Licensing Open Source Hardware

Michael Weinberg

"Gu's shoulders slumped, and Juan got a closer look at the component boxes. Every one had physical signage: NO USER-SERVICEABLE PARTS WITHIN."
—Vernor Vinge, Rainbows End

For many people, a key component of open source is the license that is attached to the project. But what is a license, and what role does licensing play in open source hardware? This chapter covers the different types of intellectual property (IP) and the basics of IP licensing with respect to open source hardware.The good news is that most hardware is born open. Unless it is explicitly covered by a patent, most hardware is available to be copied, improved, and built upon by default. This chapter is written from the perspective of U.S. law. Of course, this chapter is merely an introduction to the topic of licensing and should not serve as a substitute for legal advice. It's always a good idea to talk to a lawyer before making specific licensing decisions.

Licensing

A license is permission to do something. Usually, that permission is conditioned on some sort of behavior or action. For example, a movie theater owner obtains a license from a studio to show a new movie. In return for permission to show the movie, the theater owner promises to pay the studio a percentage of the revenue generated by ticket sales. Because the movie studio owns the copyright in the movie, if the movie theater owner decided to show a movie without a license from the studio, the studio could sue the theater owner for copyright infringement.

Critically, if you do not need permission to do something, you do not need a license. In the movie theater example, the theater owner needed permission from the studio to show the movie because the right to show a movie is one of the rights that flows from owning a copyright in a work. However, the theater owner does not need permission—a license—from the studio owner to discuss the movie with her friend. That is because copyright law does not prevent people

from discussing movies, so having such a conversa-tion without a license does not open the theater owner up to an infringement lawsuit.

There is one additional twist on needing and giving licenses. A license is only worth something from someone with the power to grant it.The theater owner does not just need permission to show the movie—she needs permission to show the movie from the studio that owns the movie. Getting permission from some random person on the street will not be much of a defense when the studio decides to sue the theater owner for copyright infringement.

Finally, although many people think of licenses in terms of intellectual property, they are actually much broader than that. A driver's license is permission to drive on roads, conditioned on passing a driving test and maintaining a reasonably safe driving record. A concert ticket can be thought of as a license to enter a venue and attend a show, condi-tioned on paying money.

In the context of open source hardware (OSHW), most of the licenses involved will relate to intellectual property rights. By and large, these licenses will grant permission to do things like copy, incorporate, and build upon existing projects.

Open Licenses in the Context of OSHW

It is probably safe to say that the GPL and Creative Commons (CC) licenses are the best-known existing licenses among people interested in open source hardware. That renown is well deserved. These licenses and their brethren (which can loosely be thought of as "open licenses") have helped to build a massive common pool of writing, photography, films, software, and countless other types of creative expression. At the same time, they have helped to raise awareness about copyright more generally. Because of this success, they are worth considering specifically in the context of open source hardware.

The real genius of open licenses is that they take something that was viewed as a barrier to sharing and turn it into an asset. As will be discussed in more detail later, one of the unique aspects of copyright is that it protects automatically. If you create a type of thing that fits within the scope of copyright (like a movie or a piece of code), it is protected as soon as it is created. This copyright protection happens regardless of the creator's interest in protecting his or her work under copyright. That is why a license is so important, even for works that no one really wanted to protect with copyright in the first place. Without an explicit license, anyone trying to build upon that work is potentially infringing upon the creator's copyright. Remember, if you didn't include an open license for your files in GitHub (or wherever you store your files), they are automatically protected by

copyright to the fullest extent possible. Merely making the files public is not enough to make them open.

Instead of seeing this automatic protection as a barrier to sharing, open licenses view it as an opportunity to promote and encourage sharing. Using an open license is an affirma-tive declaration in support of building a commons.

An open license can also be used to force others to share. Remember, open licenses are built upon a legal right: copyright. That foundation allows creators to sue anyone who does not comply with the conditions of the license for copyright infringement. Includ-ing a "share alike" provision in a CC license is not a polite request that anyone who builds upon the work contribute back to the commons; rather, it creates a legal requirement. This legal requirement helps bring people and companies that do not care about building a commons into the world of sharing. If they want to benefit from the commons by copy-ing, building upon, or integrating the commons into their own work, the open license can legally compel them to add to it as well.

Adding hardware to the mix makes things a bit more complicated. The legal require-ment that forms the foundation for GPL and CC licenses is copyright. For things that fit easily within the scope of copyright—music, movies, photographs, and so on—this does not pose a problem. But for things that do not fit neatly within the scope of copyright—hardware being the most important example in this book—that copyright basis for tradi-tional open licenses can complicate things.

