Bootloader & Kernel

Bootloader

Bootloader (부트로더)

✓ **부트로더(Bootloader)란**

운영체제가 시동되기 이전에 미리 실행되면서 커널이 올바르게 시동되기 위해 필요한 모든 관련 작업을 마무리하고 최종적으로 운영 체제를 시동시키기 위한 목적을 가진 프로그램을 뜻 한다.

Bootloader (부트로더)

✓ 부트로더(Bootloader)란

운영체제가 시동되기 이전에 미리 실행되면서 커널이 올바르게 시동되기 위해 필요한 모든 관련 작업을 마무리하고 최종적으로 운영 체제를 시동시키기 위한 목적을 가진 프로그램을 뜻 한다.

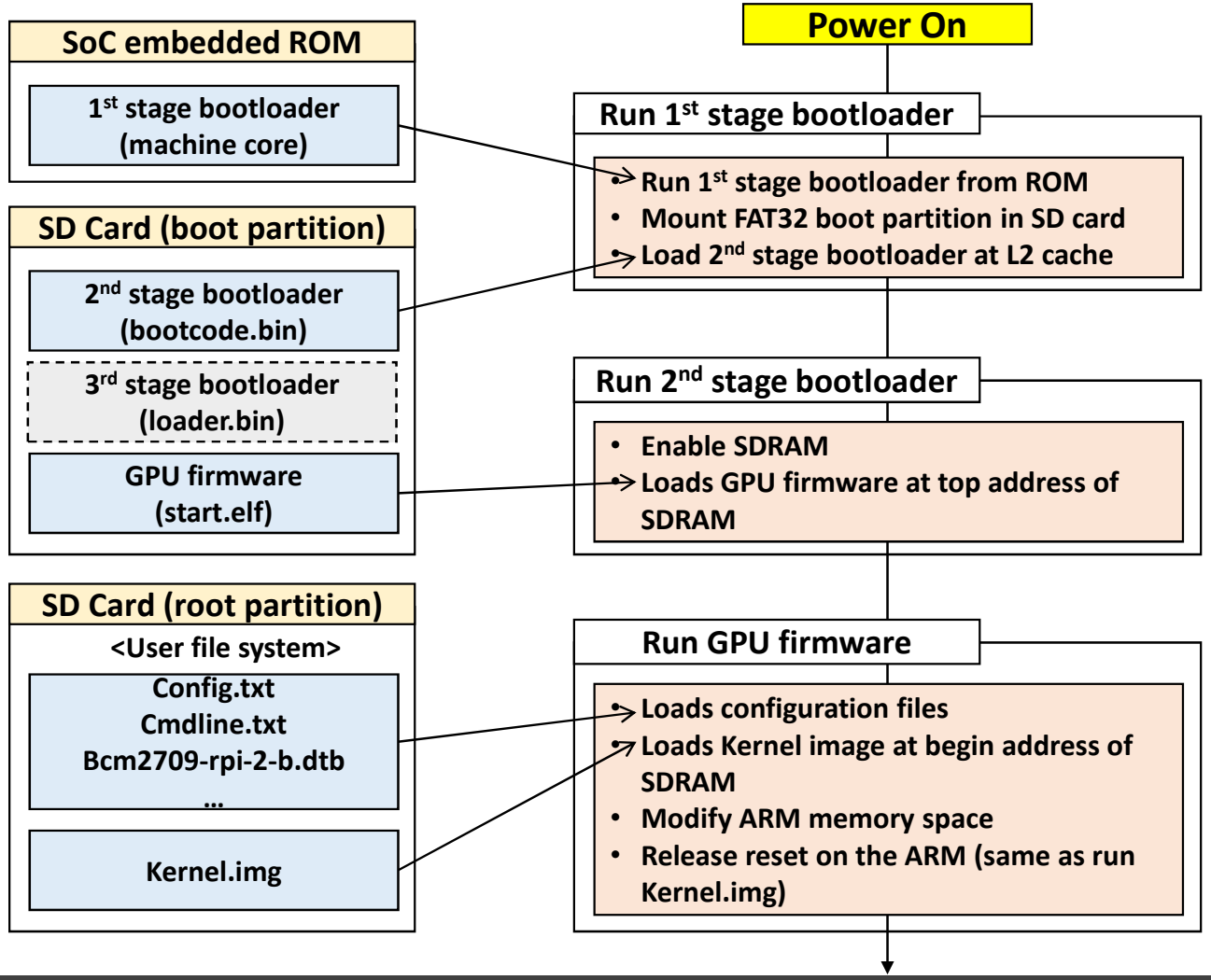
Boot Procedure (부팅절차)

