

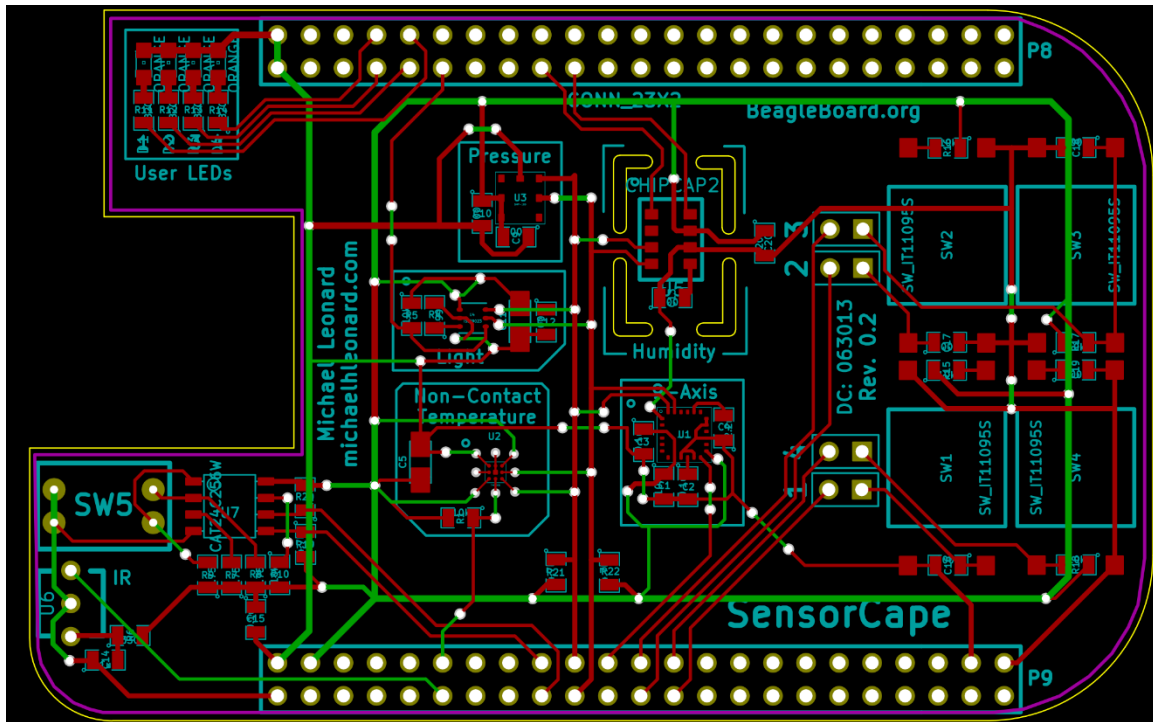
How to Design the Perfect PCB

BY MICHAEL LEONARD

Part 1 – michaelhleonard.com/how-to-design-the-perfect-pcb-part1

Part 2 – michaelhleonard.com/how-to-design-the-perfect-pcb-part2

PART 1

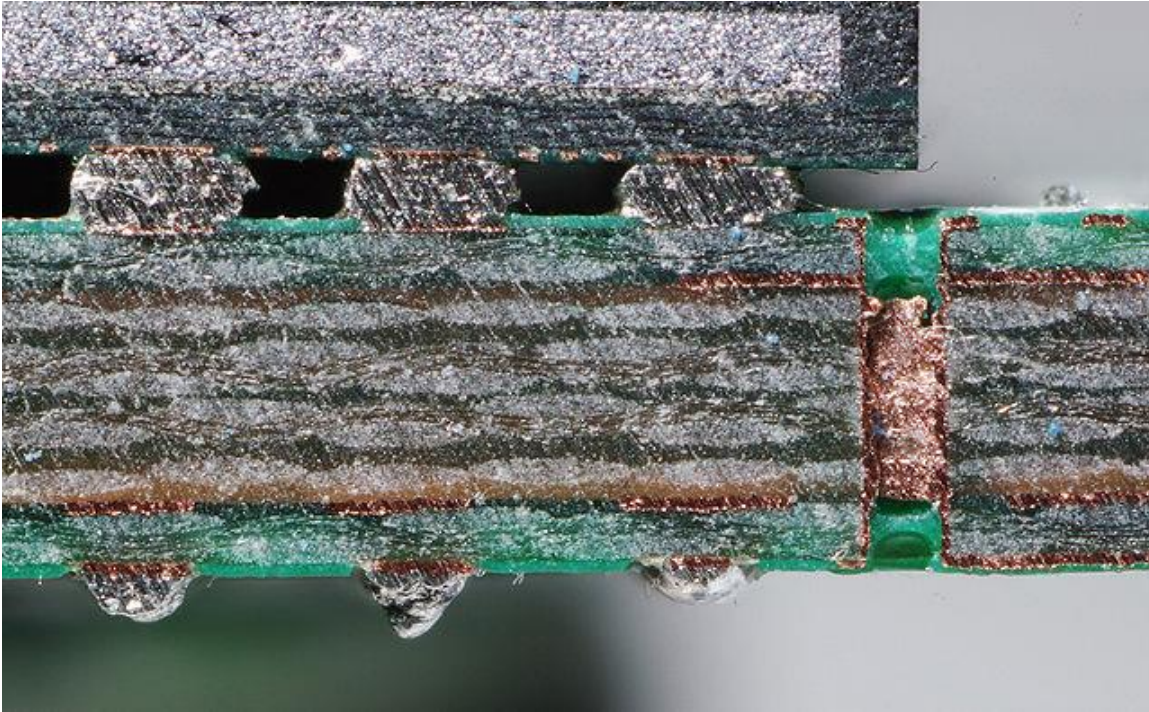


Breadboards are amazing for prototyping and are an invaluable tool to any electronics tinkerer, but when you really want to get serious you will need to learn how to create your own PCB.

Making a PCB is no simple task, however, with the right commitment, a little bit of time, and this guide you will be able to make a working PCB the first time around. If you are persistent it will even look good!

Anatomy of a PCB

When you are on your computer, everything is at kind of an abstract level and it can be easy to forget that you are working with a physical medium. Before you just start throwing together a design, I think it is useful to know what you are actually doing.



A PCB cut-away to reveal the inner layers

If you are already fairly familiar with PCBs then you can skip to “Designing Your Circuit.”

Board Materials

First we should understand what materials go into a PCB. At the most basic level, the base of the PCB is formed out of some sort of solid, non-conductive, material. This material is then laminated with a copper (or other metal) sheet, this creates the conductive surface.

The base material is usually a type of glass-reinforced epoxy known as FR-4. This is the most common material because it is flame resistant, cheap, and of course has a low conductivity.

For higher performance circuits (RF), there are other types of materials to consider such as ceramic or PTFE bases with various fillers. Since this article is more focused on general PCB design I will not go into details about designing for RF. Fortunately the EDN Network has posted a useful article on [choosing PCB materials for high frequency circuits](#).

Really these two materials are about all that goes into a bare PCB. When you send your design for manufacture (or do it yourself) the electrical connections are usually created by removing select copper portions of the bare PCB.

Layers

The **cheapest PCBs are single sided boards**. This means that they are just made of the base material, with a single sheet of metal over the top.

Single sided boards are incredibly easy to work with, if you are making your own PCB at home you will most likely be designing for a single sided board.

While the single sided boards are simple to manufacture and understand, they can also be a pain when laying out your PCB. Since you only have one layer of metal to work with, you cannot cross electrical connections without the help of some external jumper.

As a result of this complication the majority of simple commercial and hobbyist boards are created on double sided PCBs. On a double sided PCB it is a simple matter to cross electrical connections and this fact allows for more complex yet elegant designs.

For all but the simplest of designs I **recommend designing for a double sided PCB**. This is generally the most cost effective method that will leave you with the least headache possible.

As designs become even more complex, it may even become necessary to add additional layers to your design. This can be useful if your board has incredibly complex signal paths or if you are aiming for a compact design.

For the majority of users I do not recommend using more than two layers, if you do this and do not need to you will end up with an unnecessarily complex design that will cost more to produce.

Copper Traces

The copper traces on your PCB are easily the most important part of the design so it is important to understand what they're doing and what limitations you should consider.

As I mentioned earlier, copper traces are created by removing copper from the solid sheet that sits on top of the base material.

This means that the traces on your board are in fact just thin layers of copper. I don't know about you, but when I discovered that it came as a bit of a surprise (relax I didn't just find this out). For the longest time I assumed that PCBs were manufactured by pouring copper into a mold, letting it cool, and then somehow melting an insulator around the copper.

The thin sheet nature of these traces mean that there are some constraints to consider when routing your traces, most importantly, size considerations. All of these nuances will be discussed in detail in [part 2](#).

Vias

The final "main component" of a PCB would be the ever useful via. Vias are used in multi-layer boards to electrically connect one layer to another.

There are essentially three types of vias, only one of which is common in the hobbyist world. These via types are:

- **Through hole** - Most common type of via, a hole is drilled through the whole board and then electroplated so that it is conductive.
- **Blind** - A blind via is used in designs with more than two layers to connect a surface layer to an internal layer without going all the way through.
- **Buried** - Buried vias are similar to blind vias but are only used to connect internal layers.

Other Things

After discussing PCB materials, possible layering options, copper traces, and vias we have pretty much covered the basics for what makes a PCB what it is. Understanding these things certainly gives you enough information to make your own working design, but there are still some other concepts to consider.

Some other PCB concepts to explore:

- **Soldermask** - If I had to guess what you think of when I say "PCB" I would bet that it is probably something green. Did you know that PCB's are not this color as a result of what material is used? This is in fact another layer that is applied to the board after manufacturing. The purpose is to keep solder paste from spreading where it shouldn't be, but it also has the effect of giving the board a definite style.
- **Fiducials** - These are special markings on your board that allow a pick-and-place automated assembly machine to calibrate itself. Fiducials are usually just a circle where the soldermask has not been applied with copper circle in the middle. This makes the point appear fairly reflective.
- **Silkscreen** - This is also another layer of the PCB added after fabrication. Silkscreen is used to provide visual cues to the user, document board information, identify proper component placement, or for branding. There are conflicting suggestions on proper usage of silkscreen and I will address this in [part 2](#).
- **Copper fill** - Deciding whether to use a ground/power plane in your design will be an important decision to make during the design stage. The most common reasons for using a copper fill are to suppress noise on the ground circuitry, dissipate heat from a particularly active device, or because someone told you that was the way to do it.

