

原理图编辑器

The KiCad Team

Table of Contents

KiCad 原理图编辑器简介	2
描述	2
初始配置	2
原理图编辑器的用户界面	4
在编辑画布上导航	4
快捷键	5
鼠标操作和选择	5
左侧工具条显示控制	6
原理图创建和编辑	7
简介	7
原理图编辑操作	7
网格	8
捕捉	9
编辑对象属性	9
使用符号工作	9
位号和符号注释	19
电气连接	21
网络类	32
图形项目	36
原理图设置	42
Opening legacy schematics	48
层次原理图	53
简介	53
在设计中添加图框	53
原理图之间导航器	54
原理图之间的电气连接	55
层次设计实例	57
检查原理图	60
筛选工具	60
网络高亮显示	61
从 PCB 上交叉探测	61
电气规则检查	61
分配封装	71
在符号属性中分配封装	71
放置符号时分配封装	72
用封装分配工具分配封装	73
正向和反向批注	82
从原理图更新 PCB（正向批注）	82
从 PCB 上更新原理图（反向批注）	83
生成输出	86

打印	86
绘制	87
生成物料清单	88
生成网表	91
符号和符号库	98
管理符号库	98
创建和编辑符号	100
浏览符号库	124
仿真器	125
Value notation	125
分配模型	126
SPICE directives	131
Running simulations	132
高级主题	141
配置和定制	141
文本变量	141
数据基础库	143
自定义网表和 BOM 格式	146
操作参考	162
原理图编辑器	162
通用	168

参考手册

NOTE

本手册正在修订中，以涵盖KiCad的最新稳定发布版本。它包含一些尚未编写完成。我们希望您能耐心等待我们的志愿技术作者完成这项工作。我们希望您能耐心等待我们的志愿技术作者完成这项工作，同时我们也欢迎新的贡献者加入我们的行列。愿意帮助我们使 KiCad 的文档比以前更好。

版权

This document is Copyright © 2010-2024 by its contributors as listed below. You may distribute it and/or modify it under the terms of either the GNU General Public License (<http://www.gnu.org/licenses/gpl.html>), version 3 or later, or the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0/>), version 3.0 or later.

本指南中的所有商标均属于其合法所有者。

贡献人员

Jean-Pierre Charras, Fabrizio Tappero, Wayne Stambaugh, Graham Keeth

翻译人员

Liu HanCheng <buaa_cnlhc@buaa.edu.cn>, 2018.

taotieren <admin@taotieren.com>, 2019-2023.

Telegram 简体中文交流群: https://t.me/KiCad_zh_CN

反馈

KiCad 项目欢迎与软件或其文档相关的反馈、错误报告和建议。有关如何汇总反馈或报告问题的详细信息，请参阅 <https://www.kicad.org/help/report-an-issue/>

KiCad 原理图编辑器简介

描述

KiCad 原理图编辑器是作为 KiCad 的一部分分发的原理图设计软件，可在以下操作系统下使用：

- Linux
- Apple OS X
- Windows

无论操作系统如何，所有的 KiCad 文件都能 100% 地从一个操作系统兼容到另一个。

原理图编辑器是一个集成的应用程序，绘图、控制、布局、库管理和访问 PCB 设计软件的所有功能都在编辑器本身中进行。

KiCad 原理图编辑器旨在与 KiCad PCB 编辑器合作，后者是 KiCad 的印刷电路设计软件。它还可以为其他软件包输出网表文件，其中列出了所有的电气连接。

原理图编辑器包括一个符号库编辑器，它可以创建和编辑符号并管理库。它还集成了现代原理图设计软件所需的以下额外但基本的功能：

- 电气规则检查 (ERC)，用于自动控制错误和缺失的连接
- 以多种格式导出绘图文件 (Postscript, PDF, HPGL 和 SVG)
- 物料清单生成 (通过 Python 或 XSLT 脚本，允许许多灵活的格式)。

原理图编辑器以几种方式支持多张原理图：

- 扁平的层次结构 (原理图表在主图中没有明确的连接)。
- 简单的层次结构 (每张原理图只使用一次)。
- 复杂的层次结构 (有些原理图表被多次使用)。

层次结构图有详细描述《层次结构图，在手册后面》。

初始配置

当原理图编辑器首次运行时，如果在 KiCad 配置文件夹中没有找到全局符号库表文件 `sym-lib-table`，那么 KiCad 将询问如何创建该文件：

Configure Global Symbol Library Table

KiCad has been run for the first time using the new symbol library table for accessing libraries. In order for KiCad to access symbol libraries, you must configure your global symbol library table. Please select from one of the options below. If you are not sure which option to select, please use the default selection.

- Copy default global symbol library table (recommended)
- Copy custom global symbol library table
- Create an empty global symbol library table

Select global symbol library table file:

(None)



OK

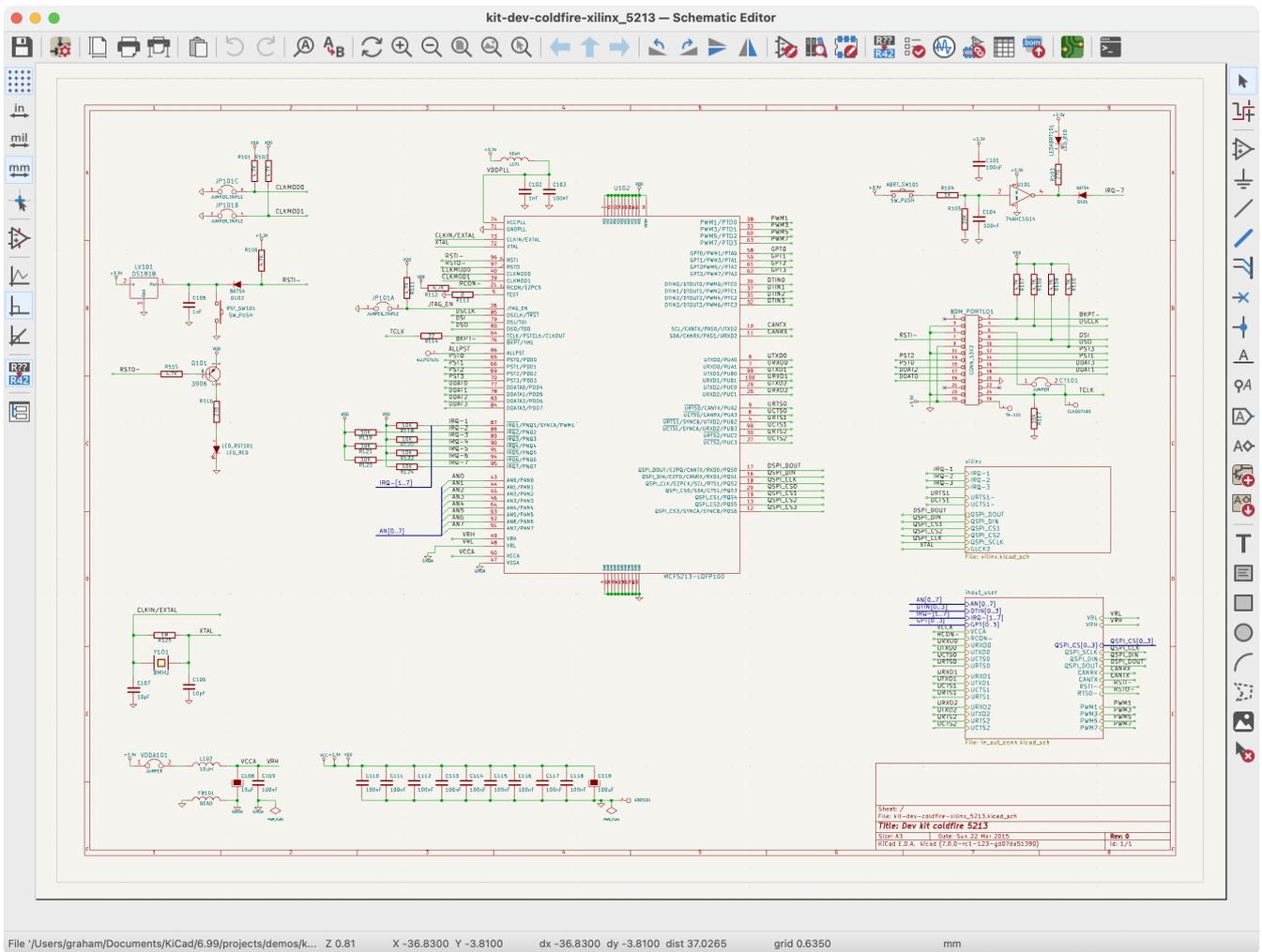
推荐使用第一个选项（**复制默认全局符号库表（推荐）**）。默认的符号库表包括所有作为 KiCad 一部分安装的标准符号库。

如果该选项被禁用，说明 KiCad 无法找到默认的全局符号库表。这可能意味着你没有和 KiCad 一起安装标准符号库，或者它们没有被安装在 KiCad 期望找到的地方。在某些系统中，KiCad 库是作为一个单独的软件包安装的）。

- 如果你已经安装了标准的 KiCad 符号库并想使用它们，但第一个选项被禁用，请选择第二个选项并浏览到安装 KiCad 库的目录中的 `sym-lib-table` 文件。
- 如果你已经有一个你想使用的自定义符号库表，选择第二个选项并浏览你的 `sym-lib-table` 文件。
- 如果你想从头开始构建一个新的符号库表，选择第三个选项。

符号库管理将在《管理符号库，待续》中详细描述。

原理图编辑器的用户界面



原理图编辑器的主要用户界面如上图所示。中间包含主编辑画布，其周围有：

- 顶级工具栏（文件管理、缩放工具、编辑工具）。
- 左侧工具栏（显示选项）
- 信息面板和底部的状态栏
- 右侧面板（绘图和设计工具）

在编辑画布上导航

编辑画布显示正在设计的原理图。你可以平移和缩放到原理图的不同部分，并打开设计中的任何原理图表。

默认情况下，用鼠标中键或右键拖动会平移画布视图，滚动鼠标滚轮会放大或缩小视图。你可以在偏好设置中的鼠标和触摸板部分改变这一行为（详见《配置和定制，配置和定制》）。

在顶部的工具栏中还有其他几个缩放工具：

- 缩小视窗的中心。
- 从视口的中心点放大。
- 放大到适合绘图原理图周围的图框。

 缩放到适合原理图中的每个项目（不包括绘图页）。例如，如果有项目放置在绘图页之外，在放大到对象之后，它们就会显现出来。

-  允许你画一个方框来确定缩放的区域。

光标的当前位置显示在窗口的底部（X 和 Y），还有当前的缩放系数（Z）、光标的相对位置（dx、dy 和 dist）、网格设置和显示单位。

按  可以将相对坐标重置为零。这对于测量两点之间的距离或对齐物体很有用。

快捷键

这个  +  快捷键显示当前快捷键列表。默认的快捷键列表包括在本手册的《原理图动作参考，动作参考》部分。

本手册中描述的热键使用了标准 PC 键盘上的按键标签。在苹果键盘布局中，使用  键来代替 ，使用  键来代替 。

许多动作默认没有分配快捷键，但可以使用快捷键编辑器（**偏好设置** → **偏好设置...** → **《偏好设置-控制，快捷键》**）分配或重新定义热键。

NOTE

许多通过快捷键进行的操作也可以在上下文菜单中使用。要访问上下文菜单，在编辑画布上点击右键。根据所选择的内容或所使用的工具，将有不同的操作。

快捷键存储在 KiCad 的配置目录下的 `user.hotkeys` 文件中。这个位置是特定于平台的：

- Windows: `%APPDATA%\kicad\7.0\user.hotkeys`
- Linux: `~/.config/kicad/7.0/user.hotkeys`
- macOS: `~/Library/Preferences/kicad/7.0/user.hotkeys`

KiCad 可以使用快捷键编辑器中的 **导入快捷键** 按钮从 `user.hotkeys` 文件中导入快捷键设置。

鼠标操作和选择

选择编辑画布中的项目是用鼠标左键完成的。单独点击一个对象将选择它。点击并拖动将执行一个框选。从左到右的框选将只选择完全在框内的项目。从右到左的方框选择将选择任何接触到方框的项目。从左到右的选择框是用黄色画的，光标表示排他性选择，从右到左的选择框是用蓝色画的，光标表示包容性选择。

在点击或拖动时按住修改键，可以修改选择动作。在点击选择单个项目时，以下修改键适用：

修改键 (Windows)	修改键 (Linux)	修改键 (macOS)	选择效果
			切换选择。
			将项目添加到现有的选择中。
			从现有的选择中删除项目。
长按	长按或 	长按或 	从弹出式菜单中清晰选择。

以下修饰键适用于拖动执行框选时：

修改键 (Windows)	修改键 (Linux)	修改键 (macOS)	选择效果
			切换选择。
			在现有的选择中添加项目。
			从现有的选择中删除项目。

选择一个对象会在窗口底部的信息面板上显示该对象的信息。双击一个对象可以打开一个窗口来编辑该对象的属性。

按 将始终取消当前的工具或操作并返回到选择工具。在选择工具处于活动状态时按下 将清除当前的选择。

左侧工具条显示控制

左边的工具条提供了改变原理图编辑器中项目显示的选项。

	打开/关闭网格显示。 注意： 默认情况下，隐藏网格将禁用网格捕捉功能。这个行为可以在偏好设置中的显示选项部分改变。
 	以英寸、密尔或毫米为单位显示/输入坐标和尺寸。
	在全屏和小的编辑光标（十字准线）之间切换。
	打开/关闭隐藏引脚的显示。
	在自由角度和水平/垂直放置新导线、总线和图形线之间进行切换。
	在自由角度、90 度模式和 45 度模式之间切换，以放置新的导线、总线和图形线。
	打开和关闭停靠的层次结构导航面板。

原理图创建和编辑

简介

用 KiCad 设计的原理图不仅仅是一个电子设备的简单图形表示。它通常是开发链的切入点，可用于：

- 根据一套规则（《ERC，电气规则检查》）进行验证，以发现错误和遗漏。
- 自动生成《创建自定义的网表和 BOM 文件，材料清单》。
- 《创建自定义网表和 BOM 文件，生成网表》用于仿真软件，如 SPICE。
- 《创建自定义的网表和 BOM 文件，定义电路》，用于转移到 PCB 布局。

原理图主要由符号、导线、标签、结点、总线和电源符号组成。为了使原理图更清晰，你可以放置纯图形元素，如总线入口、注释和折线。

符号是从符号库中添加到原理图中的。原理图制作完成后，连接和封装的集合被导入 PCB 编辑器，用于设计电路板。

原理图可以包含在一个原理图中，也可以分割成多个原理图。在 KiCad 中，多张原理图是按层次组织的，有一个根原理图和子原理图。每个原理图都是它自己的 .kicad_sch 文件，它本身就是一个完整的 KiCad 原理图。层次原理图的工作在《层次原理图，层次原理图》章节中描述。

原理图编辑操作

原理图编辑工具位于右边的工具栏中。当一个工具被激活时，它将一直处于活动状态，直到选择了另一个工具或用 **Esc** 键取消该工具。当任何其他工具被取消时，选择工具总是被激活。

	Selection tool (the default tool)
	Highlight a net by marking its wires and net labels with a different color. If the PCB Editor is also open then copper corresponding to the selected net will be highlighted as well. Net highlighting can be cleared by clicking with the highlight tool in an empty space, or by using the Clear Net Highlighting hotkey (Ctrl+H).
	Display the symbol selector dialog to place a new symbol.
	Display the power symbol selector dialog to place a new power symbol.
	Draw a wire.
	Draw a bus.
	Draw wire-to-bus entry points. These elements are only graphical and do not create a connection, thus they should not be used to connect wires together.
	Place a "no-connection" flag . These flags should be placed on symbol pins which are meant to be left unconnected. "No-connection" flags indicate to the Electrical Rule Checker that the pin is intentionally unconnected and not an error.

