Bootloader & Kernel

본 문서는 **Bootloader & Kernel 실습**을 원활하게 진행하기 위한 가이드입니다.

☑ 준비물

- 호스트 머신: VMWare
- 타겟 머신: Raspberry Pi

☑ 실습 1) 호스트 머신에 필요한 패키지를 설치

[1] Git 설치

>>> *sudo apt-get install git*

[2] Make 설치

>>> *sudo apt install make*

[3] bison flex libssl-dev bc 설치

>>> *sudo apt-get install bison flex libssl-dev bc*

[4] Arm-linux-gcc 설치

>>> *sudo apt install gcc-aarch64-linux-gnu*

☑ 실습 2) 호스트환경에서 RPI 전용 커널 소스 내려받기

[1] 현재 RPI의 커널 버전을 확인

>>> *uname -r*

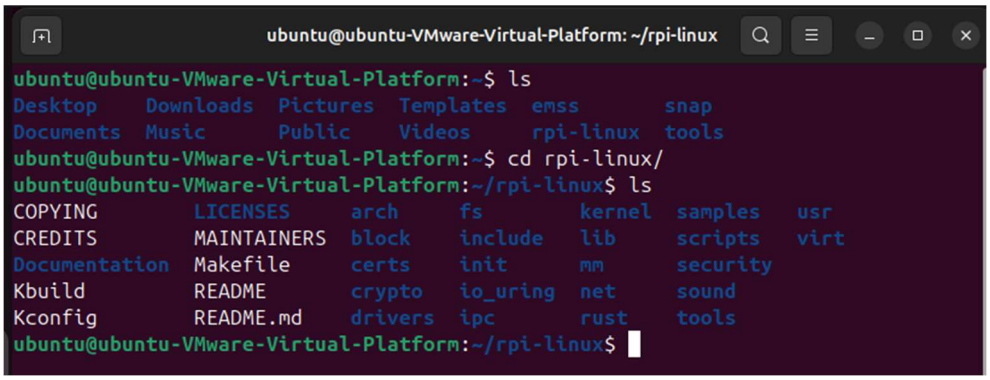
[2] RPI 커널 6.12.y 버전 소스 git clone

>>> *git clone –branch rpi-6.12.y https://github.com/raspberrypi/linux.git ~/rpi-linux*

[3] clone 확인

>>> *ls*

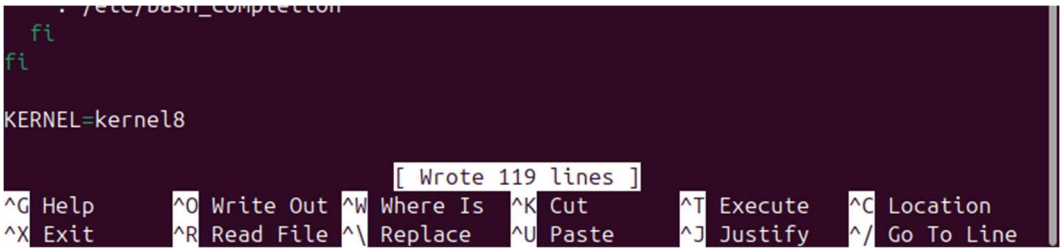
>>> *cd ~/rpi-linux*



☑ 실습 3) 호스트 머신에 bashrc 수정 및 컴파일 진행

[1] Bashrc에 kernel 버전 변경

>>> sudo nano ~/.bashrc



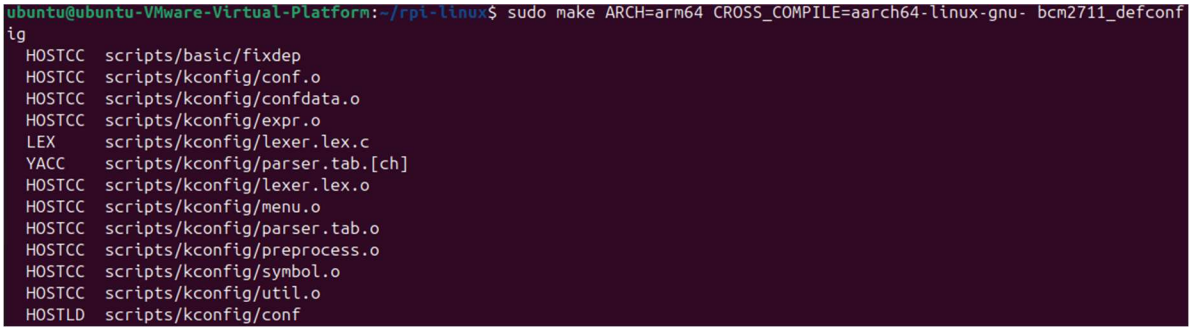
➢ 가장 마지막 줄에 KERNEL=kernel8 추가

>>> source ~/.bashrc

➢ 터미널에서 .bashrc 파일의 스크립트 수행

[2] config 파일 생성

>>> make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- bcm2711_defconfig



[3] 커널 컴파일 진행

>>> make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- Image modules dtbs