That's enough of learning what goes into PCBs though, let's discuss how to get started on your design.

Designing Your Circuit

Before you can consider any physical designs or schematic connections **you must have a clear idea of what you want your design to do**. This means taking some time to sit down and define what you want to accomplish, consider the challenges, and pick the right components for the job.

Determining Your Goals

The first step in designing the perfect PCB is to have a well-defined set of goals that you would like your design to accomplish. To steal a little bit from the business world, you should always set SMART goals for your project, this means:

- Specific
- Measurable
- Attainable
- Realistic
- Time Bound

As a personal example, I have started working on another side project for my own use. The bathroom in my apartment is too dark in the evening for me to get around in, but when I turn the light on it is way too bright and wakes me up.

To fix this little issue I thought I would just go buy a small lamp. Unfortunately, I am a little bit too picky and couldn't find a lamp that I liked. That is when I had the idea to design one of my own. Of course, I'm not talking about just any lamp. I wanted a multi-color, adjustable brightness, wirelessly controlled lamp.

Sounds cool right? Of course it does! So before the idea was able to leave my head I jotted it down in my notebook and began planning.

At this point, my goals were pretty broad, let's take a look at what I had:

- Multi-color lamp
- Adjustable brightness
- Wireless control

Unfortunately none of these goals are very specific at all. What do I mean by multi-color? Is that two colors, three, or any variable color? What is adjustable brightness? I mean technically on and off would be two different brightness settings right? Wireless control? What, do I want to use Wi-Fi, Bluetooth, infrared, RF, Zigbee, sound? Any of these options would be possible.

Revising the project goals to be SMART led me to the following list of goals:

- A continuously adjustable high-brightness RGB LED filtered through a fogged acrylic cover for even light dispersal.
- Continuously variable brightness control that will allow me to choose any brightness setting between completely off and fully on.
- Bluetooth low energy 4.0 wireless specification interface, controllable from an iOS or Android devices as well as an optional dedicated controller.

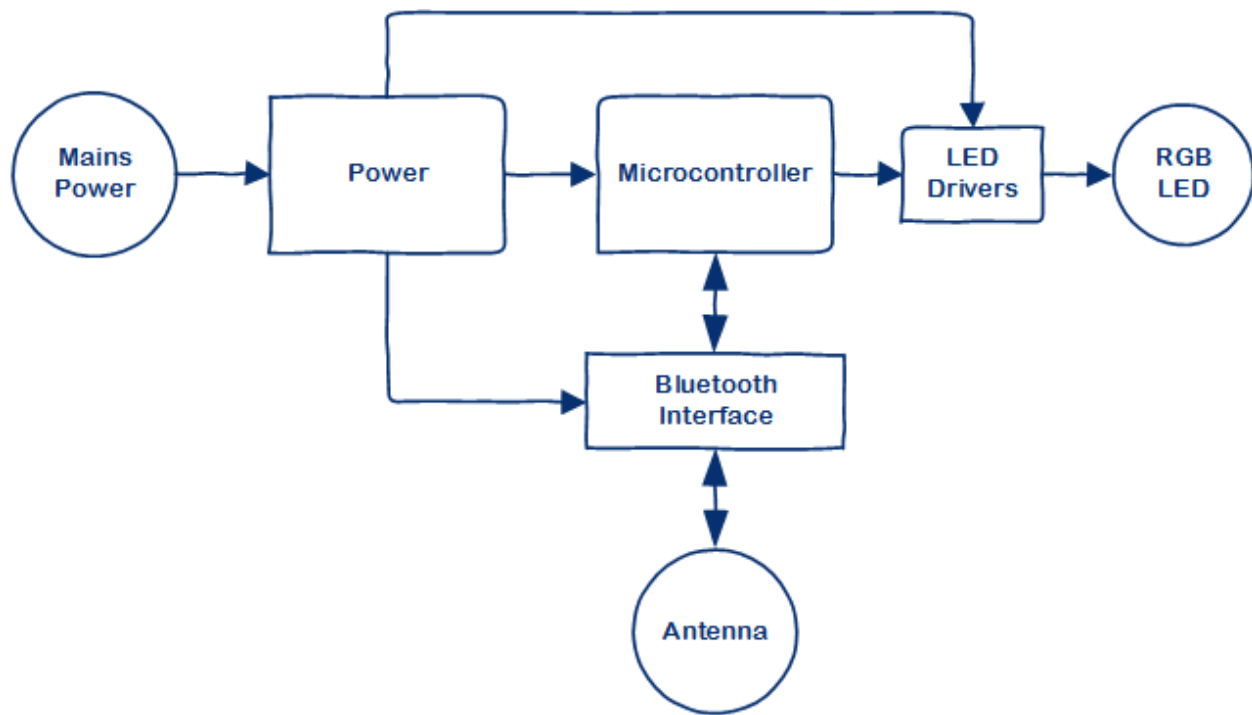
With the exception of "time bound" these goals meet all of the criteria of a SMART design and allow me to proceed forward with a clear vision of what I want to accomplish.

By doing your research first and setting SMART goals for your project you place yourself on the right track to create that perfect design.

Visualizing Your Design

Now that you have a clear idea of what you want, it is time to start designing it. Before you start scouring the internet for parts or drawing crazy schematics in your notebook, I would advise you to **take some time to develop a clear picture of how you want your final design to function.**

Try to determine how your parts will work together to achieve the goals you set. **This is a good time to be thinking about your design from a system level.**



System Level Design for My Bluetooth LED Lamp, yes I make Visio drawings for fun.

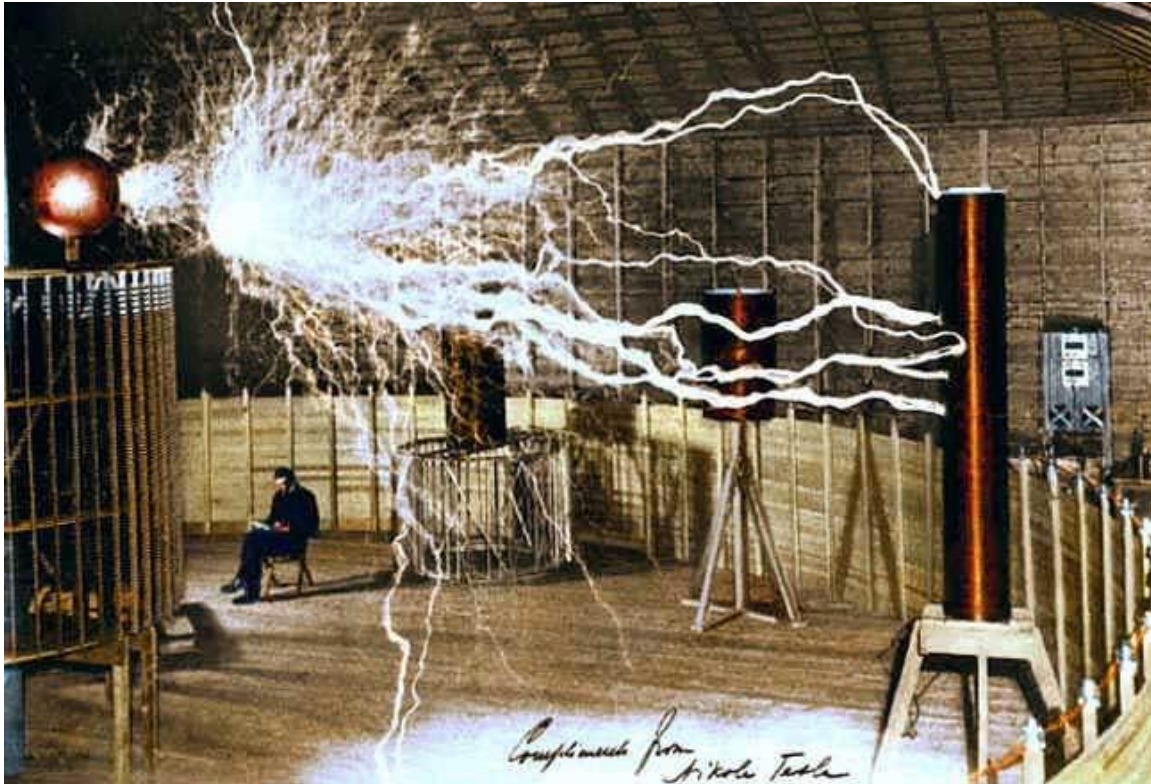
You may not know specifics like what supply voltages you will need or what connections need to be made, but you will be able to consider how each component will rely on the others and what additional components they will add to your design.

This is also a good time to consider the aesthetic aspect of your design. Are you trying to fit a certain form factor? Do you need to consider ergonomics (for example if you are designing a game controller)? Will you be able to pick up your design a year from now and understand exactly how it works? These are the types of details that, while seemingly insignificant, can be the difference between a good design and a great design.

I know all this talk of visualization may sound cheesy and like it won't really get you anywhere, but I assure you it is worthwhile. If you don't want to believe me, consider this quote from Nikola Tesla's autobiography where he describes his creative process:

My method is different. I do not rush into actual work. When I get an idea I start at once building it up in my imagination. I change the construction, make improvements and operate the device in my mind. It is absolutely immaterial to me whether I run my turbine in thought or test it in my shop. I even note if it is out of balance. There is no difference whatever, the results are the same.

Of course the vast majority of us aren't at the level of crazed genius that is Tesla, but the idea behind this method is all the same. **By visualizing your design beforehand you save time, money, and frustration.**



Tesla Turns on a Night Light

Choosing Parts

This is perhaps the most tedious step in the design process, but is crucial to a successful design. Choosing the right part for your design could be the difference between finishing your project and giving up in frustration.

All integrated circuit manufacturers work hard to make their designs robust and perform their function at the lowest price they can, but not all companies are equal. This is especially true when it comes to making their parts easy to use.

Since there are hundreds of thousands (millions?) of different components on the market, it isn't possible for me to give a complete rundown, but what I can do is provide some general guidance on how to select the best component for your purpose.

