

**Tên: Trần Minh Trí**

**MSSV: 17142054**

Lớp: TT Hệ Thống Nhúng- sáng thứ 7 tiết 1-5

Link youtube: <https://youtu.be/qFspmPxVjKU>

## **Build Kernel trên Raspberry Pi:**

### **Phần mềm sử dụng:**

+ VMware 16 Pro

+ Ubuntu 18.04.5

### **Bước 1: Cài đặt git bc:**

```
~$ sudo apt-get install git bc
```

### **Bước 2: Cài đặt file có chứa thư mục Linux và mọi thứ có trong linux để làm thư mục chứa nhân (kernel)**

(Thử xem có thư mục đó tồn tại hay chưa bằng lệnh: cd linux)

```
~$ git clone --depth=1 -b rpi-4.9.y https://github.com/raspberrypi/linux
```

```
~$ cd linux
```

### **Bước 3: Cài các tool chain sử dụng cho việc biên dịch chéo**

```
~$ git clone https://github.com/raspberrypi/tools ~/tools
```

=>> Cài thêm 1 số công cụ hỗ trợ cho biên dịch chéo

```
~$ sudo apt install crossbuild-essential-armhf
```

```
~$ sudo apt install crossbuild-essential-arm64
```

#### **Bước 4: Cài các gói và tool phục vụ cho việc build kernel và tạo biến PATH môi trường**

```
~$ sudo apt-get install linux-source
```

```
~$ sudo apt install ncurses-dev kernel-package qt4-dev-tools pkg-config build-essential
```

=>> Cài biến PATH môi trường:

```
~$ export PATH=~/.tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/bin:$PATH
```

```
~$ export TOOLCHAIN=~/.tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/
```

```
~$ export CROSS_COMPILE=arm-linux-gnueabihf-
```

```
~$ export ARCH=arm
```

#### **Bước 5: Cài đặt môi trường và bắt đầu tạo môi trường để set up các driver cần thiết**

```
~$ cd linux/
```

```
~/linux$ make mrproper
```

```
~/linux$ KERNEL=kernel7
```

```
~/linux$ make ARCH=arm bcm2709_defconfig
```

```
~/linux$ make ARCH=arm menuconfig
```

Sau lệnh ở trên sẽ có 1 bảng để hiệu chỉnh những driver cần thiết cho chúng ta chọn. Để chọn gói đó chúng ta sẽ nhấn phím 'y' để tạo dấu \*

- Chúng ta sẽ cần cài các driver như sau:

```

Device drivers >
  [*] SPI support --->
    <*> BCM2835 SPI controller
    <*> User mode SPI device driver support

Device drivers >
  I2C support --->
    I2C Hardware Bus support --->
      <*> Broadcom BCM2835 I2C controller

Device drivers >
  [*] SPI support --->
    <*> User mode SPI device driver support

Device drivers >
  [*] LED Support --->
    <*> LED Class Support
    -* LED Trigger support --->
      <*> LED Timer Trigger
      <*> LED Heartbeat Trigger

Device drivers >
  <*> Industrial I/O support --->
    -* Enable buffer support within IIO
    -* Industrial I/O buffering based on kfifo
    <*> Enable IIO configuration via configs
    -* Enable triggered sampling support
    <*> Enable software IIO device support
    <*> Enable software triggers support
      Triggers - standalone --->
        <*> High resolution timer trigger
        <*> SYSFS trigger

--

Device drivers >
  <*> Userspace I/O drivers --->
    <*> Userspace I/O platform driver with generic IRQ handling
    <*> Userspace platform driver with generic irq and dynamic memory

Device drivers >
  Input device support --->
    -* Generic input layer (needed for keyboard, mouse, ...)
    <*> Polled input device skeleton
    <*> Event interface

```

Sau khi chọn cài các driver ở trên ta save và exit ra ngoài. Sau đó chúng ta sẽ build các kernel đã chọn bằng lệnh:

```
~$ make -j4
```

