Empirically Found Drafting Guidelines

By Mike Engelhardt

[Note: Mike Engelhardt is the author of Qorvo's new simulation software, QSPICE, and, previously, of LTspice, both of which include schematic capture. In this article, he draws together drafting suggestions so that larger schematics can be handled with fewer errors.]

A schematic is ideally drafted in such a lucid manner that its operation leaps off the page for another electronic engineer. Drafting circuits properly makes design less error-prone, an important focus for any responsible design effort. My own drafting style has evolved over the years. My goal here is to document the methods I use that have been empirically found to allow me to handle larger circuits without errors.

Every style recommended here is based on functionality, but it's beneficial to know a bit of the past. As neither the transistor nor the integrated circuit were invented here in Silicon Valley(though the vacuum tube was), most historians see Silicon Valley's origins in instrumentation companies like Hewlett-Packard(1939) and not semiconductor manufacturers like Fairchild(1957). Silicon Valley was filled with instrumentation companies doing PCB level design. So much so, that the ground water became dangerous.

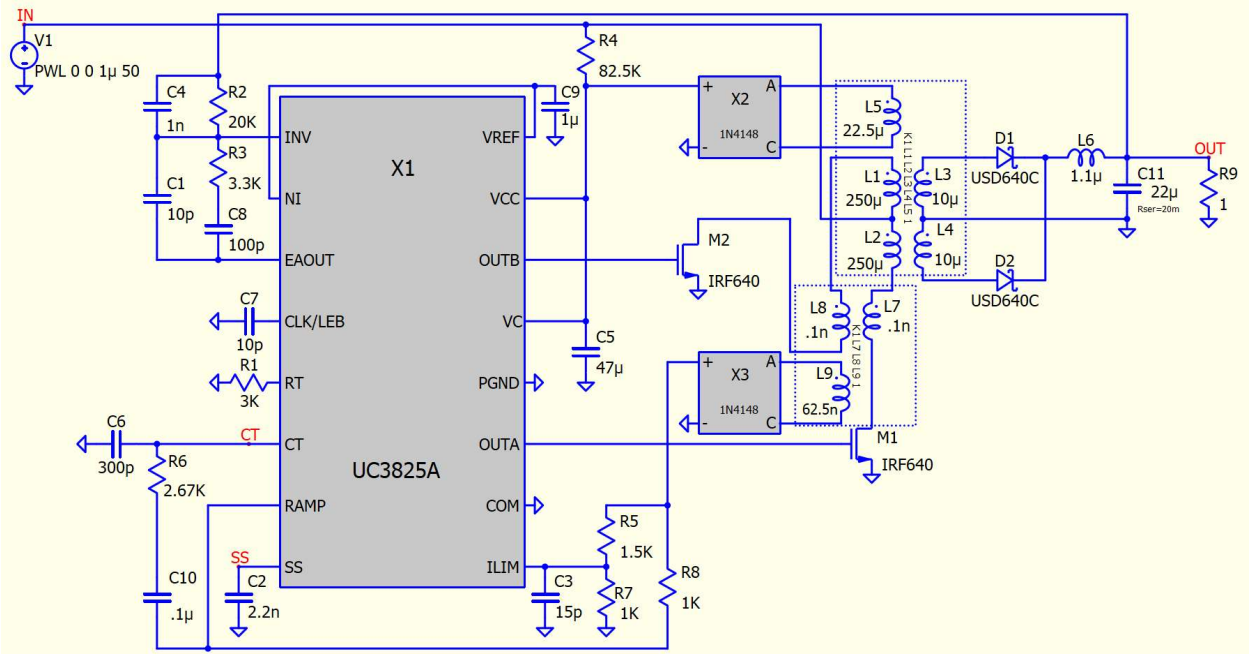
Up until the mid to late 1980s, the electronic engineer did not draft circuits as his time was too valuable. Instead, a specialized tradesman would draft under the engineer's auspices. The tradesman would draft with heavy lead pencils on special translucent paper used for making blueline prints with an ammonia-based process[1]. This predated the ubiquity of laser printing. I actually preferred the European practice I used while working at CERN at that time – India ink on true animal vellum -- since a scalpel could cleanly scrape off India ink without damaging the vellum allowing unlimited edits. The American paper "vellum" could only be erased a limited number of times before the entire circuit had to be redrafted anew.

Anyway, these tradesmen were typically temp workers. They traveled from company to company. This situation led to conventions that became fairly uniform here in Silicon Valley. And this brings me to the first drafting convention: The reference designator is drawn above the value with both to the right of the symbol. Unless the symbol is drawn horizontally. Then the reference designator goes above the symbol and the value below the symbol. If you follow that, two things happen. First you can quickly ascertain which reference designators and values go with which components, making the schematic much faster to read. And secondly, you show you know how schematics are supposed to be drawn. Basically, if you don't draft that way, you risk that a senior engineer, who actually did work at one of these early prestigious instrumentation companies, will immediately recognize that you never actually released a design to a document control department in your life. One other convention from this era is that ground on a circuit board is a triangle, not three lines. Three lines is earth ground. If you use the three line symbol to denote circuit board common, senior engineers might giggle and ask if you took your circuit outside and connected to that rod stuck in the dirt.