- **Check availability.** The last thing you want to do is put weeks or months into a design only to find out when you go to buy your parts that a crucial component is out of stock and will not be available for a few more months. Choose a part that shows a large inventory and optimally is available from multiple distributors.
- **Consider where the component is in the product life-cycle.** You generally don't want to get a component that is no longer in active production, but if your project is just a one-off build then this may not be a big deal.
- **Make good use of the parts filters.** Most distributors or part finding tools offer some way to narrow your search criteria. Make use of this not only to reduce the amount of parts you have to look at but also as suggestions for alternative components. As a simple example let's assume you have decided

you want an LED with a millicandela rating of at least 80 mcd. Instead of filtering for components that have a rating of exactly 80 mcd, filter for any component with at least 80 mcd then sort by price, forward voltage drop, or current draw. This method may save you money while also getting you a better performing component.

- **Be aware of minimum quantities.** Some components are only sold in large lots, be aware of this when choosing your components so that you aren't forced to do a redesign.
- **Know what package you are getting.** All components are delivered in some type of package that allows you to attach them to your board. Some components are offered in multiple package styles that are usually incompatible. If you are planning on making your PCB at home you should try to avoid the very small packages such as no-lead packages or chip-scale packages. These can be difficult to solder without proper equipment.
- **Understand the part!** This is the final and most important guideline that I have. You should always fully understand the part before deciding to use it in your project. Some components can require a microcontroller or microprocessor, an external clock, or a special PCB design. Being aware of these requirements beforehand will help you avoid headaches in the future.

When it comes to actually finding parts, I like to use the search capabilities provided by my favorite vendors, this way I know the product will be available and I can choose components based on the actual cost to me and available inventory.

Another way to search for parts would be to go to a particular company's website and browse their parts catalog for a solution. For example, if I know I need an ADC for a project, I may start with a company that is well-known for their ADC products such as TI. This has the advantage of often leading to a highly usable solution.

My three personal favorite avenues for finding parts are:

1. Mouser.com - Mouser is a popular worldwide component distributor that has a wide range of products. Their selection is (in my experience) not quite as large as Digi-Key's, but I prefer their website design, the better filtering system, and the more logical component organization.
2. DigiKey.com - Digi-Key is another popular worldwide component distributor. They probably have the largest component selection of any distributor, have great customer service, and are fast to ship. Overall I would put Mouser and Digi-Key about even, and certainly at the top of the list.
3. Octopart.com - This is a relatively new service that is like Google for electronics. Octopart searches through many different distributor channels for the part you want. There are many things I like about this service. They always put the datasheet in an easy to find spot, they provide a useful product summary, and show you price comparisons from different distributors. But there are drawbacks, Octopart still seems a bit unrefined and I don't think it is worthwhile to compare prices from different distributors unless you are only buying a single part or are buying a massive quantity of components and shipping costs are negligible. In a few years Octopart may become the standard for finding your parts, but for now it still has some room to grow.

This is just my preference, do you have a better method of finding parts? Let me know.

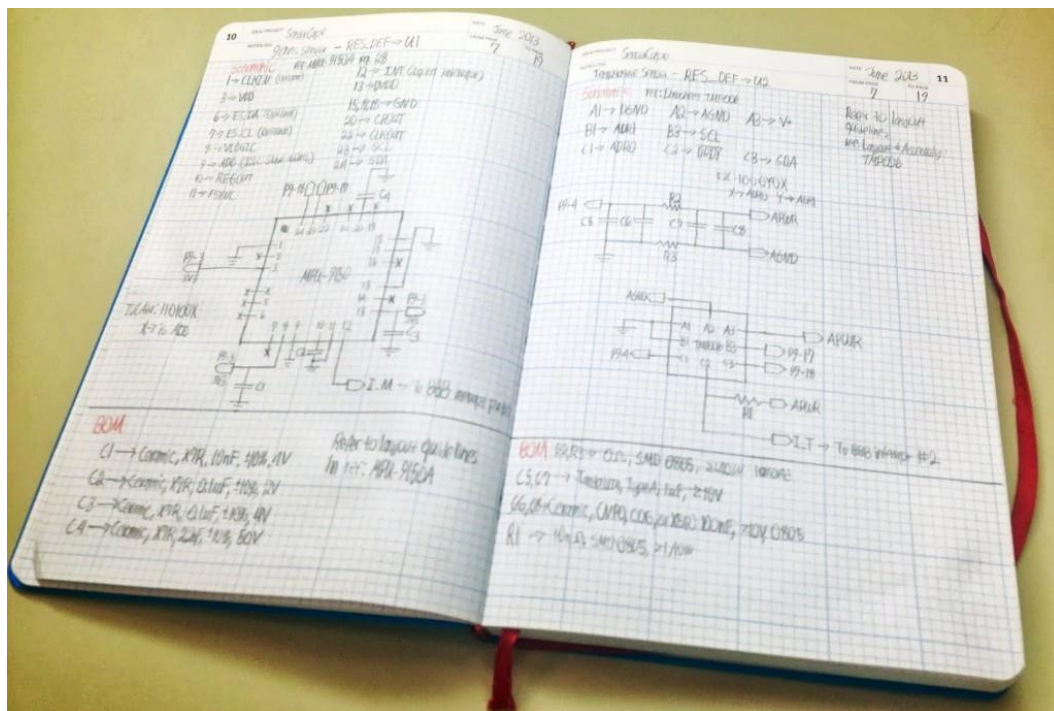
Sketching Your Connections

The final step before we switch over to software is to get a "first draft" of your design onto paper. Nikola Tesla would not approve, but he's not around to stop you so don't worry. This is a good way to get the specifics of your project organized in a coherent manner. I like to separate each system level block on a new page.

I also think it is useful to make a note here of what each important pin on the component does. It probably wouldn't hurt to get started on your bill of materials as well, this may change as your design evolves, but it at least serves as a good starting point.

In addition to the basic information, you may also want to include some more detailed info about the part that you think may be important. For example, it may become tedious to refer back to the datasheet for I2C address information or possible pin configurations, these are good details to include in your notebook.

For sketching my designs, both electrical and mechanical, I like to use this [excellent Maker's Notebook from MAKE](#).



My Maker's Notebook

With a well-defined grid, a page marker, the included organization features, and the extra pointers they include in the back, this isn't your average notebook. The Maker's Notebook is specially designed by makers, for makers, and I love mine.

After you have finished sketching your design, you have completed all of the pre-layout checks and are ready to move on to the physical design of your PCB.

PART 2



The featured image for this post is the Manga Screen designed by [Elias Bakken](#), you can learn more about it and explore his work at [Hipstercircuits](#).

When I set out to design my first PCB I was told "Well, it's your first PCB so it probably won't work anyways, but that sounds interesting." Even though this was discouraging to hear, I didn't let it stop me and I ended up with a working design. I now want to take my experiences as well as the experiences of others and make it as simple as possible for you to design your own PCB.

Putting the Design into Software

Now that you have an idea of how you want the project to turn out, it's time to start moving the design onto your computer

Choosing a CAD Package

The first step is to choose what CAD package you will be using in order to design your PCB. There are a dizzying amount of options on the market and I don't know that I can even attempt to list them all, but here are some of your options:

- [EAGLE](#) (Free with limitations, upgrades from \$70 - \$1640)- A powerhouse in the hobbyist world, EAGLE probably has the most community support, but has some quirks that led me to choose a different suite.
- [KiCad](#) (Free and open source, no limitations) - This is my current personal favorite. As a FOSS option that is quite capable, you have nothing to lose by trying it out. I think it is better than EAGLE, and you can't argue with that price.
- [gEDA](#) (Free and open source, no limitations) - Another excellent FOSS option, gEDA and KiCad are often compared. I tried them both and felt that KiCad was a bit more polished, but you may want to try gEDA for yourself.
- [ExpressPCB](#) (Free, no limitations) - It seems like good software, I have never used it personally but enough people seem to recommend it. ExpressPCB is backed by, you guessed it, ExpressPCB so this makes it very easy to send your design for manufacture.
- [DesignSpark PCB](#) (Free, no limitations) - I had never head of DesignSpark until I started writing this post. I have to say I am very impressed by what they show on the landing page so I may check it out.
- [DipTrace](#) (Non-free, prices from \$70 - \$900) - DipTrace comes highly recommend, I prefer free and open-source software (FOSS) so I have never used it, but it may be worth your time.
- [PCB123](#) (Free, more powerful packages non-free) - This software from Sunstone looks very promising, I'm sure there are some hidden limitations that I couldn't find, but I will be checking out this package when I start my next design.
- [Mentor Graphics PADS](#) (Non-free, cost unknown, likely >\$5000) - This software seems to have fairly poor user feedback, I don't know why you would choose to voluntarily use it. From what I can tell, most of its current user base is from engineers in a corporate setting that are required to use it for backward compatibility.
- [Altium Designer](#) (Non-free, >\$5000) - Altium Designer has excellent user feedback, if you are looking for a professional suite to work with and can handle the price tag, Altium may be your best option.
- [OrCad PCB Designer](#) (Non-free, >\$5000, upgrade path can push you north of \$25000) - Cadence OrCad is a behemoth in the EDA industry, unfortunately they also throw that weight around by charging insane prices. If you absolutely need the features offered by OrCad then it may be worth your investment, but for the average user this is overkill.

So I wasn't lying, it's a big list! When we narrow it down and look at what's practical though, I think the first three options are the most interesting for hobbyists.

As I said in the summary, EAGLE is a very popular package and has a huge community of hobbyists that use it for their projects. Unfortunately there are some nagging issues with EAGLE that have caused me to abandon it in search of greener pastures

For various reasons, I ultimately decided on KiCad, but since this isn't a comparison article I won't go into details. If you are curious about my reasoning, [this article](#) does a good job of highlighting what I didn't like about EAGLE, though I don't fully agree with his assessment of KiCad.

During the remainder of this article I will be using KiCad for explaining concepts of PCB design. I will do my best to cover the topics at a high level so you can easily transfer these ideas into the CAD package of your choosing, but if you are undecided **I wholeheartedly encourage you to try KiCad for your next design.** If you are interested in specifically learning how to layout a PCB in KiCad you should subscribe to this blog or check back regularly.