/\* Quá trình này sẽ khá lâu nên các bạn cứ để cho nó chạy đừng tắt nhé. \*/

**Bước 6: Vào thư mục code và chỉnh file “Make file” có sẵn thành :**



The screenshot shows a text editor window titled "Makefile" with the path "~/linux\_4.9\_rpi\_drivers". The file contains a list of object files and a Makefile structure. The object files are listed as follows:

```
obj-m += helloworld_rpi.o helloworld_rpi_with_parameters.o helloworld_rpi_with_timing.o
#obj-m += helloworld_rpi_char_driver.o helloworld_rpi_class_driver.o misc_rpi_driver.o
#obj-m += hellokeys_rpi.o ledRGB_rpi_platform.o ledRGB_rpi_class_platform.o led_rpi_UIO_platform.o
#obj-m += io_rpi_expander.o ltc3206_rpi_led_class.o
#obj-m += int_rpi_key.o int_rpi_key_wait.o keyed_rpi_class.o
#obj-m += linkedlist_rpi_platform.o
#obj-m += sdma_rpi_m2m.o
#obj-m += i2c_rpi_accel.o adxl345_rpi.o
#obj-m += ltc2607_rpi_dual_device.o ltc2422_rpi_dual.o ltc2422_rpi_trigger.o
#obj-m += adxl345_rpi_iio.o
```

The Makefile structure is as follows:

