Add Support for New Boards to Mainline Linux, U-Boot and Yocto

FOSDEM 2026

Michael Opdenacker

Root Commit

Feb. 1, 2026

Michael Opdenacker

Embedded Linux consultant and trainer

- https://rootcommit.com/about/michael-opdenacker/
- Former founder of Bootlin
- New founder of Root Commit
- Offering embedded Linux training courses with a focus on practical activities, interactivity and learning techniques. https://rootcommit.com/training/
- Happy to be contracted to add support for your boards to Linux or Yocto!
- Free Software enthusiast and advocate (member of April.org)

April
promouvoir et défendre le logiciel libre

Target Audience

root commit

You need to have some experience with:

- Embedded Linux
- Boot process
- Device Trees
- Git
- Embedded Linux build systems

If you don't...

- Pause the PDF or video and come back later
- I'd be delighted to help you learn all this 😉
- Stay in the room if you are at FOSDEM (no space to leave anyway 🤣)

Why Mainline Support?

Example 1: Tinker Board 3S

"Supported" by ASUS:

- Custom Linux 5.10 kernel tree
- Custom U-Boot 2017.09 😱
- Yocto Kirkstone (4.0) `meta-asus` layer

Issues:

- No kernel and U-Boot updates
- Yocto version no longer supported (since May 2024)
- `repo` based script to build the image 🤮
- Even if you manage to build it (manually), the kernel panics!

Who wants to create a new product with obsolete components? Who wants to invest in the past?

Asus Tinker Board 3S
Rockchip 3566 ARM64 CPU

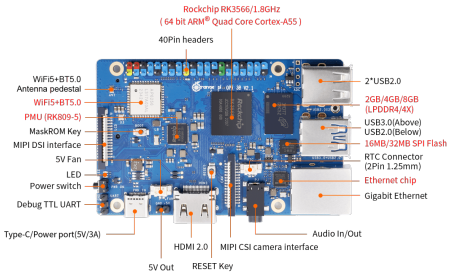
Example 2: Orange Pi 3B

"Supported" by Orange Pi:

- Custom Linux 6.6 kernel tree
- Custom U-Boot 2022.10 tree
- Orange Pi OS (Arch, Linux 5.10), Ubuntu 22.04 (Linux 6.6), Debian 12 (Linux 6.6), Android 11, OpenWRT (Linux 6.6)
- Offering a shell script to build some images (which ones?)

Issues:

- No kernel and U-Boot updates
- Just offering a shell script to build some images (which ones?)
- No support for any proper build system



Orange Pi 3B board
Rockchip 3566 ARM64 CPU
Image credits: http://www.orangepi.org

First Things to Do

root commit

- Find and hook a serial port for the board
- Find documentation for your board
  Example: Orange Pi 3B has a button to skip SPI NOR.
  Otherwise, the Rockchip SoC will always boot on it.
- Find a reference image from the vendor
- Find schematics if any.
- Create a dedicated note for your work on your board,
  consolidating all the resources you find and what you're learning.

Mainline Linux Support

Mainline Linux Support: Getting Started

root commit

- Chances are it will be pretty easy to make mainline Linux boot on your board
- Assuming other boards with the same SoC are already supported
- Just write a quick device tree with the right SoC and a UART corresponding to the primary serial / debug port of the board.
- Don't hesitate to reuse code from other DTS files with the same or similar SoC!
- Start from a reference image and replace the kernel and device tree by your own.

Basic Device Tree Example (Linux 6.18)

root commit

arch/riscv/boot/dts/spacemit/k1-orangepi-rv2.dts

```
// SPDX-License-Identifier: (GPL-2.0 OR MIT)
/*
 * Copyright (C) 2024 Yangyu Chen <cyy@cyyself.name>
 * Copyright (C) 2025 Hendrik Hamerlinck <hendrik.hamerlinck@hammernet.be>
 */

/dts-v1/;

#include "k1.dtsi"
#include "k1-pinctrl.dtsi"

/ {
        model = "OrangePi RV2";
        compatible = "xunlong,orangepi-rv2", "spacemit,k1";

        aliases {
                serial0 = &uart0;
        };

        chosen {
                stdout-path = "serial0";
        };

        leds {
                compatible = "gpio-leds";

                led1 {
                        label = "sys-led";
                        gpios = <&gpio K1_GPIO(96) GPIO_ACTIVE_LOW>;
                        linux,default-trigger = "heartbeat";
                        default-state = "on";
                };
        };
};

&uart0 {
        pinctrl-names = "default";
        pinctrl-0 = <&uart0_2_cfg>;
        status = "okay";
};
```