Best Practices for Electrical Schematics

As far as I know, there are no standard guidelines for drawing schematics. Most of us learn by trial-and-error, pick up the habits of our teachers, or just never learn. I think it's time that these guidelines found a home.

The purpose of an electrical schematic is simple, to describe how components are electrically connected. A schematic does not necessarily indicate the physical relationship between components, though a good schematic will make this obvious when necessary.

The difference between a good schematic and a bad schematic is a matter of how easy it is to understand. If another person is able to look at the schematic and simply understand it, you have made a *good* schematic, if another person has to look at your schematic and decipher it, you have made a *bad* schematic. Here are my suggestions for making a *good* schematic.

Labels and Comments

Labels and comments on your schematic come at no cost to you so you might as well use them. **You should always label your components.** Most (or all?) schematic editors will do this automatically for you, but if yours doesn't then you **must** do this. There are some generally standard designators in use today, the most common are:

- **R, C, L, and D** - These are the default designators for your most common passive components. R is for resistor, C is for capacitor, L is for inductor, and D is for diode (including LED). These are standard designators and you should not use any other designator for these components. The only exception to diode marking is if you are using a Zener diode, in that case refer to it as Z.
- **M and Q** - Standard designators for transistors. MOSFETs are referred by M while BJTs are referred by Q. M can also stand for Motor, if you are using a motor in your design then refer to all transistors as Q.
- **T or XFMR** - Transformer
- **S or SW** - Switch, this includes pretty much any type of switch, even push-buttons. You can use either designator, but you should only use one. Consistency is key, I recommend SW.
- **X or XTAL** - Generally refers to crystal oscillators, X can also be used to designate subcircuits. I recommend XTAL.

- **U or IC** - Designates an integrated circuit, U is the most common designator but IC is just as clear and well understood.
- **TP** - Test point, these are handy and you should make generous use of test points in your circuit while it is in the prototype stage.
- **JP** - Jumpers
- **J or P** - These designate jacks or plugs, respectively. Jacks are generally female while plugs are generally male.
- **F** - Fuse
- **FD** - Fiducial
- **BT or BAT** - Designates a battery or battery connector
- **X** - Parts not covered by the above rules. You should leave a label near these parts to note their function.

These rules are extremely helpful when sharing your designs with others, or when you need to come back to your design in the future. However, having correctly referenced parts is not necessarily enough to make a good schematic. It is also good practice to **comment on non-standard parts or on important performance details**.

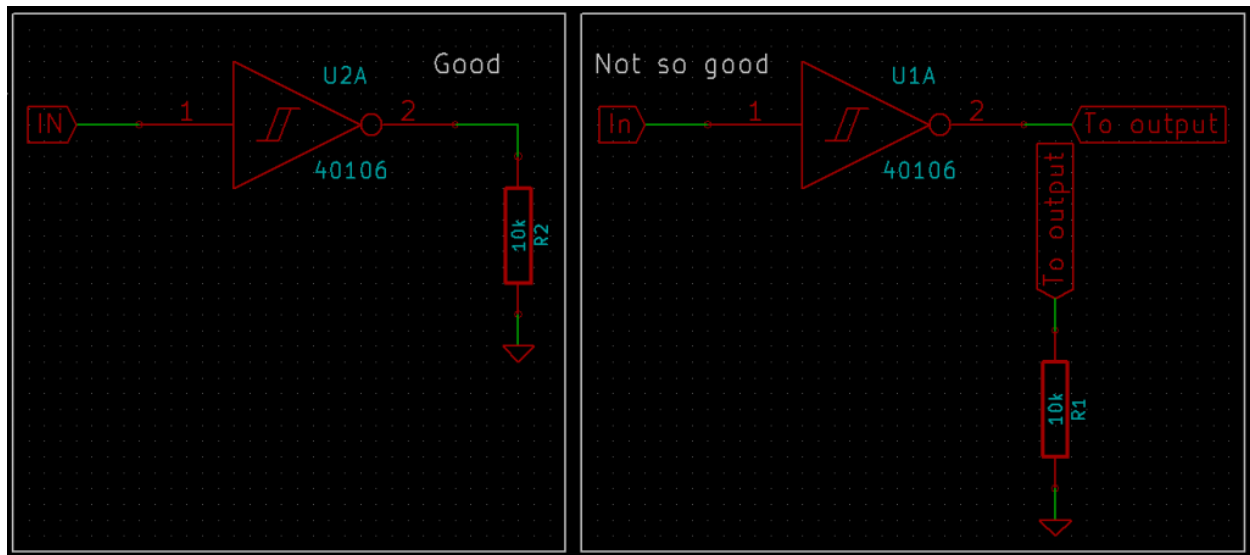
For example, if you have a component that requires a very large power trace or special shielding then you should note this on the schematic. For more advanced circuits, such as RF designs, you would also want to note required trace lengths or impedances.

There are too many examples to list all of the situations in which you should make use of comments, but a good guideline is to **ask yourself "If I were to look at this circuit in one year, would I still know what it is doing and how to take it to layout?"** If it passes this test then you are on your way to creating a successful design.

Another recommendation for making your schematics clear is to be mindful of where your labels and comments are placed. If a label is placed over another label or over a component outline then you might as well not have it there at all because it won't be readable. **Take the few extra seconds to move your labels into a logical position near the component while not overlapping other components or labels.**

The final guideline for proper labeling is to **remember to label your most important nets**. Don't go overboard with this, but a few net names to remind you of functionality is an incredibly simple way to ensure that you are making an understandable schematic. Additionally, **keep the names as short as is reasonable, use all caps, and separate words with underscores.**

This means that instead of "Pin going to output" as a label you would want to do something like "TO_OUT" or even better, just "OUT." This will ensure that you have readable schematics with signal names that are obvious and intuitive.

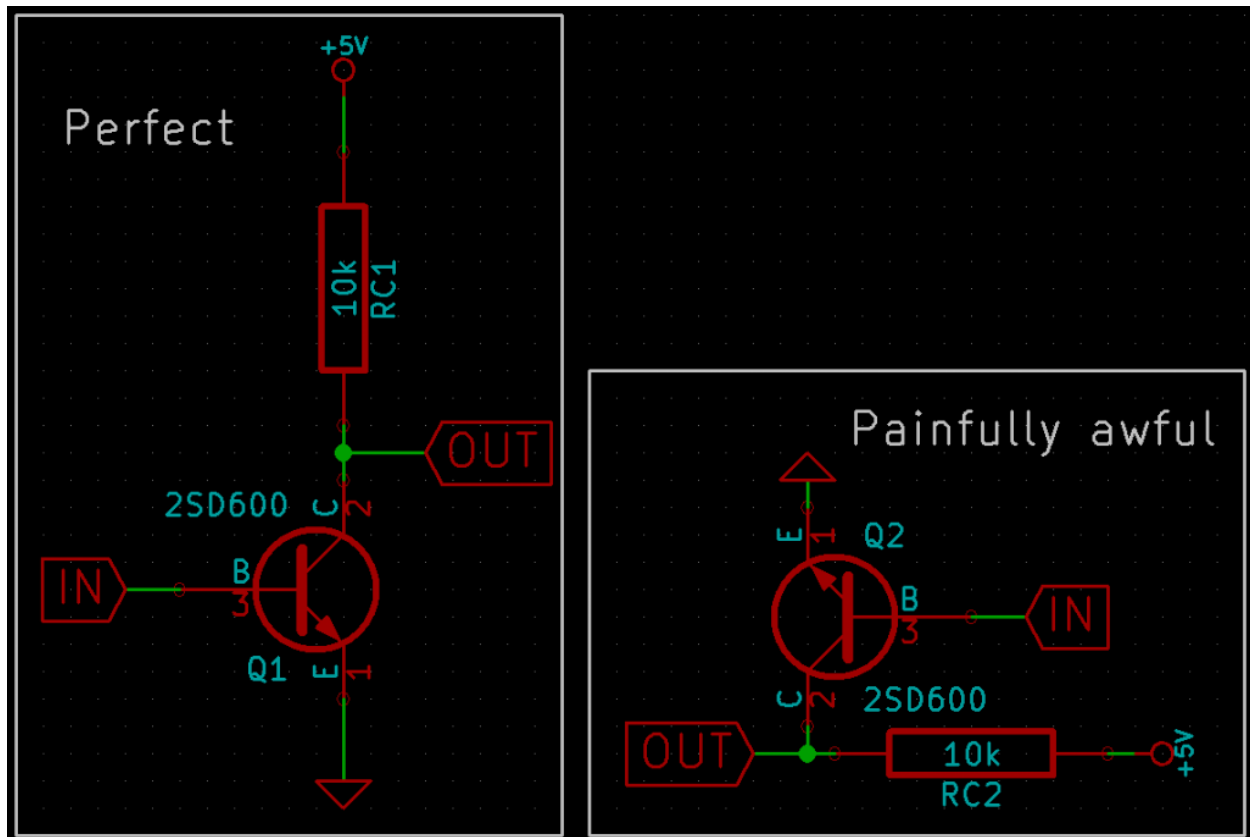


An example of good labeling against poor use of labels.

The Logical Schematic

I'm not certain where the convention came from, but it is always customary that **inputs come from the left, outputs go to the right, power comes from the top, and ground or negative voltages go to the bottom.** I recommend following this convention whenever it is possible and reasonable to do so.