	Place a junction. This connects two crossing wires or a wire and a pin, which can sometimes be ambiguous without a junction (i.e. if a wire end or a pin is not directly connected to another wire end).
	Place a local label. Local labels connect items located in the same sheet . For connections between two different sheets, use global or hierarchical labels.
	Place a net class directive label.
	Place a global label. All global labels with the same name are connected, even when located on different sheets.
	Place a hierarchical label. Hierarchical labels are used to create a connection between a subsheet and the sheet's parent sheet. See the Hierarchical Schematics section for more information about hierarchical labels, sheets, and pins.
	Place a hierarchical subsheet. You must specify the file name for this subsheet.
	Import a hierarchical pin from a subsheet. This command can be executed only on hierarchical subsheets. It will create hierarchical pins corresponding to hierarchical labels placed in the target subsheet.
	Place a text comment.
	Place a text box.
	Draw a rectangle.
	Draw a circle.
	Draw an arc.
	Draw lines. Note: Lines are graphical objects and are not the same as wires placed with the Wire tool. They do not connect anything.
	Place a bitmap image.
	Delete clicked items.

网格

在原理图编辑器中，光标总是在一个网格上移动。该网格可以自定义：

- 尺寸可以通过右键菜单或使用 **视图** → **网格属性...** 来改变。
- 颜色可以在 **偏好设置** 对话框的 **颜色** 页中改变（菜单 **偏好设置** → **常规选项**）。
- 可见性可以通过左侧的工具栏按钮进行切换。

默认的网格尺寸是 50 mil（0.050 英寸）或 1.27 mm。

NOTE

导线只有在两端完全重合的情况下才能与其他导线或引脚连接。因此，保持符号引脚和导线与网格对齐是非常重要的。建议在放置符号和绘制导线时始终使用 50 mil 的网格，因为 KiCad 标准符号库和所有遵循其风格的库也使用 50 mil 的网格。**使用 50 mil 以外的网格尺寸将导致原理图没有正确的连接！**

也可以使用较小的网格，但这只适用于文本和符号图形，不建议用于放置引脚和导线。

NOTE

符号、导线和其他没有对准网格的元素，可以通过选择它们，点击右键，并点击 **将元素对准网格**，将它们重新对准网格。

捕捉

符号、导线、文本和图形线等原理图元素在移动、拖动和绘制时都会被固定在网格上。此外，即使在禁用网格捕捉的情况下，导线工具也会捕捉到引脚。在移动鼠标时，可以通过使用下表中的修改键来禁用网格和引脚的捕捉。

NOTE

在苹果键盘上，使用 **Cmd** 键而不是 **Ctrl**。

修改键	效果
Ctrl	禁用网格捕捉。
Shift	禁用将导线捕捉在引脚上。

编辑对象属性

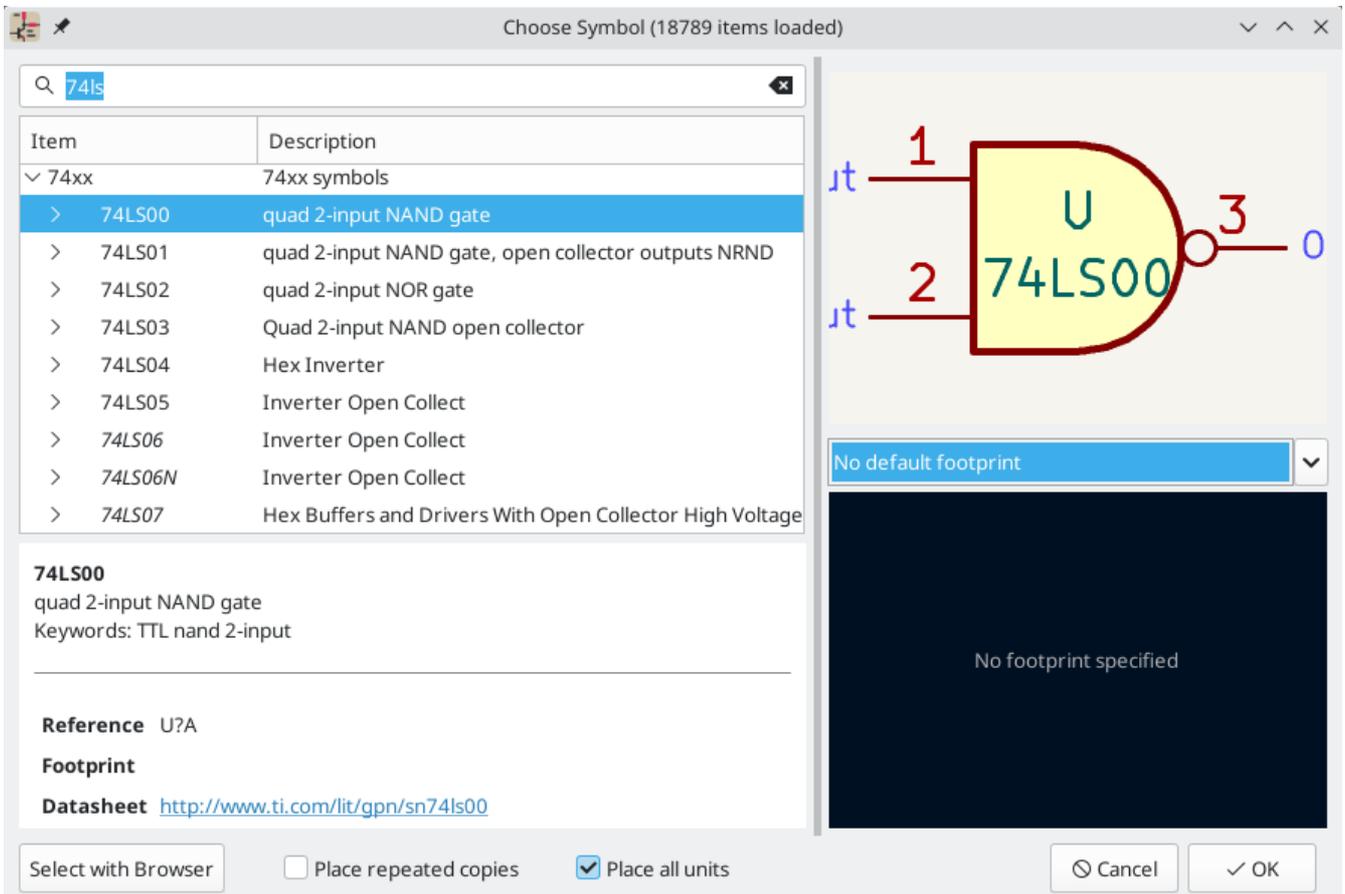
所有对象都有可在对话框中编辑的属性。使用快捷键 **E** 或从右键菜单中选择属性来编辑所选项目的属性。只有当你选择的所有项目都是同一类型时，你才能打开属性对话框。要想一次编辑不同类型的项目的属性，请参阅下面关于批量编辑工具的章节。

在属性对话框中，任何包含数字值的字段也可以接受一个基本的数学表达式，从而得到一个数字值。例如，一个尺寸可以被输入为 $2 * 2 \text{ mm}$ ，结果是 4 mm 的值。支持基本算术运算符以及用于定义运算顺序的圆括号。

使用符号工作

放置符号

To place a symbol in your schematic, use the  button or the **A** hotkey. The Choose Symbols dialog appears and lets you select a symbol to add. Symbols are grouped by symbol library.



By default, only the symbol/library name and description columns are shown. Additional columns can be added by right-clicking the column header and selecting **Select Columns**.

The Choose Symbol dialog filters symbols by name, keywords, description, and all additional symbol fields according to what you type into the search field.

有一些高级筛选器可用：

- **Wildcards:** * matches any number of any characters, including none, and ? matches any single character.
- **键值对:** 如果一个库部分的描述或关键词包含一个格式为 "Key:123" 的标签，你可以通过输入 "Key>123" (大于)、"Key<123" (小于) 等进行相对匹配。数字可以包括以下不区分大小写的后缀之一。

p	n	u	m	k	meg	g	t
10^{-12}	10^{-9}	10^{-6}	10^{-3}	10^3	10^6	10^9	10^{12}

ki	mi	gi	ti
2^{10}	2^{20}	2^{30}	2^{40}

- **Regular expressions:** if you're familiar with regular expressions, these can be used too. The regular expression flavor used is the [wxWidgets Advanced Regular Expression style](#), which is similar to Perl regular expressions.

如果符号指定了一个默认的封装，这个封装将在右下方预览。如果符号包括封装过滤器，可以在右边的封装下拉菜单中选择满足封装过滤器的备用封装。

选择一个要放置的符号后，该符号将被附在光标上。左键点击原理图中所需要的位置，将符号放入原理图中。在将符号放入原理图之前，你可以通过热键或右键菜单来旋转它、镜像它和编辑它的字段。这些操作也可以在放置后进行。

如果 **放置重复副本** 选项被选中，在放置一个符号后 KiCad 将开始放置该符号的另一个副本。这个过程一直持续到用户按下 **Esc**。

对于有多个单元的符号，如果 **放置所有单元** 选项被选中，在放置符号后 KiCad 将开始放置该符号中的下一个单元。这将持续到最后一个单元被放置或用户按下 **Esc**。

放置电源符号

一个《电源符号，电源符号》是一个代表与电源网络连接的符号。这些符号被分组在 **power** 库中，所以它们可以使用符号选择器来放置。然而，由于电源的放置很频繁， 工具是可用的。这个工具也类似，只是搜索是直接 power 库和任何其他包含电源符号的库中进行。

移动符号

符号可以用移动 (**M**) 或拖动 (**G**) 工具移动。这些工具作用于选定的符号，或者如果没有选定符号，则作用于光标下的符号。

移动 工具移动符号本身，而不保持与符号引脚的导线连接。

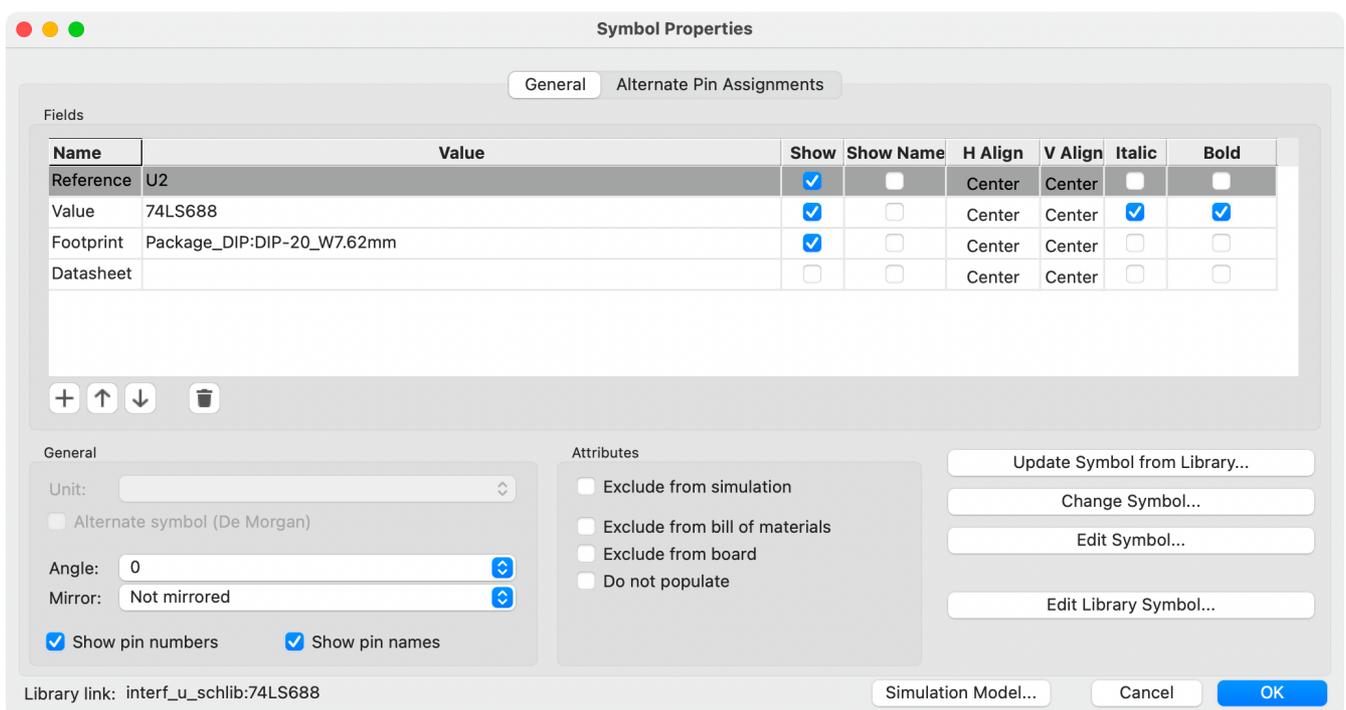
拖动 工具在移动符号时不会破坏其引脚的导线连接，因此也会移动连接的导线。

你也可以用鼠标点击并拖动符号，这取决于偏好设置中 **鼠标和触摸板** 部分的 **左键拖动手势** 设置。

符号也可以旋转 (**R**) 或在X (**X**) 或Y (**Y**) 方向上进行镜像。

编辑符号属性

符号的字段可以在符号的属性窗口中进行编辑。用 **E** 热键或通过双击符号来打开符号属性窗口。



符号属性窗口在一个表格中显示一个符号的所有字段。新的字段可以被添加，现有的字段可以被删除、编辑、重新排序、移动或调整大小。

每个字段的名称和值可以是可见的或隐藏的，并且有几个格式化选项：水平和垂直对齐、方向、位置、字体、文本颜色、文本大小和黑体/大写强调。字段的自动放置也可以在每个字段的基础上启用。对于正常显示的符号，显示的位置总是被指示的（没有旋转或镜像），并且是相对于符号的锚点而言的。

NOTE

符号字段的格式化选项可以通过右击符号字段表的标题行来显示或隐藏，并启用或禁用所需的列。默认情况下，不是所有的列都显示。

The **Update Symbol from Library...** button is used to update the schematic's copy of the symbol to match the copy in the library. The **Change Symbol...** button is used to swap the current symbol to a different symbol in the library. These functions are described [later](#).

