

# Raspberry Pi 3

## 簡易Linux系統實驗模組建置

開發學生：賴郁文

開發教師：陳鵬升

國立中正大學 資訊工程學系

# 目錄

目錄.....	1
實驗目的.....	2
實驗器材.....	2
實驗所需軟體 .....	3
Part 1: 取得 Buildroot source code .....	3
Part 2: make config (Kernel image & Bootloader) .....	3
Step 1: 進入 buildroot 資料夾內.....	3
Step 2: 確認該版本 buildroot 是否支援 Raspberry Pi 3.....	3
Step 3: make config .....	3
Step 4: 藉由 buildroot 提供的 makefile 下載所有需要的套件.....	4
Part 3: make Kconfig (Root filesystem) .....	4
Step 1: make Kconfig.....	4
Step 2: 設定 menuconfig 以編譯並打包出 root filesystem .....	4
Step 3: Save & Exit 儲存 Kconfig 之設定.....	5
Part 4: 編譯 Buildroot .....	5
Part 5: 切割 SD card.....	5
Step 1: 進入 output/images 路徑.....	5
Step 2: 掛載由 buildroot 產生的 sdcard.img 來切割 sdcard.....	5
Part 6: 將建置好的系統檔案放入切割過的 SD card 內.....	6
Step 1: 建立資料夾以便掛載使用 .....	6
Step 2: 分別掛載 mmc1 與 mmc2 至 BOOT 與 FILESYSTEM .....	6
Step 3: 將系統檔案放入 BOOT .....	6
Step 4: 將 root filesystem.gz/bz2 放入 FILESYSTEM 並解壓縮.....	6
Part 7: 開機測試.....	7

# 實驗目的

Raspberry Pi為當今熱門的單板電腦之一，為了能充分了解Linux系統相關元件如何相互作用於Raspberry Pi上。我們藉由使用Open-source編譯出最簡易的Linux元件，如：bootloader、kernel image、與root filesystem等，將其組合並實作出能在Raspberry Pi 3上執行的簡易Linux系統來達成學習目的。

# 實驗器材

1. Raspberry Pi 3 model B 一台
2. 8G SD card 一張
3. HDMI to VGA 一條
4. 個人電腦 based on Linux OS 一台
5. Ubuntu 12.04 (含)以上

# 實驗所需軟體

## 1. buildroot

### Part 1: 取得 Buildroot source code

- `git clone git://git.busybox.net/buildroot`

### Part 2: make config (Kernel image & Bootloader)

Step 1: 進入buildroot資料夾內

- `cd buildroot/`

Step 2: 確認該版本buildroot是否支援Raspberry Pi 3

- `ls configs/raspberry*`

可以找到：

`configs/raspberrypi0_defconfig`

`configs/raspberrypi2_defconfig` (for Raspberry Pi 2)

`configs/raspberrypi3_64_defconfig`

**Target→**`configs/raspberrypi3_defconfig`

`configs/raspberry3_qt5we_defconfig`

`configs/raspberry_defconfig` (for Raspberry Pi A/B A+/B+ boards)

```
configs/raspberrypi0_defconfig  configs/raspberrypi3_defconfig
configs/raspberrypi2_defconfig  configs/raspberrypi3_qt5we_defconfig
configs/raspberrypi3_64_defconfig  configs/raspberrypi_defconfig
```

