Best practices to open-source FPGA designs

Javier Serrano, with help from Alén Arias, Hamza Boukabache, Christos Gentsos, Tristan Gingold, Eva Gousiou and Dimitris Lampridis

CERN, Geneva, Switzerland

FPGA Developers' Forum
11 June 2024

Important note before we begin

Best practices are sometimes subjective. Hopefully you can find some good ideas here. Also, a bit of this material is CERN-specific. If you do not work at/with CERN, it should still illustrate issues you need to keep in mind in your particular context.

Outline

Outline

Why should I open-source my HDL code?

Pick your reason(s)

- Stop replicating each other’s work
- Increase quality through reuse and peer review
- (Depending on where you work) fulfil your mission
- Work with and learn from others, inside and outside your organisation/company

Outline

What should I be publishing?

Gateware

A single core or a complete top-level design

What should I be publishing?

Gateware

A single core or a complete top-level design

Ancillary files

Documentation (incl. reset and clocking), testbenches, scripts to generate project files and proprietary files locally. . .

What should I be publishing?

Gateware

A single core or a complete top-level design

Ancillary files

Documentation (incl. reset and clocking), testbenches, scripts to generate project files and proprietary files locally. . .

Hardware and its documentation

Power sequencing, description of I/O. . .

What should I be publishing?

Gateware

A single core or a complete top-level design

Ancillary files

Documentation (incl. reset and clocking), testbenches, scripts to generate project files and proprietary files locally. . .

Hardware and its documentation

Power sequencing, description of I/O. . .

Reference designs and their documentation

So users can build on a working system

Outline

Whom should I involve?

At CERN, in this order:

- Design team and community
- Line management
- Knowledge Transfer
  - https://kt.cern
  - kt@cern.ch
- Open Source Program Office (OSPO)
  - https://ospo.docs.cern.ch
  - open.source@cern.ch
- Back to line management for a final decision on open-sourcing

Outline

When should I go public?

Pre-requisites

- Not before doing your due diligence (see https://ospo.docs.cern.ch/key-concepts/due-diligence)
- Not before a final OK from line management (see previous slide)

When should I go public?

Pre-requisites

- Not before doing your due diligence (see https://ospo.docs.cern.ch/key-concepts/due-diligence)
- Not before a final OK from line management (see previous slide)

Then two possibilities:

- Open when ready
- Open from the start (preferred)

Outline

Outline

Reusing good ideas from FOSS

Modular design

- Try to engineer your design as a set of reusable pieces of HDL
- Take every opportunity to enrich a common kit: memories, FIFOs, sync. . .
- Hide technology-specific blocks behind common interfaces

Reusing good ideas from FOSS

Modular design

- Try to engineer your design as a set of reusable pieces of HDL
- Take every opportunity to enrich a common kit: memories, FIFOs, sync. . .
- Hide technology-specific blocks behind common interfaces

Git-friendly text files

- Reduce amount of block design
- Text-based documentation evolves with design
- If you need to use IP generators, scripts to invoke them rather than their output

Reusing good ideas from FOSS

Tools

Use good FOSS tools which automate tasks and make your life easier: Hog, HDLMake, Cheby, FuseSoC. . .

Reusing good ideas from FOSS

Tools

Use good FOSS tools which automate tasks and make your life easier: Hog, HDLMake, Cheby, FuseSoC. . .

Releases

With versioned binaries and documentation, including the versions of the tools needed to work with the core/design.

Reusing good ideas from FOSS

Tools

Use good FOSS tools which automate tasks and make your life easier: Hog, HDLMake, Cheby, FuseSoC. . .

Releases

With versioned binaries and documentation, including the versions of the tools needed to work with the core/design.

Community

Provide for a welcoming environment, foster kindness, enforce respect.

Versioning and dependencies

A convention

To identify the nature and the version (and hash) of the gateware configuring an FPGA, and ideally also of any cores inside it. See e.g. https://ohwr.org/project/fpga-dev-id/wikis.

Versioning and dependencies

A convention

To identify the nature and the version (and hash) of the gateware configuring an FPGA, and ideally also of any cores inside it. See e.g. https://ohwr.org/project/fpga-dev-id/wikis.

Packaging

Use packages to express dependencies, manage versions and check compatibility using your convention for gateware identification.