编辑符号... 打开符号编辑器，编辑原理图中的符号副本。注意，库中的原始符号将不会被修改。**编辑库中的符号...** 按钮可以打开符号编辑器来编辑库中的原始符号。在这种情况下，原理图中的符号将不会被修改，直到用户点击 **更新库中的符号...** 按钮。

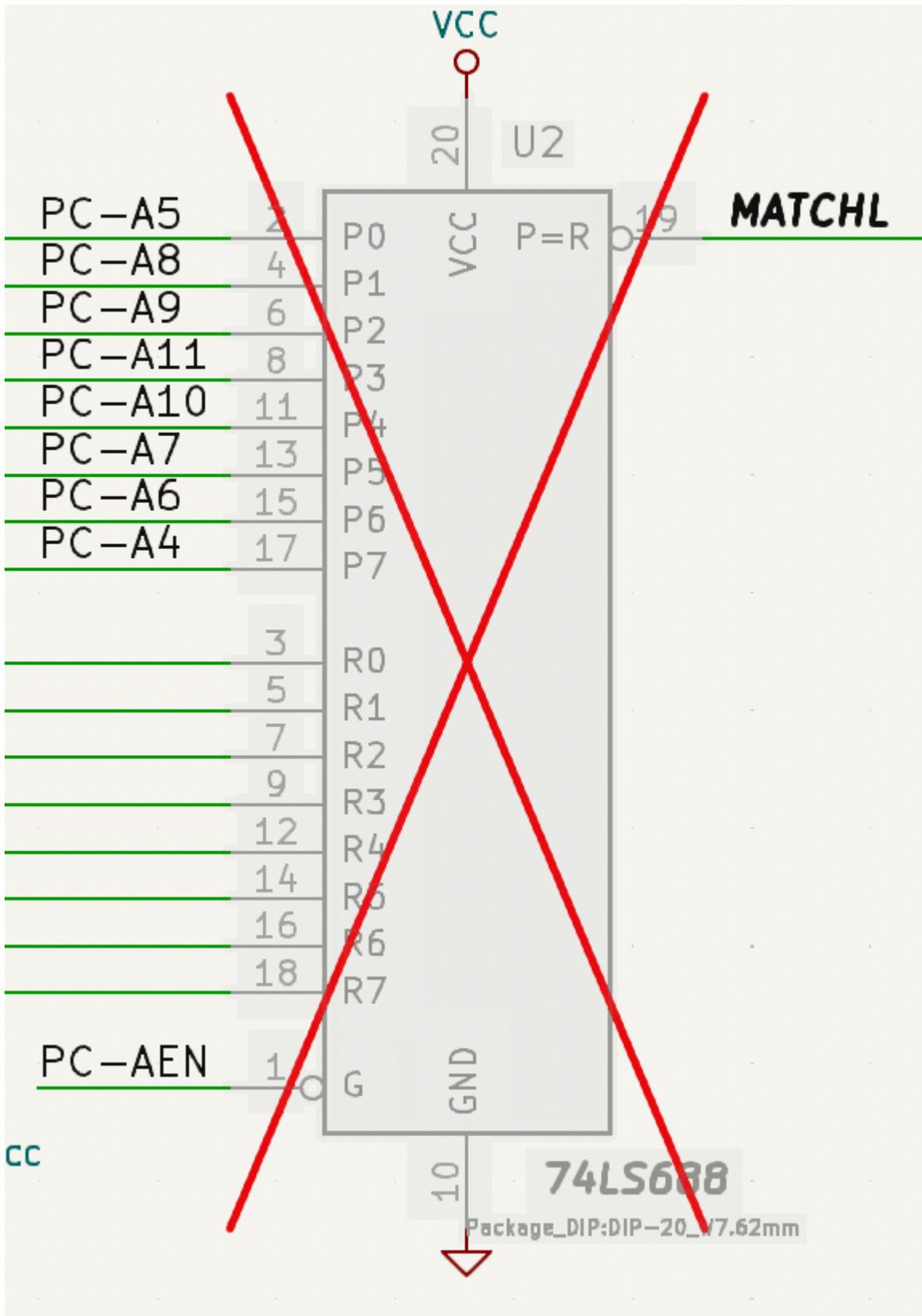
符号有几个属性，它们会影响 KiCad 其他部分对符号的处理方式。

从模拟中排除 防止该符号被包括在 SPICE 模拟中。

从 BOM 中排除 防止该元件被包括在《BOM 导出, BOM 导出》中。

从电路板中排除 意味着该符号仅用于原理图，相应的封装不会被添加到 PCB 上。

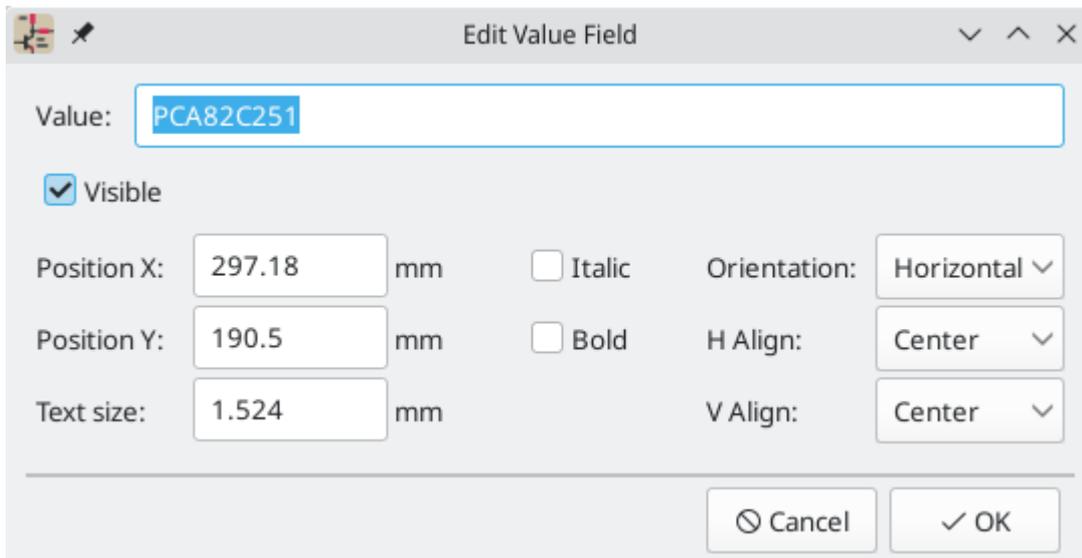
不使用 意味着该元件不应该被连接到 PCB 上，虽然相应的封装仍然应该被添加到电路板上。如下图所示，DNP 符号在原理图中显示为不饱和状态，上面有一个红色的 "X"。



单独编辑符号字段

可以用 **E** 快捷键直接编辑单个符号文本字段（选择字段而不是符号），或者双击字段。

一些符号字段有自己的快捷键，可以直接编辑它们。选定符号后，可分别用 **U**、**V** 或 **F** 快捷键编辑参考、值和封装字段。



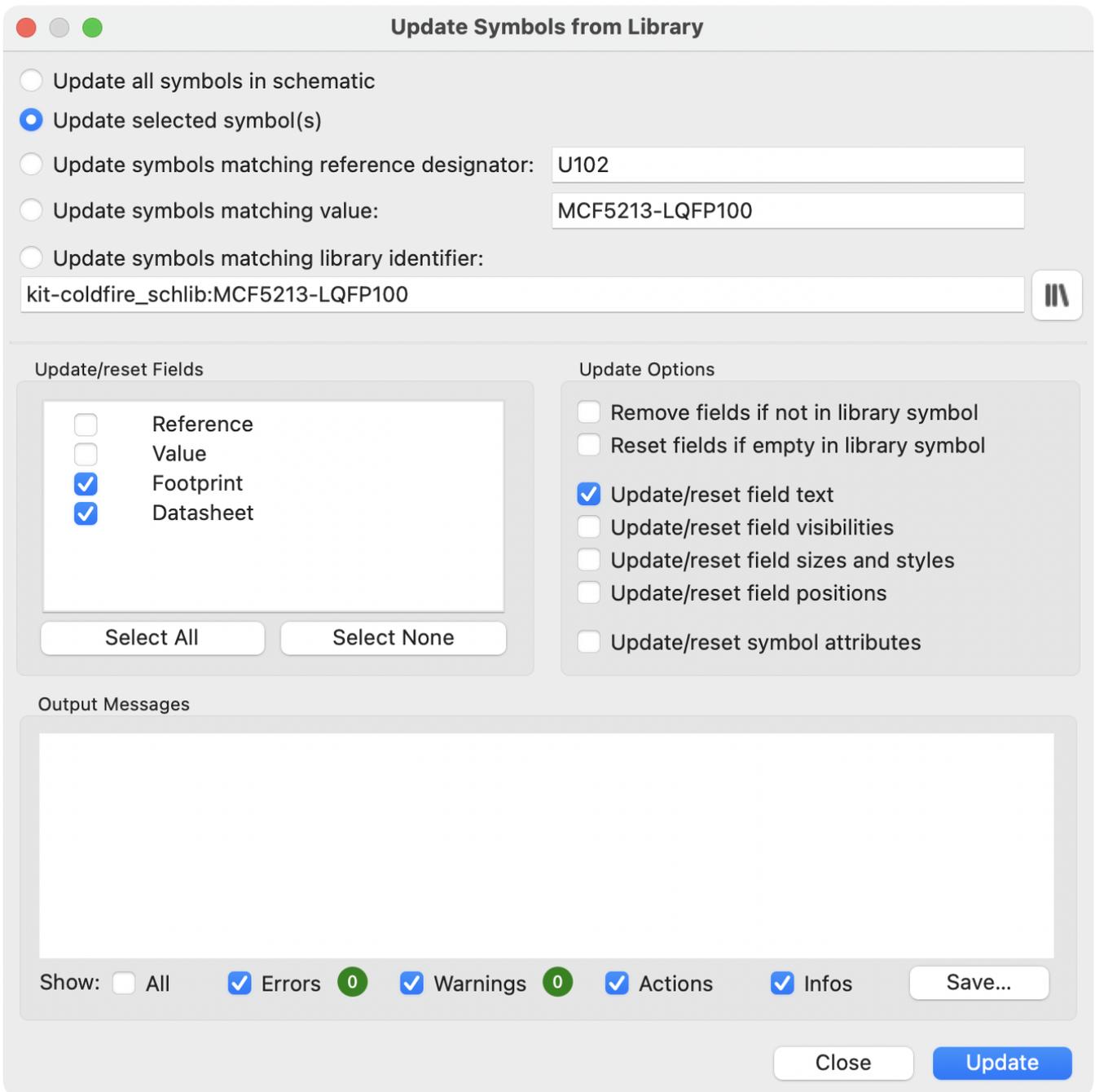
该对话框中的选项与完整的“符号属性”对话框中的选项相同，但都是针对单个字段的。

符号字段可以通过自动放置字段操作（选择一个符号并按 **0**）自动移动到一个适当的位置。字段自动放置可在原理图编辑器的编辑选项中进行配置，包括设置为总是自动放置字段。

Updating and exchanging symbols

When a symbol is added to the schematic, KiCad embeds a copy of the library symbol in the schematic so that the schematic is independent of the system libraries. Symbols that have been added to the schematic are not automatically updated when the library changes. Library symbol changes are manually synced to the schematic so that the schematic does not change unexpectedly.

To update symbols in the schematic to match the corresponding symbol, use **Tools** → **Update Symbols from Library**, or right click a symbol and select **Update Symbol from Library**. You can also access the tool from the [symbol properties dialog](#).

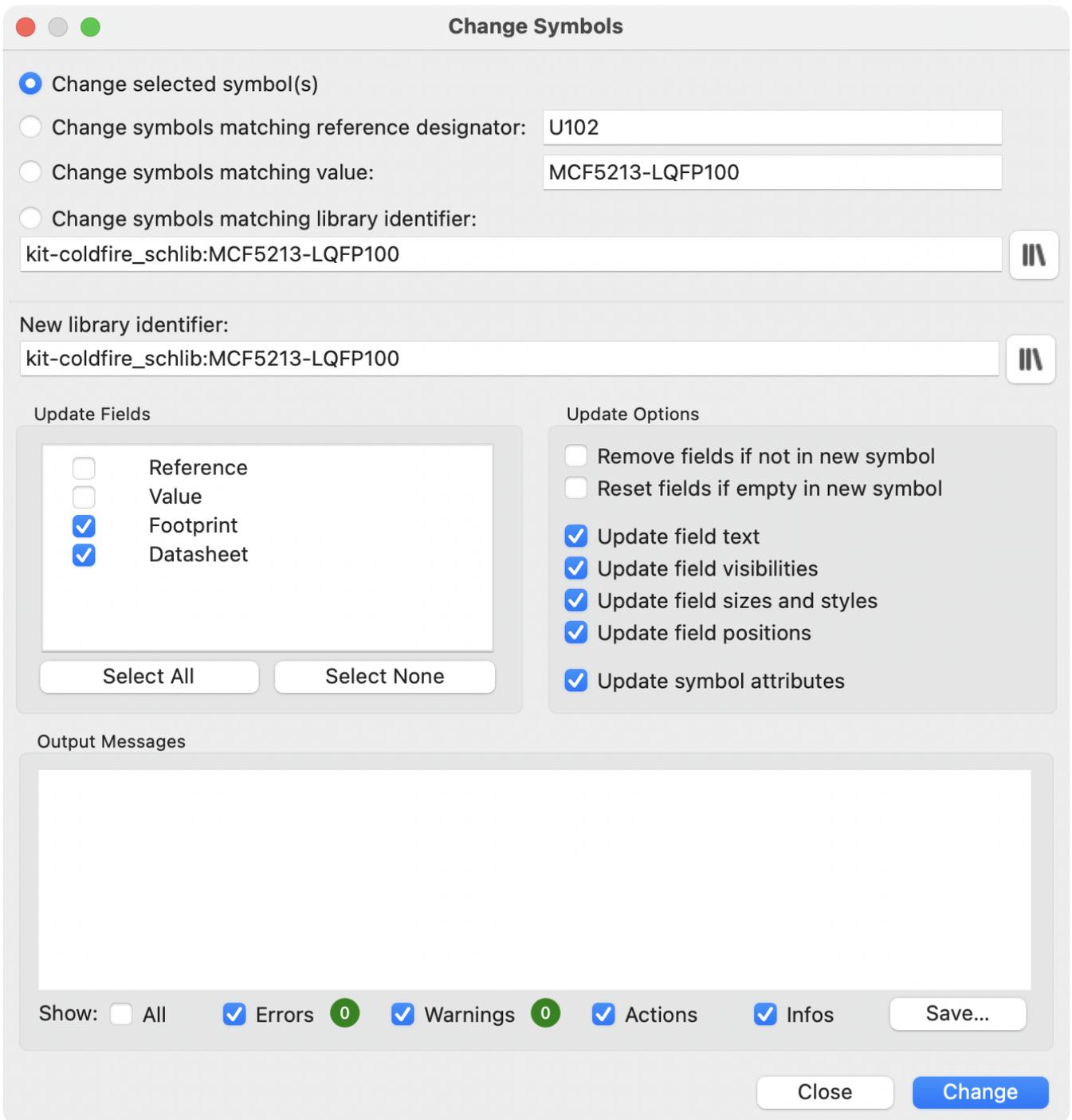


The top of the dialog has options to choose which symbols will be updated. You can update all symbols in the dialog, update only the selected symbols, or update only the symbols that match a specific reference designator, value, or library identifier. The reference designator and value fields support wildcards: * matches any number of any characters, including none, and ? matches any single character.

The middle of the dialog has options to control what parts of the symbol will be updated. You can select specific fields to update or not update, which properties of the fields to update (text, visibility, size and style, and position), and how to handle fields that are missing or empty in the library symbol. You can also choose whether to update symbol attributes, such as **do not populate** and **exclude from simulation / bill of materials / board**.

The bottom of the dialog displays messages describing the update actions that have been performed.

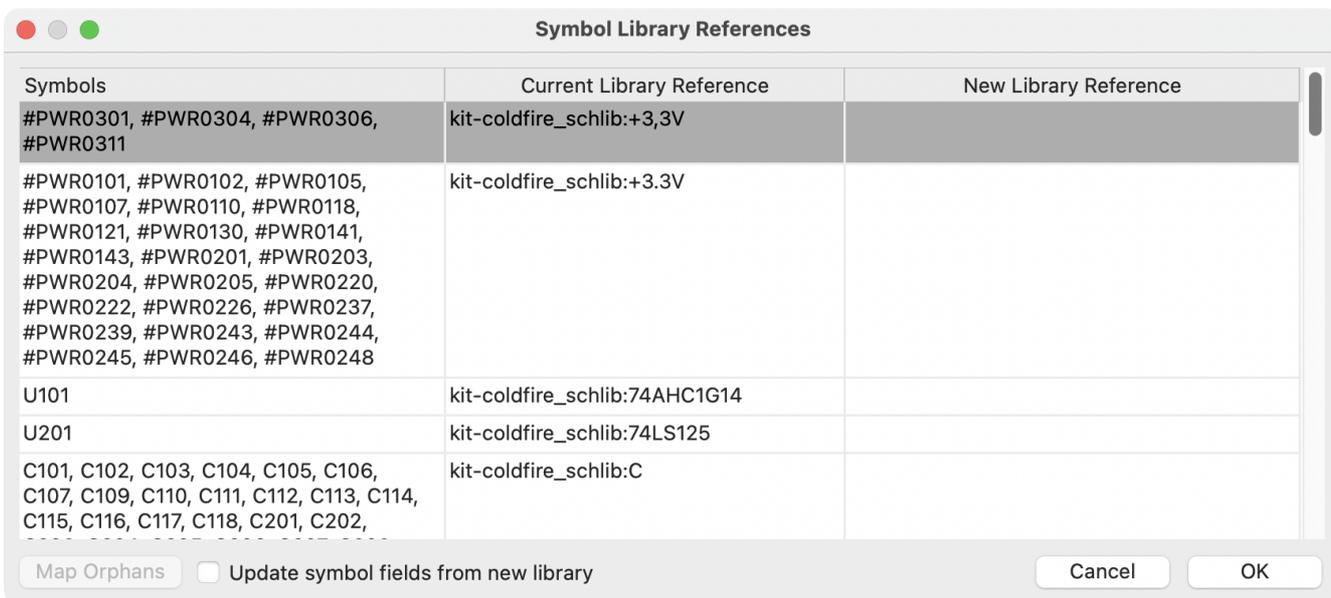
To change an existing symbol to a different symbol, use **Edit** → **Change Symbols...**, or right click an existing symbol and select **Change Symbol....** This dialog is also accessible from the [symbol properties dialog](#).