Step 3: make config

- `make raspberrypi3_defconfig`

Step 4: 藉由buildroot提供的makefile下載所有需要的套件

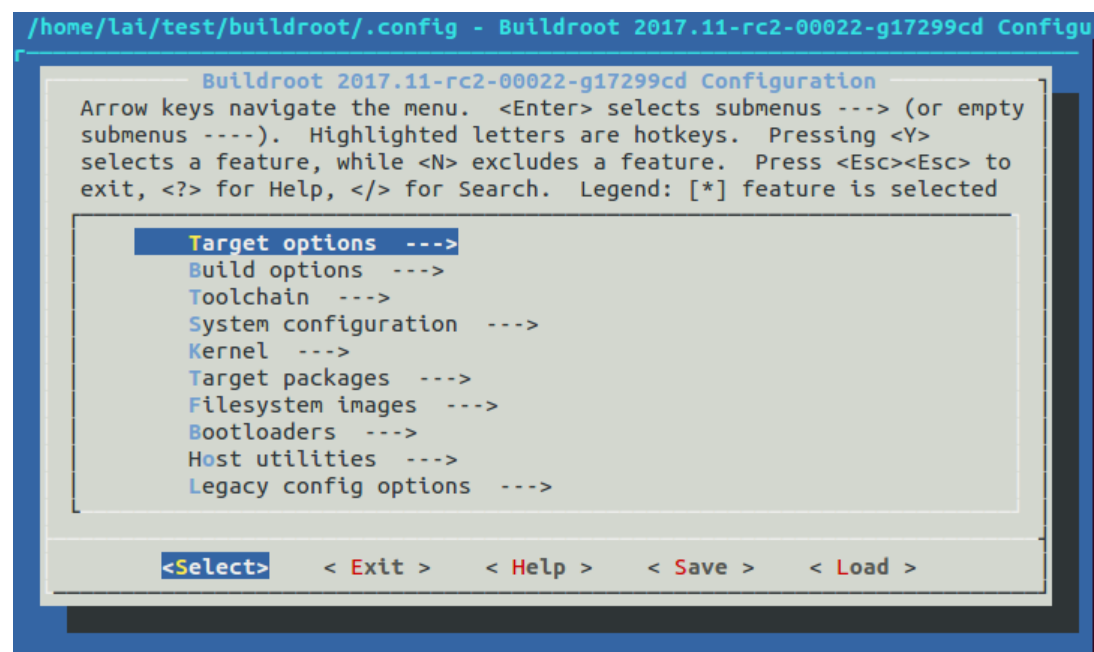
- `make source`

## Part 3: make Kconfig (Root filesystem)

buildroot 有支援 Kconfig 可供操作:

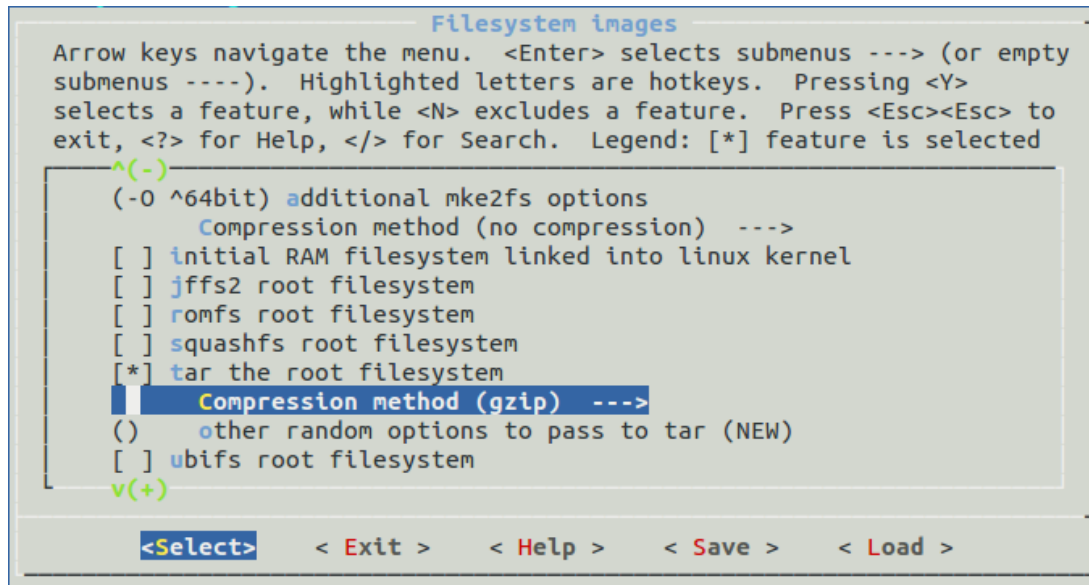
Step 1: make Kconfig

- `make menuconfig`



Step 2: 設定 menuconfig 以編譯並打包出 root filesystem

- Filesystem images --->
  - tar the root filesystem
  - Compression method( ) --->
    - gzip 或 bzip2 皆可