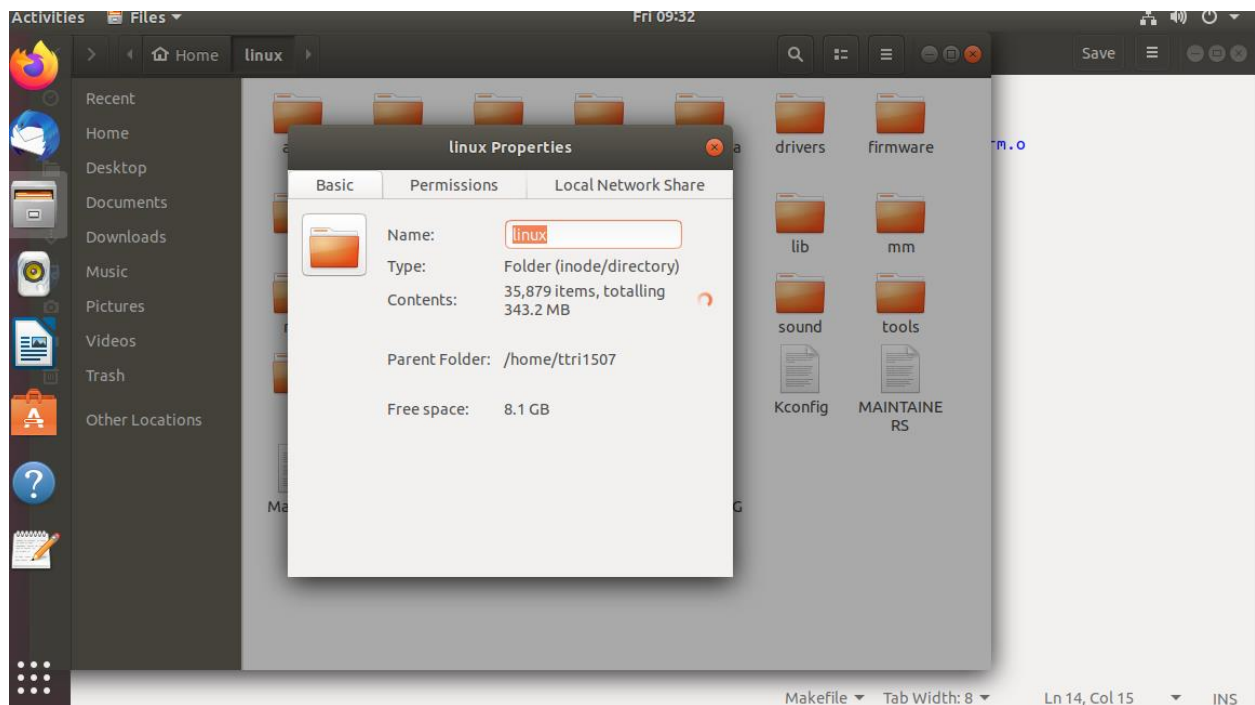
```
KERNEL_DIR ?= /home/ttri1507/linux

all:
    make -C $(KERNEL_DIR) \
        ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- \
        SUBDIRS=$(PWD) modules

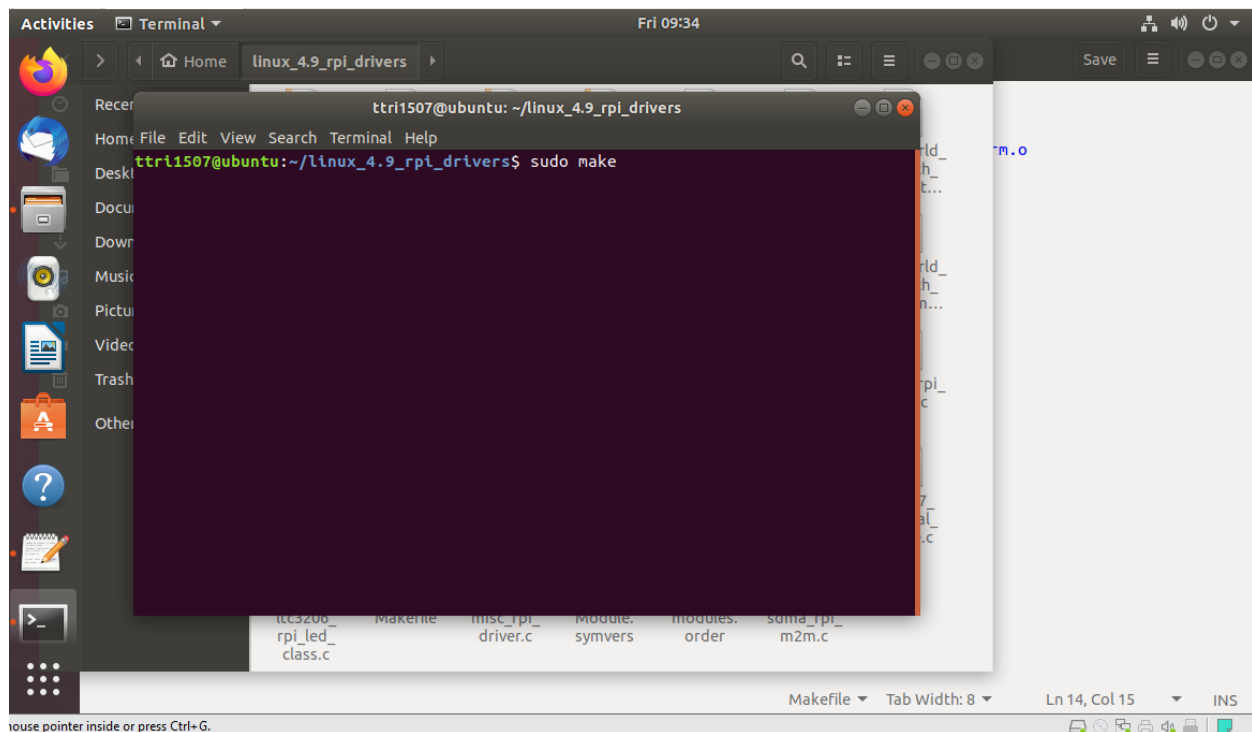
clean:
    make -C $(KERNEL_DIR) \
        ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- \
        SUBDIRS=$(PWD) clean

deploy:
    scp *.ko root@10.0.0.10:
```