The options for the Change Symbols dialog are very similar to the Update Symbols from Library dialog.

Another way to swap existing symbols for new ones is to use **Tools** → **Edit Symbol Library Links...** This dialog contains a table of every symbol in the design, grouped by current library symbol. By choosing a new symbol in the **New Library Reference** column, you can make all instances of the existing symbol instead point to the new symbol. If the **Update symbol fields from new library** option is used, the contents of the existing symbols' fields will be updated to match the new symbols' fields.

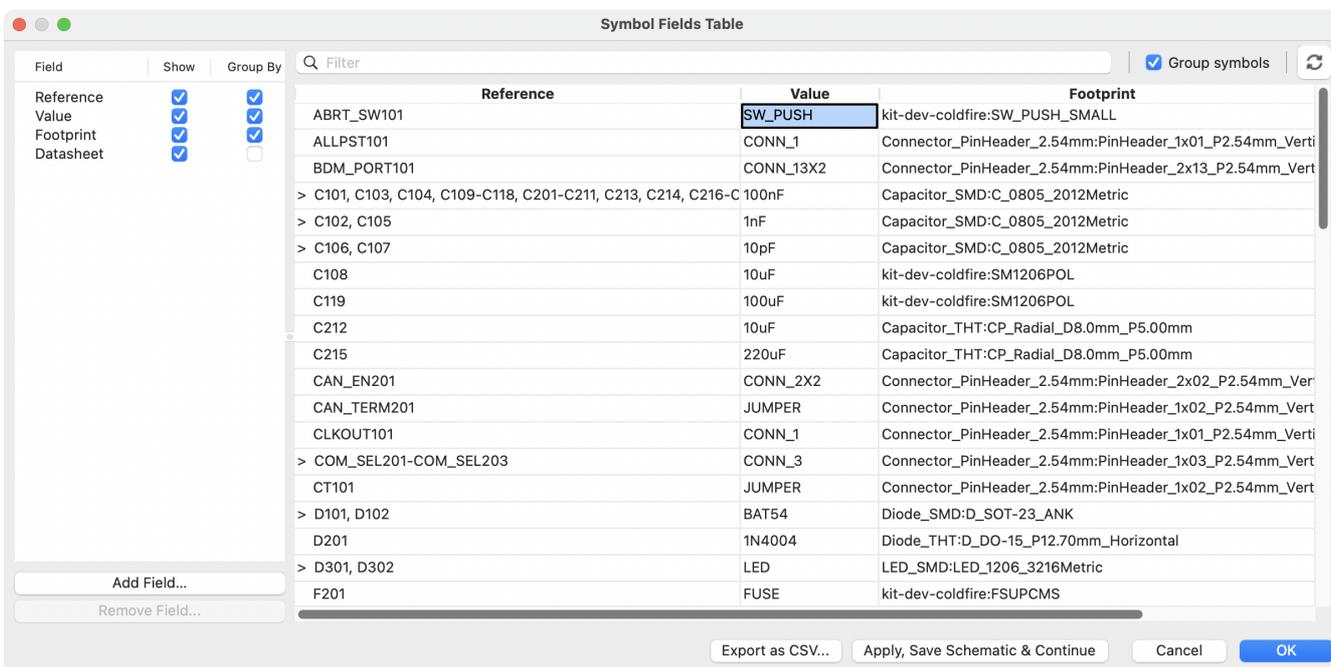
The **Map Orphans** button attempts to automatically remap orphaned symbols to symbols with the same name in an active library. For example, if there is a symbol with the current library reference `mylib:symbol123`, but the `mylib` library cannot be found, the **Map Orphans** button will attempt to find a symbol named `symbol123` in any of the libraries that are present. This button is only enabled if orphaned symbols are present in the schematic (see the [legacy schematics](#) section).



This dialog is primarily useful for managing symbols that appear in multiple libraries, when you want to switch from one library to another. For example, if a schematic uses symbols that are in both a global library and a project-specific library, the Symbol Library References dialog could be used to switch between using the global symbols or the equivalent project-specific symbols. It does not have features for fine-grained control of how fields are updated; for that, use the Change Symbols dialog.

符号字段表

符号字段表允许你在电子表格界面中查看和修改所有符号的字段值。你可以用  按钮打开符号字段表。



单元格可以用方向键导航，或者用 **Tab** / **Shift + Tab** 向右/向左移动，**Enter** / **Shift + Enter** 向下/向上移动。

通过点击和拖动可以选择一个单元格范围。选定的整个单元格范围将被复制到 (**Ctrl** + **C**) 或粘贴到 (**Ctrl** + **V**) 复制或粘贴操作上。从表中复制单元格的范围对于创建 BOM 非常有用。下面将介绍复制和粘贴单元格的更多细节。

任何符号字段都可以通过左边的 **显示** 复选框来显示或隐藏，或者通过右键点击表格的标题。新的符号字段可以用 **添加字段...** 按钮添加。

Similar symbols can optionally be grouped by any symbol field using the **Group By** checkboxes. Grouped symbols are shown in a single row in the table. The grouped row can be expanded to show the individual symbols by clicking the arrow at the left of the row. The **Group Symbols** checkbox enables or disables symbol grouping, and the  button recalculates groupings.

Symbols can be filtered using the **Filter** textbox at the top. The filter supports wildcards: ***** matches any number of any characters, including none, and **?** matches any single character.

你可以使用 **导出为 CSV...** 按钮，将符号字段保存到外部文件。这可以作为一个简单的 BOM 生成工具，尽管《BOM 导出，BOM 工具》提供了对生成的输出的更好控制。

简化填写字段的技巧

电子表格中有几种特殊的复制/粘贴方法，用于将数值粘贴到较大的区域，包括自动增加粘贴的单元格。这些功能在粘贴几个符号中共享的数值时可能很有用。

这些方法如下所示。

1. 复制 (Ctrl + C)	2. 选择目标单元格	3. 粘贴 (Ctrl + V)																																													
<table border="1"><tr><td>abc</td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	abc												<table border="1"><tr><td>abc</td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	abc												<table border="1"><tr><td>abc</td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td>abc</td><td>abc</td><td></td></tr><tr><td>abc</td><td>abc</td><td></td></tr></table>	abc						abc	abc		abc	abc										
abc																																															
abc																																															
abc																																															
abc	abc																																														
abc	abc																																														
<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12	13										<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12	13										<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td></td><td></td><td></td></tr><tr><td>11</td><td>12</td><td>13</td></tr><tr><td>11</td><td>12</td><td>13</td></tr><tr><td>11</td><td>12</td><td>13</td></tr></table>	11	12	13				11	12	13	11	12	13	11	12	13						
11	12	13																																													
11	12	13																																													
11	12	13																																													
11	12	13																																													
11	12	13																																													
11	12	13																																													
<table border="1"><tr><td>11</td><td></td><td></td></tr><tr><td>21</td><td></td><td></td></tr><tr><td>31</td><td></td><td></td></tr><tr><td>41</td><td></td><td></td></tr><tr><td>51</td><td></td><td></td></tr></table>	11			21			31			41			51			<table border="1"><tr><td>11</td><td></td><td></td></tr><tr><td>21</td><td></td><td></td></tr><tr><td>31</td><td></td><td></td></tr><tr><td>41</td><td></td><td></td></tr><tr><td>51</td><td></td><td></td></tr></table>	11			21			31			41			51			<table border="1"><tr><td>11</td><td>11</td><td>11</td></tr><tr><td>21</td><td>21</td><td>21</td></tr><tr><td>31</td><td>31</td><td>31</td></tr><tr><td>41</td><td>41</td><td>41</td></tr><tr><td>51</td><td>51</td><td>51</td></tr></table>	11	11	11	21	21	21	31	31	31	41	41	41	51	51	51
11																																															
21																																															
31																																															
41																																															
51																																															
11																																															
21																																															
31																																															
41																																															
51																																															
11	11	11																																													
21	21	21																																													
31	31	31																																													
41	41	41																																													
51	51	51																																													
<table border="1"><tr><td>11</td><td>12</td><td></td></tr><tr><td>21</td><td>22</td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12		21	22								<table border="1"><tr><td>11</td><td>12</td><td></td></tr><tr><td>21</td><td>22</td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12		21	22								<table border="1"><tr><td>11</td><td>12</td><td></td></tr><tr><td>21</td><td>22</td><td></td></tr><tr><td>11</td><td>12</td><td>11</td></tr><tr><td>21</td><td>22</td><td>21</td></tr><tr><td>11</td><td>12</td><td>11</td></tr></table>	11	12		21	22		11	12	11	21	22	21	11	12	11						
11	12																																														
21	22																																														
11	12																																														
21	22																																														
11	12																																														
21	22																																														
11	12	11																																													
21	22	21																																													
11	12	11																																													
<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td>21</td><td>22</td><td>23</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12	13	21	22	23							<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td>21</td><td>22</td><td>23</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	11	12	13	21	22	23							<table border="1"><tr><td>11</td><td>12</td><td>13</td></tr><tr><td>21</td><td>22</td><td>23</td></tr><tr><td></td><td></td><td></td></tr><tr><td>11</td><td>12</td><td></td></tr></table>	11	12	13	21	22	23				11	12										
11	12	13																																													
21	22	23																																													
11	12	13																																													
21	22	23																																													
11	12	13																																													
21	22	23																																													
11	12																																														

NOTE

这些技术在其他带有网格控制元素的对话框中也可以使用。

位号和符号注释

位号是设计中元件的唯一标识符。它们通常被印在 PCB 和装配图上，使你能够将原理图中的符号与电路板上的相应元件相匹配。

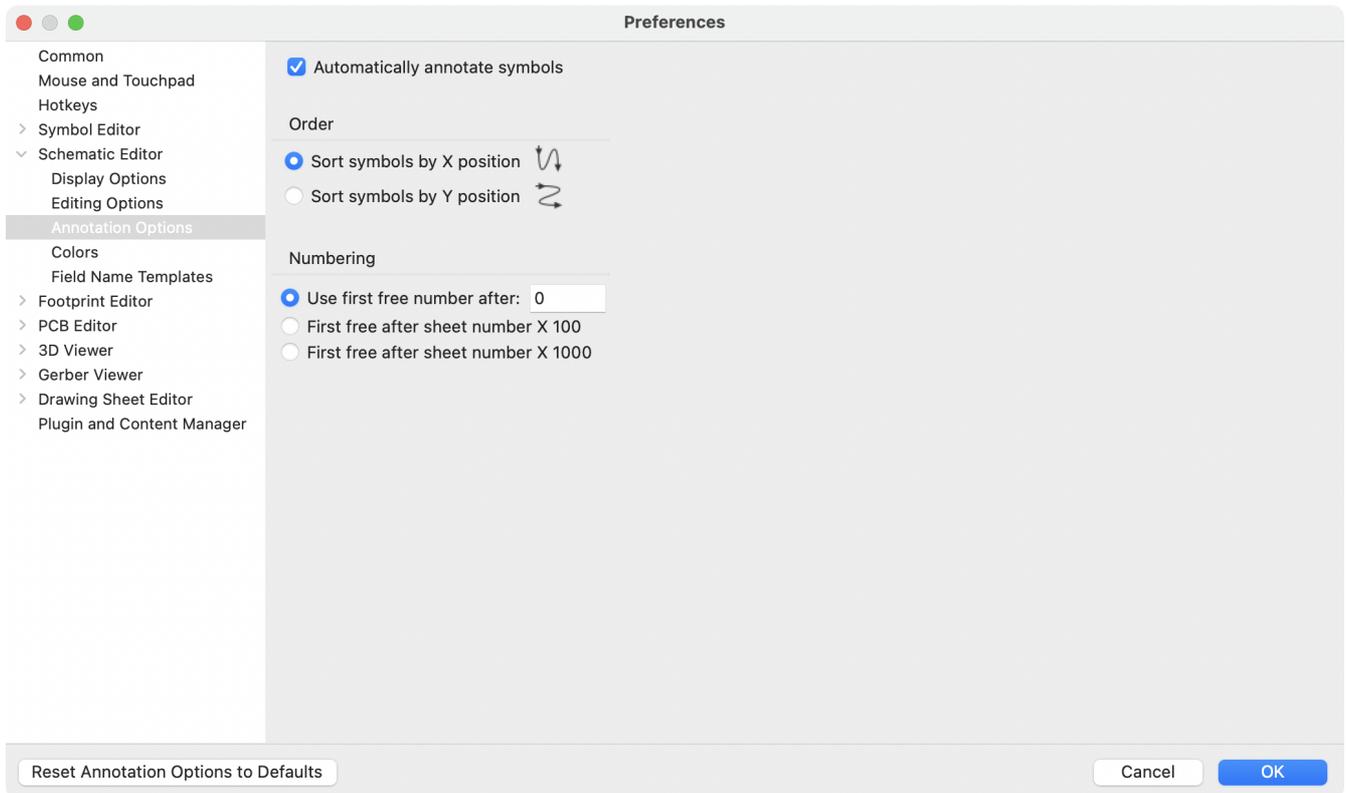
在 KiCad 中，位号由一个表示元件类型的字母组成（R 表示电阻，C 表示电容，U 表示集成电路，等等），后面是一个数字。如果符号有多个元位，那么位号也会有一个表示元位的尾部字母。没有设置位号的符号有一个 ? 字符，而不是数字。位号必须是唯一的。

位号可以在符号添加到原理图时自动设置，也可以通过手动编辑单个符号的位号字段或使用注释工具批量设置或重置位号。

NOTE 设置一个符号的位号的过程被称为 **批注**。

自动批注

启用自动批注后，符号被添加到原理图中时将被自动批注。你可以在 **偏好设置** 中 **原理图编辑器** → **批注选项** 窗格中选中 **自动批注符号** 复选框来启用自动批注。自动批注也可以通过左侧工具栏的  按钮进行切换。



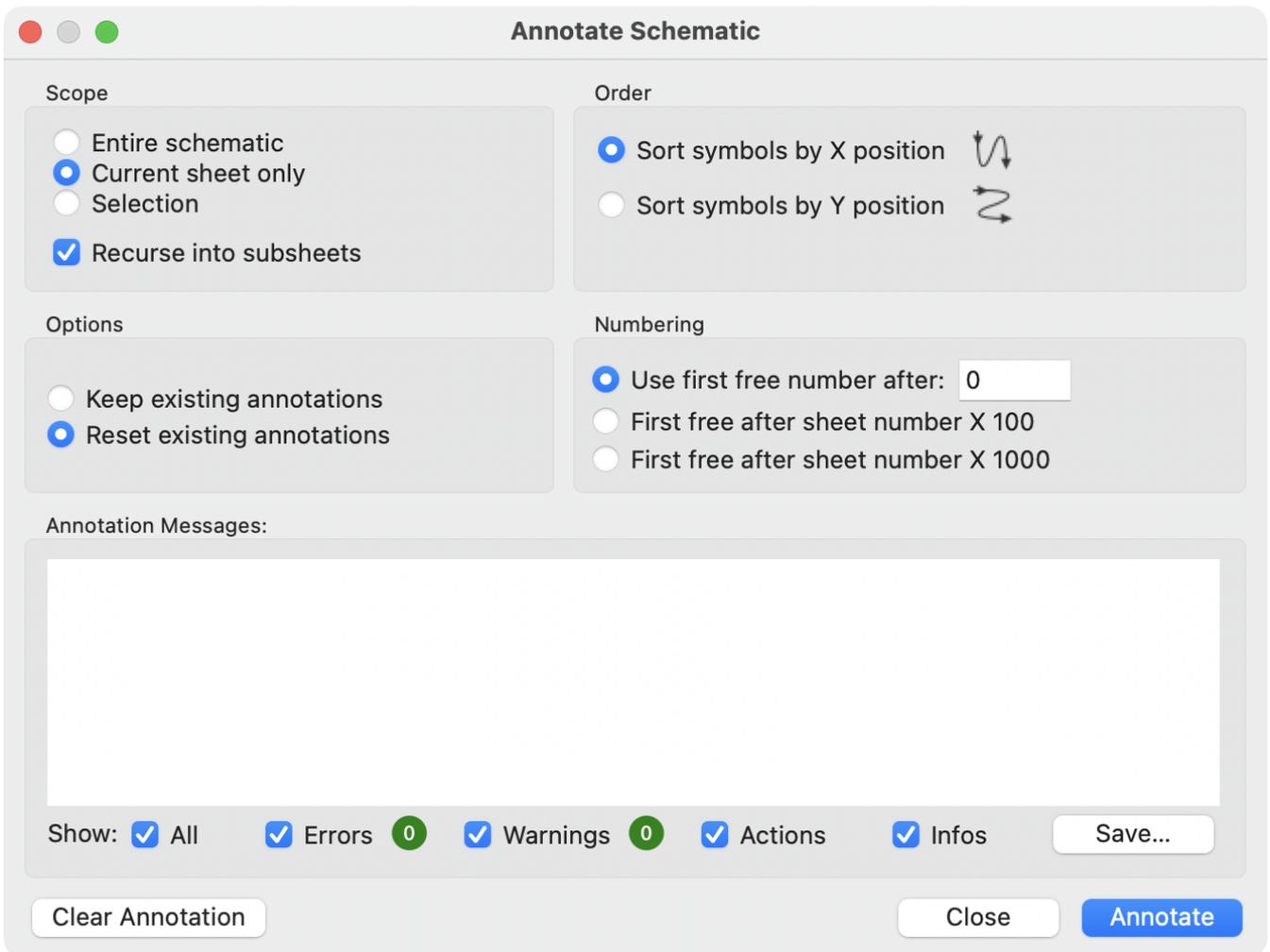
当同时添加多个符号时，它们会根据 **顺序** 设置进行批注，按 X 或 Y 位置排序。