✓ 라즈베리파이의 부팅절차

1. 보드에 전원이 들어오고 GPU가 활성화된다. (ARM Core - 비활성화 상태)
2. SoC내의 ROM에 있는 첫 번째 부트로더(Firmware)를 읽어들인다.
3. 첫 번째 부트로더가 SD카드내의 FAT32(부트 파티션)에서 두 번째 부트로더인 bootcode.bin를 호출한다.
4. 두 번째 부트로더가 SD카드에 있는 config.txt를 읽고 실행한다.
5. 두 번째 부트로더가 세 번째 부트로더 start.elf를 호출 및 실행하여, ARM Core를 활성화 시킨다.
6. ARM Core가 활성화되면 네 번째 부트로더 kernel.img를 호출, 실행한다.

✓ 따라서, 라즈베리파이를 구동하는데 필요한 최소 파일은 bootcode.bin, start.elf, kernel.img이다.

Boot Procedure (라즈베리파이 부팅절차)



HW State
- ARM core: off, GPU: off, SDRAM: disable

3rd Stage Bootloader
- Do not use loader.bin (2012.10 ~)

GPU vs. ARM in SDRAM
- GPU: loaded in 0xC0000000 of SDRAM
- Rpi divided SDRAM to GPU and ARM regions
  (GPU is Top and ARM is Bottom)

Kernel Start Position
- If there exists *DTB*, DTB is 0x100 and kernel is 0x8000
- If there isn't *disable_commandline_tag*, ATAG is 0x100 and kernel is 0x 8000
- If *disable_commandline_tag* is set, kernel is 0x0

※ DTB: Device Tree Blob

Role of Bootloader (부트로더의 역할)

- ✓ **bootcode.bin(두 번째 부트로더)**

  ARM Core와 RAM의 비활성화 상태에서 실행된다.

  그로인해, 가장 먼저 활성화되는 GPU의 L2캐시에 로드되어 GPU에서 실행된다.

  두번째 부트로더인 bootcode.bin는 램을 활성화하고 SD카드로부터 GPU펌웨어(start.elf)를 읽어들인다.

- ✓ **start.elf(세 번째 부트로더)**

  GPU의 펌웨어이며, 일종의 BIOS라고 할 수 있다.

  GPU와 ARM Core에서 사용하는 RAM을 배타적으로 할당.

  디바이스의 각종 주파수와 같은 하드웨어 제어 설정을 한다. (ARM Core를 깨우는 작업)

- ✓ **kernel.img(네 번째 부트로더)**

  ARM Core활성화 시에 RAM에 로드되며 ARM Core는 kernel.img의 명령을 실행한다.

- ✓ **config.txt**

  라즈베리파이의 하드웨어는 /boot 디렉터리에 있는 config.txt파일에 담겨있는 설정 값들에 의해 제어된다.

  이 파일은 라즈베리파이가 다양한 입/출력들을 어떻게 설정하고 SoC칩과 연결된 메모리 모듈이 실행되어야 하는 속도를 알려준다.

References

부트로더
https://toanyone.net/bare-metal-raspberry-pi
https://ko.wikipedia.org/wiki/%EB%B6%80%ED%8C%85

config.txt
http://neoze.co.kr/detail.php?docu_idx=10056&mdl_code=002&sml_code=002

부트로더 절차
http://jake.dothome.co.kr/raspberry/

Cross Kernel Compile

✓ 환경구축에 필요한 툴 설치

$ sudo apt install make

Makefile을 참조하여 빌드하는 명령어
Makefile은 컴파일시 다양한 옵션을 정리해놓은 파일로 쉘스크립트로 구성됨.