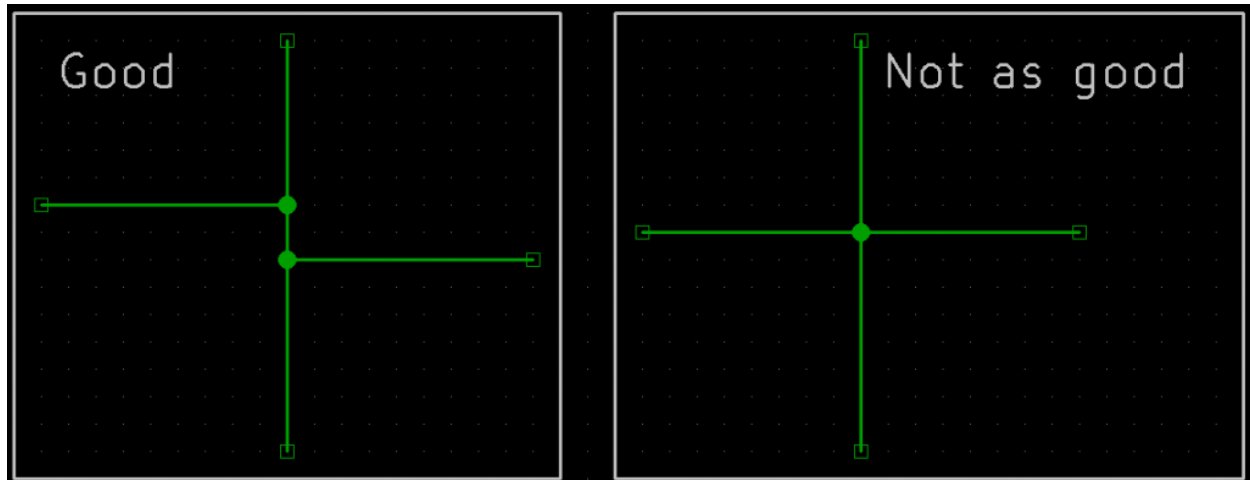
Of course you can't always do it, but at the very least try to **separate your power pins from your I/O pins.** If you have multiple voltage rails, the more positive voltages are generally higher on the schematic, though this is not a do-or-die rule.



The standard common emitter vs. a completely illogical implementation of the same circuit. It was actually difficult to draw the circuit on the right.

Dot your i's and cross your t's. Well, something like that, this is a convention that comes from the days when low resolution photocopying was common. It is universally accepted that you should **make a very clear dot where two wires form an intersection**, your CAD package will usually handle this for you but it is good to keep in mind.

Related to crossing connections, you should also try to **avoid 4-way connection points**. This is another recommendation from the days of photocopying circuits, but it never hurts to design for longevity.



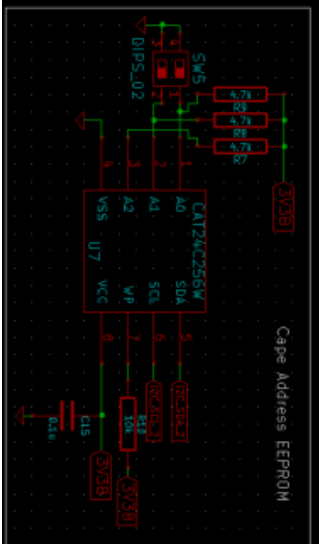
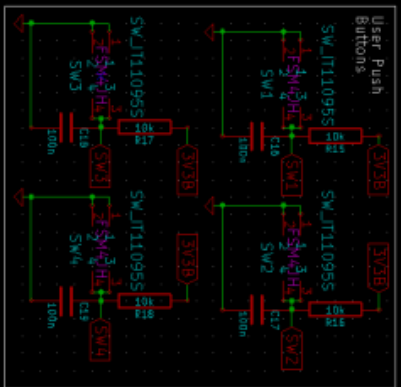
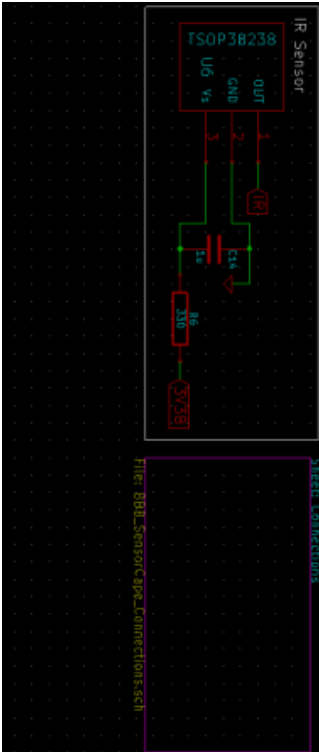
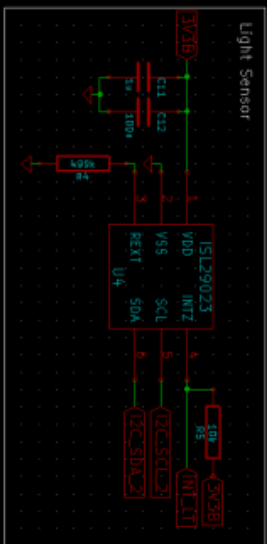
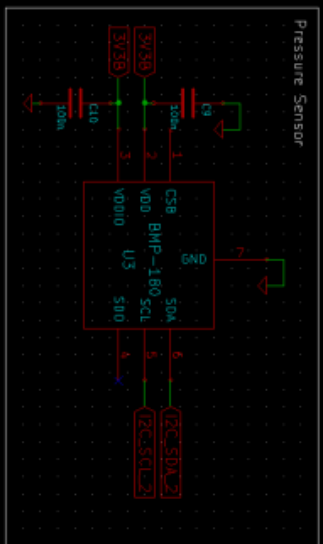
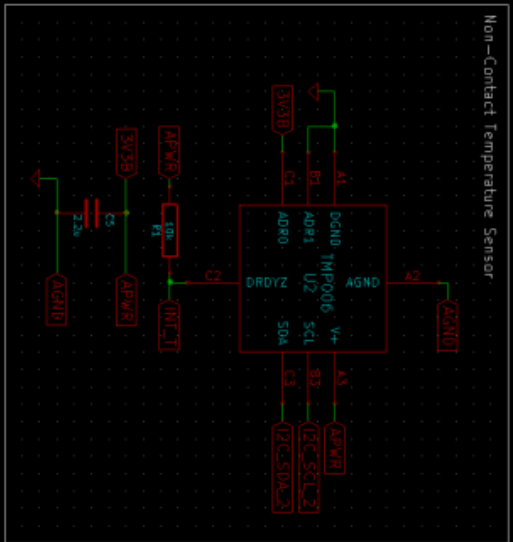
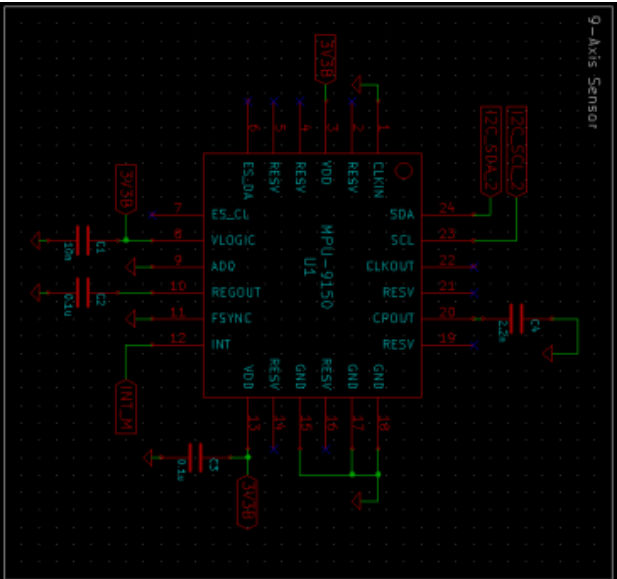
The proper way to connect three wires.

Using Hierarchy the Right Way

The final pointer in ensuring that your schematic makes sense is to utilize hierarchy effectively. This means that you **separate logically different parts into a new sheet**. By all means, if you can fit your entire design into a single sheet without cramping it together and still following the other rules, you should do that. Otherwise you might as well separate functionally different parts of the design into separate sheets.

There are different ways to do this in each CAD package, but the basic idea is the same. **By keeping related components near each other and avoiding the clutter of other components you will be able to more easily verify and debug your design.**

If your schematic doesn't necessarily require multiple sheets then you should still do your best to attempt to introduce a bit of order into the chaos that is an electrical schematic. My preferred method is to draw a box around a functional unit and then place a label inside the box to indicate what that design does.



While this schematic may not follow all of the rules in this guide, it does a good job of demonstrating functional separation.

Other Tips

In addition to the general guidelines above, I have managed to compile some other tips that will lead to a successful design. If you have some of your own, feel free to submit them to me (along with an explanation) and I will add them to the list.

- **Show decoupling capacitors near the device they are protecting.** This is one of the few devices where it is important to indicate physical presence of a component. Decoupling capacitors are used to smooth out the ripples at the power supply of a component, in order to effectively do this, they need to be placed physically close to the component. This proximity should be made clear in the schematic.
- **Design for easily printable schematics.** While most schematics today are handled electronically, it is not at all uncommon to want to print the design out, either to share it in a meeting, or to review it with a pen in hand. For these reasons, you should always make sure your schematics are designed to be easily read and analyzed at whatever paper size is common in your area. For the U.S. this is 8.5" x 11" for Europe, the most common size is A4 which comes out to about 8.3"x11.7" or 210mm x 297mm.
- **Make your schematics understandable even if they are printed in black and white.** Many office printers are not capable of printing in color. I also often find that I prefer the look of monochrome schematics when they are printed. If you design your schematic to be readable and understandable without color then it will make it easier to analyze your design later.
- **Air wires.** Use them when you have to, avoid them when it's practical. The point is to make your schematic as easy to understand as possible, so if it makes sense to put a label on a wire and connect it to others by using that label then feel free to do so. Just keep in mind that air wires can make it difficult to debug your circuit later since you must manually search for all of the connections.
- **Consider revision control.** Undoubtedly, you will end up making several versions of your original design. It is best to plan ahead for this and to manage your different versions intelligently. I like to use a combination of [git](#) with [bitbucket](#) to track my designs, but there are other methods such as manually saving version numbers or date codes into the project name.

Like I said, these are just a few tips I have picked up from my experience, if you have your own then submit them to me and I will add them to the article. Now that you know the best practices for getting your design into the computer, get to work! The next step is to start with the physical layout, this is where it starts to get interesting and you will want to have a solid schematic under your belt before you move on.

Final Preparations for PCB Layout

Now that you have your schematic entered into the computer and all the connections have been validated, you can move on to physical layout of your PCB. This is the most complex part of the design process and there are far too many different possibilities for any single guide to provide a full list of guidelines in a sensible manner. That being said, I will do my best to provide general guidelines for producing a manufacturable and electrically sound PCB.