The fact that hardware is not protected by copyright does not mean that it is impos-sible to license. It just means that the process is not as straightforward as adding an open license to the design. In thinking about licensing open source hardware, it is critical to understand what you are actually licensing—and what you do not have any power to license.

Copyright, Patent, and Trademark: Rights That You Might Be Able to License

Before considering the parts of a given project that could be licensed in an open source hardware way, you must first understand a little bit about the different types of intellectual property. Being able to identify the contours of each will help you to understand which rights you might actually have and how you might want to license them. Copyrights and patents are designed as inducements to create. The theory is that, in return for spending the time and energy creating something and sharing it with the world, the creator receives a limited monopoly on that

thing from the government. Although they are often lumped together, copyright, patent, and trademark are actually complemen-tary sets of rights.

Copyright

Copyright is a type of intellectual property that most people encounter on a daily basis (Table 3.1). In large part, this daily interaction flows from the types of things that copy-right protects and how a work qualifies for protection. Copyright is intended to protect "creative" works. Generally speaking, creative works are the types of things that you would expect an artist to produce. However, in the con-text of copyright, creative works are defined fairly broadly. The category goes well beyond things like sculptures, paintings, and songs. Copyright essentially protects any slightly creative thing that is written down—notes to yourself, doodles on a pad, the finger paint-ing of a child. It also protects software code as a "literary work" in the same way that it protects a novel.

Table 3.1 **Copyright Tip Sheet**

Types of Things Protected	Relevant Features
Painting	Automatic protection
Books	Protection lasts for life of author, plus 70 years after death
Movies	
Photographs	
Artistic sculptures	
Code	

Table 3.1 Copyright Tip Sheet
Types of Things Protected Relevant Features
Painting Books Movies Photographs Artistic sculptures Code
Automatic protection Protection lasts for life of author. plus 70 years after death

As the name implies, copyright is primarily concerned with copying of the works that it protects. While plenty of other actions are regulated by copyright (for example, publicly performing a protected work, even if it does not create a copy, can violate the copyright), as a general matter copyright violations occur when a protected work is copied without the permission of the person who owns the rights to it. This copying can take the form of literal copying (e.g., duplicating a movie file) or nonliteral copying (e.g., turning a novel into a movie).

Getting a copyright is easy. Copyright automatically protects the types of works that fall within its purview.While there are many reasons to register your copyright, a covered work is protected from the moment it exists ("fixed in a tangible medium" is the technical term). That means that everyone is the owner

of thousands, and possibly tens of thousands of copyrights—whether they want them or not.

How long does that protection last? For quite a while. For most works, protection lasts for the entire lifetime of the author plus 70 years after his or her death. The reason so many of us feel surrounded by copyrights is because they are so easy to get and last for so long.

Patent

Whereas copyright focuses on artistic works, patents focus on "useful articles," which are things that do stuff1 (Table 3.2).You can think of these as the things that engineers produce. For most potential open source hardware projects, "things that do stuff" will form the heart of the project. For that reason, it is likely that most of the important parts of the project fall within the scope of patent law, not copyright law. Unlike with copyright, just because something is protectable by patent does not mean that it will ever actually be protected by patent. To obtain a patent you need to apply for it—a process that costs both time and money. In addition to filling out paperwork, you will need to prove that the thing that you are trying to patent is novel, meaning it is actu-ally new in the world.

Table 3.2 **Patent Tip Sheet**

Types of Things Protected	Relevant Features
Microchips	Need to apply to protection
Mechanical tools	Protection lasts for 20 years
Processes	Must be truly novel for protection

Table 3.2 Patent Tip Sheet
Types of Things Protected Relevant Features
Microchips Mechanical tools Processes
Need to apply to protection Protection lasts for 20 years Must be truly novel for protection

If and when you make it through the patent application process and are granted a patent, that patent will last for 20 years. While 20 years is a long time, it is significantly shorter than a copyright's protection (the creator's lifetime plus 70 years).

Finally, patent law and copyright law are mutually exclusive. In other words, something either fits within the scope of patent law or it fits within the scope of copyright law. In cases where an object seems to combine both creative and functional parts, the law does its best to separate the two elements out. The goal

of this process is to avoid giving copy-right protection to functional items outside of its traditional scope.

Trademark

In contrast to copyrights and patents, trademarks are all about identifying goods in the mar-ket and giving consumers' confidence in what they are buying (Table 3.3). If you have a headache and run to the pharmacy in search of some painkillers, you want to be sure that the bottle marked "Tylenol" actually came from the Tylenol people. Perhaps just as important, if you buy the Tylenol bottle and something goes wrong, it is important to know that you could sue the Tylenol people for harming you. What is critical about trademark is that it is a way for a manufacturer to identify itself in the marketplace.