$ sudo apt install git

외부(다른서버) 저장소에 위치한 파일/디렉터리 복사

$ sudo apt-get install bison flex libssl-dev bc

크로스컴파일시 시스템에 필요한 툴

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

크로스컴파일에 필요한 툴체인 설치

✓ 환경구축

$ vi ~/.bashrc

가장 끝 줄(명령모드 G)에 밑의 한 개의 line추가

: KERNEL=kernel7

rpi 2이상 kernel7 , rpi 1 이하 kernel

$ echo PATH=\$PATH:~/tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-Raspbian/bin 〉〉 ~/.bashrc

PATH 환경변수를 업데이트 시켜 크로스 컴파일에 필요한 파일 위치를 인식시키기.

호스트 시스템이 32bit일시 gcc-linaro-arm-linux-gnueabihf-Raspbian

호스트 시스템이 64bit일시 gcc-linaro-arm-linux-gnueabihf-Raspbian-x64

$ getconf LONG_BIT 〉〉 OS의 명령어형식 bit수 확인 명령어

$ source ~/.bashrc

.bashrc파일의 스크립트 수행.

해당 명령어를 사용한 뒤 echo $KERNEL과 echo $PATH를 사용하여 정상등록 됐는지 확인

✓ 커널소스 내려받기

- 1번째 방법

  $ git clone --depth=1 --branch rpi-[vir] https://github.com/raspberrypi/linux.git ~/linux

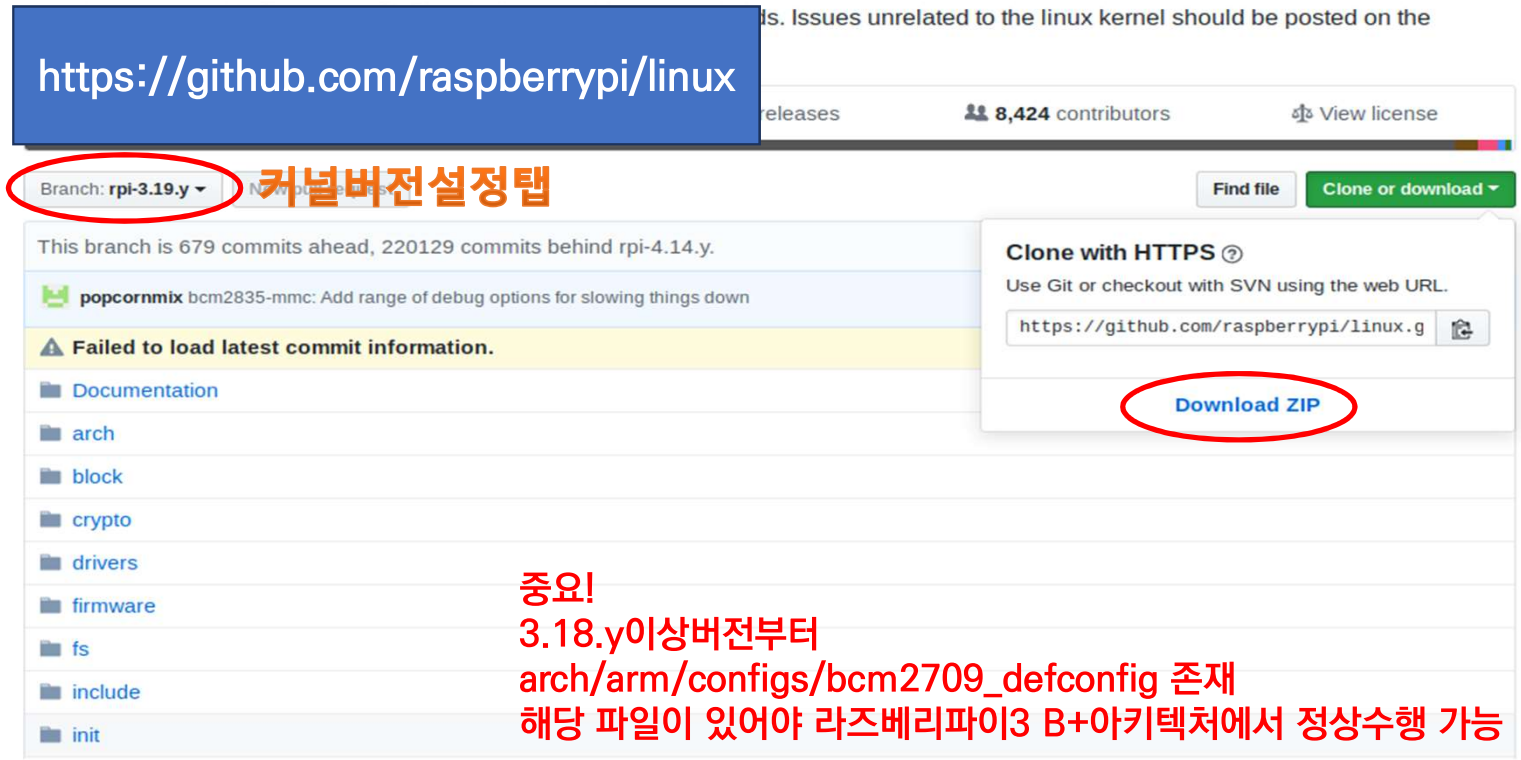
  depth=1 : 최신 커널소스를 내려받기
  해당 옵션이 없을시 저장소 전체를 내려받음
  branch[vir] : [vir]의 커널소스를 내려받기