Device Tree Binding for New Board

Before submitting a DTS for a new board, you **have** to add a new *binding* for it.

```
# SPDX-License-Identifier: (GPL-2.0-only OR BSD-2-Clause)
%YAML 1.2
---
$id: http://devicetree.org/schemas/riscv/spacemit.yaml#
$schema: http://devicetree.org/meta-schemas/core.yaml#

title: SpacemiT SoC-based boards

maintainers:
  - Yangyu Chen <cyy@cyyself.name>
  - Yixun Lan <dlan@gentoo.org>

description:
  SpacemiT SoC-based boards
```

```
properties:
  $nodename:
    const: '/'
  compatible:
    oneOf:
      - items:
          - enum:
              - bananapi,bpi-f3
              - milkv,jupiter
              - spacemit,musepi-pro
              - xunlong,orangepi-r2s
              - xunlong,orangepi-rv2
          - const: spacemit,k1

additionalProperties: true

...
```

Otherwise, your new DTS won't validate.

Build Your Own Device tree ... and kernel!

- Clone the `master` branch of the official Linux tree:

```
$ git clone https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
$ cd linux
```

- Create your own Device Tree Source (DTS):
  `arch/<arch>/boot/dts/<vendor>/soc-board.dts`
- Add this new file to `arch/<arch>/boot/dts/<vendor>/Makefile`
- Set the target architecture and cross-compiler:

```
$ export ARCH=<subdirectory under arch/>
$ export LLVM=1
```

Build Your Own Device Tree and Kernel (continued)

root commit

- Compile the Device Tree:

  `$ make dtbs`

  This generates `arch/<arch>/boot/dts/<vendor>/soc-board.dtb`
- Compile the kernel:

  `$ make -j16 Image # or Image.gz`

  This produces `arch/<arch>/boot/Image`

Replacing the Original Kernel — Simple Case

If you are lucky...

- You will find the vendor kernel and DTB on a partition on the SD card.
- First make a copy of these in a `backup` directory
- Then replace these files with the new ones you generated (keep the same file names)

Replacing the Original Kernel — FIT Image Case

The bootloader or firmware could be loading a FIT image instead

- A FIT image bundles the Kernel, DTB and possibly other files (initramfs, firmware...) in a single file.
- Such a file (`.itb`) is generated from a description in Device Tree syntax (`.its`), using U-Boot's `mkimage` command.
- Fortunately, it's not very complicated to update.

How to "Decapsulate" a FIT Image

Typically to replace the original kernel by a mainline or recompiled one

- Dump the initial FIT image:

  ```
  dtc beaglev_fire.itb > beaglev_fire.its
  ```

- Modify the `.its` file with a text editor:
  Replace "data" binary sections with:

  ```
  data = /incbin/("Image.gz");
  ...
  data = /incbin/("mpfs-beaglev-fire.dtb");
  ```

  Also remove the hash values (they are optional)

- Copy your kernel and DTB to the current directory
- Update the fit Image:

  ```
  mkimage -f beaglev_fire.its beaglev_fire.itb
  ```

How to Boot Without Storage?

If you only have RAM so far

- Boot on an *initramfs*
- Root filesystem loaded in RAM, either by the bootloader or included in the kernel image.
- Easy to build a very small one using BusyBox and a few scripts.
- See my Embedded Linux from Scratch in 50 minutes presentation for details.