编号 选项为新的位号设置起始编号。这可以是最低的可用数字，也可以是基于原理图编号的数字。

关于注解选项的更多信息，请参见《批注工具，批注工具》的文档。

批注工具

批注工具会自动为原理图中的符号指定位号。要启动批注工具，请点击顶部工具栏上的  按钮。



该工具提供了几个选项来控制符号的批注方式。

范围：选择批注是否适用于整个原理图、仅适用于当前原理图或仅适用于选定的符号。如果选择了 **递归到子原理图** 的选项，所选范围的子原理图中的符号将被重新批注；否则子原理图中的符号将不会被重新批注。例如，如果 **递归到子原理图** 和 **仅选择**，任何选定的子原理图中的符号将被重新批注。

选项：选择批注是否应适用于所有符号和重置 **现有的位号**，或仅适用于未批注的符号。

顺序：选择编号的方向。如果符号按 X 位置排序，原理图页面左侧的所有符号将比页面右侧的符号编号低。如果符号按 Y 位置排序，则原理图顶部的所有符号的编号将低于原理图底部的符号的编号。

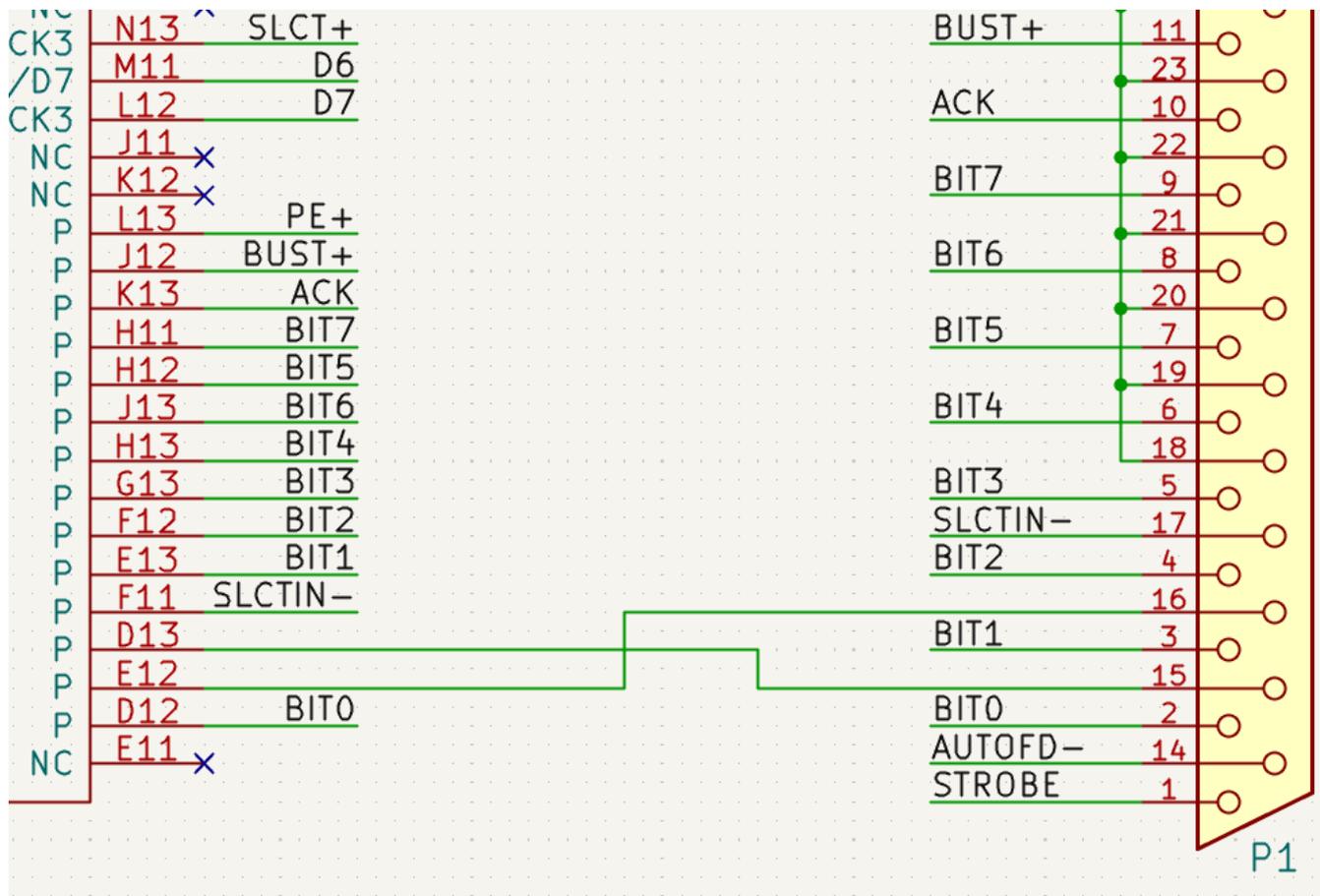
编号：选择编号位号的起点。每一个位号都会选择在起点之上的最低的未使用的数字。起始点可以是一个任意的数字（通常是 0），也可以是纸张编号乘以 100 或 1000，这样每个零件的位号就与它所在的原理想图页相对应。

清除批注 按钮可以清除所选范围内的所有位号。

批注信息可以用底部的复选框进行过滤，或用 **保存...** 按钮保存到报告中。

电气连接

有两种主要的方式来建立连接：导线和标签。导线进行直接连接，而标签则与具有相同名称的其他标签连接。下面的原理图中显示了导线和标签的情况。



也可以用总线进行连接，并通过隐藏电源引脚进行隐性连接。

本节还将讨论两种特殊类型的符号，可以用右侧工具栏上的"电源符号"按钮添加：

- **电源符号**：用于将电线连接到电源或地网络的符号。
- **PWR_FLAG**：一个特定的符号，用于表示一个网络在没有连接到电源输出引脚时是有电源的（例如，一个由板外连接器提供的电源网络）。

导线

导线用于在两点之间直接建立电气连接。要建立连接，必须将一段导线的末端与另一段导线或一个引脚相连。只有导线的末端才能建立连接；如果一根导线穿过另一根导线的中间，就不会建立连接。

未连接的导线端有一个小方块，表示连接点。当连接到导线端时，这个方块就会消失。未连接的引脚有一个圆圈，当进行连接时也会消失。

NOTE 导线只有在两端完全重合的情况下才能与其他导线或引脚连接。因此，保持符号引脚和导线与网格对齐很重要。建议在放置符号和绘制导线时始终使用 50 mil 的网格，因为 KiCad 标准符号库和所有遵循其风格的库也使用 50 mil 的网格。

NOTE 符号、导线和其他没有对准网格的元素，可以通过选择它们，右键点击，并选择 **将元素对准网格**，来将它们重新对准网格。

绘制和编辑电线

要开始用导线连接元素，请使用右侧工具栏中的导线工具 。也可以通过点击未连接的符号引脚或导线末端来自动启动导线。

你可以用左边工具栏上的  按钮将导线限制在 90 度角，或者用  按钮限制在 45 度角。 按钮允许你以任何角度放置导线。你可以使用 `Shift + Space` 在这些模式中循环，或者在 **偏好设置** → **原理图编辑器** → **编辑选项** 中选择所需的模式。这些模式除了影响导线外，还影响《图形线，图形线》。

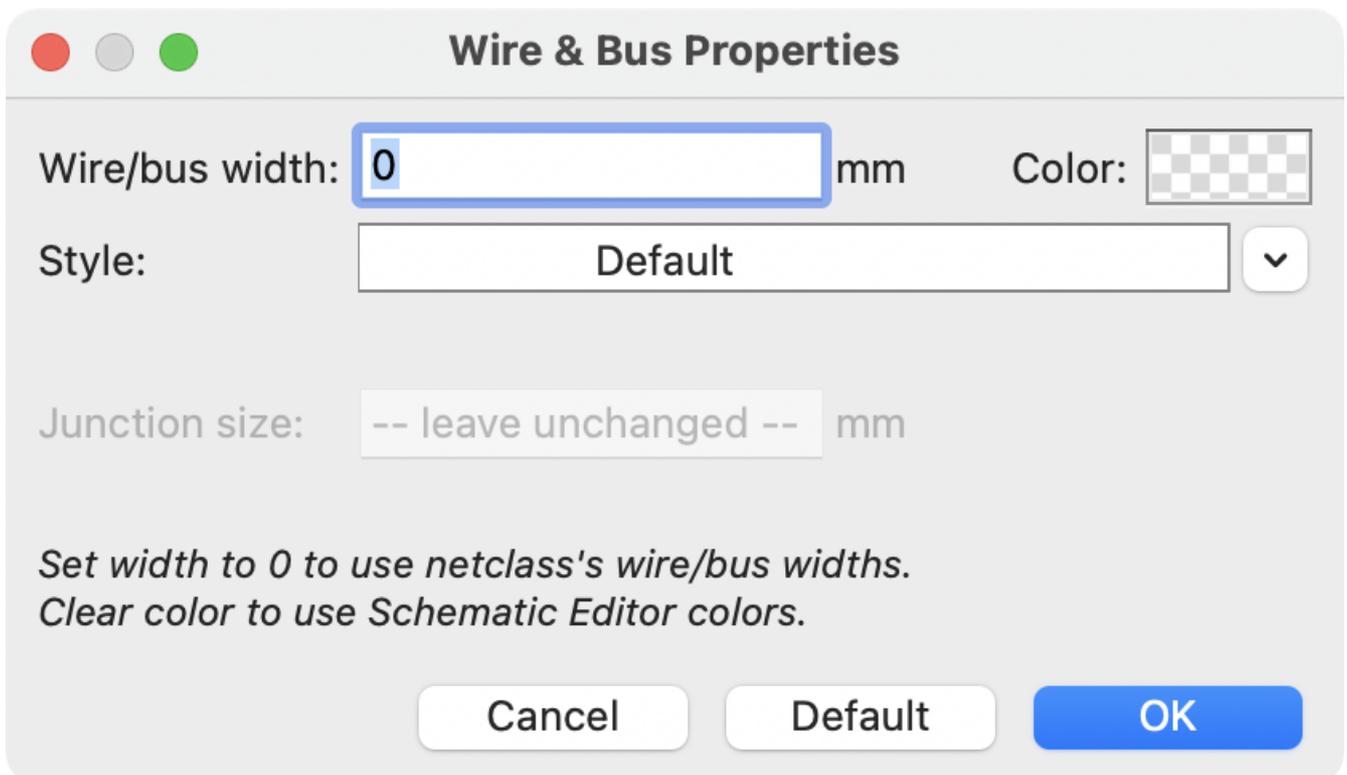
As in the PCB editor, the `/` hotkey switches wire posture.

可以使用移动 (`M`) 或拖动 (`G`) 工具来移动和编辑导线。与符号一样，**移动** 工具只移动选定的线段，而不保持与其他线段的现有连接。而 **拖动** 工具则保持现有的连接。

你可以使用 **选择连接** 工具 (`Alt + 4`) 选择连接的导线。这个工具可以选择所有连接的线段，直到到达一个结点，从所选线段或光标下的线段开始。再次使用该工具可以将现有的选择范围扩大到下一个结点。

你可以通过右击导线并选择 **切片** 将一个线段分成两块。该线段将在当前的鼠标位置被分开。你也可以通过右击一个线段并选择 **断开** 将其与相邻的线段分开。

通常情况下，线的线型是按照网的《原理图设置网络类，网络类设置》（如果没有指定其他网络类，网就处于默认网络类）。然而，所选线段的线型可以在线段的属性对话框中被覆盖（当线段被选中时 `E`）。可以设置线的宽度、颜色和线的样式（实线、虚线、点线等）。将宽度设置为 0，清除颜色，并使用默认线条样式，分别使用网络类设置中的默认宽度、颜色和样式。如果在选择中包含了一个导线结点，结点的大小也可以在这里进行编辑。

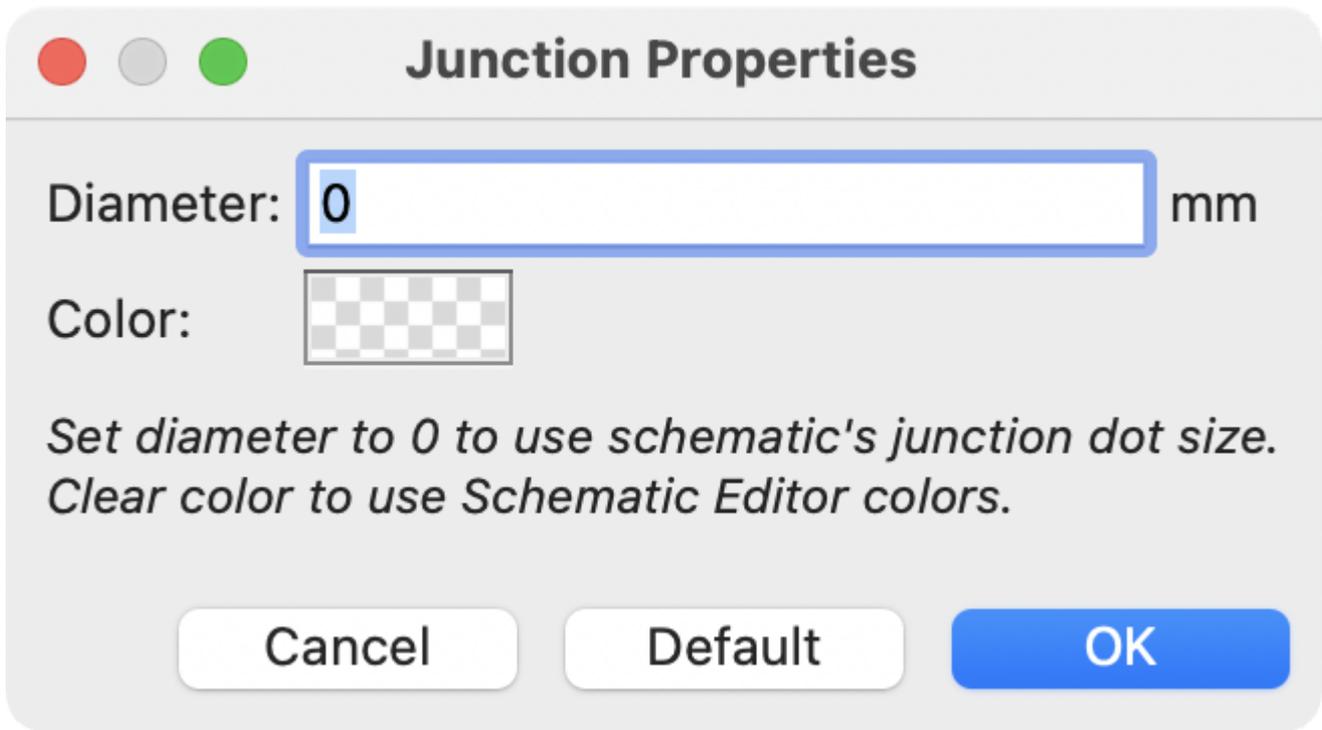


导线结点

交叉的导线不会被隐含地连接起来。如果需要连接，有必要通过明确地添加结点来连接它们 ( 按钮在右侧工具栏)。交叉点将被自动添加到开始或结束于现有导线之上的导线。

在上面的原理图中，连接到 P1 引脚 18、19、20、21、22 和 23 的导线上使用了结点。

结点大小自动遵循原理图的 **结点大小** 设置，在 **原理图设置** → **通用** → **格式化**。颜色遵循《原理图设置网络类，网络类设置》。自动尺寸和颜色可以在每个连接点的属性中被覆盖；尺寸为 0 相当于原理图的默认尺寸，而清除颜色则使用网络类颜色。