Step 3: Save & Exit 儲存 Kconfig 之設定

## Part 4: 編譯 Buildroot

設定完畢所有設定值後，使用 make 指令編譯出所有目標系統檔案

- `make -j4` (run with 4 cores)

## Part 5: 切割 SD card

Step 1: 進入output/images路徑

- `cd buildroot/output/images`

Step 2: 掛載由buildroot產生的sdcard.img來切割sdcard

- `sudo dd if=sdcard.img of=/dev/sdx`

註：可由下列指令觀察sdcard編號：

- `df -a`

若sdcard為/dev/sdd，則sdx為sdd

```
/dev/sdd2          55397    54911         0 100% /media/79290c31-5
9-8b7ba0be66e8
/dev/sdd1          32686     7670    25016   24% /media/5CB9-F0E5
```

## Part 6: 將建置好的系統檔案放入切割過的 SD card 內

Step 1: 建立資料夾以便掛載使用

- `cd ~`
- `mkdir mmc1 mmc2`

Step 2: 分別掛載 mmc1 與 mmc2 至 BOOT 與 FILESYSTEM

- `sudo mount /dev/sdd1 ~/mmc1`
- `sudo mount /dev/sdd2 ~/mmc2`

Step 3: 將系統檔案放入 BOOT

- `cd buildroot/output/images`
- `sudo cp -rf bcm2710-rpi-3-b.dtb zImage ~/mmc1`
- `sudo cp -rf buildroot/output/images/rpi_firmware/* ~/mmc1`

- **bcm2710-rpi-3-b: device tree blob**
- **zImage: kernel image**
- **bootcode.bin: second stage bootloader**

Step 4: 將 root filesystem.gz/bz2 放入 FILESYSTEM 並解壓縮

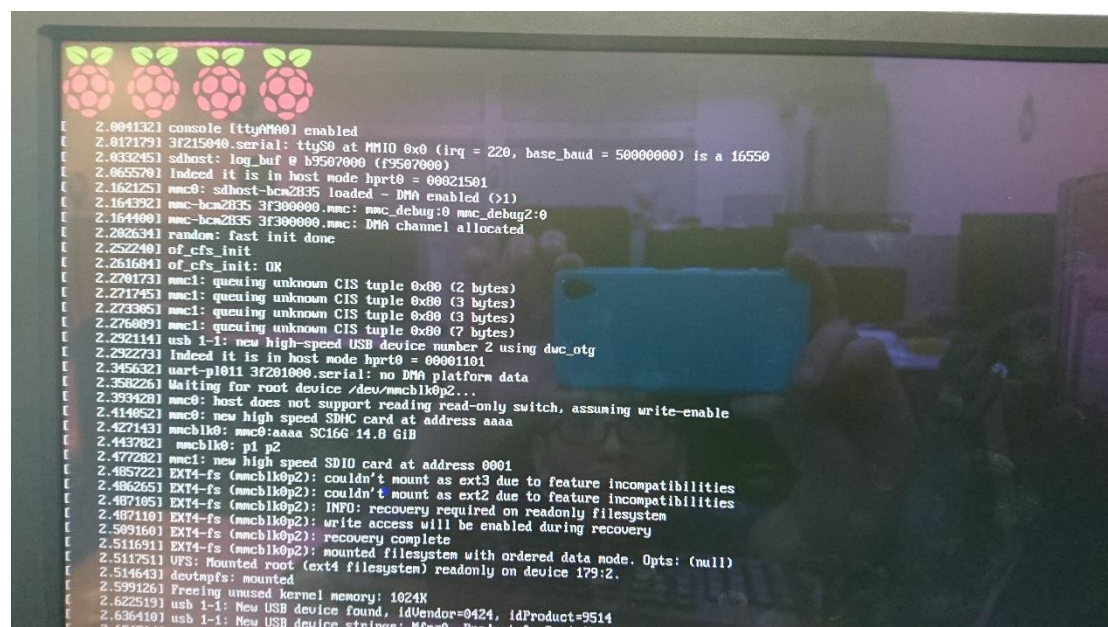
- `sudo cp -rf buildroot/output/images/rootfs.tar.gz ~/mmc2`  
OR  
`sudo cp -rf buildroot/output/images/rootfs.tar.bz2 ~/mmc2`
- `cd ~/mmc2`
- `sudo tar -zxvf rootfs.tar.gz` OR `sudo tar -jxvf rootfs.tar.bz2`
- `sudo rm rootfs.tar.gz` OR `sudo rm rootfs.tar.bz2`