If you are interested in the specifics of specialized PCB design techniques, how to use the various CAD tools, and other general workflow improvements then subscribe to this blog (in sidebar on desktop, below on mobile) to continue receiving articles and guest posts on improving your designs.

Choosing a Manufacturer

I'm sure it may seem like a strange suggestion to choose your manufacturer before you begin board layout, but I assure you there is a good reason for this. No two PCB manufacturers are the same and each one has different limitations.

Definitions: A "mil" refers to a thousandth of an inch. While most engineering processes have standardized on the far more understandable metric system, physical products still rely on imperial units. "Clearance" refers to the separation from the edge of one trace to the nearest edge of another.

While one manufacturer may be able to produce 6 mil traces with 6 mil clearances (3/3) another manufacturer may be limited to a (8/8) setup. **You do not want to design your board and route all of your traces at a very small width only to find that your manufacturer cannot produce the board at that specification.** This could leave you resorting to much more expensive manufacturing options, or receiving a non-working PCB when the manufacturing house attempts to manufacture your design anyways.

If you thought there were a lot of CAD packages to choose from, then you would be truly overwhelmed by the number of different manufacturers and assembly houses there are. Rather than compiling a huge list of all of the PCB manufacturers and assemblers, I am just going to post a short list of those that I either have personal experience with or that have made a strong impression on me.

- [OSH Park](#) (Low cost fabrication, U.S. based company) - If you just need to do a small order of PCBs, don't need assembly, and your design fits within their limitations, I fully recommend OSH Park. The turnaround time isn't stellar, but in the 2 weeks you may end up waiting for a board, you can always start working on other projects.
- [Advanced Circuits](#) (Incredible flexibility and turnaround time, also offer assembly, U.S. based operations) - I have never used their services personally but I have to say I am impressed with what they offer and others seem to be happy with their results. Their prices for short run fabrication are much higher than OSH Park, but their medium run prices (50+ boards) are perfectly reasonable and they offer a wide range of fabrication options. They also offer a "Barebones" option that is excellent for prototyping.
- [Gold Phoenix](#) (Cheap higher quantity orders, non U.S. based) - I have also never used Gold Phoenix but they are apparently often used by SparkFun and offer good prices on larger orders. Their services are based out of China and their boards are not as high of quality as Advanced Circuits, but they are worth a shot for medium run production.
- [ExpressPCB](#) (Software integrated solution, reasonable pricing) - ExpressPCB is the only one on this list that also showed up on the CAD list as well. From what I can tell, their quality seems to be about on par with Gold Phoenix with a similar pricing structure to Advanced Circuits.
- [Seeed Studio](#) (Reasonable pricing, not as many options as Advanced Circuits, but more freedom than OSH Park) - Seeed studio has made a point to focus on hobbyist work and their dedication is clear. You can get a 2-layer 5cm x 5cm board starting at \$9.99, and they include free testing which is a great deal.

These are my top five, but as I noted earlier, there are many, many more out there.

Defining your Design Rules

After choosing a manufacturer you should make note of their manufacturing constraints. As an example, at the time of writing the minimum specifications for OSH Park were:

- 6 mil copper traces
- 6 mil spacing between traces
- 13 mil drill diameter
- 7 mil annular rings [Defined as (diameter of the pad - diameter of the hole) / 2]

This means that these should be the smallest features you use under any circumstances. **Using smaller features will likely result in broken traces, overlap of traces and copper fills, or busted vias.**

Once you determine these rules, you should head into your CAD software and define them. This will enable some design for manufacture (DFM) checks to take place as you design so your program will not allow you to perform operations that will cause you to have non-manufacturable boards.

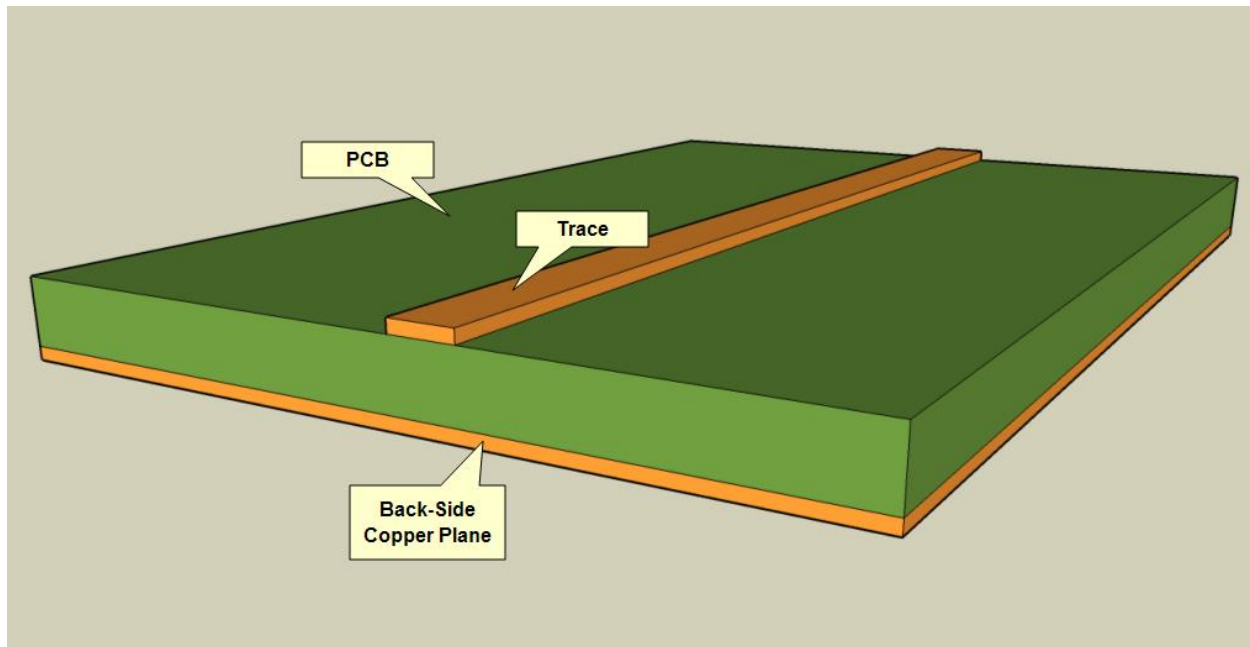
This is one step of the process where EAGLE generally has a distinct advantage over KiCad, most board houses provide a design rules file that can be imported directly into EAGLE. This saves a few steps in defining design rules, and reduces the chances that you will end up with incorrectly defined rules.

Note: Just because your fabrication house can do a 6 mil trace doesn't mean you should use exclusively 6 mil traces. You should use the largest traces possible that will still allow you to fit your design in the required space. Using larger traces improves reliability, decreases parasitic resistance, and as a whole results in a better circuit.

Do I Need a Ground Plane?

One of the more debatable topics in designing a PCB is deciding **whether to include a ground plane or not**. While it is nearly standard practice to include a ground plane in your design, this is not always required and can in some cases actually result in worse performance. But how can you know when you should and when you shouldn't?

First, it helps to know what a ground plane is. Simply put, **a ground plane is a copper layer on your PCB that acts as a common ground to many devices**. It is called a ground plane because it often occupies an entire layer, which creates a planar surface to conduct charge.



This image from [Circuit Calculator](#) shows the difference between a trace, the PCB, and the copper plane that could act as ground.

What are the benefits of a ground plane? There are several benefits to using a ground plane in your design, the most common are to provide electromagnetic shielding, lower the resistance of the path to ground, and to assist with heat dissipation across the board. These benefits are excellent for the vast majority of designs, but there are also some drawbacks to using a ground plane in your design.

What are the drawbacks of a ground plane? Perhaps the largest drawback of using a ground plane is the increase in parasitic capacitance. Parasitic capacitance is an undesirable effect that will essentially cause your circuit to be less "responsive" than intended. For most applications this is just fine, but for exceptionally quick response circuits it may be worth removing the ground plane.

Ultimately, it is up to you to decide whether you need a ground plane or not. Here are a few guidelines to help you make the decision:

- If your design is not particularly high performance, **it's your call**.
- If your design includes RF range signals, you should **always use a ground plane**.
- If you have components that rely on fast changing input signals you may choose to **not use a ground plane at all** or to **remove part of the ground plane around those inputs**.
- If you're just not sure, **use a ground plane**. The chances are in your favor that the circuit will behave as expected with a ground plane, even if that is not required. The opposite is not quite true.

Special Considerations

In addition to considering the ground plane and adhering to design rules there are some other special cases in which you may need to consider other effects.

- **Designing for RF** - If your design will be operating in the radio frequency range or using similar high frequency components, there are many special design factors to consider. This topic is too in depth to cover right here, right now, but this [post from EEWeb](#) outlines some good notes to get you started.
- **Mixed-signal designs** - If your PCB carries both analog and digital signals then you will want to make certain that you have fully separated these signal paths. The fast changing voltages used in digital circuitry can cause your analog circuitry to behave erratically. Mixed-signal design is a whole field of study on its own (as-is RF) so if you are working with these designs it may be best to seek the help of an experienced designer.
- **High voltage work** - High voltage circuits require extra care when designing and testing so as to avoid exciting outcomes like heart-stopping electrocutions, electrical fire starting mishaps, or other disasters. If you are designing for high voltage applications then stop reading and seek the assistance of a grizzled old electrical engineer experienced in high-voltage designs.

Once you're absolutely certain that you should be capable of designing the circuit yourself, you can move on to the next step.

Get to Work!

Finally, what we've been working towards the whole time. Now that you are a few hours (or days, or weeks) into the design process, you can finally start working on what you have been planning so carefully for. But you do have one more decision to make first...

To Autoroute or Not?

Should you use the wonderful autorouting features of your CAD package or not? For those who don't know what an autorouter does, it automatically connects the traces in your board in a pattern that the software deems is most efficient. Some CAD packages also include an "Autoplacer" which will automatically place your components for you before routing.

