### Install toolchain

Use the following command to download the toolchain to the home folder:

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

Updating the $PATH environment variable makes the system aware of file locations needed for cross-compilation. On a 32-bit host system you can update and reload it using:

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

source ~/.bashrc

If you are on a 64-bit host system, you should use:

echo PATH=\$PATH:~/tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/bin >> ~/.bashrc

source ~/.bashrc

### Get sources

To download the minimal source tree for the current branch, run:

git clone --depth=1 https://github.com/raspberrypi/linux

### Build sources

To build the sources for cross-compilation, make sure you have the dependencies needed on your machine by executing:

sudo apt-get install git bison flex libssl-dev

If you find you need other things, please submit a pull request to change the documentation.

Enter the following commands to build the sources and Device Tree files:

For Pi 1, Pi Zero, Pi Zero W, or Compute Module:

cd linux

KERNEL=kernel

make ARCH=arm CROSS\_COMPILE=arm-linux-gnueabihf- bcmrpi\_defconfig

For Pi 2, Pi 3, Pi 3+, or Compute Module 3:

cd linux

KERNEL=kernel7

make ARCH=arm CROSS\_COMPILE=arm-linux-gnueabihf- bcm2709\_defconfig

Then, for both:

make ARCH=arm CROSS\_COMPILE=arm-linux-gnueabihf- zImage modules dtbs

### Install directly onto the SD card

Having built the kernel, you need to copy it onto your Raspberry Pi and install the modules; this is best done directly using an SD card reader.

First, use lsblk before and after plugging in your SD card to identify it. You should end up with something like this:

sdb

sdb1

sdb2

mkdir mnt

mkdir mnt/fat32

mkdir mnt/ext4

sudo mount /dev/sdb1 mnt/fat32

sudo mount /dev/sdb2 mnt/ext4

Next, install the modules:

sudo make ARCH=arm CROSS\_COMPILE=arm-linux-gnueabihf- INSTALL\_MOD\_PATH=mnt/ext4 modules\_install

Finally, copy the kernel and Device Tree blobs onto the SD card, making sure to back up your old kernel:

sudo cp mnt/fat32/$KERNEL.img mnt/fat32/$KERNEL-backup.img

sudo cp arch/arm/boot/zImage mnt/fat32/$KERNEL.img

sudo cp arch/arm/boot/dts/\*.dtb mnt/fat32/

sudo cp arch/arm/boot/dts/overlays/\*.dtb\* mnt/fat32/overlays/

sudo cp arch/arm/boot/dts/overlays/README mnt/fat32/overlays/

sudo umount mnt/fat32

sudo umount mnt/ext4

Another option is to copy the kernel into the same place, but with a different filename - for instance, kernel-myconfig.img - rather than overwriting the kernel.img file. You can then edit the config.txt file to select the kernel that the Pi will boot into:

kernel=kernel-myconfig.img

This has the advantage of keeping your kernel separate from the kernel image managed by the system and any automatic update tools, and allowing you to easily revert to a stock kernel in the event that your kernel cannot boot.

Finally, plug the card into the Pi and boot it!