FreeBSD on the Pinebook Pro

BY JESPER SCHMITZ MOURIDSEN

The Pinebook Pro is a Rockchip rk3399-based arm64 laptop. It is not currently for sale (according to Pine64) due to global components shortage. You might already own one though, and have missed running FreeBSD on it. In this article, I will describe how to get FreeBSD running on it as a useful desktop. If you do not happen to have a PineBook Pro, the test image that I provide and the build steps also apply to the RockPRO64 board. (Except for U-Boot, do use the stock FreeBSD one for RockPRO64).

As I said, it is rk3399 based, but it is not RockPRO64 in a casing — it does have its own specific mainboard. So the official rockpro64 builds are not the way to go. There are some patches in review that do make the Pinebook Pro useful as a desktop, most notably, the drm-sub-tree of Emmanuel Vadot and the work of Ruslan Bukin who wrote about panfrost in an earlier issue. Also patches for working sound have been written by Alexander Tymoshenko. A lot of work has been done by the posters on the Pinebook Pro thread on forums.freebsd.org started by SleepWalker. It's worth a read if you want a closer look at the development history of getting FreeBSD to run on Pinebook Pro.

Supported Hardware

- The graphics stack with accelerated graphics by the panfrost driver work of Ruslan Bukin and related work by Emmanuel Vadot in the drm-subtree.
- Sound recording and playback by Alexander Tymoshenko.
- emmc and sd-card are also fully supported.
- The SPI flash is detected, but is probably still hit by bug 244146.
- The PCI bridge is supported with an SSD, although ufs shows some weirdness in my tests.
- All CPUs are supported, but FreeBSD cannot make full use of big.LITTLE. i.e., the faster cores have to follow the frequency of the slower ones.
- USB-2 ports.
- Touchpad and keyboard — the touchpad is only working as a simple mouse though.
- Webcam works with webcamd.

Unsupported Hardware

- Wi-Fi and Bluetooth, and DP over USB-C.
- USB-c works as USB-c, if you turn it on with gpioctl. Do not try this if you are not totally sure on where to look in the device tree for the right pin setting and how to set it.

What Software Runs?

I have tested both sway and hikari / wayland and X11 with a couple of DEs. In the process, I found that openbox has an issue with the graphics stack. It does not really render anything correctly within the windows' frames. Luckily xfwm4 does not have any issues. Thus LXQt required a change from the default openbox to xfwm4. LibreOffice runs nicely. Electron stuff is amd64 only on FreeBSD, so you might miss some electron based applications e.g., vscode-oss. Sway needs to be reverted to version 14.1 otherwise you hit *"Cannot use DRM dumb buffers with non-primary DRM FD."* I suspect it is due to the setup where there are two card entries namely /dev/dri/card0 /dev/dri/card1 where only card1 has a render device, but I have not looked into it more closely. I choose instead to revert sway and wlroots to the last known good version on the Pinebook Pro.

Firefox on arm64 and perhaps other applications crashes often unless started with ASLR (address space layout randomization) disabled with proccontrol `-m aslr -s disable firefox`. Also to test webgl in Firefox, you might have to set webgl.force-enabled to true in about:config. Since webcamd runs well and supports the built-in camera, I tested a webrtc call in Nextcloud talk with Firefox and it worked well. Screen sharing in the call worked as well, but only one window at a time. Vlc also does not like the screen:/// capturing, so something with full-screen capturing is an issue at the moment. I also watched YouTube full-screen video without glitches. Note that since this work is based on 14-CURRENT, the default package repository does not build quarterly — so some packages might fail to build once in a while, and thus be missing.

There are some patches in review that do make the Pinebook Pro useful as a desktop

The Booting Process

U-boot versions without video support i.e., without showing anything on the display right at boot time are too old be useful. Also, I skipped a panel driver in the test image, so the panel must be turned on by u-boot. Backlight working also apparently relies on a recent u-boot. The one I have used most successfully is 2021.7 which is the one in the FreeBSD ports tree at the time of writing.

NetBSD noted that the panel breaks on warm reboot. They have a patch that I highly recommend since the panel seems to start in an awfully wrong — and probably very bad for it — state on a warm reboot. The NetBSD patch is adapted to the FreeBSD ports tree here and is in the test image.