- 2번째 방법(웹브라우저를 사용하여 내려받기)

  URL : github.com/raspberrypi/linux 접속
  다음 슬라이드에 계속 ⟩⟩

✓ 커널소스 내려받기(2번째 방법)

https://github.com/raspberrypi/linux

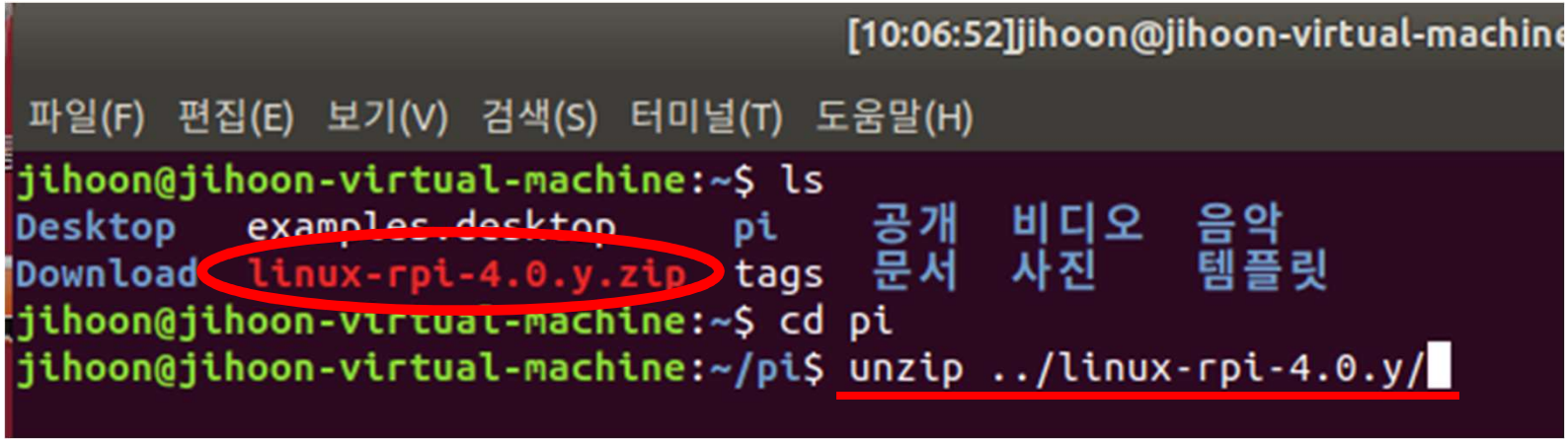


중요!
3.18.y이상버전부터
arch/arm/configs/bcm2709_defconfig 존재
해당 파일이 있어야 라즈베리파이3 B+아키텍처에서 정상수행 가능

