Schematic & Programmable Logic Coding Standards

Using standards for electronics design schematics and programmable logic coding allows multiple people to create schematics and code with clarity and the same look and feel.

The guidelines below are limited in scope so that they can apply to many different companies. In addition to other forms of revision control, adding a revision date visible in the title block for each schematic sheet or each programmable logic code file can be very helpful.

Table of Contents:

Page 1: Schematic Standard
Page 5: Programmable Logic Coding Standard

A. Schematic Standard

Drawing schematics for electronics and electrical design is an imprecise art. However, most users agree on the relative quality of various schematics!

Users for schematics vary widely in background. They include engineers, technicians, assemblers, and management. The most important purpose for an electronics design schematic is to tell the story of the circuit operation.

Schematics should not be confused with wiring diagrams used as part of manufacturing work instructions. Wiring diagrams are often mechanically-oriented and show such information as which color wire attaches to each screw of a terminal block.

1) Cover Sheet: It can be useful to reserve the first sheet of a multi-page schematic for notes, a table of contents, engineering change order releases, etc.

2) Try to complete a single idea on a sheet of a schematic where practical. (This sometimes requires efficient, thoughtful use of the space.) It is almost always possible to draw schematics on “A” size sheets. This allows easy printing & copying without having to use magnifying glasses, special paper sizes, etc. Very few designs cannot be clearly shown on “A” size sheets. Avoid using different sized sheets for different pages of the schematics (e.g., both “A” and “B” sizes).

3) Parts with large numbers of pins (such as field programmable gate arrays or FPGAs) can be drawn as heterogeneous parts so that they are separated into multiple, separate boxes that can show data flows, control pins, power and ground pins. Multiple boxes for these larger parts allow use of “A” size sheets.

4) For clarity, avoid excessive length as well as unnecessary crossing and bends in wires. It is often helpful to draw versions of symbols that allow nets to interconnect between ICs without too many crossed wires.

5) With color printers, it is no longer necessary to show inputs and outputs from and to other sheets on the lefts side and right sides, respectively. Instead, these inputs and outputs should be shown close to their respective connection points. A separate color preference can be set for these inter-sheet connections so that they are easy to spot. For clarity and error-checking, inter-sheet references should include reference page numbers that can be automatically updated on request.

6) Place bypass capacitors near the integrated circuits (ICs) they're bypassing. The proximity of the capacitors shows which ICs they're bypassing. At the same time, make liberal use of global power and ground symbols on these bypass capacitors so that massive nets of these do not cover the schematic.

7) Useful suffixes to show differential pairs are _n & _p. An "n" prefix is a useful way to show a negative-logic signal. A "g" prefix is a useful way to show a global clock. Use both upper and lower case on net names for clarity. All-uppercase lettering was originally used on hand-drawn schematics for printing uniformity. This is not necessary with computer-aided drawing.

8) Generally, buses should be numbered from most to least significant bits. This is usually from large to small numbers, e.g. Data[15:0].

9) Schematics should be cleaned up so that all errors and most warnings go away during a design rules check. This significantly lowers the chances of actual mistakes. Digital ICs should usually be drawn with input and output pins explicitly defined. This allows the design rules check to spot connection errors more easily.

10) Brief notes should be added to most schematics to clarify the purpose of the various sections, e.g., "Two-Pole, Low-Pass, 100-Hz Butterworth Filter." Most connectors should include a "Note" attribute stating what the connector is intended to be connected to.

11) Power and ground pins should be shown explicitly on part symbols. This is because of the prevalence of multiple power and grounds within most modern designs. Errors are likely without this practice since someone looking at a print-out of the schematic can't see how some of the pins are connected. Hidden power and ground pins are a frequent source of errors. Heterogeneous part symbols allow power and ground pins to appear a single time on part symbols with multiple sections. Parts with multiple ground pins can be named with the pin

number as part of the name to avoid confusion, e.g, Gnd12 and Gnd31 for grounds on pins 12 and 31.

12) Symbols should be drawn for clarity of use, not so the symbol pins are shown in the same order as the actual package. Adding brief descriptive text to a symbol can add useful clarity such as “2048x16 EEPROM” for a memory chip.