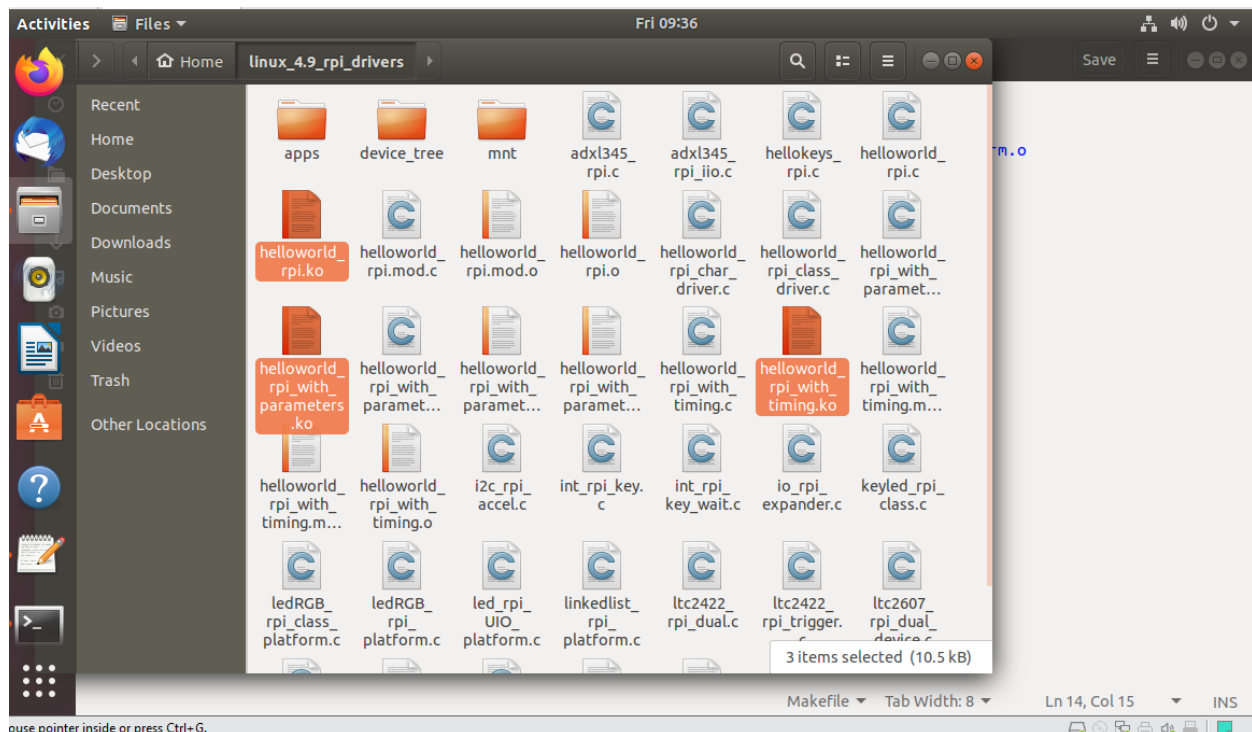
Với ký tự bôi đen là đường dẫn vào thư mục linux của bạn:



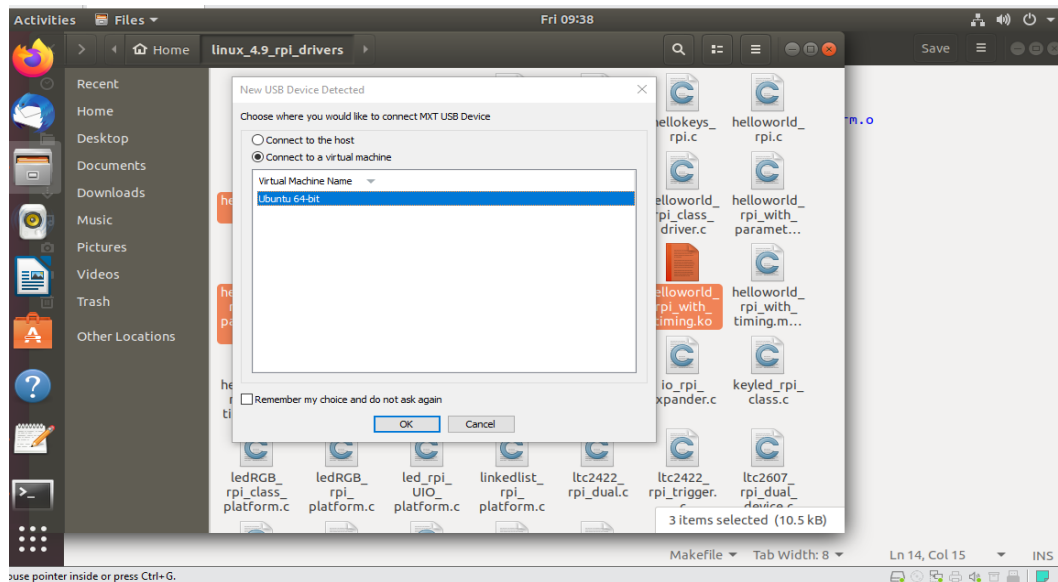
### Bước 7: Vào thư mục code và mở terminal lên, gõ lệnh `~$ sudo make`



Ở đây các file .o và được chọn để make file sẽ biến thành file .ko là file chạy chứa các driver trên chip BCM2708. Đến đây nếu ra được file .ko chúng ta đã thành công trong việc build kernel.



