Tools Used in the Creation of Open Source Work

Open Research Institute, Inc.

October 4, 2020

“Our goals are to produce open source designs that anyone is free to adapt to their specific needs. Some parts may require specialized tools to use - we wish to minimize that, but non-free non-open source tools may sometimes be required to utilize the sources. We can limit IP and endeavor to avoid purchasing IP beyond that included with the standard tools. But I don't think we can afford not to use commercial tools when they are the best choice for completing a particular set of tasks. If we need Matlab to run a vendor's filter generation tools - so be it. If someone wants to take on converting this to Octave - please have at it. If you can modify it and prove it's valid we'll be happy to use it. Otherwise, we will just suck it up and use Matlab.”

- Wally Ritchie 11 August 2020

Open source vs. proprietary tools.

Many open source tools are rapidly approaching the point where they are equal to or superior to closed source tools. Some tools are well beyond that point. We stop arguing about them and proprietary competitors fade away when that happens. For example, Linux for servers.

This is not a uniformly distributed reality. Cutting-edge open source work often relies on closed-source/proprietary tools and frameworks. Being explicit about when to use proprietary tools improves the process of creating quality open source work within an organization or group.

This paper discusses the use of two different open source tools within Open Research Institute and related policy implications. The first, Xilinx Vivado, is a closed source tool necessary for the production of the digital multiplexing transponder. Xilinx Vivado is the current recommended baseline for field programmable gate array (FPGA) work using Xilinx’s high-end devices. The second, KiCad, is an open source framework that is now the recommended baseline for printed circuit board assembly (PCBA).

Capabilities of both proprietary open source tools constantly change to address the needs of the user base. Advantages and disadvantages change over time. A decision to endorse, require, or baseline a tool will need to be regularly revisited.

There shall be no penalty for selecting a closed source/proprietary tool when it gives superior performance and it is not the primary product under investigation or development.

The assertion of a moral requirement to require the use of open source.

> "Well, I think we fail at the mission of being an Open Research Institute that produces open stuff for everyone if they can't utilize it without proprietary tools."
>
> -Bruce Perens 11 August 2020

Supporting open source work is the mission of an open source research and development organization like Open Research Institute. Rigidly requiring all peripheral tools and activity to also either be or use open source in order to utilize a design is a policy choice that must be weighed and considered. The unintended consequences of enforcing an *only open source* mandate can be significant.

It is impossible to be pure. For example, rejecting a presentation about an improvement to forward error correction because it's in PowerPoint hurts people wanting to learn about codes. Refusing to accept a donation because PayPal uses proprietary Javascript hurts fundraising. Failing to file taxes because the accountant used commercial software will result in legal repercussions. Refusing to use test equipment because it's not open source hardware limits test coverage. Refusing to simulate an open source antenna in HFSS because HFSS is not open source misses out on powerful and useful results that may dramatically improve the design. Delaying a build because no manufacturer will agree to use open source production software can wreck a product schedule. Using inferior communications platforms because they are open source makes for missed connections.

These examples illustrate some unreasonable ways to support the production of an open source creative work that has nothing do with presentation software, financial transition software, tax preparation software, test equipment, simulation software, manufacturing process controls, or group chat.

When designing a communications system for amateur radio use, the focus must remain on the primary product and secondarily on the tools used to get there. Insisting that only open source tools be used to create hardware will result in painful and possibly permanent delays. We should go where the best tools take us. If this means using proprietary tools to create a useful open source work, then so be it.

<u>We shall use the right tools for the job of producing the highest possible quality of work in the shortest possible amount of time.</u>

It does harm to delay the primary product indefinitely if one cannot produce or verify the primary product with open source tools. For example, in the case of hardware description language work, if the firmware produced would be identical in every other way, it is a superior open source work if it is produced sooner rather than later (or never).

Addressing the assertion of a license requirement to use open source in every possible way on a project.

Under the CERN open hardware license version 2.0, there is no requirement that open source tools must be used to produce an open source work. This license was written with work such as FPGA technology in mind because of the particular challenges with licensing firmware within the dominance of proprietary toolchains.

The key concept is “available component”. This concept has been carefully tailored specifically to permit use of proprietary FPGA tools because there is no other viable choice yet.

From:

https://ohwr.org/project/cernohl/wikis/home

1.7 'Available Component' means any part, sub-assembly, library or code which:

a) is licensed to You as Complete Source under a Compatible Licence; or

b) is available, at the time a Product or the Source containing it is first Conveyed, to You and any other prospective licensees

i) as a physical part with sufficient rights and information (including any configuration and programming files and information about its characteristics and interfaces) to enable it either to be Made itself, or to be sourced and used to Make the Product; or
ii) as part of the normal distribution of a tool used to design or Make the Product.

