Building Kernels

If you are only interested in the kernel, you may use this guide to download and build the appropriate kernel.

The following instructions assume that you have not downloaded all of AOSP. If you have downloaded all of AOSP, you may skip the git clone steps other than the step to download the actual kernel sources.

We will use the Pandaboard kernel in all the following examples.

IN THIS DOCUMENT

Figuring out which kernel to build

This table lists the name and locations of the kernel sources and binaries:

Device	Binary location	Source location	Build configuration
shamu	device/moto/shamu-kernel	kernel/msm	shamu_defconfig
fugu	device/asus/fugu-kernel	kernel/x86_64	fugu_defconfig
volantis	device/htc/flounder-kernel	kernel/tegra	flounder_defconfig
hammerhead	device/lge/hammerhead-kernel	kernel/msm	hammerhead_defconfig
flo	device/asus/flo-kernel/kernel	kernel/msm	flo_defconfig
deb	device/asus/flo-kernel/kernel	kernel/msm	flo_defconfig
manta	device/samsung/manta/kernel	kernel/exynos	manta_defconfig
mako	device/lge/mako-kernel/kernel	kernel/msm	mako_defconfig
grouper	device/asus/grouper/kernel	kernel/tegra	tegra3_android_defconfig
tilapia	device/asus/grouper/kernel	kernel/tegra	tegra3_android_defconfig
maguro	device/samsung/tuna/kernel	kernel/omap	tuna_defconfig
toro	device/samsung/tuna/kernel	kernel/omap	tuna_defconfig
panda	device/ti/panda/kernel	kernel/omap	panda_defconfig
stingray	device/moto/wingray/kernel	kernel/tegra	stingray_defconfig
wingray	device/moto/wingray/kernel	kernel/tegra	stingray_defconfig
crespo	device/samsung/crespo/kernel	kernel/samsung	herring_defconfig
crespo4g	device/samsung/crespo/kernel	kernel/samsung	herring_defconfig

You will want to look at the git log for the kernel binary in the device project that you are interested in.

Device projects are of the form device/<vendor>/<name>.

```
$ git clone https://android.googlesource.com/device/ti/panda
$ cd panda
$ git log --max-count=1 kernel
```

The commit message for the kernel binary contains a partial git log of the kernel sources that were used to build the binary in question. The first entry in the log is the most recent, i.e. the one used to build that kernel. You will need it at a later step.

Identifying kernel version

To determine the kernel version used in a particular system image, run the following command against the kernel file:

```
$ dd if=kernel bs=1 skip=$(LC_ALL=C grep -a -b -o $'\x1f\x8b\x08\x00\x00\x00\x00\x00' kernel | cut
```

For Nexus 5 (hammerhead), this can be accomplished with:

```
$ dd if=zImage-dtb bs=1 skip=$(LC_ALL=C od -Ad -x -w2 zImage-dtb | grep 8b1f | cut -d ’ ’ -f1 | he
```

Downloading sources

Depending on which kernel you want,

```
$ git clone https://android.googlesource.com/kernel/common.git
$ git clone https://android.googlesource.com/kernel/x86_64.git
$ git clone https://android.googlesource.com/kernel/exynos.git
$ git clone https://android.googlesource.com/kernel/goldfish.git
$ git clone https://android.googlesource.com/kernel/msm.git
$ git clone https://android.googlesource.com/kernel/omap.git
$ git clone https://android.googlesource.com/kernel/samsung.git
$ git clone https://android.googlesource.com/kernel/tegra.git
```

- The `goldfish` project contains the kernel sources for the emulated platforms.
- The `msm` project has the sources for ADP1, ADP2, Nexus One, Nexus 4, Nexus 5, Nexus 6, and can be used as a starting point for work on Qualcomm MSM chipsets.
- The `omap` project is used for PandaBoard and Galaxy Nexus, and can be used as a starting point for work on TI OMAP chipsets.
- The `samsung` project is used for Nexus S, and can be used as a starting point for work on Samsung Hummingbird chipsets.
- The `tegra` project is for Xoom, Nexus 7, Nexus 9, and can be used as a starting point for work on NVIDIA Tegra chipsets.
- The `exynos` project has the kernel sources for Nexus 10, and can be used as a starting point for work on Samsung Exynos chipsets.
- The `x86_64` project has the kernel sources for Nexus Player, and can be used as a starting point for work on Intel x86_64 chipsets.

Downloading a prebuilt gcc

Ensure that the prebuilt toolchain is in your path.

```
$ export PATH=$(pwd)/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6/bin:$PATH
```

or

```
$ export PATH=$(pwd)/prebuilts/gcc/darwin-x86/arm/arm-eabi-4.6/bin:$PATH
```

On a linux host, if you don't have an Android source tree, you can download the prebuilt toolchain from:

```
$ git clone https://android.googlesource.com/platform/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6
```

Building

As an example, we would build the panda kernel using the following commands:

```
$ export ARCH=arm
$ export SUBARCH=arm
$ export CROSS_COMPILE=arm-eabi-
$ cd omap
$ git checkout <commit_from_first_step>
$ make panda_defconfig
$ make
```

To build the tuna kernel, you may run the previous commands replacing all instances of "panda" with "tuna".

The kernel binary is output as: `arch/arm/boot/zImage` It can be copied into the Android source tree in order to build the matching boot image.

Or you can include the `TARGET_PREBUILT_KERNEL` variable while using `make bootimage` or any other make command line that builds a boot image.

```
$ export TARGET_PREBUILT_KERNEL=$your_kernel_path/arch/arm/boot/zImage
```

That variable is supported by all devices as it is set up via device/common/populate-new-device.sh