➢ 호스트 PC 성능에 따라 소요되는 시간 상이 (최소 20분 이상)

>>> ls .arch/arm64/boot/

```
ubuntu@ubuntu-VMware-Virtual-Platform:~/rpi-linux$ ls ./arch/arm64/boot/
Image  Makefile  dts  install.sh
ubuntu@ubuntu-VMware-Virtual-Platform:~/rpi-linux$
```

- Image 생성 확인

[4] 타겟 시스템의 모듈 인스톨

>>> mkdir ~/modules

>>> sudo make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- INSTALL_MOD_PATH=~/modules modules_install

```
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/openvswitch/vport-vxlan.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/openvswitch/vport-gre.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/openvswitch/vport-gre.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock_diag.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock_diag.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vmw_vsock_virtio_transport_common.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vmw_vsock_virtio_transport_common.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock_loopback.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/vmw_vsock/vsock_loopback.ko.xz
  INSTALL /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/nsh/nsh.ko
  XZ      /home/ubuntu/modules/lib/modules/6.12.81-v8+/kernel/net/nsh/nsh.ko.xz
```

>>> ls ~/modules/lib/modules/

```
ubuntu@ubuntu-VMware-Virtual-Platform:~/rpi-linux$ ls ~/modules/lib/modules/
6.12.81-v8+
```

- 모듈 생성 확인

>>> sudo rm ~/modules/lib/modules/6.12.81-v8+/build

- 불필요한 파일 삭제

☑ 실습 4) 호스트 머신에서 타겟 머신으로 전송

[1] 타겟 머신으로 전송

>>> scp arch/arm64/boot/Image pi@ 타겟 IP :/home/pi

>>> scp arch/arm64/boot/dts/broadcom/bcm2711-rpi-4-b.dtb pi@ 타겟 IP :/home/pi

>>> scp arch/arm64/boot/dts/overlays/.dtbo pi@ 타겟 IP :/home/pi*

>>> scp -r ~/modules/lib/modules/6.12. pi@ 타겟 IP :/home/pi*

```
crc8.ko.xz                                    100% 1412   586.0KB/s   00:00
lz4hc_compress.ko.xz                          100% 6132   781.8KB/s   00:00
lz4_compress.ko.xz                            100% 8288     1.4MB/s   00:00
crc7.ko.xz                                    100% 1400   585.6KB/s   00:00
cordic.ko.xz                                  100% 1424   566.5KB/s   00:00
ts_kmp.ko.xz                                  100% 2112   734.1KB/s   00:00
ts_fsm.ko.xz                                  100% 2384   351.9KB/s   00:00
ts_bm.ko.xz                                   100% 2224   908.2KB/s   00:00
lru_cache.ko.xz                               100% 4608   790.4KB/s   00:00
configs.ko.xz                                 100%   56KB   4.6MB/s   00:00
modules.builtin                               100%   16KB   2.3MB/s   00:00
modules.alias                                 100%  688KB   8.8MB/s   00:00
modules.order                                 100%   74KB   6.8MB/s   00:00
ubuntu@ubuntu-VMware-Virtual-Platform:~/rpi-linux$
```

[2] 타겟 머신에서 확인 및 파일 이동

>>> *sudo mv ~/Image /boot/firmware/kernel8.img*

>>> *sudo mv ~/*.dtb /boot/firmware/*

>>> *sudo mv ~/*.dtbo /boot/firmware/overlays/*

>>> *sudo mv ~/6.12.* /lib/modules/*

```
pi@pi: ~
File  Edit  Tabs  Help
justboom-digi.dtbo                vc4-kms-dsi-waveshare-panel.dtbo
ltc294x.dtbo                      vc4-kms-dsi-waveshare-panel-v2.dtbo
max98357a.dtbo                    vc4-kms-kippah-7inch.dtbo
maxtherm.dtbo                     vc4-kms-v3d.dtbo
mbed-dac.dtbo                     vc4-kms-v3d-pi4.dtbo
mcp23017.dtbo                     vc4-kms-v3d-pi5.dtbo
mcp23s17.dtbo                     vc4-kms-vga666.dtbo
mcp2515-can0.dtbo                 vec-gpio-pi5.dtbo
mcp2515-can1.dtbo                 vga666.dtbo
mcp2515.dtbo                      Videos
mcp251xfd.dtbo                    vl805.dtbo
mcp3008.dtbo                      w1-gpio.dtbo
mcp3202.dtbo                      w1-gpio-pi5.dtbo
mcp342x.dtbo                      w1-gpio-pullup.dtbo
media-center.dtbo                 w1-gpio-pullup-pi5.dtbo
merus-amp.dtbo                    w5500.dtbo
midi-uart0.dtbo                   watterott-display.dtbo
midi-uart0-pi5.dtbo               waveshare-can-fd-hat-mode-a.dtbo
midi-uart1.dtbo                   waveshare-can-fd-hat-mode-b.dtbo
midi-uart1-pi5.dtbo               wifimac.dtbo
midi-uart2.dtbo                   wittypi.dtbo
midi-uart2-pi5.dtbo               wm8960-soundcard.dtbo
```

