How to Draw Circuits for Presentations and Reports

Matthew Spencer, Harvey Mudd College, 2018, rev 1

I'm often asked how I draw circuits on a computer. The short answer is "Google Drawings with a custom circuit library," but this document aims to answer that question in exhaustive detail and give any schematic drafter a wide choice of tools to choose from.

This document is not about entering schematic designs into CAD tools (see "How to Make a PCB for Me" for that), but rather about drawing schematic figures for presentations or documents. Some of the aesthetic considerations are the same, but using a program designed for figures to make designs or vice versa often has poor results.

"How to Draw Circuits" is a conveniently ambiguous title for this document that lets me rhapsodize both about best practices for making a good looking schematic and the computer programs to enact those techniques. This document is divided into two sections, one on each of those topics.

Making a Schematic Figure Look Good

Making a schematic figure look good is reasonably simple because the implicit rules governing readable schematics are strict. The best practices listed here should make almost any schematic look good. The rules are presented in a rough priority order, so you can infer that that following schematic conventions is more important than only drawing straight lines, for instance.

Of course, it's worth quoting Orwell: "Break any of these rules sooner than say anything outright barbarous."

Only include enough detail to make your point

The biggest challenge in drawing a schematic, and the area where a figure maker has the most influence, is deciding how much detail to include. Crowded schematics are hard to understand: If there are lots of connections then it is hard to follow them visually, and if there are many symbols then it is difficult to tell which one is important. Omit as much as you are able while still making your point. Only label nodes that will be referred to, and only include element values or labels that are needed for discussion. It is common to find that the minimum level of detail is still quite high: that's alright, just do your best to reduce clutter.

It's important to be aware of context when deciding what information to cut out. For instance, bypass capacitors are essential in a PCB schematic design because amplifiers are often unstable if they are omitted (though they're enough of a visual distraction that schematic designers put complicated bypass networks on separate pages with symbolic connections). However, in E153 reports students are tasked with sharing their novel arrangement of transistors into amplifiers, so bypass caps just get in the way. In E80, students are trying to convey that they have made a thorough and careful design, so including bypass caps shows that they understand the importance of the component. Similarly, resistor values are crucial in almost any class at Mudd (and obviously on actual designs), but are often omitted in conference presentations where all the practitioners know how to calculate the relevant values.

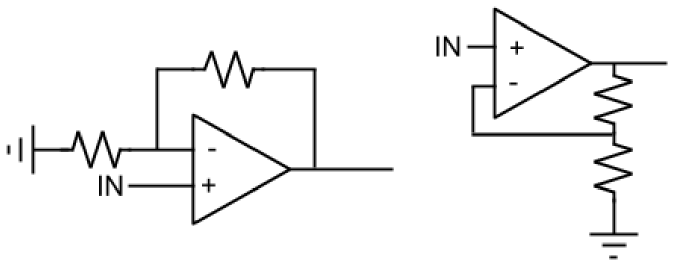
If you can't cut out enough detail to make your circuit easy to understand, then consider splitting it into multiple schematics (possibly spread over multiple slides or pages). It is better to explain connections between schematics to the audience than to have one schematic that no one understands.

Follow schematic conventions

Power flows from the top to the bottom of a schematic and signal flows left to right. Feedback flows from right to left. Low voltages are lower on the page than high voltages. Never, for instance, make a voltage divider or RC low pass filter where the input is at the bottom and ground is at the top. It's worth making a few clunky jogs in a wire to adhere to these principles.

Draw circuits in a form that audiences will recognize

Both of the circuits below depict non-inverting amplifiers, but one is drawn according to convention and the other is not. A casual observer could confuse the non-traditional rendition for an inverting amplifier. These conventional circuit arrangements are a visual shorthand that lets trained engineers quickly parse complicated schematics, so violating them only makes your circuit hard to read.



Occasionally you will come up with a cute idea for a schematic that you feel lays an important idea bare but requires that you violate convention. Don't do it! Convention is very powerful, so your idea would have to be revolutionary to justify violating it.

A few common conventions to meditate on:

- Voltage dividers can be drawn as two vertical resistors or as dog-legs: a horizontal resistor for the upper branch of the divider and a vertical resistor (usually attached to ground for the lower leg). Use the vertical drawing for references and the dog-leg to show signal flow. Consequently, you should almost always use the dog leg for unusual impedance dividers like RC filters.
- Potentiometer symbols have three terminals. The top and bottom of the resistor have a fixed resistance between them, and the arrow pointing at the side indicates the wiper, which has a variable resistance to both top and bottom. Indicate a variable resistance by connecting to the wiper and one other terminal.

Use straight lines wherever possible and don't have 4 way junctions

Your audience will follow the lines on your schematic to trace connections. You want this tracing process to be effortless, or else the audience will spend lots of time tracing and little time understanding your point.

- Snarls in your wires confuse the tracing process, so keep traces straight.
- Only turn routes at ninety degree angles.

- If at all possible, group busses into a single route rather than running many wires in parallel.