标签

标签是用来给导线和引脚分配网络名称的。具有相同网络名称的导线被认为是连接在一起的，所以标签可以用来进行连接，而不需要直接画线连接。

一个网络只能有一个名称。如果有两个不同的标签放在同一个网络上，将产生一个 ERC 违规。在网表中只使用其中一个网络名称。最终的网络名称是根据下面描述的《网络名称分配规则》决定的。

有三种类型的标签，每种都有不同的连接范围。

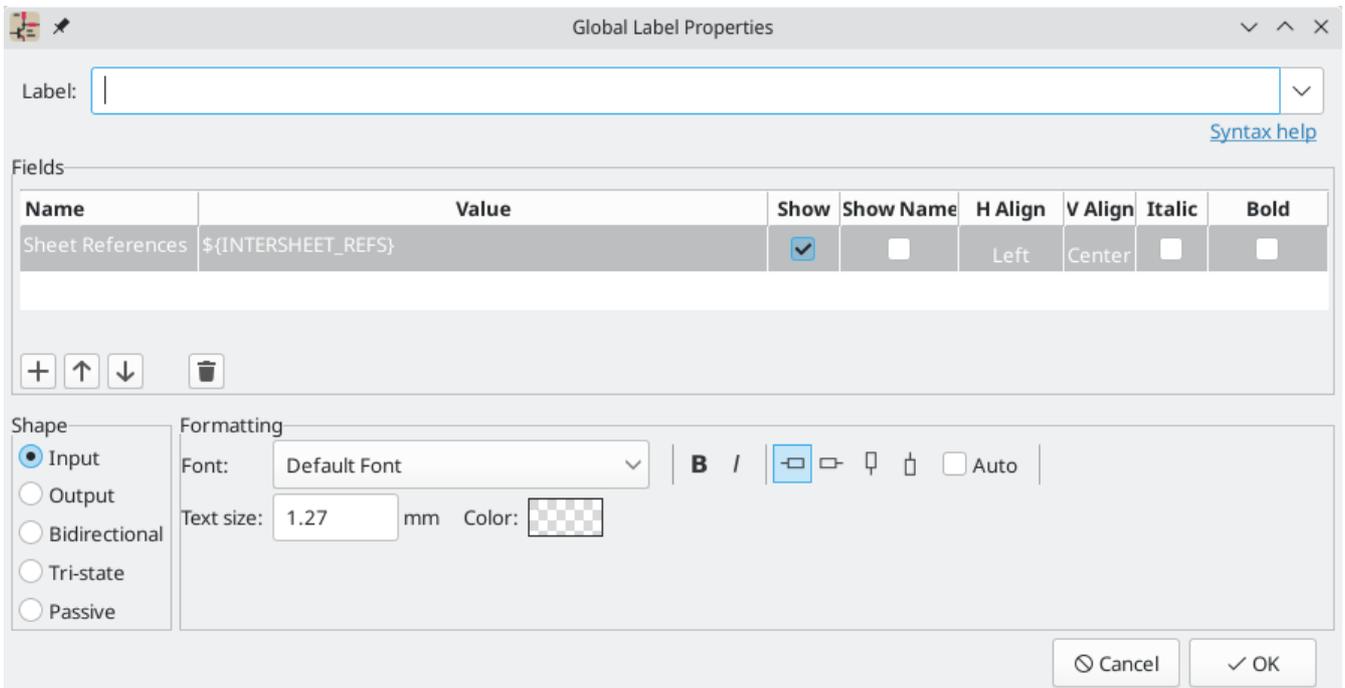
- **局部标签**，也被简单地称为标签，只在一个原理图内进行连接。用右边工具栏上的  按钮添加一个局部标签。
- **全局标签** 可以在原理图的任何地方进行连接，而不考虑原理图。用右边工具栏上的  按钮添加一个全局标签。
- **层次标签** 连接到层次的原理图页码，在《层次原理图，层次原理图》中用于连接子原理图和其父原理图。用右边工具栏上的  按钮添加层次标签。

NOTE

具有相同名称的标签将被连接，无论标签类型如何。如果它们在同一张原理图上。

添加和编辑标签

使用适当的按钮或快捷键创建标签后，会出现标签属性对话框。



标签 字段设置了标签的文本，它决定了标签分配给其连接线的网络。标签文本支持《文本标记，标记》，用于覆盖条、下标等，以及《文本变量，变量替换》。使用对话框中的 **语法帮助** 链接来获得摘要。

有几个选项可以控制标签的外观。你可以改变文本的《字体，字体》、大小和颜色，并设置粗体和斜体强调。你还可以设置文本相对于标签连接点的方向。层次和全局标签有几个额外的选项：**自动** 选项根据连接的原理图元素自动设置标签方向，**形状** 选项控制标签轮廓的形状（**输入**、**输出**、**双向**、**三态** 或 **被动**）。轮廓形状纯粹是视觉上的，没有电气上的影响。

NOTE

The default text size can be set for a schematic in [Schematic Setup](#), and the default font can be set in [Preferences](#).

NOTE

Global labels have additional settings to control margins around the label text in the [Schematic Setup dialog](#).

标签也可以添加字段。有两个字段有特殊意义（“网络类”和“位表间参考”，下面会介绍），但也可以添加任意字段。标签字段的行为类似于《编辑符号属性，符号字段》：你可以显示或隐藏它们的名称和值，调整对齐方式、方向、位置、大小、字体、颜色和强调。

NOTE

通过右击标签字段表的标题行，启用或禁用所需的列，可以显示或隐藏标签字段的格式化选项。默认情况下，不是所有的列都显示。

与符号字段一样，标签字段可以通过从原理图中打开特定标签字段的属性进行单独编辑（双击标签字段，或使用 **E**）。

接受标签属性后，标签将被连接到光标上进行放置。标签的连接点是标签角落里的小方块。当标签被连接到导线或引脚的末端时，这个方块就会消失。

unconnected_label



连接点相对于标签文本的位置可以通过在标签的属性中选择不同的标签方向，或通过镜像/旋转标签来改变。

通过选择一个标签并使用 **E** 快捷键、双击该标签或在右键菜单中使用 **属性...**，可以随时访问标签属性对话框。

用标签分配网络类

In addition to assigning net names, labels can be used to assign net classes. A label field named **Net Class** assigns the specified netclass to the net associated with the label. To make it easier to assign net classes in this way, **Net Class** is the default name for new label fields, and **Net Class** fields present a dropdown list of all the net classes in the design. Net classes must be created in the **Schematic Setup** or **Board Setup** windows before they can be assigned with a label field.

关于分配网络类的更多信息，请参见《原理图网络类，网络类文档》。

表间参考

全局标签可以显示表间参考，这是原理图中出现同一全局标签的其他地方的页码列表。点击表间参考就可以进入所列出的页面。如果列出了多个位号，点击位号列表会出现一个菜单来选择所需的页面。

Inter-sheet references are globally controlled in the **Schematic Setup** window's Formatting page. References can be enabled or disabled, and the displayed format for the list can be adjusted, including with optional prefix or suffix characters.

下图显示了一个全局标签，其中有对其他两个原理图页面的之间位号。在原理图设置中，分别添加了前缀和后缀 [和]。

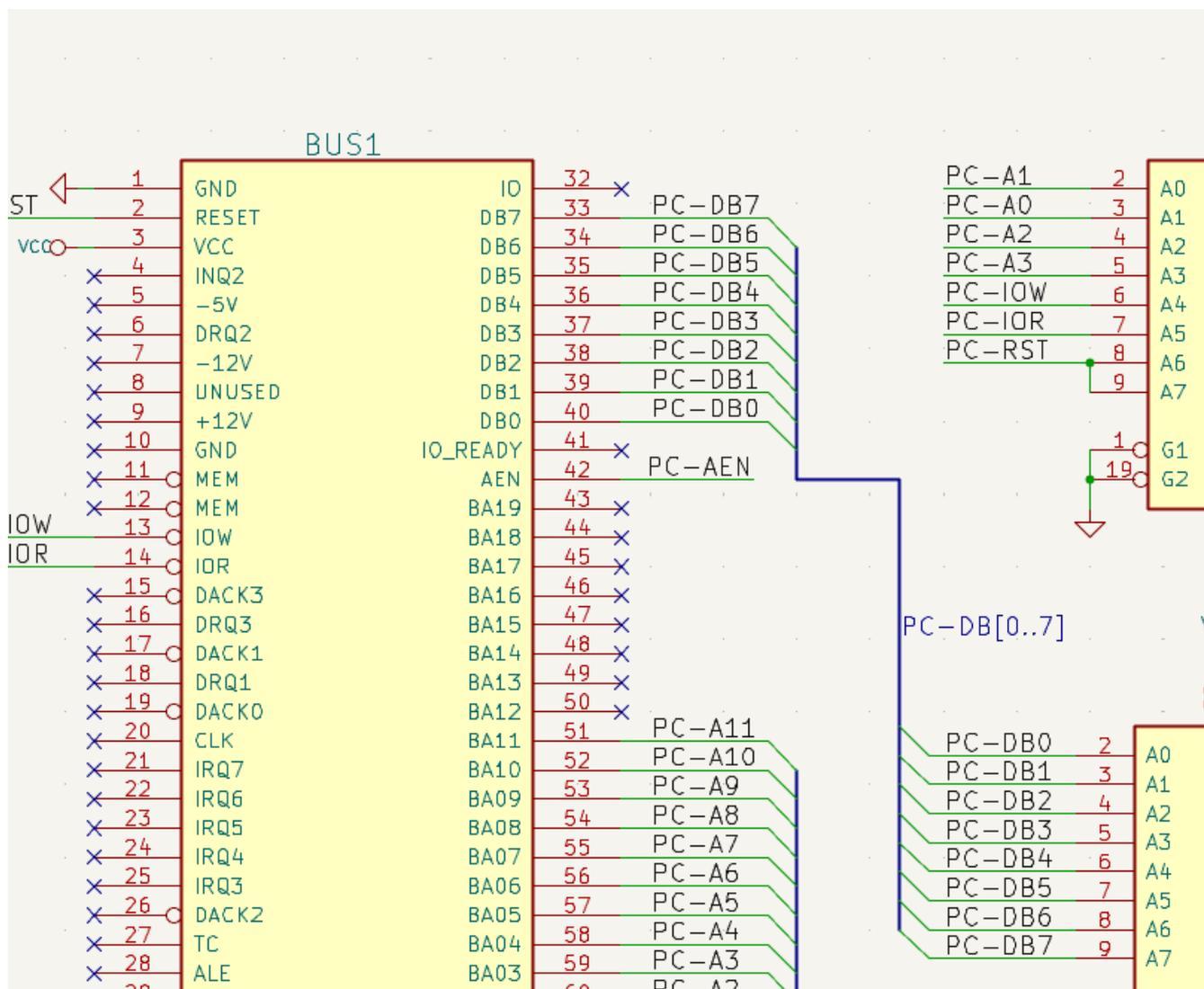


全局标签会自动添加一个“表间参考”字段，其值为“`#{INTERSHEET_REFS}`”，用于控制该标签的表间参考。`#{INTERSHEET_REFS}` 文本变量会被扩展为全局标签的表间参考的完整列表，如在原理图设置中所配置。表间参考的可见性在“原理图设置”中是全局控制的，而不是通过“表间参考”字段的可见性控制。对于其他类型的标签，“表间参考”字段没有意义。

总线

总线是一种在原理图中分组相关信号的方法，以简化复杂的设计。总线可以像导线一样用总线工具画出来，并像信号线一样用标签命名。

在下面的原理图中，许多引脚都连接到总线上，也就是中间的蓝色粗线。



总线编号

在 KiCad 6.0 及以后版本中，有两种类型的总线：矢量总线和分组总线。

一个**矢量总线**是一个信号的集合，以一个共同的前缀开始，以一个数字结束。矢量总线被命名为 <前缀>[M..N]，其中`前缀`是任何有效的信号名称，M 是第一个后缀数字，N 是最后的后缀数字。例如，总线 DATA[0..7] 包含信号 DATA0、DATA1，以此类推，直到 DATA7。指定 M 和 N 的顺序并不重要，但两者必须是非负数。

一个**分组总线**是一个或多个信号和/或矢量总线的集合。分组总线可以用来把相关的信号捆绑在一起，即使它们有不同的名字。分组总线使用一种特殊的标签语法：

```
<OPTIONAL_NAME>{SIGNAL1 SIGNAL2 SIGNAL3}
```

该分组的成员被列在大括号 ({}) 内，用空格字符隔开。在大括号的后面有一个可选的分组名。如果分组总线没有命名，PCB 上的结果网络将只是该分组内的信号名称。如果分组总线有名字，产生的网路将以名字为前缀，用句号 (.) 分隔前缀和信号名称。

例如，总线 {SCL SDA} 有两个信号成员，在网表中这些信号将是 SCL 和 SDA。总线 USB1{DP DM} 将产生名为 USB1.DP 和 USB1.DM 的网表。对于在几个类似电路中重复出现的较大的总线的设计，使用这种技术可以节省时间。

分组总线也可以包含矢量总线。例如，总线 MEMORY{A[7..0] D[7..0] OE WE} 同时包含了矢量总线和普通信号，并将在 PCB 上形成 MEMORY.A7 和 MEMORY.OE 这样的网络。

总线的绘制和连接方式与信号线相同，包括使用结点来创建交叉线之间的连接。与信号线一样，总线不能有一个以上的名称—如果在同一条总线上有两个冲突的标签，将产生一个 ERC 违规。

总线成员之间的连接

在总线的相同成员之间连接的引脚必须通过标签连接。不可能将一个引脚直接连接到总线上；这种连接方式将被 KiCad 忽略。

在上面的例子中，连接是通过放置在连接到引脚的导线上的标签来实现的。通往总线的总线入口（45 度的线段）只是图形化的，并不是形成逻辑连接的必要条件。

事实上，使用重复命令（`Insert`），如果元件引脚以递增的顺序排列，可以非常快速地进行连接（在实践中，这种情况常见于存储器、微处理器等元件）：

- 放置第一个标签（例如：PCA0）。
- 根据需要使用重复命令来放置成员。KiCad 将自动创建下一个标签（PCA1，PCA2...）垂直对齐，理论上是在其他引脚的位置上。
- 在第一个标签下画线。然后使用重复命令将其他导线放在标签下。
- 如果需要，以同样的方式放置总线入口（放置第一个入口，然后使用重复命令）。