- 파일 이동 시 보존 실패 경고 발생 (무시)
- mv: failed to preserve ownership for '/boot/firmware/overlays/act-led.dtbo': Operation not permitted

[3] 재부팅 후 버전 확인

>>> *sudo reboot*

>>> *uname -r*

```
pi@pi:~ $ uname -r
6.12.81-v8+
pi@pi:~ $
```

☑ 부록 1) 호스트 머신 용량 확인

[4] 현재 용량 확인

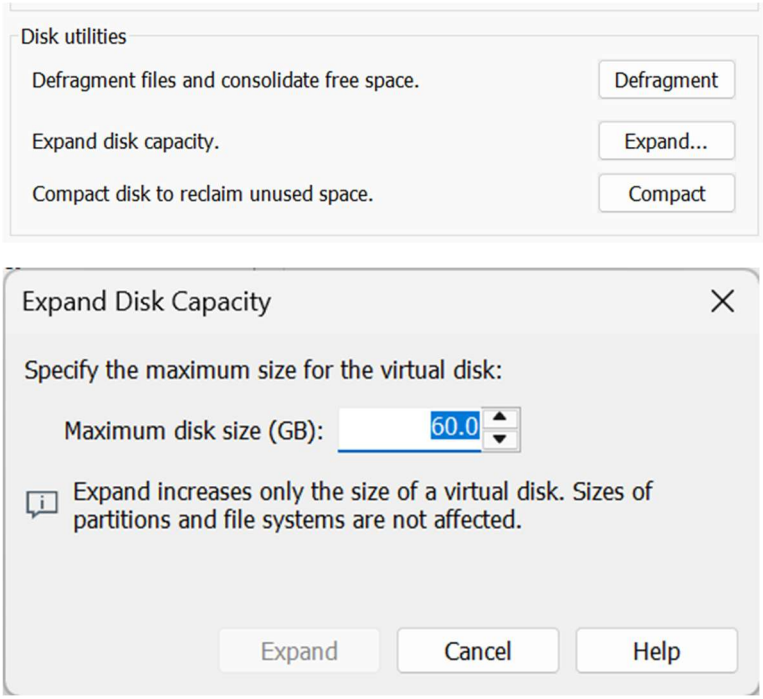
>>> df -h

```
ubuntu@ubuntu-VMware-Virtual-Platform:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           790M  2.0M  788M   1% /run
/dev/sda2        20G   13G  5.6G  71% /
tmpfs           3.9G     0  3.9G   0% /dev/shm
tmpfs           5.0M  8.0K  5.0M   1% /run/lock
tmpfs           790M  120K  790M   1% /run/user/1000
/dev/sr0        100M  100M     0 100% /media/ubuntu/CDROM
/dev/sr1        6.2G  6.2G     0 100% /media/ubuntu/Ubuntu 24.04.4 LTS amd64
```

[5] / (루트)가 Used 하고 있는 용량을 확인 70% 이상 → 용량을 추가로 할당하여 진행

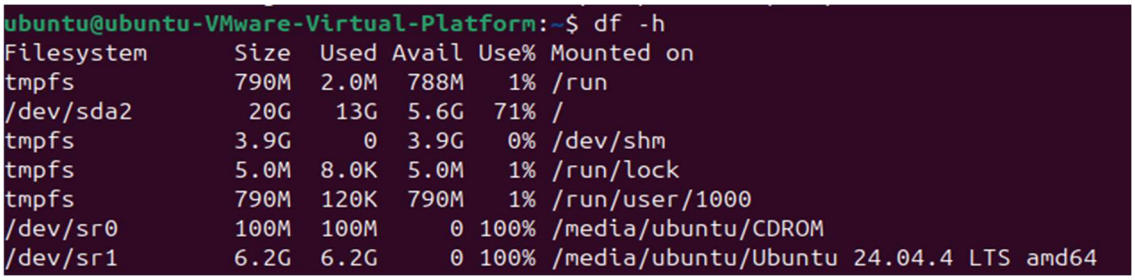
[6] 용량 할당

1. 호스트 머신 power off
2. Settings → Hard Disk 클릭
3. Expand 클릭 → 용량을 60GB로 증가



4. 하지만 Settings 화면에서 Current size가 변하지 않는걸 확인
   가상 디스크의 최대 크기는 증가, 가상 PC 내부의 하드 디스크에는 할당 되지 않은 상태, 가상 PC 안에서 파티션 작업을 별도로 진행
5. 파티션 할당

   >>> dh -f



>>> / (루트)의 파티션 번호를 확인 → sda2

6. 파티션 확장하기

   >>> sudo growpart /dev/sda 2

7. 파일 시스템 크기 조정하기

   >>> sudo resize2fs /dev/sda2

8. 최종 용량 확인

   >>> df -h