Wire crossings present special hazards for the tracing process: Does the eye follow the branching wires or not? Minimize complexity carefully so that you don't present too many wires or junctions for the audience to trace. Use dots to differentiate junctions from wire crossings, and don't draw four way junctions: use two adjacent three-way junctions instead.

Use heavy line weights and large fonts

All of this advice is useless if your audience can't see your figures. Make them visible to the nearsighted professors who stubbornly refuse to use their glasses by making everything big.

Programs which are Useful for Drawing Schematic Figures

There are a wealth of programs that can be used to draw figures. They fall into the general categories of PCB CAD tools, general purpose drawing programs, and specialized circuit drawing programs.

PCB CAD Tools

Many students (and some practitioners) opt to use the schematic entry tool of a PCB CAD suite to draw schematic figures. I think this is often bad practice because PCB schematic entry programs offer little control over color choices, fonts, and line weights and because designs entered in schematic CAD programs contain many details that are unnecessary in most figures.

Of course, a user can work around these limitations by, for instance, making schematics small enough to fit on one screenshot, but this often comes at the expense of a crowded schematic that is difficult to read. The chief argument in favor of using PCB CAD tools for making figures is that the design only needs to be entered once: your design can be your figure. However, the graphical constraints of making a schematic figure look good in a presentation result in a schematic design that doesn't leverage the features of a CAD tool like zooming, symbolic connections and multiple pages.

Though I don't recommend using PCB CAD tools for designing figures, a list is included below for completeness sake.

- KiCAD – free and open source
- Eagle – a free student version exists
- PADS – installed in the ECF
- Altium – the industry standard
- OrCAD – also prominent in industry

General Purpose Drawing Programs

Most of the figures that I have put in papers and books were made with programs of this type. These programs are designed to make good looking figures and often do an admirable job. The effort of making schematic figures with these programs can be greatly reduced by using libraries of pre-generated schematic symbols. Find these where possible

Two features are particularly desirable in these programs: a strong grid system and ninety-degree line snapping. These features help keep wires straight and cleanly routed. No coincidence that these features are standard in PCB entry tools. Some programs allow lines to be able to snap to points on symbols, which is usually helpful but can backfire if the symbol snap points are not located on the drawing grid: this causes unsightly angled lines.

- Microsoft Visio – This is a decent drawing program, and one of the few that I've seen used by serious figure makers. A circuit library of moderate usefulness is built into Visio, but I have to look up how to find and use it. I also have to look up how to make my own symbols and set up the grid: be prepared to sink some time into setup.
- Adobe Illustrator – I've never used this but live under the impression that it's the land of milk and honey. I'm sure regular users can dissuade me of that idea, but Illustrator figures generally look very good, and it seems like a powerful tool. Circuit symbols aren't built in, but do seem easy to find. There is a student version of this software that may be more affordable.
- Google Draw – This is the software that I use. The program isn't perfect: line ends don't have snap points and Google recently did away with some of the grid support that I preferred. So be prepared to sink some time into making sure the corners of your routes line up exactly right and work at high zoom levels to be sure there aren't tiny imperfections in the schematic. In spite of these wrinkles, I use it because it is very fast and doesn't require installation if I'm working on an unusual machine. I use a schematic symbol library which can be found here.
- Dia – Free and open source and comically clunky last time I checked (admittedly years ago). It has promising grid and snapping behavior, but it wasn't faster for me than Google Draw.
- Inkscape – Free and open source and generally useful for drawing cartoons of physics phenomena. Possibly useful for circuits, but I've never found a library I like.

Specialized Schematic Drawing Programs

There are a handful of programs that I have found which specialize in making schematic figures. These tend to have the desirable properties of general purpose drawing programs with circuit libraries built in.

- Xcircuit – This is my favorite circuit drawing software. It doesn't have a snap system, but the grid is strong enough that you don't need it. It renders lovely schematics. Unfortunately it only runs on Linux, and I don't have the patience to figure out Cygwin. Maybe worth a look now that Windows 10 supports Bash and Ubuntu Canonical. It's not easy to work on a design across multiple computers because multiple files are required for each design and because it is difficult to install.
- SchemeIt – This is a favorite of Mudd students. It is an online schematic drawing program provided by Digikey, so no installation is required. Library support is OK, but I find making new symbols onerous and don't like the visual style of labels. This also doesn't offer a lot of control over line weight or font. schematics.com (by TI) and EasyEDA (which also has a handy simulator) are similar creatures.
- Circuit-Diagram – Free and open source. Doesn't bring anything special to the table. The grid and routing system isn't strong enough for my taste and the custom symbol support was weak. Like many open source projects, it was a bit awkward to use the last time I checked. TinyCAD is a similar creature.
- Fritzing – This program is used to draw breadboards, not schematics, but many students have found it useful for planning out E80 projects before launching into them.
- smartdraw – I haven't tried it. It has a circuit diagram drawing program, but requires you to pay. It doesn't appear to have any particularly desirable features, but it's an option.