Trademark is also limited in important ways. At their core, copyrights and patents are about copying or reproducing. Trademark law does not really care about copying for the sake of copying. Rather, trademark law cares about using marks in commerce and confus-ing consumers.

That means that not every use of a trademark qualifies as trademark infringement. Using someone else's trademark in an attempt to pass off your product as theirs is trademark infringement. But using someone else's trademark in a comparison or as a descriptor is not trademark infringement.

Table 3.3 **Trademark Tip Sheet**

Types of Things Protected	Relevant Features
Company/product names	Need to apply for protection
Company/product logos	No explicit term limit
Nonfunctional elements of appearance that help identify product	Does not prevent unauthorized use for comparison/compatibility purposes

Table 3.3 Trademark Tip Sheet
Types of Things Protected
Company/product names Company/product logos Nonfunctional elements of appearance that help identify product
Relevant Features
Need to apply for protection No explicit term limit Does not prevent unauthorized use for comparison/compatibility purposes

For example, using the "Arduino" trademark on a microprocessor that I create myself will be trademark infringement: A consumer might (wrongly) think that Arduino was behind my microprocessor as well. In contrast, describing my microprocessor as "Arduino-compatible" may not be trademark infringement. In

this second example, I am using Arduino's trademark to explain a feature of my own board, not to suggest that Arduino made it. It would be pretty hard to tell a potential customer that my microprocessor is compatible with Arduino without using the word "Arduino," and the law recognizes that fact. Similarly, I can use the Arduino trademark for comparison—"My microprocessor is five times slower and ten times harder to use than Arduino"—without running into trade-mark trouble.

Finally, the process of getting a trademark is something like an easier version of getting a patent.You still need to apply for the trademark and fill out forms, but the process will probably be easier and faster than a patent application.You may still need to hire a lawyer to help you with this process, but the bill should be significantly lower than the bill for a patent application.

Actually Licensing a Copyright, Patent, or Trademark

The previous discussion of the general types of intellectual property is all well and good, but if you are reading this book, you are probably a bit more interested in their application.

Licensing a Copyright

As mentioned earlier in this chapter, copyrights are the type of intellectual property that most people think of first. Copyrights are easy to get, so almost everyone has some. Also, because software is protected by copyright, copyright forms the core of the open source software movement. It is only natural to try and draw parallels when thinking about open source hardware.

But hardware is different, and not just because it is tangible. Because the core of most open source hardware projects is some sort of functionality—which is excluded from the world of copyright—copyright may not actually protect very much of an open source hardware product.

Of course, this does not mean that nothing in an open source hardware project will be protectable by copyright. Obviously, any software you include in your project is protect-able by copyright and should be licensed accordingly. Many (but not necessarily all) design files will be protectable by copyright as well. Also, in most cases, nonfunctional, decorative flourishes are exactly the type of thing hat is protectable by copyright.

For example, the Evil Mad Scientist iavolino development board has a cool de-sign screened onto its backside (http://sh p.evilmadscientist.com/productsmenu itinykitlist/180-diavolino).That design does not contribute to the actual working of

the board; it would work fine without it. But it is a nice artistic flourish—exactly the type of nonfunctional flourish that is protectable by copyright. In contrast, the mostly functional designs on the front of the board that identify the pins and various components are much less likely to be protected by copyright.

Thus the nonfunctional design elements of your project may very well be protected by copyright and, therefore, licensable under existing open licenses, such as Creative Com-mons. That protection will not extend to the functional parts of the project, however.

If you are looking for a good rule of thumb about which parts of your project might be protectable by copyright, ask yourself what would happen if the part in question disap-peared. If the project still works as expected, the part is probably protected by copyright (or totally unnecessary and should be removed, but that's an entirely different discussion). If the project stops working, or stops working as well, then the part may be the kind of functional element that is protectable by patent.

Licensing a Patent

At first glance, patents seem to be at the core of open source hardware. After all, patents protect things that do things—that's hardware! And if patents are to hardware as copyrights are to software, then the key to licensing open source hardware is to find a way to license the patents.

While reasonable on its face, this impulse breaks down in practice. First, as discussed earlier, just because something is the type of useful object that falls within the scope of patent does not mean that it is actually patentable. To get a patent, you need to prove that the thing itself is novel. For many projects, that simply will not be possible.While it might be a new open source hardware project, there may be plenty of earlier examples that would prevent it from actually being patentable.