## Part 7: 開機測試

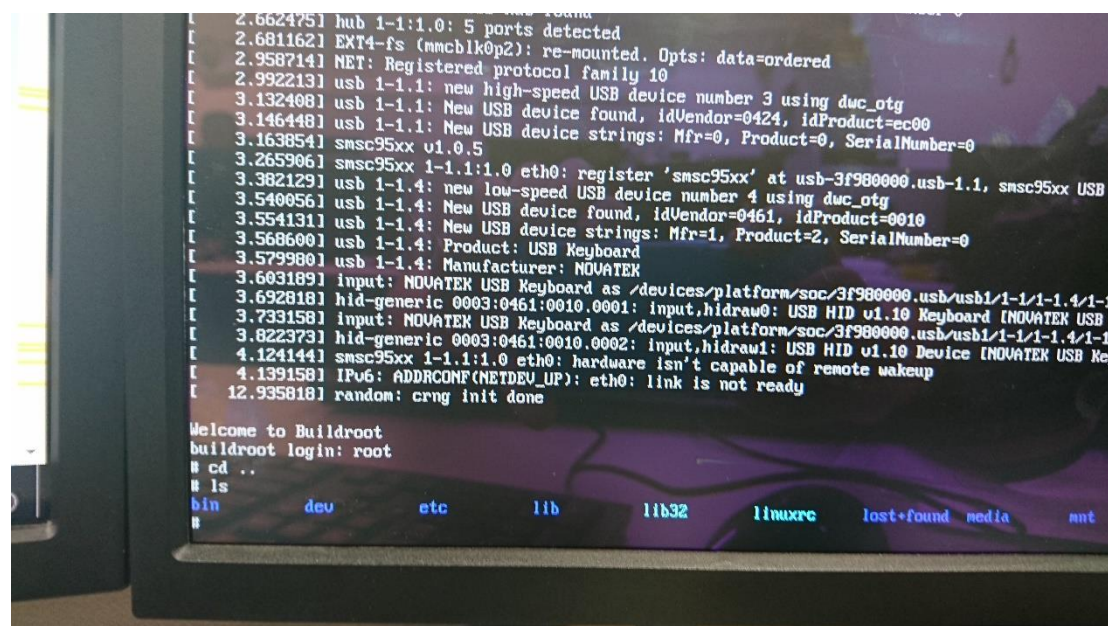
此最簡易 Linux 系統只有 root，login 處鍵入 root 即可取得 root 權限