If you also have network access

- Boot on a directory on an NFS server (Network File System)
- Best solution: you have as much space for kernel modules, tools and applications as needed.
- Very easy to update without rebooting
- Build your image with Buildroot (easier to get started) or Yocto (if you are familiar with it)
- See `https://github.com/riscv/meta-riscv/blob/master/docs/bananapi-f3.md` for an example.

Board Device Tree Coding Guidelines

root commit

- Follow the DTS coding style:
  https://docs.kernel.org/devicetree/bindings/dts-coding-style.html
  - Nodes on any bus should be in ascending address order
  - If no address, order alphanumerically by node name
  - `status` property right before child nodes
  - Properties of the same type (like aliases) in "natural" order (1, 3, 12...)
- When you submit first DTS support for one board, submit the DTS in one commit. https://lore.kernel.org/all/20251010-confider-raven-0ad7a810e5de@spud/
- Very important: run `make dtbs` and then `make dtbs_check` to validate your changes.
- Then run `scripts/checkpatch.pl -f <modified_files>`
- You will never feel be embarrassed if you already passed these first two checks 😉.
- Caution: `make dtbs_check` won't stop coding style issues.

See Krzysztof Kozlowski's LPC presentation for reasons why DTS validation is important.

Thanks to Krzysztof for always being there to educate us when we make mistakes! ❤️

Writing Good Commit Messages

In commit messages...

- Explain **why** you're doing something (Krzysztof Kozlowski)
- Don't comment on where the DT is coming from (device tree is similar to...)
- References should be like this, followed by a blank line before `Signed-off-by`:

```
[1] https://tinker-board.asus.com/series/tinker-board-3.html
[2] https://tinker-board.asus.com/series/tinker-board-3s.html

Signed-off-by: Michael Opdenacker <michael.opdenacker@rootcommit.com>
```

Generic advice:
https://www.kernel.org/doc/html/latest/process/submitting-patches.html

Belgian policeman from Gaston Lagaffe (Franquin)

Sending and Resending

There are lots of things you need to know to get changes accepted:

- Generate patches
- Write a cover letter and update it with changes
- Sending to the right people and lists (`./scripts/get_maintainer.pl`)
- Add received `Reviewed-by`, `Tested-by` endorsements
- Track your submissions ("revisions")

I used to do all this manually but this can be pretty tedious.

What does it mean if someone replies this to you?

`2 < v`

Answer:

„Please send a V2"

Use b4!

b4 makes life easier for maintainers, contributors and reviewers

- Did you know it can fetch a branch from mailing lists archives in a single command?
  `b4 shazam <message-id>`
  `b4 shazam <message-url-on-lore.kernel.org>`
- It can also automatically collect received `Reviewed-by` and `Tested-by` endorsements

https://b4.docs.kernel.org/en/latest/

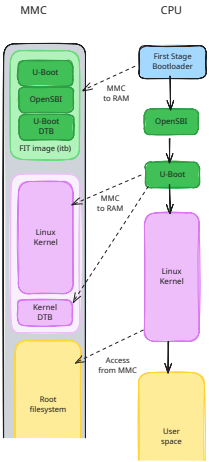
1. Check out the reference branch (typically `master`)
   `git checkout master`
2. Create a new branch:
   `b4 prep -n my-changes`
   Or enroll the current branch (specify the reference branch)
   `b4 prep -e master`
3. Implement and commit your changes
4. Test your changes
5. Create / edit the cover letter
   `b4 prep --edit-cover`
6. Prepare the list of recipients
   `b4 prep --auto-to-cc`
7. Send the patches to yourself
   `b4 send --reflect --no-sign`
8. Send the patches to the lists
   `b4 send --no-sign`

Be patient, if your code depends on other submissions (driver updates in particular)

- Late in the `-rc` series, so much code has been published in various trees (drivers updates in particular), that new code may have too many many dependencies.
- Better to wait for the next `-rc1` until you know for sure what code has landed in mainline.