Perhaps more importantly, even if your project could qualify for a patent, getting a patent is expensive in terms of both time and money. Obviously, this is a barrier to open source hardware projects that are bootstrapping or that do not have easy access to patent attorneys. Of course, these descriptions apply to many hardware start-ups.

There is also another barrier to obtaining a patent that is unique to open source hard-ware. One of the things that makes open licenses like those maintained by Creative Com-mons so popular is that they allow people to give away rights that they acquired for free. Everything that yo can license with a CC license is

protected by copyright, whether you like it or not. For manypeople, the fact that they made no effort to secure the copyright protection makes it that much easier to release it to the larger community.

Patent fundamentally shifts that calculus.You need to take affirmative, expensive steps to get that patent. Obtaining a patent with the specific purpose of broadly licensing it to the community can make the financial commitment a hard one to justify. This is especially true when other ways to share with the community—namely, not patenting your hard-. ware at all and doing a good job of documenting the project—are so much easier. For many projects, obtaining a patent will be like building an expensive cage to trap a wild animal just so that you can turn around and set the animal free again. It is probably easier for everyone to just keep the animal—and the invention—free from IP restrictions in the first place.

In many ways, this lack of automatic protection is a great strength of open source hard-ware. Open licenses were originally designed to circumvent a defect in the law—namely, the problem that copyright was automatically locking up creativity for the life of the author plus 70 years. In many cases, that defect does not exist for hardware. Most hardware is born free, and anyone is free to copy it unless the creator goes out of his or her way to protect it.

Unfortunately, this freedom inherent in hardware—in so many ways a positive thing—also comes at a cost: virality. Because there is an existing right (copyright) that requires permission for copying, that permission can be 'given conditionally. You are allowed to copy a copyrighted work under an open source license as long as you allow people to copy your work on the same terms. This factor has helped the idea of sharing spread be-yond communities that care about sharing for its own sake and into communities that just need to access the stuff. Without an underlying patent, however, it will be harder to pull that second community—the one that doesn't care about sharing but wants access to the stuff—into the world of open source hardware.

Ultimately, only two types of open source hardware projects are likely to be interested in getting patents. The first are projects attached to institutions that have a process in place to patent everything as a matter of course. In those cases, because the patents already ex-ist, it makes sense to find a way to openly license them.The second are projects backed by individuals or companies with lots of money who feel truly passionately about open source hardware. In those cases, the value of having a viral license that can spread the ethos of open source hardware will balance out the costs of actually obtaining the patent. If you choose to license your patent, you should talk to a patent because each case is different.

Licensing a Trademark

In the context of open source hardware, a trademark may become the most important type of intellectual property. This is because of the trademark's ability to identify the source of a product in the marketplace. The source is not only who conceived of it, but who actually assembled it and stands behind its quality. Even if you cannot control how people copy or incorporate your project into their own, you can control how they identify it.

Let's turn again to the world of open source software to see how this works. Mozilla owns the trademark for the Firefox web browser. Because Firefox is an open source browser, anyone can take the code and adapt it to their own purposes (e.g., Ice Weasel is a version of Firefox for the Debian operating system). However, while anyone is free to take the code, they cannot take the name along with the code. Only Mozilla's version of Firefox can use the Firefox trademark.

The result of this is that if you, as a user, find a Firefox installer online, you can be con-fident that it will install the Mozilla version of Firefox. It does not really matter where you find the installer—on Mozilla's own website or on some third-party site. As long as it is branded as "Firefox,"2 you know that Mozilla stands behind the browser.

The same is true, but potentially even more so, in the world of hardware. Someone who is getting ready to buy a piece of open source hardware wants to know who de-signed the product, just as someone who is getting ready to download some open source software wants to know who designed the software. But that open source hardware cus-tomer also cares about who actually assembled the product. While anyone can compile a software package with essentially the same result, two different people can assemble an identical piece of hardware with very different results.

For this reason, controlling the trademark of your open source hardware project can be very important. Although you will have to come to terms with people creating poorly made versions of your product (it will happen), passing those poorly made versions off as coming from ou should be a different matter. Registering your trademark helps you to build a reputati n for quality and reliability by giving you the ability to make sure that only products t at are up to your standards get to use the name.

This does not mean that you cannot license your trademark to others. In theory, you could license your trademark under the same types of terms that you license copyrights or patents.You could allow other parties to use your trademark as

long as they complied with conditions that forced them to share their derivative in the same way, or to not use the mark on commercial products.

In practice, it probably makes more sense to hold your mark a bit closer to the chest. Remember, a trademark is your project's identity in the marketplace. People will rightly assume that anything identified with the mark came from you and is up to your standards. If they find that not to be the case, it may undermine their confidence in everything that you do. If you choose to license your trademark, you should talk to a trademark attorney, because each case is different.