Outline

github.com vs gitlab.com vs. . .

Elements to take into account

- Familiarity and wishes of main designers
- Easiness to create an account and start contributing
- Features of the platform

github.com vs gitlab.com vs. . .

Elements to take into account

- Familiarity and wishes of main designers
- Easiness to create an account and start contributing
- Features of the platform

Desirable features

- Git repository
- Issue management
- Wiki
- Forums – sometimes communicating over issues and PR/MR is not enough
- CI/CD

Guidelines for CERN members of personnel

Check https://ospo.docs.cern.ch. In particular:

- Use your cern.ch address for your account in the collaboration platform and for your Git commits.
- Do not use gitlab.cern.ch to host projects with non-CERN contributors.
- Use CI/CD for linting (enforcing a coherent and documented style), pass/fail testbenches and packaging.
- Mirror projects into gitlab.cern.ch using the pull method when you need to run proprietary tools as part of your CI/CD pipelines.
- Provide a good README at the top directory of your project with a short description, ways to contribute, licensing and contact information.
- Adopt REUSE (https://reuse.software) for machine-readable and checkable licensing information and coherency.

Outline

What is a licence?

A permission you give someone to do something (s)he would otherwise not have the right to do.

Software licensing: our starting point

Mostly copyright licences

- Very uniform legal landscape worldwide
- Modern licences also deal with patents

Software licensing: our starting point

Mostly copyright licences

- Very uniform legal landscape worldwide
- Modern licences also deal with patents

Three licensing regimes

- Permissive (BSD, MIT, Apache v2)
- Weakly reciprocal (MPL v2, LGPL v3)
- Strongly reciprocal (GPL v3, AGPL v3)

Issues with reciprocal software licences when applied to HDL

Language

Uncertain meaning of terms in an HDL context

Issues with reciprocal software licences when applied to HDL

Language

Uncertain meaning of terms in an HDL context

There is no equivalent to “linking”

Hard/impossible to interpret section 4d of LGPL v3

Issues with reciprocal software licences when applied to HDL

Language

Uncertain meaning of terms in an HDL context

There is no equivalent to “linking”

Hard/impossible to interpret section 4d of LGPL v3

Reciprocity

What should a reciprocal licence do for a hardware design? What is the scope of reciprocity?

Issues with reciprocal software licences when applied to HDL

Language

Uncertain meaning of terms in an HDL context

There is no equivalent to “linking”

Hard/impossible to interpret section 4d of LGPL v3

Reciprocity

What should a reciprocal licence do for a hardware design? What is the scope of reciprocity?

The HDL design ecosystem

Dominated by proprietary tools, parts of which sometimes go into the design itself

The CERN Open Hardware Licence v2

- Based on rights mainly applying to the design sources (e.g. circuit schematics, CAD drawings or HDL files)

The CERN Open Hardware Licence v2

- Based on rights mainly applying to the design sources (e.g. circuit schematics, CAD drawings or HDL files)
- Specifies conditions for:
  - Copying designs
  - Modifying designs
  - Distributing modified or unmodified designs
  - Making hardware out of those designs
  - Distributing that hardware

The CERN Open Hardware Licence v2

- Based on rights mainly applying to the design sources (e.g. circuit schematics, CAD drawings or HDL files)
- Specifies conditions for:
    - Copying designs
    - Modifying designs
    - Distributing modified or unmodified designs
    - Making hardware out of those designs
    - Distributing that hardware
- Drafted by Myriam Ayass, Andrew Katz and Javier Serrano

The CERN Open Hardware Licence v2

- Based on rights mainly applying to the design sources (e.g. circuit schematics, CAD drawings or HDL files)
- Specifies conditions for:
  - Copying designs
  - Modifying designs
  - Distributing modified or unmodified designs
  - Making hardware out of those designs
  - Distributing that hardware
- Drafted by Myriam Ayass, Andrew Katz and Javier Serrano
- Comes in three variants:
  - CERN-OHL-P-2.0 (permissive)
  - CERN-OHL-W-2.0 (weakly reciprocal)
  - CERN-OHL-S-2.0 (strongly reciprocal)