✓ 압축풀기(2번째 방법으로 받았을 시 해당)

$ unzip linux-rpi-[ver].zip

unzip명령어로 zip파일 압축해제



✓ .config파일 생성

$ sudo make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcm2709_defconfig

ARCH변수는 커널을 빌드할 타겟시스템의 아키텍처를 명시하는 부분
CROSS_COMPILE변수는 타겟시스템의 크로스 컴파일러를 명시하는 부분
명령어를 수행시 .config파일이 생성

```
jihoon@jihoon-virtual-machine:~/pi/linux-rpi-4.0.y$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- bcm2709_defconfig
#
# configuration written to .config
#
jihoon@jihoon-virtual-machine:~/pi/linux-rpi-4.0.y$ ls -a
.         .gitignore  CREDITS        Kconfig      README          block    firmware  init    lib  samples   sound  virt
..        .mailmap    Documentation  MAINTAINERS  REPORTING-BUGS  crypto   fs        ipc     mm   scripts   tools
.config   COPYING     Kbuild         Makefile     arch            drivers  include   kernel  net  security  usr
jihoon@jihoon-virtual-machine:~/pi/linux-rpi-4.0.y$
```

✓ .config파일의 내용

Arm아키텍처에 올라갈 Linux kernel버전이 명시되어 있다.

```
#
# Automatically generated file; DO NOT EDIT.
# Linux/arm 4.10.17 Kernel Configuration
#
CONFIG_ARM=y
CONFIG_ARM_HAS_SG_CHAIN=y
CONFIG_MIGHT_HAVE_PCI=y
CONFIG_SYS_SUPPORTS_APM_EMULATION=y
CONFIG_HAVE_PROC_CPU=y
CONFIG_STACKTRACE_SUPPORT=y
CONFIG_LOCKDEP_SUPPORT=y
CONFIG_TRACE_IRQFLAGS_SUPPORT=y
CONFIG_RWSEM_XCHGADD_ALGORITHM=y
CONFIG_FIX_EARLYCON_MEM=y
CONFIG_GENERIC_HWEIGHT=y
CONFIG_GENERIC_CALIBRATE_DELAY=y
CONFIG_NEED_DMA_MAP_STATE=y
CONFIG_ARCH_SUPPORTS_UPROBES=y
CONFIG_FIQ=y
CONFIG_VECTORS_BASE=0xffff0000
CONFIG_ARM_PATCH_PHYS_VIRT=y
CONFIG_GENERIC_BUG=y
CONFIG_PGTABLE_LEVELS=2
CONFIG_DEFCONFIG_LIST="/lib/modules/$UNAME_RELEASE/.config"
CONFIG_IRQ_WORK=y
CONFIG_BUILDTIME_EXTABLE_SORT=y
```

✓ 타겟시스템의 커널컴파일 실행

$ sudo make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage modules dtbs

Target시스템의 img file, modules.* file, *.dtb *.dtbo 생성

✓ dtb(device tree blob)

임베디드보드 하드웨어 설정내용을 커널로 넘기기 위한 구조체

```
lyc@lyc-linux:~/pi/linux-rpi-4.10.y$ ls
COPYING       Kbuild      Makefile       System.map  certs     firmware  init    lib           modules.order  scripts   tools  vmlinux
CREDITS       Kconfig     Module.symvers  arch        crypto    fs        ipc     mm            net            security  usr    vmlinux.o
Documentation  MAINTAINERS  README         block       drivers   include   kernel  modules.builtin  samples        sound     virt
lyc@lyc-linux:~/pi/linux-rpi-4.10.y$ cd arch/arm/boot
lyc@lyc-linux:~/pi/linux-rpi-4.10.y/arch/arm/boot$ ls
Image  Makefile  bootp  compressed  dts  install.sh  zImage
lyc@lyc-linux:~/pi/linux-rpi-4.10.y/arch/arm/boot$
```