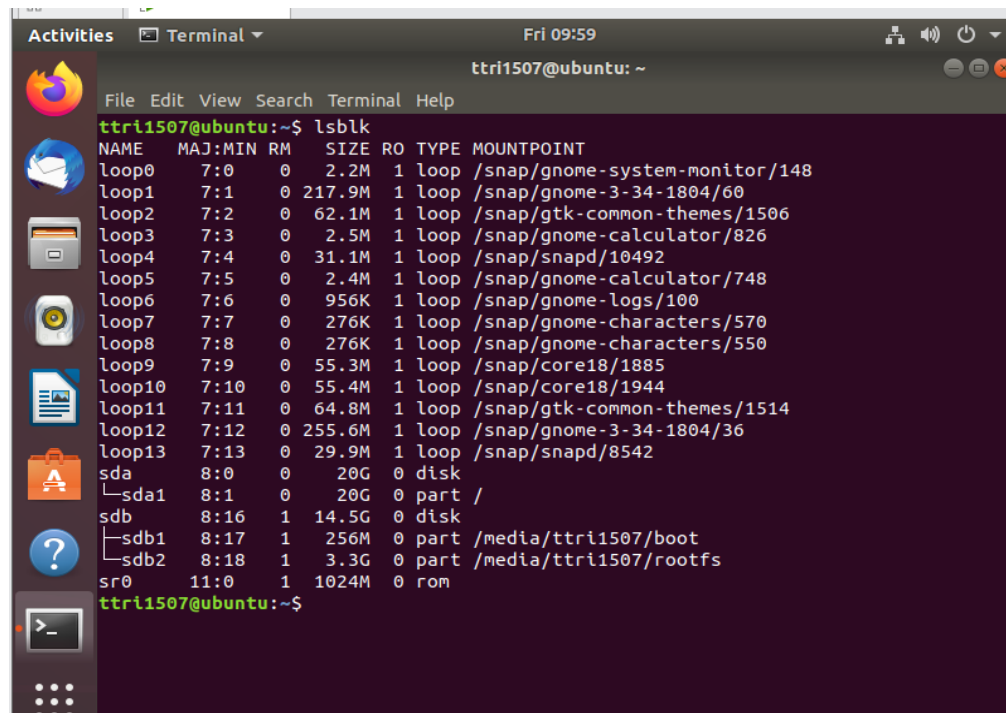
**Bước 8: Dùng thẻ nhớ cắm vào đầu đọc thẻ. Dùng các lệnh để chép dữ liệu từ driver đã lưu sang thẻ để đưa vào Raspberry Pi. Khi cắm xong nhớ chọn máy ảo (Ubuntu) của chúng ta để sao chép dữ liệu sang:**



**Dùng các lệnh:**

`~$ lsblk`

ở lệnh này sẽ hiện cho ta 1 thư mục để boot và 1 thư mục để root. (Như mình là sdb1 và sdb2).



**Tiếp theo chúng ta đánh các lệnh:**

```
~$ mkdir ~/mnt  
~$ mkdir ~/mnt/fat32  
~$ mkdir ~/mnt/ext4  
~$ sudo mount /dev/sdb1 ~/mnt/fat32  
~$ sudo mount /dev/sdb2 ~/mnt/ext4  
~$ ls -l ~/mnt/fat32/
```

**Sau đó chúng ta cần update config của thẻ nhớ bằng lệnh:**

```
~$ cd mnt/fat32/  
~/mnt/fat32$ sudo gedit config.txt  
Ở đây hiện ra 1 bảng. chúng ta nhập:
```

```
dtparam=i2c_arm=on  
dtparam=spi=on  
dtoverlay=spi0-cs  
# Enable UART  
enable_uart=1  
kernel=kernel-rpi.img  
device_tree=bcm2710-rpi-3-b.dtb
```