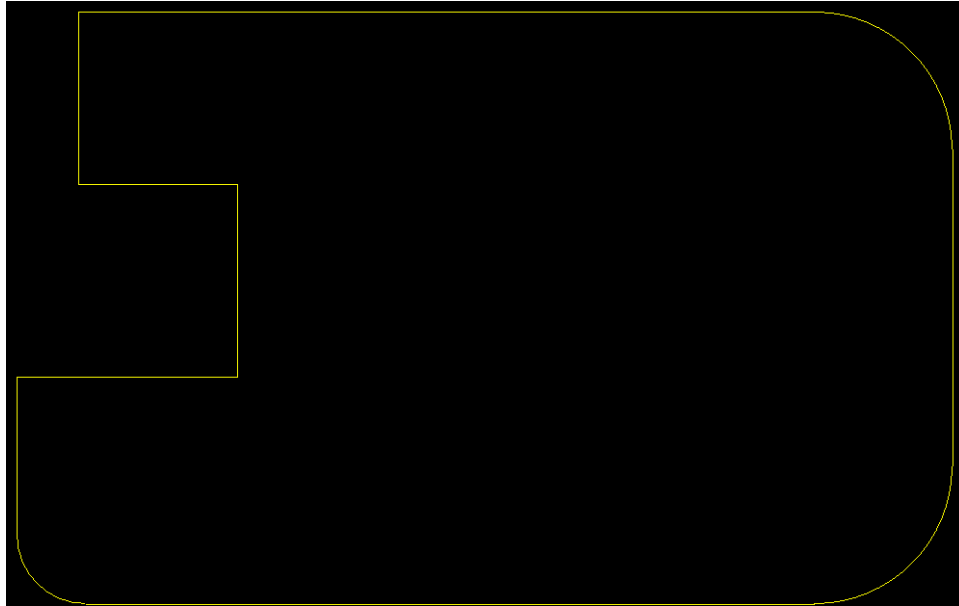
In general you are probably better off avoiding both of these tools for simple hobby work or even moderately complex designs. Honestly, if you have a thorough understanding of your circuit, and you should by now, then you will be able to do a better job of placing and routing components.

There are of course a few exceptions to this guideline. If the design you are working on is complex and it would take you weeks or months to perform a proper layout by hand, then you should try to get access to an advanced CAD package to make use of the significant research these companies have put in to autorouting algorithms.

In addition, if you are just dreading the idea of sitting at a computer ensuring that your design is perfect, then you may just wish to place your necessary components, lock them into place, route the critical connections, and then run the autorouter. **This is the recommended way to use the autorouter if you choose to do so.**

Drawing the Board Outline

The first step is to draw the outline of your board. This layer tells your fab house where to cut to give you the right sized PCB.



BeagleBone Black cape, board outline layer. Yes it is boring looking, but that's all it is.

To draw the board edge you will begin by switching to the layer that is designated for cutouts. Depending on your CAD package, this could be "Edges", "Board", "Cutout", or something else along those lines.

After selecting the board edge layer make good use of your drawing grid to ensure that you have straight lines of a well-defined length. If your board needs to be a specific size then make sure you are using the correct measurement units for your grid. If you draw your board in mm instead of inches, you will get a little surprise when you try to place your parts and can't fit them all on the board.

A few tips for drawing your board edges:

- **Set your drawing grid at a reasonable spacing** and make sure your cursor is set to **snap to the grid**.
- If you are using EAGLE or another program that supports it, **make use of the keyboard commands for defining lengths of segments**.
- **Consider using alternate axes** (plural of axis). Most CAD software supports some method of setting an alternative origin point, this will allow you to draw lines of specific length without needing to subtract coordinates in your head. If your CAD package supports #2 then you may not need to use this feature, but I have found it helpful in KiCad.
- **Ensure that all the edges line up exactly**. If you have a spot on the board where two edges meet but don't quite touch, then you should fix that now. Trying to send your board to the fab house like this will result in them responding to you with a solid "No thanks." Fix this issue before it becomes a headache to fix.
- **Alternative Method:** If your board doesn't need to be any specific size or shape then you may want to wait until the end to draw your board edge. But if you have any predetermined requirements for the board, you will want to start with this step.

Placing Components

Next up, you will want to place all of your components inside the board outline. **You should start with components that have a set physical location** such as connectors or sensors that can't be blocked.

After that, you will want to **begin placing your ICs. Start with the largest ICs first and then place the smaller ICs as you go.** ICs with more pins will require more room around them for routing traces and placing auxiliary components. **Try to leave extra room around devices that have many pins.**

Another thing to consider when placing devices is to **try to keep all your ICs oriented in the same direction on the board.** This is not a strict rule, but it can sure make assembly much simpler and is generally not a bad rule to follow. If it just isn't possible then don't worry about it.

Once all of your physical components and your ICs are in place you can place the supporting devices. Things like resistors, capacitors, diodes, etc. This is a good time to refer back to your design notes and make sure that any components which need to be physically close to an IC are in a good position. If you have components that have these requirements then I would recommend locking them in position after placement.

One final thing, you will want to consider **leaving space for annotations and markings on the board.** I'll discuss these things in more detail in a bit, but you'll want to make sure there is enough space around your components to leave some lettering near anything that may need it. For more info on this, jump down to the "Adding Some Style" section.

Making Connections

So, everything is in position and the board is starting to look like it will come together. The next step is to make it all work! You could just start by connecting pins all willy-nilly as you see fit, but taking some time to do it right will pay off in the end, which is coming sooner than you think.

There are two recommended ways of starting out laying your traces. You can begin by routing your power traces first and then focus on everything else, or if your design has high frequency signals you can begin by routing those first. Other than that, the rest of the connections are up to you.

While there isn't any single "right" way to lay-out the rest of your PCB traces, there are some methods that are "more right" than others. You can see my recommendations below, and I hope to see a few more additions come from the community.

1. **Make use of thick power supply traces.** The power supply rail will most likely be the most active trace on your PCB and since it will be supplying the more current than any other trace it also deserves some special attention when determining how wide it should be. As a general guide I like to use 20 mil power traces. For low power circuits this is probably overkill and you could get away with less, for higher power circuits this may not always be enough. If you are designing circuitry that is going to draw current in the Ampère range as opposed to milliAmpère then I suggest taking a closer look at your trace width. [This handy calculator](#) will help you calculate the correct width to use for your traces.
2. **Avoid routing two (or more) high frequency signal traces in parallel with each other.** According to Ampère's law (with Maxwell's correction) we know that a changing current induces a magnetic field around the wire, we also know that a changing magnetic field induces a current perpendicular to the direction of the magnetic field. This effect can cause two parallel wires to couple together so that a

change on one wire can induce a change on the other. You can avoid this by keeping high frequency traces separate and only crossing them in perpendicular alignment.

3. **Try to group similar signals together.** If you have a bunch of wires coming from a single device that all perform a similar function then you should try to keep them neatly grouped until it is absolutely necessary to split them up. This will allow you to more easily follow the signal path and will help you end up with a cleaner looking board after fabrication.
4. **Minimize your use of vias, but not at the cost of dramatically increasing signal paths.** This recommendation may just come as a result of my fascination with aesthetic design, but there is also a practical reason to use fewer vias. The simple point is that vias are a manufacturing risk. While it isn't likely that your board house will mess up and drill a hole too large or break the conduction ring, it is completely possible. Having fewer vias on the board reduces this risk. Another benefit of reducing vias is that it results in a shorter signal path (even if only a tiny reduction). This type of caveat is really only important in RF designs, but if there is a way to improve a circuit, I'm always looking for it.
5. On the other hand you could, **use two inline vias at every signal pin.** I know this may make me look like a hypocrite or an otherwise very confused person but there are also situations in which more vias could help. If you are working on a prototype circuit, extra vias allow you to easily create test points, and to simply cut and reconfigure traces. This recommendation comes courtesy of a reader, James Edwards I believe, but I couldn't find the original comment.

There are probably some more useful pointers that I'm just not thinking of, but nothing comes to mind right now. I will add more as I think of them.

Adding Some Style

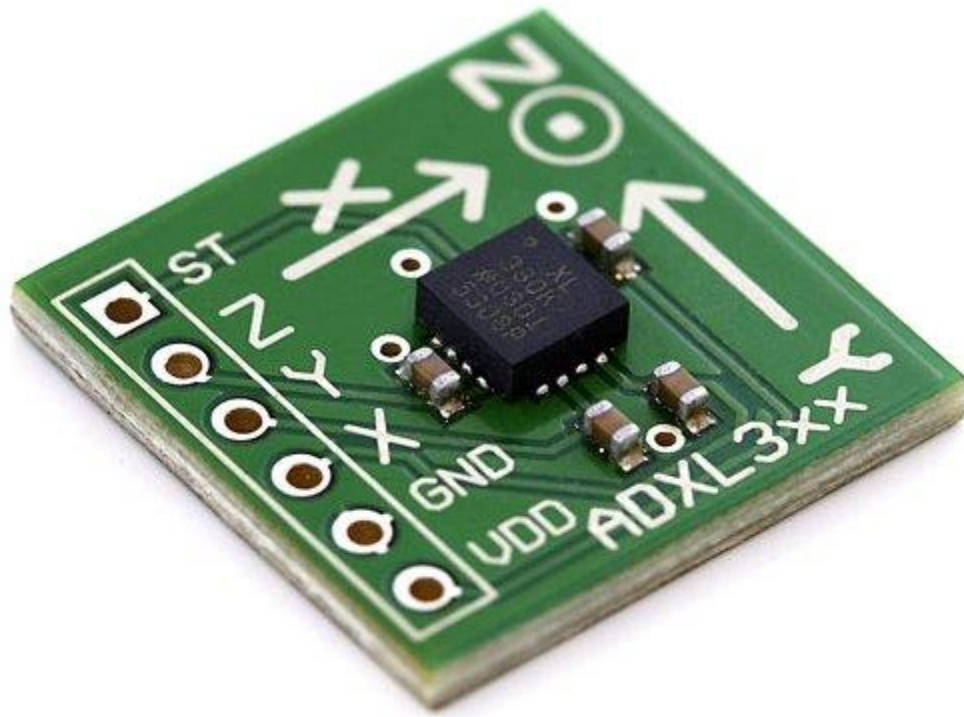
You're almost there, the last major step before moving on to manufacture is to add the finishing touches to the board. This may seem like it's just aesthetics, but there are several practical reasons to include some extra markings on your board.

The first decision you need to make is whether or not you should include component reference numbers or values. As an example, do you want your final design to indicate that "this" resistor is R11 and has a value of 4.7k or do you want to just mark it as R11? Maybe you don't want to mark it at all?