The next legacy drafting convention is one you're probably familiar with: Signal flow should be from left to right and current should flow from top to bottom. Follow this as much as possible.

Around 1985, management bought us CAD tools. By 1990 the temp workers were gone and management told us to draft our own schematics. The change of medium from bluelines to laser-printed CAD drawings allowed drafting style to evolve in the interest of being able to humanly handle larger schematics in a less error-prone manner. The following conventions are not historically based but purely functionally based. The goal is to be concise so it's easier to handle larger schematics.

1. Use ground symbols and not return paths. This dramatically reduces the number of lines on a schematic making it much easier to read. It also reflects the fact that CAD tools do a very good copper pour these days so the return path of least inductance is found automatically.
2. A properly implemented CAD tool scales junction dots differently that either fonts or graphic objects in the interest of being certain they are always visible no matter the scale on the screen or paper. This means that it is incorrect to stagger wire tees in a misplaced attempt to make it more clear where the connections are. The problem with a staggered cross is that it obfuscates the symmetry of common analog blocks making them harder to identify.
3. Draft compactly. Schematics are not printed as often as they are viewed on the screen. But as even a 4K screen has nowhere near the resolution of even a $100 laser printer, you can't always read the component values on the screen. But if you draft compactly, you'll be able to read the component values on the screen without having to zoom up on one section at a time. Drafting compactly is so important in my own work, that I will break the other guidelines. The example below has ground symbols drawn at angles that don't reflect current flowing from top to bottom in the interest of being compact.



The following convention addresses schematics used for SPICE simulations:

4. Do not use unnecessary characters. Ever. It's just more clutter. Here are some examples:

i) Units. SPICE is MSKA². You don't get to pick the units. A 3.3V voltage source should have a value of "3.3", not "3.3V". Similarly, the syntax is simply "PULSE 0 1 0 1µ 1µ .5m 1m" not "PULSE(0V 1V 0 1us 1us 0.5ms 1ms)" This isn't just a matter of reducing clutter. If you specify units, you read

them but the computer does not. So, if you make a syntax error, it's less obvious why the simulation is thinking 10 Amps while you're thinking 10 Volts. If you don't use the units, you're more likely to see things as they get parsed. Another common mistake is to give a 2.2 Farad super cap a value of 2.2F, but that is parsed as 2.2e-15, not 2.2.

ii) Parenthesis. Traditionally, parentheses were alternative whitespace characters to SPICE parsers. However, in coding, they have very special meaning. Don't use parenthesis when the special meaning isn't what's meant.

iii) Commas. Commas were also treated as whitespace characters in early SPICE parsers. When people discovered that they were ignored, people wrote ".model dname D(Is=1e-13,Cjo=10p,N=1.1)" instead of ".model mydiode D Is=1e-13 Cjo=10p N=1.1" Don't use commas when their special meaning to modern compilers isn't what's meant.

iv) Meaningless zeros. Use ".1" instead of "0.1" The meaningless zero is just more clutter for your eye to parse before you can see the meaning. The leading zero makes no more sense than writing "1.0" for "1." But there's more. The sooner you see your schematic as the computer sees it, the sooner you make fewer drafting mistakes. For example, QSPICE allows you to seamlessly spill over to C++ code. There, "1" and "1." are different things. Specifically, if you code "float x = 1 / 3; float y = 1. / 3;" x will equal zero and y will equal .333333 because "1" is an integer and "1." is floating point. Integer 1 divided by integer 3 is 0. Like parenthesis, the period decimal point has a very special meaning. Attempts to circumvent the need to be able to see it are harmful to your productivity. Note also, that the European practice of using a comma as the point of decimation is incorrect. Computers in Europe use the period as the radix point. No meaningless zeros no matter how many knuckles got slapped by nuns in parochial school for not leading the mantissa with a zero.

v) Use metric multipliers, not scientific notation, because "1µ" is more concise than "1e-06" Note also that no space is allowed between the digits and the metric multiplier.

Any worthy presentation on guidelines, best practices or standards, will invoke all sorts of spirited comments from people who haven't been there and done that. Knock yourselves out. These are the guidelines empirically found to make drafting less error-prone, allowing larger simulations to be handled.

1] Blueprints were a different, albeit also ammonia-based, process from an earlier decade. Calling a blueline a blueprint is akin to saying "outer space" instead of "space" in the aerospace industry.

2] There are exceptions, e.g., band gap energy is given in eV, not Joule.