1.8 'Complete Source' means the set of all Source necessary to Make a Product, in the preferred form for making modifications, including necessary installation and interfacing information both for the Product, and for any included Available Components. If the format is proprietary, it must also be made available in a format (if the proprietary tool can create it) which is viewable with a tool available to potential licensees and licensed under a license approved by the Free Software Foundation or the Open Source Initiative. Complete Source need not include the Source of any Available Component, provided that You include in the Complete Source sufficient information to enable a recipient to Make or source and use the Available Component to Make the Product.

From the rationale document at https://ohwr.org/project/cernohl/wikis/uploads/0be6f561d2b4a686c5765c74be32daf9/CERN_OHL_rationale.pdf

If you're designing a development board that contains an FPGA which the user can load with their own bitstream, then you are under no obligation to distribute any bitstream at all, if the FPGA is not loaded with one on distribution. You may choose to load a simple sample bitstream, in which case you should distribute either the bitstream itself, alongside the FPGA, or the FPGA preloaded with a bitstream with sufficient description to enable someone to recreate the functionality of the FPGA as configured.

The bitstream itself does not have to be licensed under the CERN-OHL (although it may be), but it must be provided with sufficient rights to enable the recipient of the FPGA to use it to Make the Product.

However, we encourage people to release the HDL (hardware description language) file for the bitstream under the CERN-OHL-S, -W or -P. That essentially makes it a Product (the bitstream) within a Product (the FPGA). Where a bitstream is released under CERN-OHL-W or -S, what's the Complete Source for the bitstream?

It will consist of the custom code that you have written together with details of the libraries you are using to create it. If they are the standard libraries that are provided with Xilinx's ISE or Vivado Design Suite, then they will fulfill the definition of Available Component: they are libraries or code generally available (in this case, at a cost), and are part of the normal distribution of a tool used to design or Make the Product. So you get the bitstream you need, and you are not required to release the Xilinx libraries if you distribute the bitstream. This is intended to balance the legitimate interests of the FPGA companies as well as retaining the essential openness in the design.

There is no license requirement that open source tools must be used to produce open source work under CERN OHL.

Even GPL allows proprietary compilers.

We should strive whenever we can to use, support, and develop open source tools. We should impose no unnecessary friction on producing the primary open source product by requiring secondary or support products to also be open source, when they are not the best tool for the job or secondary component.

For example, Open Research Institute actively supports Applied Ion Systems. This is an open source thruster project for spacecraft. This is critical technology for open source space.

However, Open Research Institute baselines a proprietary thruster for cubesat projects, as the open source thruster cannot yet provide enough thrust for the planned missions. The current primary product of Open Research Institute is the communications payload, and not the propulsion system.

ORI support of AIS work does not require including AIS thrusters, at this point, in ORI designs. That is the end goal of ORI's support for AIS. Supporting AIS work does not mean forgoing a launch opportunity because the technology cannot yet provide enough reliable thrust.

<u>Xilinx Vivado is the baseline toolchain recommendation for Open Research Institute.</u>

Open Research Institute actively supports the ICEBreaker LiteX work and similar initiatives that are making it easier to do open source FPGA work. We believe this line of work will eventually produce an FGPA toolchain superior to Xilinx Vivado for any particular level of project complexity.

> "…what we will see, and soon, is a disappearance of proprietary tools for FPGA and ASIC layout. We've seen a rapid catchup of open tools in this space in just 12 months - you can now, for example, synthesize and simulate or program relatively complex OpenPOWER and RISC-V cores/SoCs using only these tools."
>
> - Hugh Blemings 11 August 2020

For communications work where open source tools are not yet at the level where they replace proprietary vendor tools, then baseline recommendation is to use those proprietary tools.

Open Research Institute follows a both/and, not an either/or, approach to technical work. Selecting the right tool for the job means constant mindfulness of requirements, specifications, organizational values, and volunteer time. When, and not if, open source tools catch up in this space, they will be the baseline recommendation in support of the primary product.