What to Do Now

By this point in the chapter, you may have concluded that the licensing issues surround-ing open source hardware are a bit more complicated than those surrounding open source software. If you haven't, you haven't been reading that closely. Unlike software, which is automatically and completely protected by copyright from the moment it is typed out, hardware is a mix of possibly copyright-protected elements, patent-protected elements, and entirely unprotected elements. As a consequence, it is unlikely that we will see an easy-to-understand, widely applicable, commonly agreed-upon license for open source hardware soon.

Fortunately, this does not mean that all hope is lost. Remember, the core of open source hardware is about sharing. Regardless of the license you do or do not use, sharing means documenting and engaging with the community. The truth of the matter is that your time may be better spent documenting your project than trying to figure out how to license each and every part of it.

That being said, licensing is important. It can make your intentions clear and give people confidence that they can build upon your project without creating a legal trap for themselves. The easiest part of licensing your hardware will probably be finding an open copyright license that has terms with which you are comfortable. To the extent that there are copyrightable parts of your project (including software), using the license you find will give people permission to use those parts of the project with legal clarity. This is a good first step, but you need to be realistic about what the copyright license does and does not cover.

No copyright license can cover the functional parts of your project. That means com-ing to terms with the fact that people may copy or build upon the functional parts of your project without complying with your copyright license. If they do, it is your responsibility to respond in a way that accurately represents your actual

ability to control the use of your project. If someone is copying functional parts of your project in a way you do not ap-prove of and all you have are a bunch of copyrights, don't threaten them with a copyright lawsuit. That's a threat that you can't back up—and it is an obnoxious one to make. Don't be that person.

Once you have your copyright house in order, you will need to think long and hard about patents. In most cases, patenting something just so you can license it openly will not make financial sense. But if you decide to go that route, make sure you choose an open license that is written specifically for patents. Simply adding a copyright license to a patent will just create confusion.

Finally, consider obtaining a trademark. In many ways, your trademark will become your identity in the marketplace. It will allow you to control how others perceive your project, and possibly whether people develop confidence in it. It takes some money and effort to apply for a trademark, but in many cases it is well worth it.

Summary

Where does this leave us? The good news is that most hardware is born open. Unless it is explicitly covered by a patent, all hardware is available to be copied, improved, and built upon. That puts hardware well ahead of software. The less good news is that licensing hardware is more complicated than licensing software. It may be a long time before there is something as simple and all-encompassing for hardware as a Creative Commons or GPL license, because the basis for licensing software is so much more straightforward than the basis for licensing hardware.

Where does that leave you? First, take the time to document your project. Regardless of whether the hardware is protected by intellectual property regulations, documenta-tion is a big part of what separates open source hardware from hardware that merely lacks intellectual property protections (and go read Chapter 14,Taxonomy of Hardware Documentation). Second, if your project has parts that are protected by copyright, pick a permissive license. This makes it easy for people to use your project, and it gives them guidance on how you would prefer they use it. Third, a trademark may end up being the most important type of protection for open source hardware projects looking to scale up. Being able to show the world that your hardware comes from you can go a long way in building your reputation and an enthusiastic community for your work.

Resources

Here are resources from which to learn more about the topics discussed in this chapter.

General IP News Sources

Public Knowledge: www.publicknowledge.org

Peer to Patent: wwwpeertopatent.org

Article One: www.articleonepartners.corn

Ask Patents: www.patents.stackexchange.com

EFF Chair to Eliminate Stupid Patents: www.eff.org/about/staff/daniel-nazer

Resources from the Copyright Office Copyright Office

circular: www.copyright.gov/circs/circOl.pdf

Resources from the U.S. Patent and Trademark Office about Trademarks

Trademark Basics: http://www.uspto.gov/trademarks/basics/index.jsp

Trademark Process: http://vvww.uspto.gov/trademarks/process/TMIN.jsp

---

1. In addition to traditional "utility" patents, you can get design patents, which are something of a middle ground between a copyright and a patent. They protect nonfunctional parts of objects, but only for 14 years.

2. And the person who put the installer there is not a trademark infringer peddling counterfeit ver-sions of Firefox.

About the author

Michael Weinberg is aVice President at Public Knowledge, a nonprofit digital advocacy group in Washington, D.C. As part of its advocacy mission, Public Knowledge has pushed to introduce the concept of open source hardware to policymakers and members of Congress. Michael oversees PK Thinks, Public Knowledge's in-house think-tank, and is involved in a wide range of issues focusing primarily on copyright issues before the Federal Communications Commission (FCC) and emerging technologies such as 3D printing and open source hardware.

License