13) The following attributes are very useful to create in the part symbols. *Attributes in italics are specific to OrCad*.

a) *Reference* = Reference designator needed on any bill of material (BOM) and on the schematic (e.g., R151). The reference designator is always visible on the schematic. Use standard prefixes such as R, L, C, D, Q, U etc., for resistors, inductors, capacitors, diodes, transistors, integrated circuits, respectively. The “P” prefix should be used for all plugs (male pins). The “J” prefix should be used for all jacks (female pins). WL is useful for wire-loop test points to distinguish these from test pads (denoted with a TP prefix) that are features of the PCB but not actual parts to stuff. LED is sometimes useful to distinguish LEDs from other diodes.

b) Value = Part number or value appearing on schematic. This attribute is generally needed on the schematic for readability, e.g.: 100k or 100, 1/2W or 5.1Vreverse.

c) Mfg = manufacturer of part (or one of them). This attribute is useful for ordering prototype parts and for reference even if a purchase specification based on the PN exists.

d) Mfg_PN = manufacturer’s part number (or one of them). The Mfg_PN is sometimes made visible on the schematic if the “Value” attribute is being used for other useful information such as “2x8Mbit Flash.” The Mfg_PN is normally not shown for resistors and capacitors. It is optional for inductors. It is normally shown for transistors, diodes, and ICs (either in the Value or the Mfg_PN attributes). This is useful for ordering prototype parts and for reference even if a company purchase specification based on the PN exists.

e) Description = part description (e.g., 1206 resistor, 1%, thick film, 49.9 ohms). Useful for reading BOMs and emphasizing a few important part features for writing company purchase specifications.

f) Note: This should be added to every connector to indicate what the connector is normally or potentially connected to.

g) *PCB Footprint* = footprint name to be net-listed for printed circuit board layout. Use of all lower-case avoids case issues.

h) *Group* = unique identification number (within a single or multi-page schematic). This identification number is entered by hand for all of the sections that form a symbol. For example, a quad-AND gate is often drawn as four separate parts. The “Group” identification allows re-numbering of reference designators without losing the integrity of each overall part. In OrCad, the “Group” attribute must be called out by name during reference designator re-annotation.

14) The following attributes are optional but often useful:

a) Sq_cm = square centimeters of maximum dimensions of part on board (length times width). This attribute is useful for estimating the minimum board area (once routing room is taken into account).

b) Cost1k = budgetary price in 1000-piece quantities. This is useful for estimating large production quantity pricing.

c) mA_5V, mA_-5V, mA_3.3V, etc. = milli-amperes at each of these power levels. These are generally only entered for integrated circuits. They are useful for power supply selection & design. Usually, these numbers represent the highest likely current draw.

d) PN = company part number—e.g., a five-digit number such as 05431. This attribute is useful in production BOMs. It is not typically added to BOMs that Redgarden makes unless specifically requested.

The engineers at Redgarden Engineering have used many schematic capture packages. One of our preferred packages is Cadence OrCad (http://www.cadence.com/products/orcad/pages/default.aspx). OrCad is widely known among electronics engineers and technicians. It has the substantial advantage of supporting design caches of parts. This avoids the significant hassle of maintaining strict global libraries along with version numbers. If you are working with us on schematics with copies of our libraries, it is useful to put them in: c:\Orcad\

We also have recent experience using DX Designer and Altium.

Shared libraries can be very helpful. However, it is generally useful to maintain local libraries with parts copied from shared libraries. The pin numbers and part numbers (along with other important attributes) for these parts must be checked before net-listing the schematic for PCB layout. OrCad schematic capture interfaces well with most layout packages including Mentor Graphics Pads. It is a simple process to automatically back-annotate re-sequenced reference designators from Pads layout back into OrCad schematic capture.

B. Programmable Logic Coding Standard

Like drawing schematics, coding for programmable logic (as for many other purposes) is an art. However, the following practices will add clarity.

1) Language: The original Verilog language was a significant improvement for code clarity and brevity compared to the older VHDL and ABLE languages. The 2001 version of Verilog had some addition improvements that make this language the strong choice for programmable logic coding and simulation. (However, Redgarden Engineering will code a design in VHDL if specifically requested by a customer.)

2) Clarity versus Brevity: In most cases, code should be written first for clarity, then for brevity, in that order. Comments should be added to the code particularly if the author himself is unlikely to remember how it works.

3) Tab Stop: A standard tab stop should be used for readability. Tabs composed of three spaces have been a good choice over the years.

4) Maximum Line Length: A standard length of line of eighty characters is useful for viewing source code in most text editors.

5) Signal Names: It is useful to have a mixture of "camel-back" notation (e.g., TerminalCnt) and under-scored notation for emphasis (e.g., DataBus_A).

6) Constants: It is traditional to show constants in all-capital letters with underscore separations (e.g., RED_ALARM). Constants should either be defined in one area of a file or in separate "include" file(s).

7) Signal Descriptions: It is very useful to show descriptions of all signals passing in and out of a module.

8) Grouping logic statements is very useful to put them in context. Carefully and thoughtfully separating the code into separate files is useful for code exceeding five-ten pages. Re-naming signals in different files should be done with care to avoid confusion!

9) Global clocks should be used whenever possible to avoid race conditions. Interfaces between clock domains should be carefully designed and simulated.