This is really up to you. My thought is that you will be one of the few people who actually looks at the board components and if you are looking at the board you will be able to easily pull up a schematic for reference. For that reason, **I do not include reference numbers or component values on my final designs.**

Having said that, there are some components that can benefit from a bit of labeling. **Generally you will want to label any LEDs, buttons, switches, connectors, or otherwise important devices.** I guess the argument could be made that ALL the components are important, but I am specifically referring to parts that would be good to know when using the device.

SparkFun gives some good advice on labeling in [their PCB guide](#). I specifically like the example shown below.



From SparkFun: Proper Labeling of an Accelerometer Breakout Board. Notice this includes the part number, pin functions, and even orientation axes.

After all of the functionally important components have been labeled, you may have a bit of room left over for more labeling. Don't overcrowd the board, but you may want to include your name or some sort of branding. Perhaps a logo? You can do this easily in [EAGLE](#) or KiCad using the built in tool or this [online tool from Wayne & Layne](#).

Final Design Checklist

The design checklist is included as Appendix A.

Manufacturing the Board

Since you should have chosen a manufacturer by now, this part of the process is going to be relatively easy.

The first step in preparing your design for manufacture is of course to perform the final design checklist. Since we covered that in the last section, it's okay to go ahead with the rest of the manufacturing process.

Run a Design Rules Check

Before going any further, you will want to run one final DRC. This generally checks that your PCB layout matches the schematic and that your layout follows all of the design rules that you defined. If the DRC catches errors, you should review each one individually and either fix the problem or mark it as a false warning.

Another thing to check is that all of your connections have been made. Sometimes the DRC tool does not check connections, or your ratsnest may be too small to notice. You want to be extra certain that all of the board connections are complete before moving on.

Generate the Bill-of-Materials (BOM)

A bill-of-materials (BOM) is used by the assembly house, or for your own use. Either way, it is important to have a good list of your parts. Here are the important details to include in a BOM:

- Component reference designator
- Component package
- Quantity required for one PCB
- Description
- Manufacturer
- Manufacturer reference number
- Supplier
- Supplier reference number
- Cost per unit
- Alternative parts allowed, if applicable
- Comments

Most CAD software can export a BOM automatically, but you will generally want to format it and make sure that everything is correct.

One reason to make a good BOM, even if you are not using assembly services, is because many parts suppliers will allow you to upload this file directly to their website and automatically purchase components. I know Mouser and Digi-Key support this, others may as well. **This method will save you hours of time searching for components and adding them to your order.**

Export the Board Files

Each fabrication house will have different requirements for how you should submit your design but nearly all of them accept one common file format called "Gerbers." Some will even accept your CAD files directly, but this isn't universal and you shouldn't rely on it.

Each software has a different process to follow in order to get Gerber files out, here are some tutorials that show you how to export Gerber files from EAGLE, KiCad, and gEDA.

- [Prepare EAGLE files for manufacture](#) from Hack-a-Day
- [KiCad Tutorial: Gerber file generation](#) from Wayne & Layne
- [PCB basics and gEDA/PCB Tips and Tricks](#) from 5 Man Conspiracy

Final Manufacturing Checklist

The manufacturing checklist is included as Appendix B.

Send it to Your Manufacturer

That's it! Depending on the service you chose for manufacturing, the instructions will vary, but from here on out the process is relatively straightforward. Upload your files, cross your fingers, and hit submit.

The End

That brings us to the conclusion of this series, I know you've been hit with a lot of information. If you read both part 1 and part 2 in their entirety then you soaked up over 11,000 words of PCB designing goodies. That's more information than a silverback gorilla retains in its whole life*!

*Maybe, I have no data to back this up.

I hope you've found at least parts of this guide to be helpful and that you will make use of it in your future designs. I attempted to make it general enough to be useful for any CAD package at any point in time. When searching how to do stuff like this it is incredibly easy to stumble across information that just isn't relevant anymore, so I hope this will help.

I worked on this article off-and-on for about two weeks, even though I reviewed it several times before posting, **I don't make any claim that it's perfect**. If you see issues in my writing or think my advice is bad then feel free to let me know.

If you have your own projects, articles, or reviews that you would like to write and you are a clear writer then I would love to add you as a contributor to this site. You can request contributor status by contacting me in the contact form at the bottom of my [About Me](#) page.

Appendix A: Final Design Checklist

1. All unused inputs terminated
2. All outside world I/O lines filtered for RFI and protected against static
3. Bypass cap for each IC power supply
4. Voltage ratings of components checked – This is important
5. File name on each sheet
6. Dot on each connection
7. Minimum number of characters in values, within reason
8. Consistent character size for readability
9. Schematics printed at a readable scale
10. All components have reference designators and values
11. Special PCB or parts list information entered for each component, if required
12. All polarized components checked for reverse voltages
13. Title block completed for each sheet
14. Pull-up resistors on all open collector or open drain outputs
15. Sufficient power rails sizes
16. Consider signal rate-of-rise and fall for noise radiation
17. Separate analog signals from noisy or digital signals
18. Sufficient capacitance on low dropout voltage regulators
19. Check the data sheet fine print and app notes for strange IC behaviors
20. Automotive powered devices must withstand 60 to 100 volt surges
21. Check maximum power dissipation at worst-case operating temperatures
22. Check time delays and slew rates of OpAmps used as comparators
23. Check failure modes and effects of failed power semiconductors
24. Estimate total worst case power supply current
25. Check pin numbers of all custom-generated parts
26. For buses, ensure bus order matches device order
27. Ensure resistors are operating within their specified power range
28. Resistor power ratings derated for elevated ambient temperatures
29. Use of baud rate friendly clock source for devices that have serial ports
30. ROHS compliance requirement review if you are planning to sell your product
31. Review parts to make sure they aren't obsolete
32. All no-connect pins on IC's should be labelled NC
33. Text should not overlap wire or symbol graphics on schematics
34. Page title present and consistent on all pages if not in title block
35. Off board connectors identify all signals even if not used on this design
36. Unpopulated parts annotated and enclosed by dashed-line box on schematics
37. Wires exist between all connected pins/ports (no direct pin/pin connections)
38. Pin names and attributes on symbols with multi-function pins should match actual design usage
39. Connect DIP switches and other grouped I/O to ports in a logical way, LSB to LSB, MSB to MSB
40. Use preferred component reference designators

Appendix B: Final Manufacturing Checklist

1. Hole diameter on drawing are finished sizes, after plating.
2. Finished hole sizes are ≥ 10 mils larger than lead
3. Silkscreen legend text weight ≥ 10 mils
4. Pads ≥ 15 mils larger than finished hole sizes
5. Place through-hole components on 50 mil grid
6. No silkscreen legend text over vias (if vias not soldermasked) or holes
7. All legend text reads in one or two directions
8. Components labeled left-right, top-bottom
9. Company logo in silkscreen legend, company logo in foil, copyright notice on PCB, date code on PCB, PCB part number
10. Assembly part number on PCB, for assembled boards
11. Components ≥ 0.2 " from edge of PCB
12. Ground planes where possible
13. Test pad or test via on every net to allow in circuit test
14. Test pads 200 mils from edge of board
15. All polarized components checked
16. No acute inside angles in foil
17. Traces ≥ 20 mils from edge of PCB
18. PCB revision on silkscreen legend
19. Mounting holes matched 1:1 with mating parts
20. Automated netlist check
21. Manual netlist check
22. Check netlist for nodes with only one connection
23. CAD design rule check
24. Tools on drill plot and NC drill file cross checked
25. Soldermask over bare copper noted if needed
26. PCB thickness, material, copper weight noted
27. Trace width sufficient for current carried
28. Minimum component body spacing
29. SMD pad shapes checked
30. Visual references for automated assembly
31. Tooling holes for automated assembly
32. Sufficient clearance for high voltage traces
33. Component and trace keepout areas observed
34. High frequency circuitry precautions observed
35. Thermal relief pads for internal power layers
36. Blind and buried vias only allowed on multilayer PCB
37. Sufficient clearance for socketed ICs
38. SMD component orientation arbitrary or consistent
39. Ensure pin 1 interpretation and orientation consistent among all connectors of a given type on the board

40. Standoffs on power resistors or other hot components
41. Digital and analog signal commons joined at only one point
42. EMI and RFI filtering as close as possible to exit and entry points in shielded areas
43. Layout PCB so that any rework or repair of a component does not require removal of other components
44. Extra connector and IC pins accessible on prototype boards, just in case
45. Check all power and ground connections to ICs
46. Provide ground test points, accessible and sized for scope ground clip
47. Potentiometers should increase controlled quantity clockwise
48. Bypass capacitors located close to IC power pins
49. All silkscreen text located to be readable when the board is populated
50. All ICs have pin one clearly marked, visible even when chip is installed
51. High pin count ICs and connectors have corner pins numbered for ease of location
52. Silk screen tick marks for every 5th or 10th pin on high pin count ICs and connectors
53. Check for traces running under noisy or sensitive components
54. Check IC pin count on layout vs. schematic
55. No vias under metal-film resistors and similar poorly insulated parts
56. Check for traces which may be susceptible to solder bridging due to low clearances
57. Maximize distances between features where possible
58. Check for dead-end traces
59. Check for power not shorted to ground
60. Provide multiple vias for high current and/or low impedance traces

Some References

I used many references when compiling this guide, I can't possibly list everything, and much of it was my own knowledge. A few of my more prominent sources are listed below.

Electrical Engineering StackExchange, specifically:

- <http://electronics.stackexchange.com/questions/28251/rules-and-guidelines-for-drawing-good-schematics>
- <http://electronics.stackexchange.com/questions/1024/good-tools-for-drawing-schematics>

Better PCB's in EAGLE from SparkFun

- <https://www.sparkfun.com/tutorials/115>

Atlantic Quality Design Electronics Checklist (Used for Design and Manufacturing checklists)

- <http://www.aqdi.com/check.htm>

Electronic Products Design Good PCB Design Checklist (A simplified checklist that should cover most uses)

- <http://www.electronic-products-design.com/geek-area/electronics/pcb-design/general-pcb-design/a-good-pcb-design-checklist>