**Tiếp đến chúng ta update kernel, device tree files and modules:**

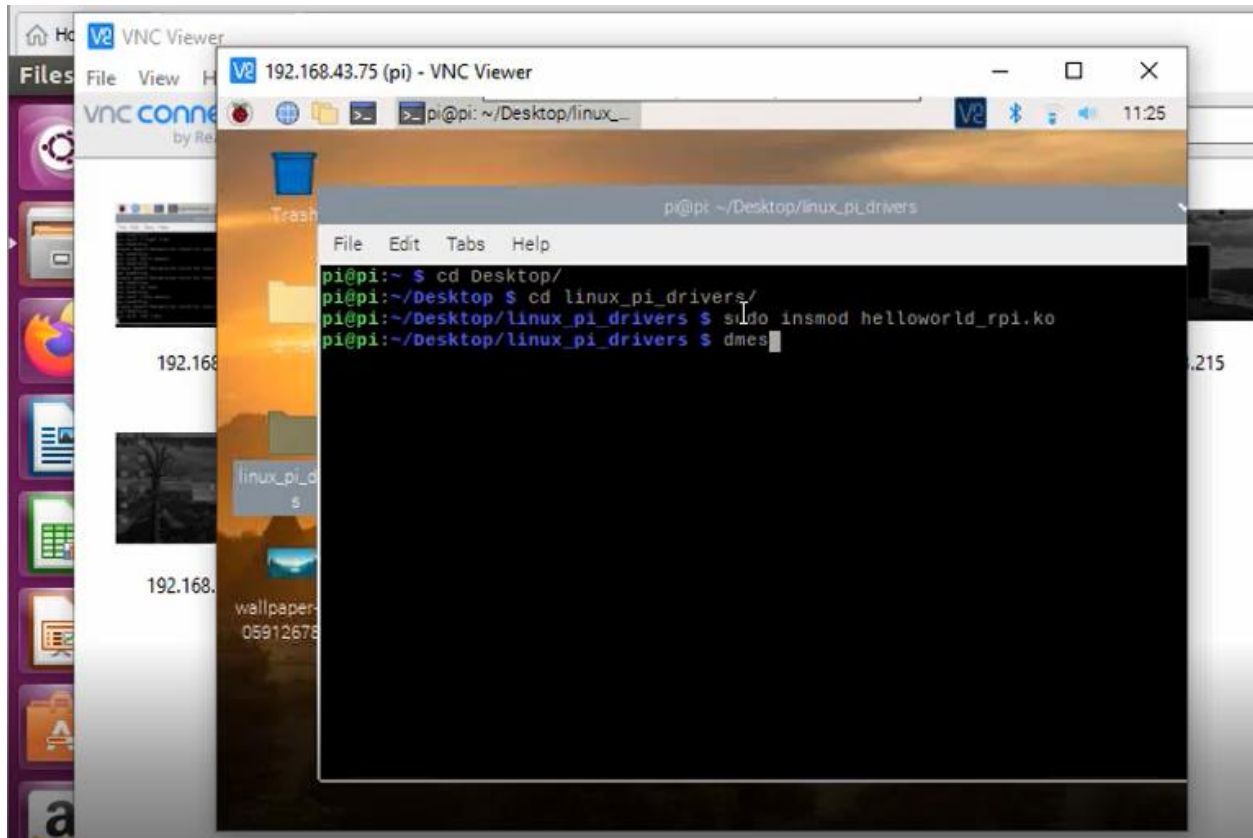
```
~/linux$ sudo cp arch/arm/boot/zImage ~/mnt/fat32/kernel-rpi.img  
~/linux$ sudo cp arch/arm/boot/dts/*.dtb ~/mnt/fat32/  
~/linux$ sudo cp arch/arm/boot/dts/overlays/*.dtb* ~/mnt/fat32/overlays/  
~/linux$ sudo cp arch/arm/boot/dts/overlays/README ~/mnt/fat32/overlays/  
~/linux$ sudo make ARCH=arm INSTALL_MOD_PATH=~/mnt/ext4 modules_install
```

**Cuối cùng chúng ta Unmount dữ liệu và bắt đầu chuyển sang Raspberry pi để chạy kernel đã build.**

```
~$ sudo umount ~/mnt/fat32
```

```
~$ sudo umount ~/mnt/ext4
```