Challenges in hardware licensing

How CERN OHL v2 deals with them

Language

Uses terms which are easy to relate to in an HDL context

Challenges in hardware licensing

How CERN OHL v2 deals with them

Language

Uses terms which are easy to relate to in an HDL context

Linking

No equivalent of section 4d of LGPL v3 in CERN-OHL-W-2.0

Challenges in hardware licensing

How CERN OHL v2 deals with them

Language

Uses terms which are easy to relate to in an HDL context

Linking

No equivalent of section 4d of LGPL v3 in CERN-OHL-W-2.0

Reciprocity

Have URL travel with object and use concepts of Product and Available Component to establish limits of reciprocal obligations

Challenges in hardware licensing

How CERN OHL v2 deals with them

Language

Uses terms which are easy to relate to in an HDL context

Linking

No equivalent of section 4d of LGPL v3 in CERN-OHL-W-2.0

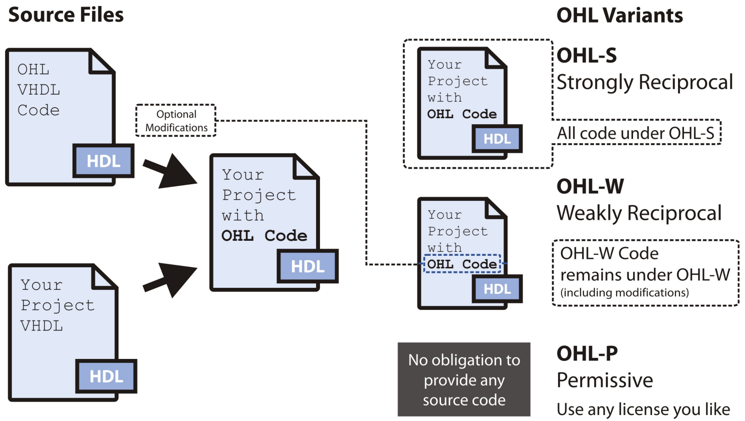
Reciprocity

Have URL travel with object and use concepts of Product and Available Component to establish limits of reciprocal obligations

The hardware design ecosystem

Components which are shipped with design tools qualify as Available Components

CERN OHL v2 for HDL/FPGA/ASIC designs



Courtesy Tomasz Włostowski

Licensing your software and documentation

Use appropriate licences for software (see earlier slide and your favourite guidelines website or OSPO) and documentation. CC-BY and CC-BY-SA are good licences for documentation.

Outline

Conclusion

- Most of these best practices are (I think) largely uncontroversial.
- Tooling a matter of taste.

CERN | OSPO

CERN OSPO technical website: https://ospo.docs.cern.ch
OSPO forum: https://ospo.web.cern.ch
OSPO email address: open.source@cern.ch
Get in touch if you want to discuss further!

Backup slides

Backup slides

The Holy Trinity of “Intellectual Property”

Copyright

generally deals with the right to make copies

Trademarks

define under what circumstances you can use a recognisable brand or logo

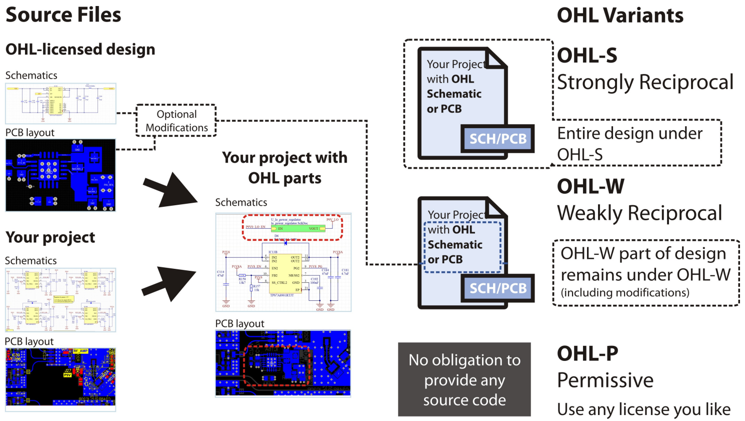
Patents

allow an inventor to exclude others from making, using or selling an invention

Are the reciprocal variants of CERN OHL v2 compatible with GPL?

No. This was a tough decision, but making the W and S variants compatible with GPL would have offered licensees an easy way to escape certain obligations, such as the need to keep a visible URL on an object at which to find the design files from which the object was created.

CERN OHL v2 for PCB designs



Courtesy Tomasz Włostowski