If you start from emmc with a FreeBSD u-boot, you should know that it does not default to booting the SD-card. To boot from the SD card in that case, you should press any key to stop autoboot and type run bootcmd_mmc1. Note that the keyboard under u-boot is not too well supported. It does type, but you better type slowly. One can also disable the emmc totally and just test the test image from an sd-card directly. You then avoid u-boot version issues, but it is a little inconvenient to use the internal laptop emmc (fragile) kill switch. It is easy to open the laptop — the switch is just badly placed in my opinion. So if you have a stock manjaro or debian installation, you might have to upgrade in a way that also updates u-boot or to install FreeBSD to the emmc or use its kill switch as mentioned.

Also remember that the u-boot that boots should be the patched one.

The Quick Start

I will later describe how to install from source, but you can easily fetch the test image from github along with modified packages described later.

If you have disabled your emmc or installed the patched u-boot for pinebook pro to it--you can find it here. You are ready to go. Note that RELEASE(7) has per default created two unsafe users with password root, and the user freebsd with password freebsd. ssh is enabled as well. All defaults for arm boards but not so appropriate for the laptop setup. Also do not forget to add your own user to the video group, otherwise the graphics stack would not be permitted to access the graphics device nodes.

Getting Online

Network connectivity is, of course, a must, but built in Wi-Fi is unsupported as of now. I choose to use a wifi dongle with a chipset from man rtwn_usb. You might also install a wi-fi card to m.2 slot if you have the pci-bridge expansion.

I did not try the later myself though. Your cell phone in usb tethering mode can also bring your Pinebook Pro online. For a Wi-Fi dongle, you will have to know how to connect via command-line interface. I recommend you edit /etc/rc.conf right away. See the handbook for more information about that. Using your phone a dhclient ue0 should be enough.

How to Cross Build From Source on amd64

The test image is based on 14-current at commit c9e023541aef. To build it, you can use my PINEBOOKPRO.conf and release.sh at people.freebsd.org/~jsm/pbp

```
./release.sh -c arm64/PINEBOOKPRO.conf
```

But it takes a while to build. (About 1.5 to 2 hours on a GEN10 10 core Intel CPU.

Panfrost is still work in progress at the time of this writing, so I modified Ruslan Bukin's latest PR against the drm-subtree to not use continuous memory since the Pinebook Pro apparently runs low on free continuous memory under relatively heavy loads such as the webglsamples. org aquarium in firefox. I am building panfrost as a module since it crashes at boot if compiled as a device. To build the module you can chroot to the scratchdir of release.sh and in /usr/src with the following environment variables set

```
setenv TARGET arm64
setenv WORKSPACE /usr
setenv MAKEOBJDIRPREFIX $WORKSPACE/obj/
setenv ROOTFS $WORKSPACE/rootfs
setenv SRC /usr/src
setenv MAKESYSPATH $SRC/share/mk
```

use

```
make buildenv TARGET_ARCH=aarch64 BUILDENV_SHELL=/bin/sh
```

I did a Makefile and some compile time error fixing consisting only of adding prototypes and printf format string fixes. You change dir to `/usr/src/sys/dev/drm/panfrost` when in the buildenv. Then you can execute

```
make
make DESTDIR=$ROOTFS install
```

You can then easily experiment with continuous memory by changing

```
-       if (1 == 1)
+       if (1 == 0)
                panfrost_alloc_pages_iommu(bo);
        else
                panfrost_alloc_pages_contig(bo);
```

in panfrost_gem.c. It does not have a sysctl knob, it is a code change. See the discussion on drm-subtree, pull #13.

Ports and Modified Packages

As stated before, I reverted sway wlroots and hikari to earlier versions. libdrm is modified in order to detect the panfrost with this patch. Hikari also needs a small patch to fix its argument parsing issue mesa-dri and mesa-libs are also modified to compile the panfrost driver and enable gles1 and gles2, i.e., to compile in the way Ruslan Bukin described in his article.

The packages are prebuilt for download as is a patch for the ports tree if you prefer to build from source. You can take advantage of the pkg lock feature to not get the modified packages reinstalled by pkg commands. Simply run `pkg lock <pkgname>`. On my system I have the following packages locked:

```
hikari-2.3.2
libdrm-2.4.109,1
mesa-dri-21.3.6
mesa-libs-21.3.6
sway-1.6.1_2
wlroots-0.14.1_2
```

Note for RockPRO64 owners, do not forget to reinstall the RockPRO64 u-boot from ports to the test image, and note that sound is not patched in.

JESPER SCHMITZ MOURIDSEN is a self-taught system administrator and developer currently employed as system administrator working with OpenStack. He is a FreeBSD ports committer, with LXQt as his main focus and co-author of rtsx(4). AFK he likes a ride on his bicycle and is fan of cycling.