**Bước 9. Sang Pi truyền file .ko sang và chạy:**



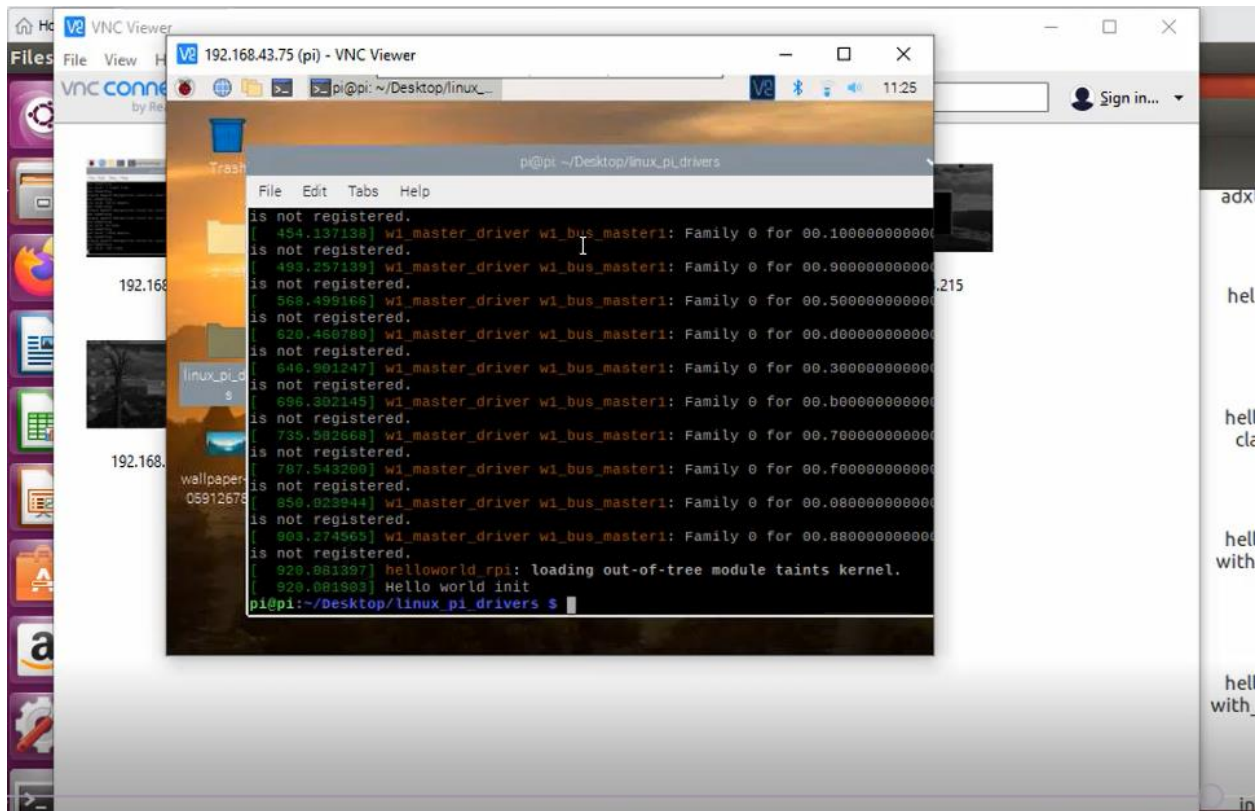
Chuyển đường dẫn vào thư mục chứa file .ko sử dụng lệnh:

```
~$ sudo insmod tenfile.ko
```

```
~$ dmesg
```