NOTE

在偏好设置菜单的 **原理图编辑器** → **编辑选项** 部分，你可以设置重复的参数：

- 水平间距
- 垂直间距
- 标签递增（标签可以递增或递减 1、2、3，等等）。

总线展开

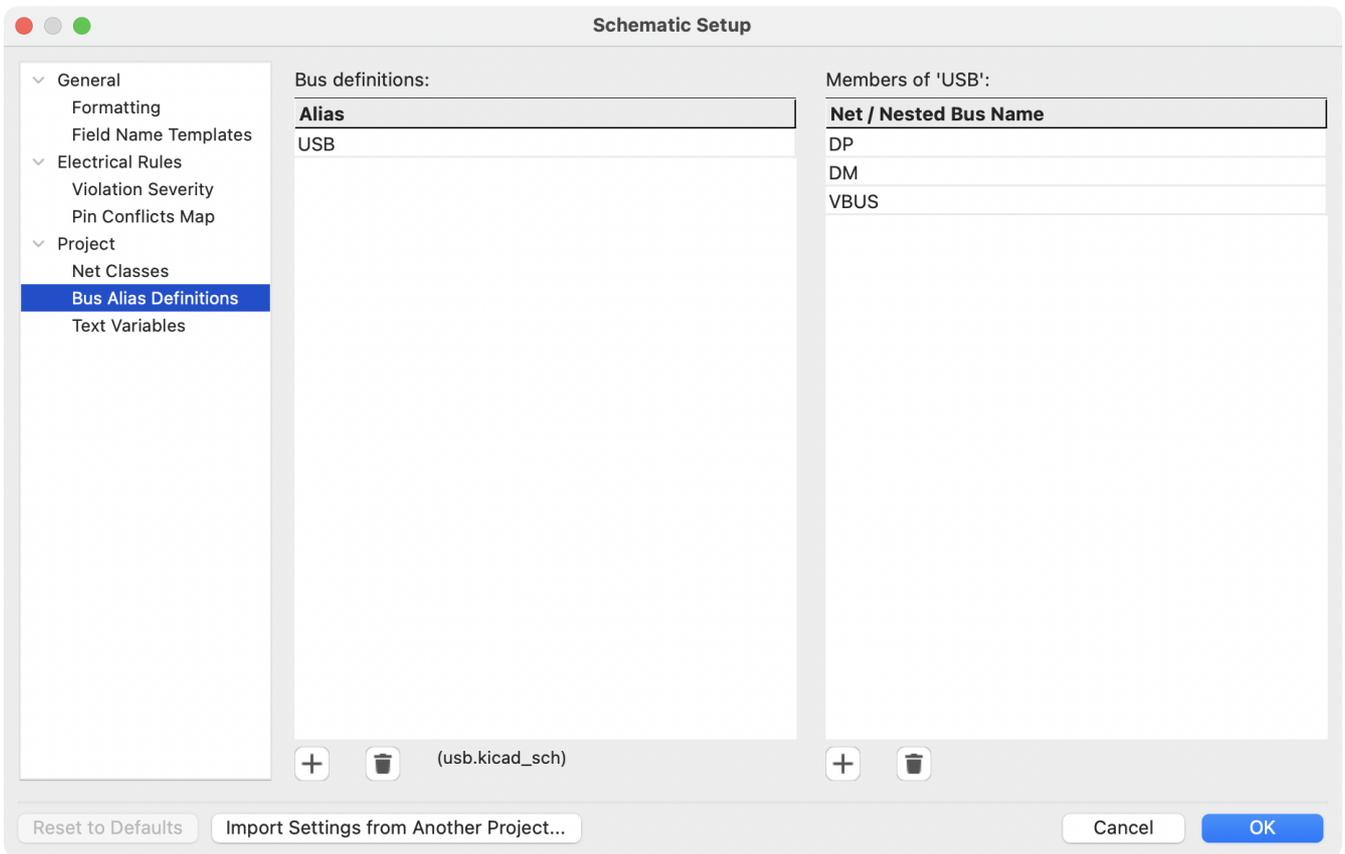
展开工具允许你快速地从总线上分离出信号。要展开一个信号，请右击一个总线对象（一个总线导线等），并选择 **从总线上展开**。或者，当光标在一个总线对象上时，使用 **展开总线** 快捷键（默认：`C`）。该菜单允许你选择要展开的总线成员。

在选择总线成员后，下一次点击将把总线成员的标签放在所需的位置。该工具会自动生成一个总线入口和通向标签位置的导线。放置完标签后，你可以继续放置额外的线段（例如，连接到一个元件引脚），并以任何正常方式完成布线。

总线别名

总线别名是一种快捷方式，可以让你更有效地处理大型组总线。它们允许你定义一个组总线，并给它一个简短的名称，然后可以在整个原理图中代替完整的组名称。

要创建总线别名，请在《原理图设置，原理图设置》中打开 **总线别名定义** 窗格。



一个别名可以被命名为任何有效的信号名称。使用该对话框，你可以向别名添加信号或矢量总线。作为一种快捷方式，你可以输入或粘贴信号和/或总线的列表，用空格隔开，它们将全部被添加到别名定义中。在这个例子中，我们定义了一个名为 USB 的别名，成员为 DP、DM 和 VBUS。

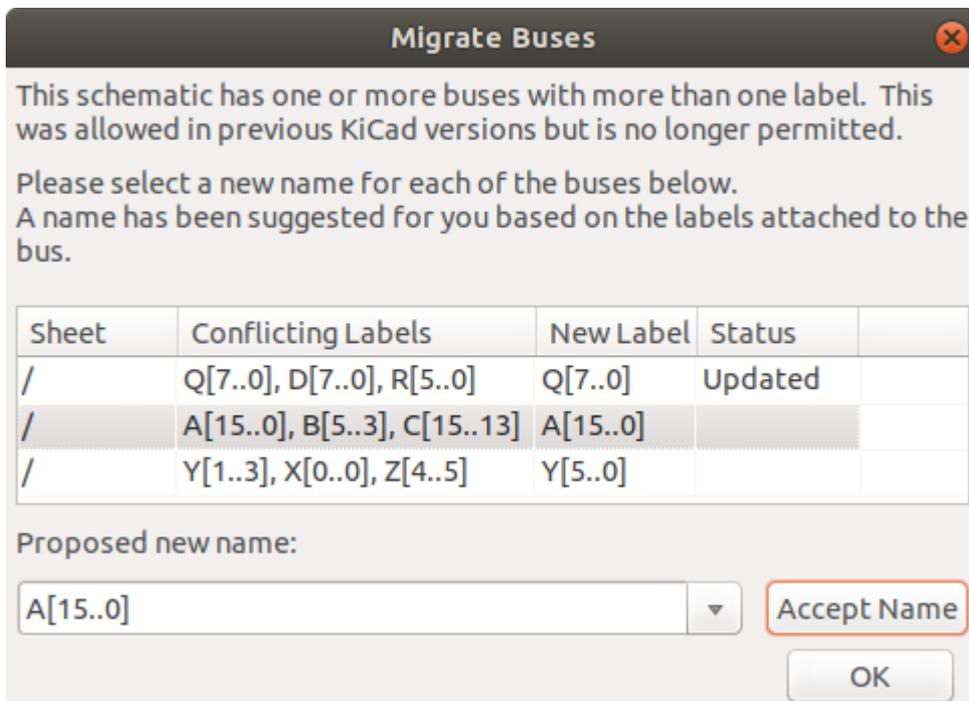
定义别名后，可以在组总线标签中使用，方法是将别名放在分组总线的大括号内：`{USB}`。这与给总线贴上`{DP DM VBUS}`标签的效果相同。你也可以给分组添加一个前缀名，比如 `USB1{USB}`，这样就会产生如上面所说的 `USB1.DP` 这样的网路。对于复杂的总线，使用别名可以使原理图上的标签短得多。请记住，别名只是一个快捷方式，别名的名称不包括在网表中。

总线别名被保存在创建别名时打开的原理图文件中。在 **总线别名定义** 窗口中，与所选别名相关的原理图文件显示在别名列表的底部。在一个给定的原理图工作表中创建的任何别名都可以在同一层次设计中的任何其他原理图工作表中使用。如果一个层次设计中的多个原理图包含相同名称的总线别名，这些别名必须都有相同的成员。如果多个同名的总线别名没有一致的成员，《ERC 检查列表，ERC 将报告违规情况》。

有多个标签的总线

KiCad 5.0 和更早的版本允许将具有不同标签的总线连接在一起，并在网表编制时将这些总线的成员连接起来。这种行为在 KiCad 6.0 中被删除了，因为它与分组总线不兼容，而且还会导致网表的混乱，因为一个给定的信号将得到的名称不容易预测。

如果您在现代版本的 KiCad 中打开使用此功能的设计，您将看到“迁移总线”对话框，该对话框将指导您更新原理图，以便在任何给定的总线线路上只存在一个标签。



对于具有多个标签的每组总线，您必须选择要保留的标签。下拉名称框允许您在设计中存在的标签之间进行选择，或者您可以通过手动将其输入新名称字段来选择其他名称。

隐藏电源引脚

当一个符号的电源引脚是可见的，它们必须被连接，就像其他信号一样。然而，诸如门和触发器等符号有时会画出隐含的电源输入引脚，这些引脚是隐性连接的。

KiCad 会自动将类型为 "电源输入" 的隐形引脚连接到与该引脚同名的全局网络。例如，如果一个符号有一个名为 VCC 的隐藏电源输入引脚，这个引脚将被全局连接到所有原理图上的 VCC 网络。

NOTE

隐藏的引脚可以在原理图中显示，方法是在 **原理图编辑器** → 偏好设置的 **显示选项** 部分勾选 **显示隐藏的引脚**，或者选择 **视图** → **显示隐藏的引脚**。在左边的工具栏上还有一个切换图标 。

可能有必要将不同名称的电源网络连接起来（例如，TTL 元件中的 GND 和 MOS 元件中的 VSS）。为了达到这个目的，为每个网络添加一个《电源符号,电源符号》，并用一根导线连接它们。

如果使用隐藏的电源引脚，不建议使用本地标签进行电源连接，因为它们不会连接到其他片上的隐藏电源引脚。

NOTE

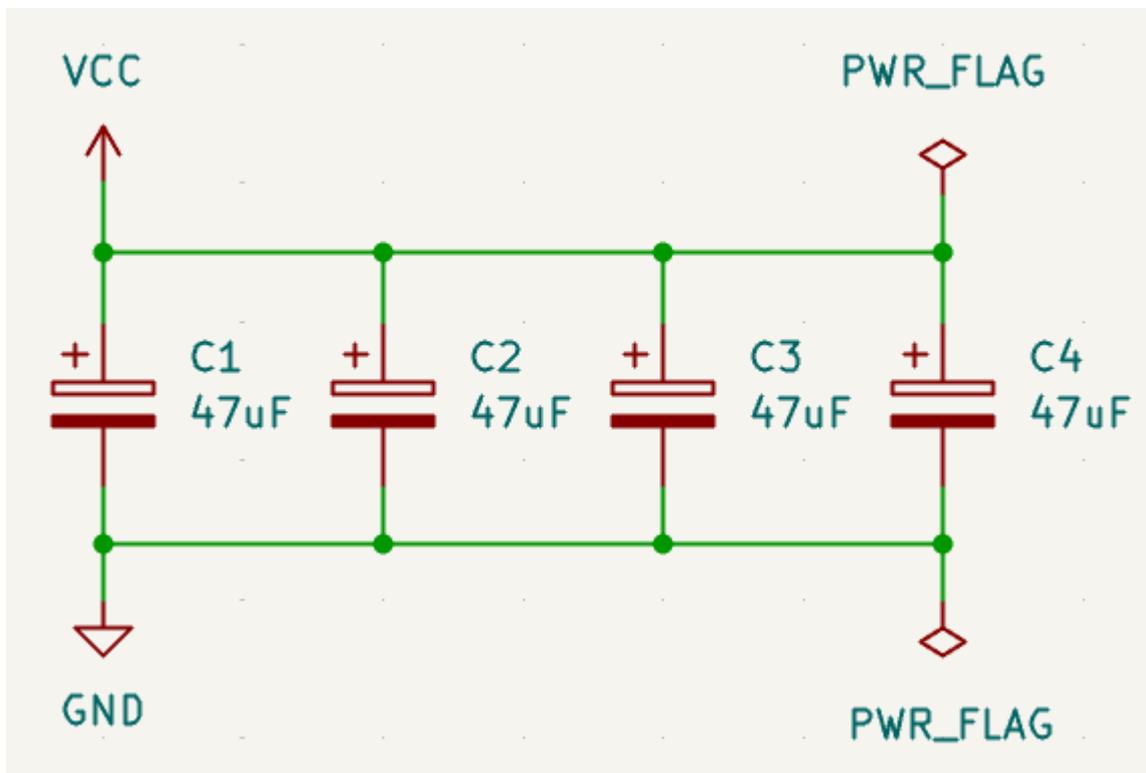
对隐藏的电源输入引脚必须小心，因为它们可能会产生无意的连接。从本质上讲，隐藏的引脚是不可见的，不显示其引脚名称。这使得很容易意外地将两个电源引脚连接到同一个网络。由于这个原因，不建议 **在电源符号之外** 使用隐形电源引脚，只支持与传统设计和符号兼容。

电源符号

电源符号是传统上用来表示与电源网连接的符号，如 `VCC` 或 GND。除了作为连接网络是电源布线的视觉指示外，电源符号还可以进行全局连接：两个具有相同引脚名称的电源符号在原理图中的任何地方都可以相互连接，而不用考虑原理图。

Power symbols are virtual: they do not represent a physical component on the PCB.

在下图中，电源符号用于将电容器的正负极分别连接到 VCC 和 GND 网络上。



In the KiCad standard library, power symbols are found in the `power` library, but power symbols can be created in any library.

To create a custom power symbol, make a new symbol with a power input pin that is set to be invisible. Name the pin according to the desired power net. In addition, set the "Define as power symbol" symbol property. As described in the [hidden power pins section](#), invisible power input pins make global connections based on the hidden power pin's name. The process of creating a power symbol is described in more detail in the [Symbol Editor section](#).

NOTE

连接的网名是由电源符号的 **引脚名称** 决定的，而不是符号的名称或值。这意味着电源符号的网名只能在符号编辑器中改变，不能在原理图中改变。

网络名分配规则

原理图中的每个网都被分配了一个名称，无论这个名称是由用户指定还是由 KiCad 自动生成。

当多个标签附加到同一个网时，最终的网络名将按以下顺序确定，从最高优先级到最低：

1. 全局标签
2. 《电源符号, 电源符号》
3. 局部标签
4. 层次标签
5. 测试原理图页码

如果一个网络有多个同一类型的标签，则按字母顺序排序，使用第一个。

如果一个网络穿过《层次原理图,层次》的多张原理图，它将从它有层次标签或局部标签的层次结构的最高层取名。通常，局部标签优先于层次标签。

如果上面的标签类型都没有附加到网络中，那么网络的名称将根据连接的符号引脚自动生成。

PWR_FLAG

在上面的截图中可以看到两个 PWR_FLAG 符号。它们向 ERC 表明，两个电源网络 VCC 和 GND 实际上连接到一个电源上，因为没有明确的电源，如连接到任何一个网络的电压调节器输出。

如果没有这两个标志，ERC 工具会诊断出：*错误：输入电源引脚没有被任何输出电源引脚驱动。*

PWR_FLAG 符号可以在 power 符号库中找到。通过将任何电源输出引脚连接到网络上，可以达到同样的效果。

无连接标志

No-connection flags (→✕) are used to indicate that a pin is intentionally unconnected. These flags prevent "unconnected pin" [ERC warnings](#) for pins that are intentionally unconnected.