✓ 타겟시스템의 모듈인스톨

$ mkdir ~/modules

모듈 인스톨 시 저장해둘 dir생성
해당 dir는 ~(home)에 생성되어 있음

$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf INSTALL_MOD_PATH=~/modules modules_install

Target시스템 모듈인스톨 수행
INSTALL_MOD_PATH변수는 명령어수행시 생성파일들 저장할 경로

✓ 타겟시스템의 모듈인스톨

$ rm build source

불필요한 링크파일 삭제
삭제하지 않아도 정상수행 가능하나 시간단축을 위해 삭제하는 것

```
jihoon@jihoon-virtual-machine:~/modules/lib/modules/4.0.9-v7$ ls
build   modules.alias      modules.builtin      modules.dep      modules.devname  modules.softdep  modules.symbols.bin
kernel  modules.alias.bin  modules.builtin.bin  modules.dep.bin  modules.order    modules.symbols  source
jihoon@jihoon-virtual-machine:~/modules/lib/modules/4.0.9-v7$ rm build source
jihoon@jihoon-virtual-machine:~/modules/lib/modules/4.0.9-v7$ ls
kernel         modules.alias.bin  modules.builtin.bin  modules.dep.bin  modules.order    modules.symbols
modules.alias  modules.builtin    modules.dep          modules.devname  modules.softdep  modules.symbols.bin
jihoon@jihoon-virtual-machine:~/modules/lib/modules/4.0.9-v7$
```

✓ 타겟시스템으로 컴파일된 파일 및 디렉터리 전송

```
$ scp linux-rpi-[ver]/arch/arm/boot/zImage pi@xx.xx.xx.xx:/home/pi
$ scp linux-rpi-[ver]/arch/arm/boot/dts/*.dt pi@xx.xx.xx.xx:/home/pi
$ scp linux-rpi-[ver]/arch/arm/boot/dts/overlays/*.dtb* pi@xx.xx.xx.xx:/home/pi
$ scp linux-rpi-[ver]/arch/arm/boot/dts/overlays/README pi@xx.xx.xx.xx:/home/pi
$ scp -r ~/modules/lib/modules/[ver]/ pi@xx.xx.xx.xx:/home/pi
```

- ✓ scp명령어의 형식
  scp [파일 및 디렉터리경로] [Target user이름]@[주소]:[Target내부의 저장경로]
  디렉터리를 보낼시 -r옵션 추가