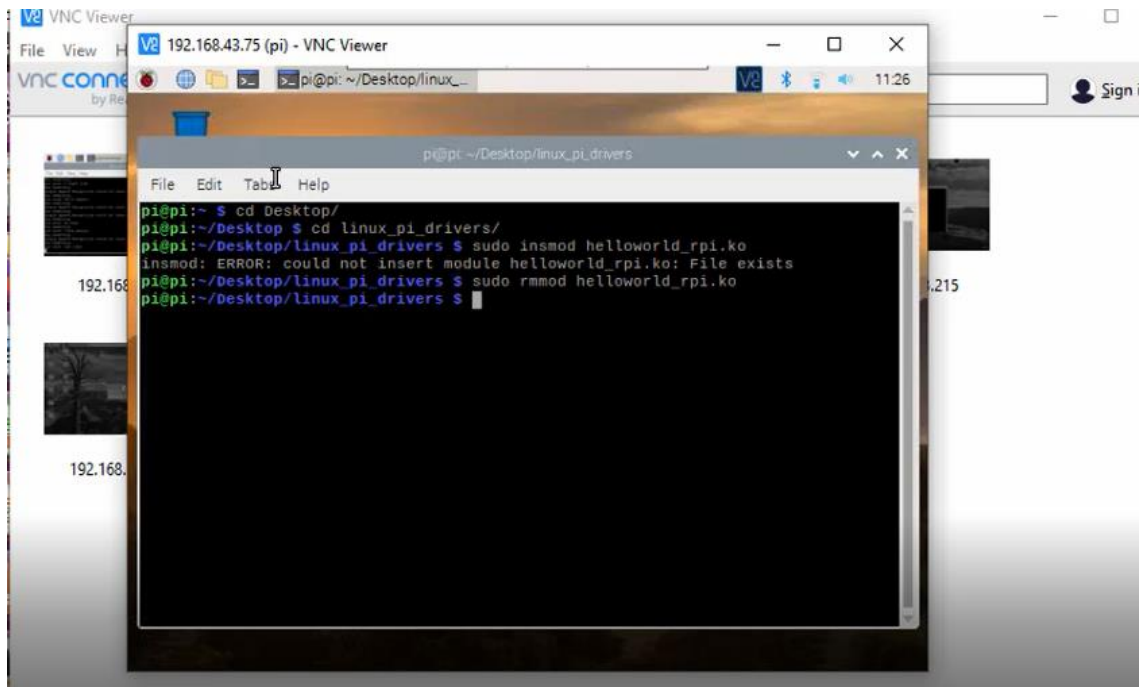


Ở đây chúng ta sẽ chạy 1 file chứa driver cho ra được kết quả:



```
pi@pi: ~/Desktop/linux_pi_drivers
File Edit Tabs Help
is not registered.
[ 454.137138] w1_master_driver w1_bus_master1: Family 0 for 00.100000000000
is not registered.
[ 493.257139] w1_master_driver w1_bus_master1: Family 0 for 00.900000000000
is not registered.
[ 568.499166] w1_master_driver w1_bus_master1: Family 0 for 00.500000000000
is not registered.
[ 620.460780] w1_master_driver w1_bus_master1: Family 0 for 00.d00000000000
is not registered.
[ 646.901247] w1_master_driver w1_bus_master1: Family 0 for 00.300000000000
is not registered.
[ 696.382145] w1_master_driver w1_bus_master1: Family 0 for 00.b00000000000
is not registered.
[ 735.582668] w1_master_driver w1_bus_master1: Family 0 for 00.700000000000
is not registered.
[ 787.543200] w1_master_driver w1_bus_master1: Family 0 for 00.f00000000000
is not registered.
[ 850.023944] w1_master_driver w1_bus_master1: Family 0 for 00.000000000000
is not registered.
[ 883.274565] w1_master_driver w1_bus_master1: Family 0 for 00.880000000000
is not registered.
[ 920.081397] helloworld_rpi: loading out-of-tree module taints kernel.
[ 920.081903] Hello world init
pi@pi:~/Desktop/linux_pi_drivers $
```

Vì nó insmod chỉ 1 lần nên dùng lệnh insmod lại sẽ không được. các bạn có thể dùng lệnh để xóa rồi insmod lại file khác bằng cách:



```
pi@pi:~ $ cd Desktop/
pi@pi:~/Desktop $ cd linux_pi_drivers/
pi@pi:~/Desktop/linux_pi_drivers $ sudo insmod helloworld_rpi.ko
insmod: ERROR: could not insert module helloworld_rpi.ko: File exists
pi@pi:~/Desktop/linux_pi_drivers $ sudo rmmod helloworld_rpi.ko
pi@pi:~/Desktop/linux_pi_drivers $
```

```
~$ sudo rmmod tenfile.ko
```

Sau đó các bạn có thể chạy bất kì 1 file kernel nào khác bằng lệnh `sudo insmod tenfilekhac.ko`

---

Đến đây phần build kernel cho Raspberry Pi đã hết. Mong rằng có thể giúp các bạn build 1 kernel thành công.