- `cd ..`
- `ls`

即可看到整個 root filesystem



```
2.004132] console [ttyAMA0] enabled
2.017129] 3f215040.serial: ttyS0 at MMIO 0x0 (irq = 220, base_baud = 5000000) is a 16550
2.033245] sdhost: log_buf @ b9507000 (f9507000)
2.065570] Indeed it is in host mode hprt0 = 00021501
2.162125] mmc0: sdhost-bcm2035 3f300000.nmc: DMA enabled (>1)
2.164392] mmc-bcm2035 3f300000.nmc: mmc_debug:0 mmc_debug2:0
2.164400] random: fast init done
2.202634] of_cfs_init
2.252240] of_cfs_init: OK
2.261604] mmc1: queuing unknown CIS tuple 0x00 (2 bytes)
2.271745] mmc1: queuing unknown CIS tuple 0x00 (3 bytes)
2.273395] mmc1: queuing unknown CIS tuple 0x00 (3 bytes)
2.276091] mmc1: queuing unknown CIS tuple 0x00 (7 bytes)
2.292141] usb 1-1: new high-speed USB device number 2 using dwc_otg
2.292273] Indeed it is in host mode hprt0 = 00001101
2.345632] uart-pl011 3f201000.serial: no DMA platform data
2.350225] Waiting for root device /dev/mmcblk0p2...
2.393420] mmc0: host does not support reading read-only switch, assuming write-enable
2.414052] mmc0: new high speed SDHC card at address aaaa
2.427143] mmcblk0: mmc0:aaaa SC16G 14.8 GiB
2.443702] mmcblk0: p1 p2
2.477282] mmc1: new high speed SDIO card at address 0001
2.485722] EXT4-fs (mmcblk0p2): couldn't mount as ext3 due to feature incompatibilities
2.486265] EXT4-fs (mmcblk0p2): couldn't mount as ext2 due to feature incompatibilities
2.487105] EXT4-fs (mmcblk0p2): INFO: recovery required on readonly filesystem
2.487110] EXT4-fs (mmcblk0p2): write access will be enabled during recovery
2.509160] EXT4-fs (mmcblk0p2): recovery complete
2.511691] EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode. Opts: (null)
2.511751] VFS: Mounted root (ext4 filesystem) readonly on device 179:2.
2.514643] devtmpfs: mounted
2.599126] Freeing unused kernel memory: 1024K
2.622519] usb 1-1: New USB device found, idVendor=0424, idProduct=9514
2.636410] usb 1-1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
2.651564] hub 1-1:1.0: USB found
```



```
2.662475] hub 1-1:1.0: 5 ports detected
2.681162] EXT4-fs (mmcblk0p2): re-mounted. Opts: data=ordered
2.958714] NET: Registered protocol family 10
2.992213] usb 1-1.1: new high-speed USB device number 3 using dwc_otg
3.132400] usb 1-1.1: New USB device found, idVendor=0424, idProduct=ec00
3.146440] usb 1-1.1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
3.163854] smsc95xx v1.0.5
3.265906] smsc95xx 1-1.1:1.0 eth0: register 'smc95xx' at usb-3f980000.usb-1.1, smc95xx USB 2
3.382129] usb 1-1.4: new low-speed USB device number 4 using dwc_otg
3.540056] usb 1-1.4: New USB device found, idVendor=0461, idProduct=0010
3.554131] usb 1-1.4: New USB device strings: Mfr=1, Product=2, SerialNumber=0
3.568600] usb 1-1.4: Product: USB Keyboard
3.579980] usb 1-1.4: Manufacturer: NOVATEK
3.603189] input: NOVATEK USB Keyboard as /devices/platform/soc/3f980000.usb/usb1/1-1/1-1.4/1-1.4.1/1-1.4.1.1/input/input0
3.692818] hid-generic 0003:0461:0010.0001: input,hidraw0: USB HID v1.10 Keyboard [NOVATEK USB Keyboard]
3.733158] input: NOVATEK USB Keyboard as /devices/platform/soc/3f980000.usb/usb1/1-1/1-1.4/1-1.4.1/1-1.4.1.1/hid/hidraw1
3.822373] hid-generic 0003:0461:0010.0002: input,hidraw1: USB HID v1.10 Device [NOVATEK USB Keyboard]
4.124144] smc95xx 1-1.1:1.0 eth0: hardware isn't capable of remote wakeup
4.139158] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
12.935818] random: crng init done

Welcome to Buildroot
buildroot login: root
# cd ..
# ls
bin      dev      etc      lib      lib32    linuxrc  lost+found  media  mnt
```