANG=en_US
```

2.然后更改设置

```
xdg-user-dirs-gtk-update
```

此时会弹出界面，选择带有“update”字样的选项或者是“更新名称”



将标准文件夹更新到当前语言吗？

您已经以一种新语言登入。您可以将主文件夹下的某些标准文件夹名按照新语言进行自动更新。该更新将会更改以下文件夹：

当前文件夹名称	新的文件夹名称
/home/bingwu/Desktop	/home/bingwu/桌面
/home/bingwu/Downloads	/home/bingwu/下载
/home/bingwu/Templates	/home/bingwu/模板
/home/bingwu/Public	/home/bingwu/公共的
/home/bingwu/Documents	/home/bingwu/文档
/home/bingwu/Music	/home/bingwu/音乐
/home/bingwu/Pictures	/home/bingwu/图片
/home/bingwu/Videos	/home/bingwu/视频

请注意，现有内容不会被移动。

☐ 不要再次询问我(D)

如果需要更改回中文语言环境：

```
export LANG=zh_CN.UTF-8
```

再次执行：

```
xdg-user-dirs-gtk-update
```

此时再次弹出界面，选择“保留旧的名称”

## 附录D：Ubuntu用vi编写代码的常用指令

### 常用命令(vi下)

进入之后vi编辑器之后，提供三种模式，第一种普通模式，第二种插入模式，第三种命令行模式。

默认普通模式，按下i进入插入模式，按下：进入命令行模式

进入插入模式按下esc退回普通模式

**插入：**

普通模式下

i: 在光标位置之前输入字符  
a: 在光标位置之后输入字符  
I: 在光标所在行行首输入字符  
A: 在光标所在行行末输入字符  
o: 在光标所在行的上面插入一行  
O: 在光标所在行的下面插入一行  
s: 删除光标后的一个字符，然后进入插入模式；  
S: 删除光标所在的行，然后进入插入模式；

## 复制：

### 普通模式下

yw: 复制当前光标到行末  
nyw: 复制当前光标和光标后n+1个字符  
yy: 复制当前行  
nny: 复制当前光标后n行  
p: 在光标行下一行粘贴  
P: 在光标行上一行粘贴

## 删除：

### 普通模式下

x: 删除当前光标位置的字符  
nx: 删除当前光标位置后n个字符(不含空格)  
dw: 剪切当前光标位置的单词  
ndw: 剪切当前光标位置后n个单词  
d\$: 剪切当前光标位置到行尾的内容  
J: 删除光标所处行与下一行之间空格，让两行连接起来  
dd: 剪切当前行  
nnd: 剪切当前光标位置下n行，包括此行

## 替换：

### 普通模式下

r: 替换光标所在处的字符  
R: 替换光标所在处的字符，直到按下「ESC」键为止  
cw: 删除目标单词，再进入输入模式

## 选中：

### 普通模式下

v: 选中部分代码  
V: 选中行代码  
v ==: 将部分代码对齐  
gg+v+G+=: 所有代码对齐  
选中代码后  
y: 复制  
d: 剪切

## 退出：

## 普通模式下

: 进入命令行模式  
:wq: 保存代码并且退出  
:x: 保存代码并且退出  
shift+zz: 保存并且退出  
:q!: 强制退出不保存代码  
:q: 退出  
:w <文件路径>: 将文件另存为  
:seveas <文件路径>: 将文件另存为

## 常用快捷键(vi下)

### 光标移动

#### 普通模式下

:n: 跳转到第n行  
h或←键: 光标左移一格  
j或↓键: 光标下移一格  
k或↑键: 光标上移一格  
l或→键: 光标右移一格  
n<space>: 光标向右移动n个字符  
n<delete>: 光标向左移动n个字符  
n<enter>: 光标向下移动n行  
O或者HOME键: 移动到光标行第一个字符处  
\$或者END键: 移动到光标行最后一个字符处  
gg: 移动到代码第一行  
G: 移动到代码最后一行  
nG: 移动到代码第n行  
ctrl+f: 屏幕向下翻一页  
ctrl+d: 屏幕向下翻半页  
ctrl+b: 屏幕向上翻一页  
ctrl+u: 屏幕向上翻半页

### 查找与修改

#### 普通模式下

u: 撤销上次操作  
ctr+r或者.: 重做上一个动作  
/word: 光标向下移动到代码中word字符串位置  
?word: 光标向上移动到代码中word字符串位置  
n: 执行完/word之后, 或者?word之后按下n继续执行上次/word或者?word操作  
N: 类似于n操作, 只是执行的是上一次操作的相反操作  
:n1,n2s/word1/word2/g: 在第n1行到第n2行的word1替换为word2  
:1,\$s/word1/word2/g: 在第1行到最后一行的word1替换为word2  
:1,\$s/word1/word2/gc: 在第1行到最后一行的word1替换为word2, 每次取代之前会询问

### 环境设置

#### 普通模式下

```
:set nu: 设置行号
:set nonu
:set autoindent: 自动对齐
:set noautoindent: 不自动对齐
:set all: 显示目前所有环境参数设置
:set: 显示与系统默认值不同的环境参数
```

更多vi下的常用指令请移步至这里:

[Linux--Vim的使用以及快捷键大全 ubuntu linux vim多文件切换快捷-CSDN博客](#)

## 附录E: Ubuntu基本设置及快捷键

参考博客: [Ubuntu基本设置及快捷键 ubuntu 设置快捷键-CSDN博客](#)

## 附录F: Ubuntu系统汉化

参考博文: [Ubuntu系统汉化 ubuntu里面找不到language support-CSDN博客](#)

## 附录G: Ubuntu系统网络连接

参考博文: [Ubuntu系统如何进行网络连接-连接电脑局域网-物联网开发-Ubuntu系统维护 ubuntu虚拟机网络连接-CSDN博客](#)