✓ 타겟내부에서 호스트에게서 받은 파일들을 해당경로로 이동

```
pi@raspberrypi:~ $ ls
4.10.17-v7                          bmp085_i2c-sensor.dtbo          i2c-mux.dtbo                    piscreen.dtbo                   smi-dev.dtbo
adau1977-adc.dtbo                   Desktop                         i2c-pwm-pca9685a.dtbo           pisound.dtbo                    smi.dtbo
adau7002-simple.dtbo                dht11.dtbo                      i2c-rtc.dtbo                    pitft22.dtbo                    smi-nand.dtbo
ads1015.dtbo                        dionaudio-loco.dtbo             i2c-sensor.dtbo                 pitft28-capacitive.dtbo         spi0-cs.dtbo
ads1115.dtbo                        dionaudio-loco-v2.dtbo          i2s-gpio28-31.dtbo              pitft28-resistive.dtbo          spi0-hw-cs.dtbo
ads7846.dtbo                        Documents                       iqaudio-dac.dtbo                pitft35-resistive.dtbo          spi1-1cs.dtbo
akkordion-iqdacplus.dtbo            Downloads                       iqaudio-dacplus.dtbo            pps-gpio.dtbo                   spi1-2cs.dtbo
allo-boss-dac-pcm512x-audio.dtbo    dpi18.dtbo                      iqaudio-digi-wm8804-audio.dtbo  Public                          spi1-3cs.dtbo
allo-piano-dac-pcm512x-audio.dtbo   dpi24.dtbo                      justboom-dac.dtbo               pwm-2chan.dtbo                  spi2-1cs.dtbo
allo-piano-dac-plus-pcm512x-audio.dtbo  dwc2.dtbo                   justboom-digi.dtbo              pwm.dtbo                        spi2-2cs.dtbo
at86rf233.dtbo                      dwc-otg.dtbo                    lirc-rpi.dtbo                   python_games                    spi2-3cs.dtbo
audioinjector-addons.dtbo           enc28j60.dtbo                   MagPi                           qca7000.dtbo                    spi-gpio35-39.dtbo
audioinjector-wm8731-audio.dtbo     enc28j60-spi2.dtbo              mcp23017.dtbo                   raspidac3.dtbo                  spi-rtc.dtbo
audremap.dtbo                       fe-pi-audio.dtbo                mcp23s17.dtbo                   rpi-backlight.dtbo              Templates
bcm2708-rpi-0-w.dtb                 googlevoicehat-soundcard.dtbo   mcp2515-can0.dtbo               rpi-cirrus-wm5102.dtbo          tinylcd35.dtbo
bcm2708-rpi-b.dtb                   gpio-ir.dtbo                    mcp2515-can1.dtbo               rpi-dac.dtbo                    uart1.dtbo
bcm2708-rpi-b-plus.dtb              gpio-poweroff.dtbo              mcp3008.dtbo                    rpi-display.dtbo                vc4-fkms-v3d.dtbo
bcm2708-rpi-cm.dtb                  hifiberry-amp.dtbo              midi-uart0.dtbo                 rpi-ft5406.dtbo                 vc4-kms-v3d.dtbo
bcm2709-rpi-2-b.dtb                 hifiberry-dac.dtbo              midi-uart1.dtbo                 rpi-proto.dtbo                  vga666.dtbo
bcm2710-rpi-3-b.dtb                 hifiberry-dacplus.dtbo          mmc.dtbo                        rpi-sense.dtbo                  Videos
bcm2710-rpi-cm3.dtb                 hifiberry-digi.dtbo             Music                           rpi-tv.dtbo                     w1-gpio.dtbo
bcm2835-rpi-a.dtb                   hifiberry-digi-pro.dtbo         mz61581.dtbo                    rra-digidac1-wm8741-audio.dtbo  w1-gpio-pullup.dtbo
bcm2835-rpi-a-plus.dtb              hy28a.dtbo                      pi3-act-led.dtbo                sc16is750-i2c.dtbo              wittypi.dtbo
bcm2835-rpi-b.dtb                   hy28b.dtbo                      pi3-disable-bt.dtbo             sc16is752-spi1.dtbo             zImage
bcm2835-rpi-b-plus.dtb              i2c0-bcm2708.dtbo               pi3-disable-wifi.dtbo           sdhost.dtbo
bcm2835-rpi-b-rev2.dtb              i2c1-bcm2708.dtbo               pi3-miniuart-bt.dtbo            sdio-1bit.dtbo
bcm2835-rpi-zero.dtb                i2c-bcm2708.dtbo                Pictures                        sdio.dtbo
bcm2836-rpi-2-b.dtb                 i2c-gpio.dtbo                   piscreen2r.dtbo                 sdtweak.dtbo
pi@raspberrypi:~ $
```

》타겟으로 전송완료된 파일들 (작업 디렉터리 : /home/pi)

✓ 타겟내부에서 호스트에게서 받은 파일들을 해당경로로 이동(작업 디렉터리 : /home/pi)

$ mv ./zImage /boot/kernel7.img
  생성한 커널이미지를 boot내의 kernel7.img로 교체
$ mv ./[modules ver]/ /lib/modules/.
  생성한 모듈(directory) 이동
$ mv ./*.dtb /boot/.
$ mv ./*.dtb* /boot/overlays/.
$ mv ./README /boot/overlays/.
$ sudo reboot
  타겟머신 리부팅
$ uname -r
  변경된 커널버전 확인하기

E N D