> "If you can get a carpenter to build something for free labor - you wouldn't ask him to make his own tools - just because he can. We won't build our own spectrum analyzers or oscilloscopes - although we could. We focus on our project and we will beg, borrow, steal, or buy if necessary the tools and test equipment to complete the jobs we are doing."
>
> - Wally Ritchie 11 August 2020

<u>KiCad is the baseline toolchain recommendation for Open Research Institute.</u>

> "Conducted well, the Ground Station project will likely outlive any commercial tools we use. It is our choice as to whether it outlives the file formats chosen. That is to say we'll be building and launching iterations of todays work when many of the existing proprietary tools are fading, or have faded from view. Ensuring we can use accumulated design work products over a couple of decades becomes important.
>
> We've already seen basic multilayer designs using at least DDR3 and similarly specified SERDES done using open tools such as KiCAD - one recent example is OrangeCrab This board is of course simpler than what we will need for the project, but I believe illustrates the feasibility of the endeavour.
>
> In a (recent) previous life I worked with an organisation doing open source high performance computing hardware - their next generation designs will be multilayer (10 I think) and support PCIeGen4, DDR4 and likely OpenCAPI with a server class CPU. This will be done in its entirety with open tools - KiCAD among them.

> We have already seen a strong trend of the PCB fab houses embracing these open tools, it's not unreasonable to think this trend will continue to increasingly sophisticated board materials, layer options etc.
>
> I don't think we'll see proprietary tools like Allegro, Altium, OrCad disappear overnight, but as open offerings become increasingly capable their days surely become more difficult. Couple this with a new generation of EE who cut their teeth on open tools expecting to use them in industry and we may well find established engineers favouring an opportunity to learn how to use this open source KiCAD thing they've heard so much about."

- Hugh Blemings 11 August 2020

In order to fulfill the point of having the CERN OHL v2.0 license, that the published work allows the work to be recreated, close attention should be paid to (among many other things) the longevity and ease of use of the file formats describing the design. Open source designs are increasingly superior with respect to establishing an enduring archive.

KiCad functionality meets or exceeds the requirements.

> "We are asking such hams and non-hams to contribute to our projects on a volunteer basis. We don't expect them to have to bring or buy their own software tool licenses, test equipment, $3000 eval boards, or $2000 chips. We want their skills - not their stuff. For these projects to succeed we will find ways to supply the stuff - and the volunteers will supply the much higher value engineering labor. Whatever investments we make in tools will be those that are highly leveraged."

- Wally Ritchie 11 August 2020

KiCad in particular does allow us to leverage donated skilled labor with a manageable learning curve. This fully acknowledges John Ackerman's point about the "unavoidable learning curve where different team members are used to different tools." KiCad's learning curve is manageable, the functionality meets our needs, and it's open source.

There have been and will be cases where a mix of different tools will used within the project. There are some cases where tool diversity may be unavoidable. It's up to the team members to negotiate the correct tool for the job, taking the baseline recommendations strongly into consideration, and justifying and isolating exceptions.

For example, a component may be more quickly and easily done in Altium by a volunteer. The component is then ready and waiting for the other volunteers that are doing the rest of the design using KiCad. The component integrates later without difficulty, as it's already been designed, documented, reviewed, and built. The component might not have been built at all, or built with errors, if the volunteer had been forced to use KiCad.

Verification and validation are especially important for the exacting reliability requirements for space. If an open source approach cannot deliver the failure modes analysis, verification, and validation, then those functions must be provided by whatever means necessary.

This extends to the components required for a communications payload. The notable uptake of RISC-V processors over the past year for space applications is of great interest at Open Research Institute. Our baseline recommendation for a flight computer is the Vorago M0 or M4. The Vorago is an ARM processor made extremely radiation resistant with a custom proprietary process.

Like the ARM processor in the Vorago product, RISC-V has substantial and growing terrestrial use. Experienced volunteers for either the Vorago or the RISC-V can be recruited “from the wild”. Expertise does not have to be created within the team, as it would in the case of custom space hardware. Leveraging existing familiar platforms for space products reduces risk and allows development to begin arbitrarily early in the design flow.

Unlike Vorago, if an issue was discovered, or an innovation developed, it could be spun into the RISC-V space hardware with fewer impediments than with a proprietary product. One would not have to rely on a vendor to disclose an issue or incorporate an improvement. This does not make improving an open source space processor design easy. But, it does reduce at least one major impediment: the friction of secrecy.

Tool choice depends on what is being done and what skill sets are available to the project at any particular time. The opinions that matter most are those of the people who will be doing the critical work and what they need to perform their job, given the requirement that it must be possible for others to recreate the work. Allowing consensus to appear as to what is appropriate for the task at hand is good management. Having an open source tools as a baseline is an ideal situation. Open source tools are not a rigid mandate in order to create open source work.