Upstreaming hardware support in the Linux kernel: why and how?
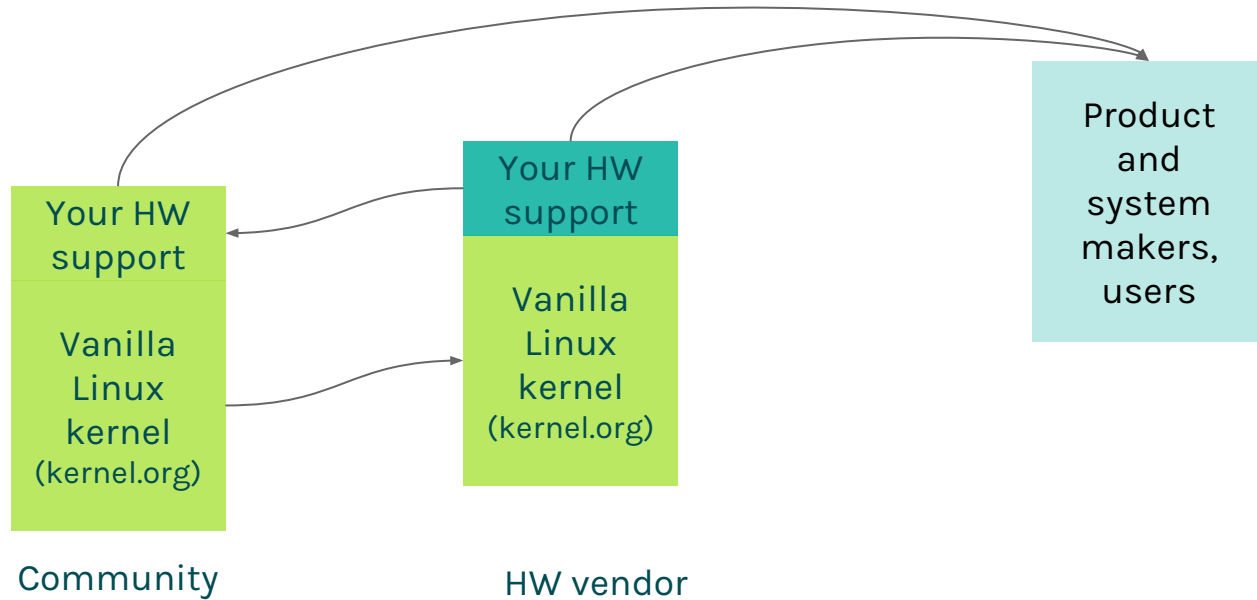
Thomas Petazzoni - CTO Bootlin

Bootlin

- **Engineering** and training company
- **Focused** on embedded Linux
- 11 people, **8 engineers**
- Strong **contribution** to the Linux kernel and other OSS projects
- Freely available training materials

What is upstreaming ?

Your HW support

Vanilla Linux kernel (kernel.org)

Community

Your HW support

Vanilla Linux kernel (kernel.org)

HW vendor

Product and system makers, users

Bootlin kernel upstreaming

- As of v4.5: more than **3000 patches** merged, by 9 persons
- In the **top 20** contributing companies
- Collaboration with HW vendors
- **Focus on ARM**: Marvell, Atmel, Allwinner…
- **6 (co-)maintainer** positions by Bootlin engineers

bootlin

Why?

Reducing software debt

- Kernel is a **very fast moving** target
- **Very costly** to maintain significant changes on up-to-date kernels
- Upstreaming allows to have your changes in the official release
- Code gets **updated** by the community

Higher quality

- Review from **experts** in each kernel subsystem
- Use of **common infrastructures**, code sharing
- Coding standards
- **Contributions** from downstream: tests, bug reports/fixes, improvements

Better experience for integrators/users

- **Long term** availability of kernel **updates**: security fixes, LTS releases
- Benefit from new kernel **features**
- **Standard** interfaces
- Community/third-party **support**
- Allows **downstream** participation
- HW supported by OS vendors/distributions

Increased credibility

- Shows **commitment** to proper software support
- Both in terms of **quality** and **cost**
- **Positive image** in the open-source community
- **Easier to hire** open-source engineers

Higher control

- If you don't do it, others might do it
- In which case you have **less control**
- By initiating the kernel support for a feature, you have more **control** over it
- You are the one driving rather than being driven

How?

Small & focused team

- **Small** team inherently needs less communication overhead
- **Focused** on upstreaming only
- **No distraction** from products, customer support or bring-up
- **Motivated** team, engineers who like to contribute

Community involvement

- Engineers must **be part of** the community
- **Understand** the community as a special actor, not your employees
- Learn the **rules** of the community
- Your solution may not be the one accepted by the community
- Allow your engineers **to contribute** beyond your own HW

Empower the community

- Enable others to work on your HW
- Will give you **free** bug fixes, performance improvements, and additional features
- **Datasheets**, as open as possible
- **Be present** on the mailing lists, **answer** questions
- Leave enough room for others to **engage**
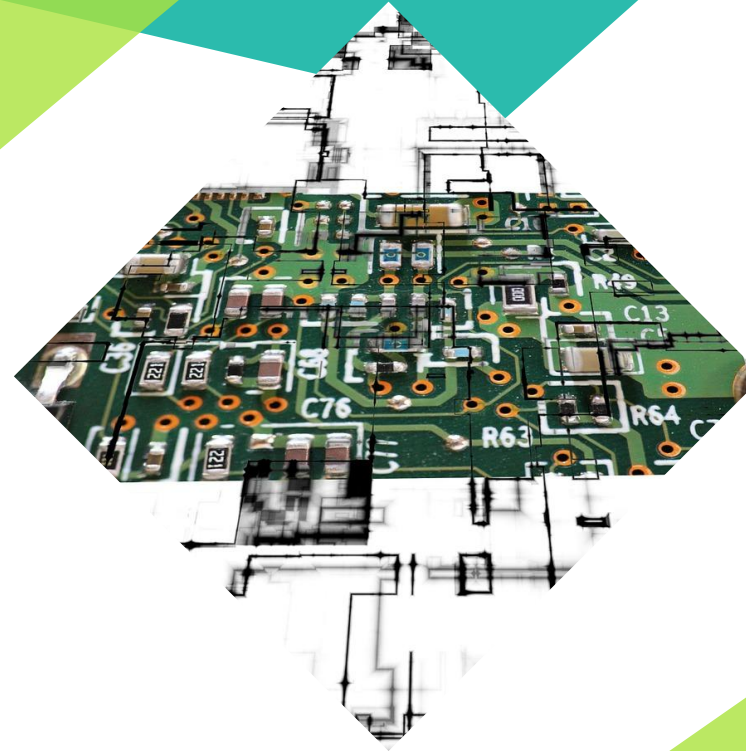
Management buy-in

- Recognize upstreaming as a **special** activity
- Difficult to do planning, you don't control the community
- **Reduce** the administrative and legal **overhead**
- Must be a **long-term** strategy

Conferences and networking

- Good way to **be part of** the community
- **Meet** other developers and kernel maintainers
- **Learn** about the latest Linux developments
- **Talk/discuss** about the issues to support your HW

Provide/use the right tools

- **Lots of time lost** in big companies to fight against inappropriate tooling
- Outlook, Word and Windows **are not** the right tools for kernel engineers
- A Linux machine, standard SMTP server to send e-mails/patches, IRC access.

Thanks!

Any questions?

You reach me at thomas@bootlin.com