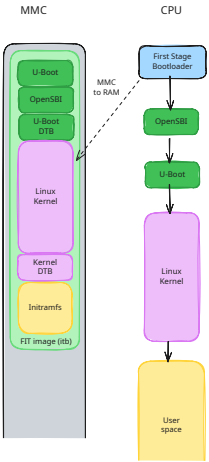
Mainline U-Boot Support

Using Mainline U-Boot without MMC support



In early stages, you may not have access to MMC from U-Boot (and maybe Linux) (currently the case for SpacemiT RISC-V K1 based boards)

- If the first stage bootloader (FSBL) supports MMC, you can load everything to RAM from there!
- Just put more things into the FIT image loaded by the FSBL.
- This way this sequence works:
  Vendor FSBL →**Mainline OpenSBI** →**Mainline U-Boot** →**Mainline Linux**
- But limited to 64 MB according to my experiments



Yocto Support

Usefulness of Standard Build Systems

root commit

In addition to all their benefits (availability of updates, reproducibility...)

- Hide the SoC specific ways of booting
- Help bootloader, kernel and application developers to focus on the "interesting" parts.
- Also help with the adoption of new boards.
  Who wants to learn how to tweak an unmaintained, custom built system?

Build Systems: Priority to the mainline Linux Kernel

root commit

- A full mainline stack (firmware, bootloader, kernel, userspace) is the ultimate goal.
- But at the beginning, focusing on mainline Linux support is more important.
  Even U-Boot depends on the Linux kernel for the Device Tree.
- Therefore, initially keeping the vendor bootloader and firmware can be acceptable.

Yocto: Board Related Definitions

root commit

Here are the main variables to defined in a new `conf/machine/<machine>.conf` file

- `KERNEL_DEVICETREE`: list of kernel device trees
  (the bootloader should find which one to load).
- `UBOOT_MACHINE`: U-Boot configuration file
- `PREFERRED_PROVIDER` for `virtual/kernel` amd `virtual/bootloader`
  if not using the default recipes for kernel and bootloader.
- `WKS_FILE`: specification of flash or block storage image.
  Can be shared with boards with the same SoC.

Example MACHINE definition

root commit

conf/machine/orangepi-3b.conf (meta-rockchip)

```
#@TYPE: Machine
#@NAME: Orange Pi 3B
#@DESCRIPTION: Raspberry Pi sized SBC designed by Kunlong Co.,Limited.
#http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/Orange-Pi-3B.html

require conf/machine/include/rk3566.inc

KERNEL_DEVICETREE = "rockchip/rk3566-orangepi-3b-v2.1.dtb rockchip/rk3566-orangepi-3b-v1.1.dtb"
MACHINE_EXTRA_RRECOMMENDS += "kernel-modules"

UBOOT_MACHINE = "orangepi-3b-rk3566_defconfig"
```

When other similar boards are already supported, that's all you need!

Yocto Resources

- `meta-rockchip` is a great and clean example to imitate
  https://git.yoctoproject.org/meta-rockchip
- Many more details in my "Adding Support for New Boards" presentation at the OpenEmbedded Workshop 2026 (tomorrow!)
  https://rootcommit.com/pub/conferences/2026/oe-workshop/yocto-new-boards/

Wrapping Up

Key Takeaways

root commit

- Not very difficult to start contributing support for new boards
- Start by imitating and reusing code for similar boards:
  a good way to build experience!
- A challenging, rewarding and even addictive experience
- You feel like celebrating at each kernel release
- Allows to get in touch with a small community
  of people with shared interests (and hardware!)
- Could open the door to career opportunities

Call For Contributions

root commit

- I hope you're excited and you have hardware
  to support and co-maintain with future friends
- Don't hesitate to CC me when you submit your board support code.
  I'll review it whenever I can.
- That's also a way for me to learn more 😉

Thank you

root commit

Questions? Comments?

- mo@rootcommit.com
- https://fosstodon.org/@MichaelOpdenacker
- XMPP: omichael@conversations.im
- Signal: rootcommit.01
- Slides available under the CC-By-SA 4.0 license
  https://rootcommit.com/pub/conferences/2026/fosdem/support-new-boards/
- Sources (LaTeX):
  https://gitlab.com/rootcommit/support-new-boards

I ❤ Minneapolis