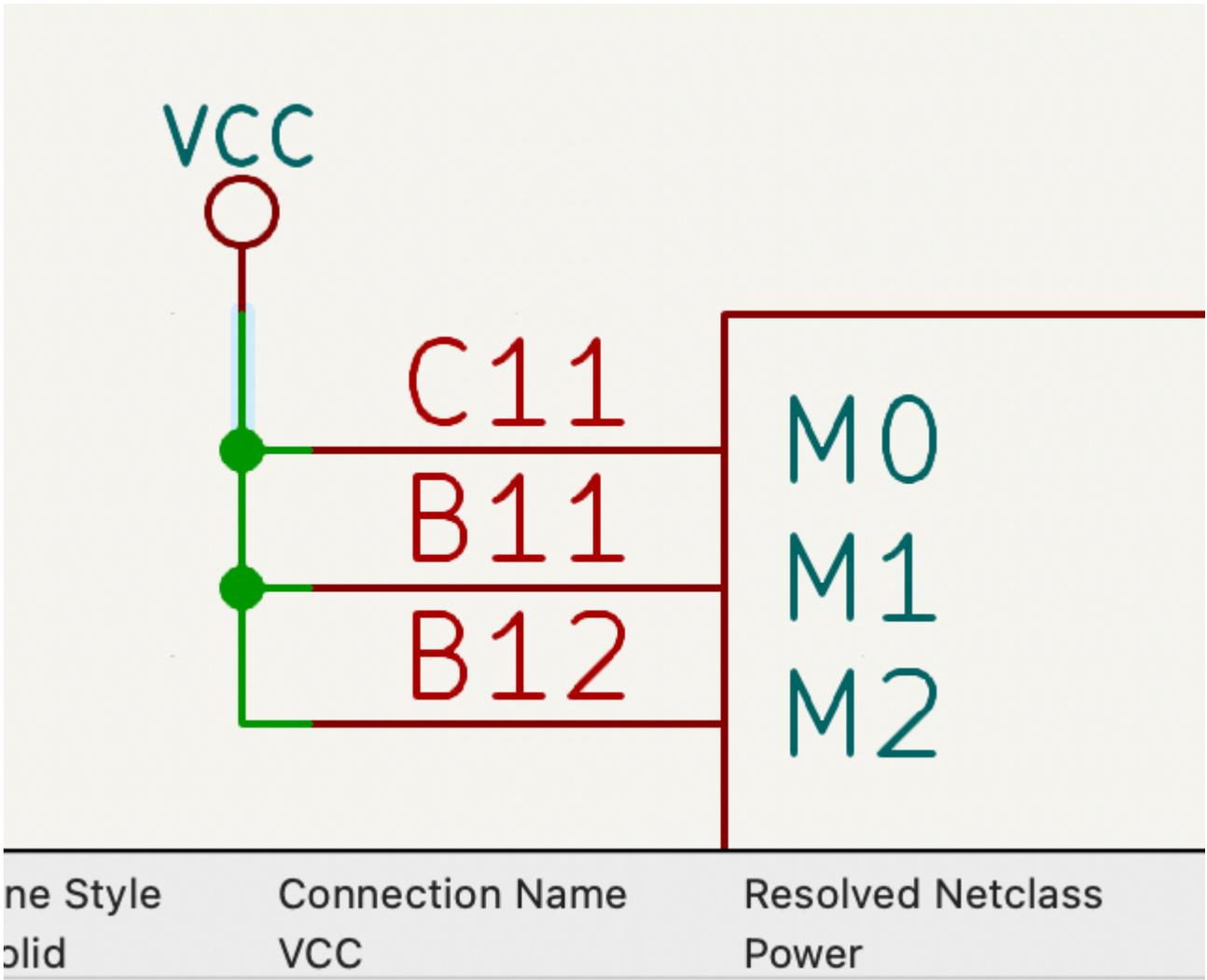
Note that no-connection flags are distinct from the ["unconnected" symbol pin type](#), although they both prevent "unconnected pin" ERC warnings on the pin in question.

网络类

网络类是可以被分配设计规则（用于 PCB）和图形属性（用于原理图）的网络类。在 KiCad 中，每个网络都是一个网络类的一部分。如果你不把一个网络添加到一个特定的类中，它将是默认类的一部分，它总是存在。

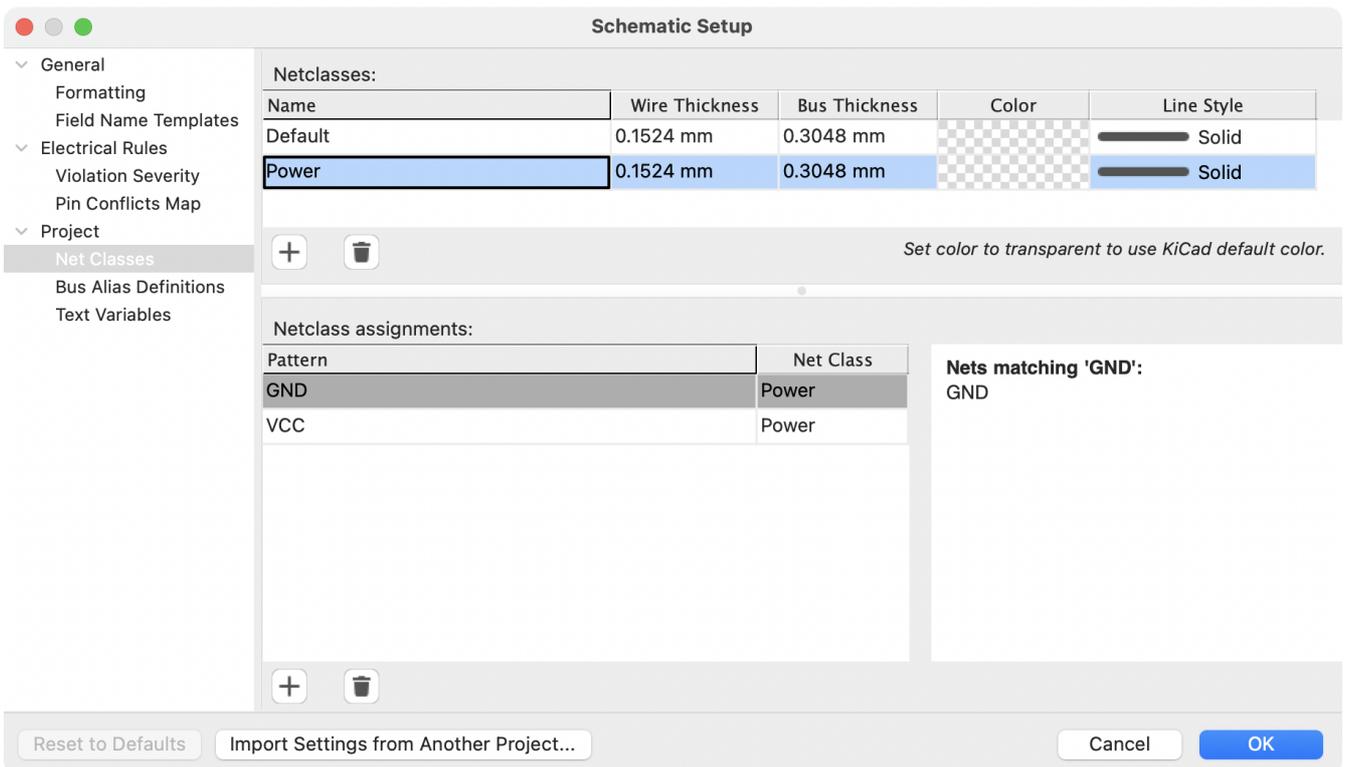
网络类可以在原理图或电路板设置对话框中创建和编辑。可以使用下面描述的基于模式的分配将网络添加到原理图或电路板的网络类中。网络也可以用网络类指令或《[标签网络类](#)，[网络标签](#)》的图形分配到原理图中的网络类。

选择一个导线或标签会在窗口底部的信息面板中显示该网络的网络类。



在原理图设置中管理网络类

网络类在 **原理图设置** 对话框的 **网络类** 面板中管理。



顶部的窗格列出了设计中存在的网络类。默认网络类总是存在的，您可以通过  按钮添加额外的网络类，或者通过  按钮删除选定的网络类。

每个网络类可以有独特的图形属性，决定该网络类的导线如何在原理图中显示。导线和母线的厚度、颜色和线条风格（实线、虚线、点线等）都可以调整。将颜色设置为透明将使用主题的默认导线/总线颜色，该颜色可在《偏好设置颜色，偏好设置》中配置。

You can also set board design rules for each netclass, although the DRC fields are hidden by default. Right click the header row to show or hide additional columns. For more information about setting netclass design rules, see the [PCB editor documentation](#).

The bottom pane lists pattern-based netclass assignments. Each row has a net name pattern and a netclass; nets with names that match the pattern are assigned to the specified netclass. If a net matches multiple patterns, the first match is used. Pattern-based netclass assignments are dynamic: when a new net is added that matches an existing pattern, it will be assigned to the associated netclass automatically. Net patterns can use both wildcards (* to match any number of any characters, including none, and ? to match any character) and [regular expressions](#). The nets that match the selected pattern are displayed to the right of the pattern list.

For example, the net* pattern matches nets named net, net1, network, and any other net name beginning with net. Because * has a slightly different meaning in a regular expression (* matches zero or more of the preceding character), the net* pattern would also match a net named ne.

NOTE

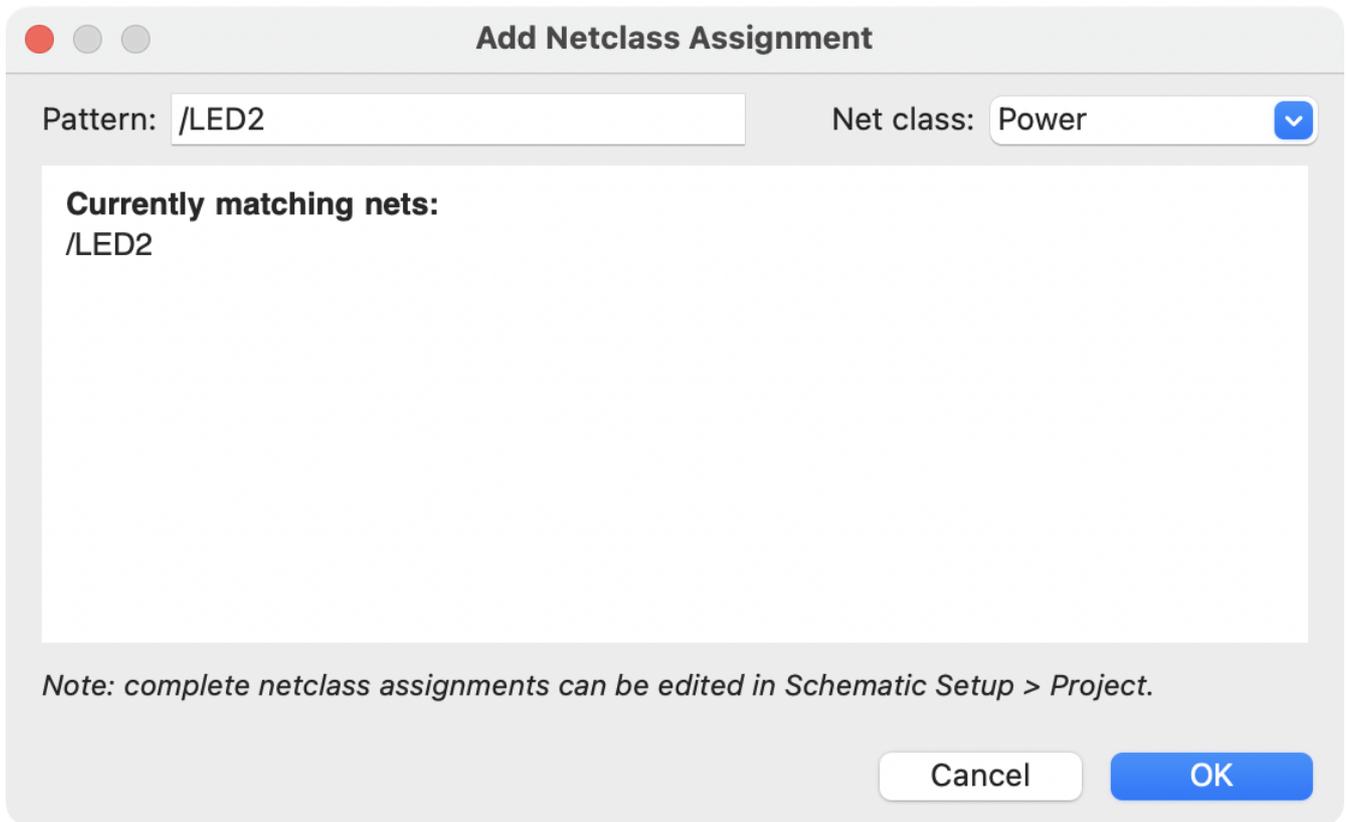
记住，网络名必须包括完整的工作表路径。例如，一个在根原理图中的本地标记的网有一个以 / 为前缀的名称。

使用  按钮来添加网络类分配模式，或  按钮来删除模式。

NOTE

A netclass pattern containing only the * wildcard will match all explicitly named nets, but will not match unlabeled nets. To match unlabeled nets, you can include more of the net name before the wildcard character. All unlabeled nets have names that begin with Net-, so the pattern Net-* will match all unlabeled nets. You can also assign a netclass to an unlabeled net using a [net class directive](#).

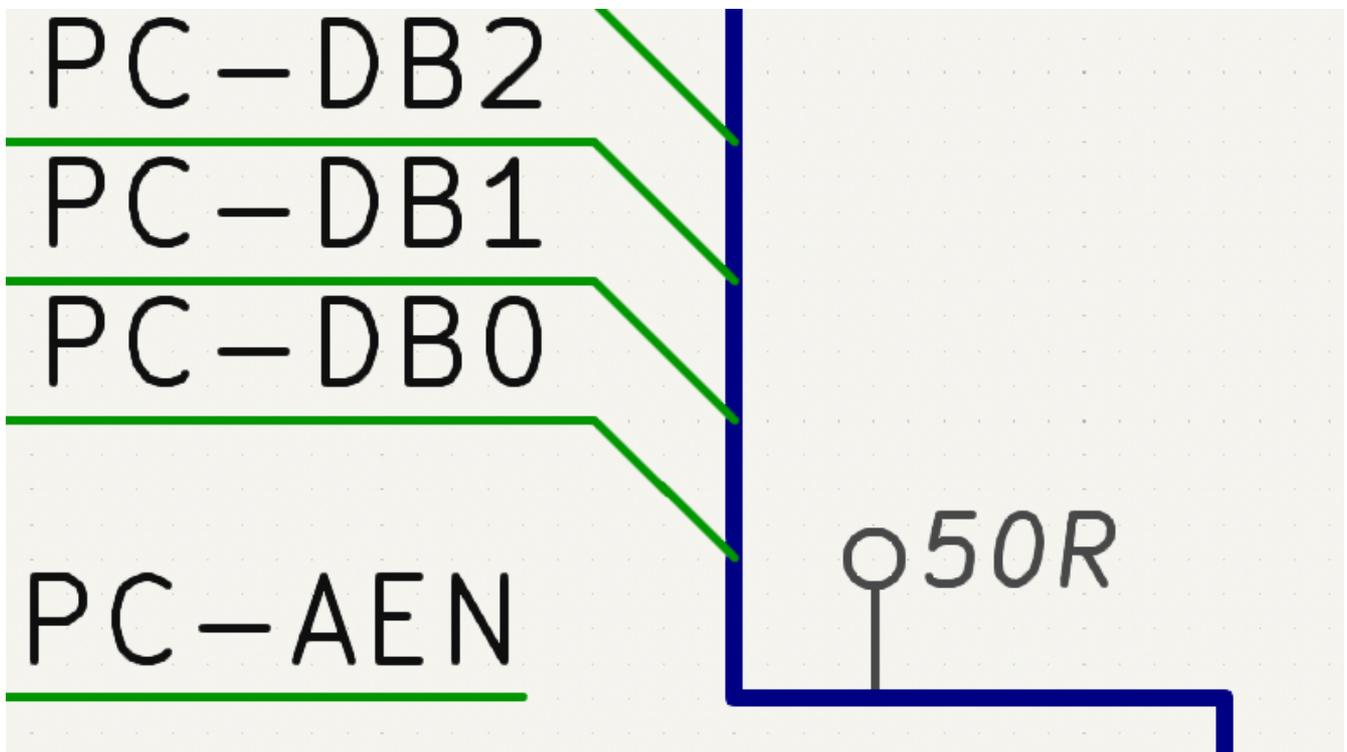
您可以从原理图画布上直接创建网络类图案，而不是在原理图设置对话框中添加网络类图案。右击一个网络，选择**分配网络类...**，就会出现**添加网络类分配**对话框。网络类模式会预先填上所选网络的名称，但如果需要，也可以改变模式。所有符合该模式的网都显示在对话框中。



在原理图中以图形方式分配网络类

作为基于模式的网络类分配的替代方法，网络类可以用 **网络类指令** 或 **标签** 以图形方式分配给原理图中的网络。网络类必须在《原理图设置网络类，原理图设置》中创建，然后才能以图形方式分配。

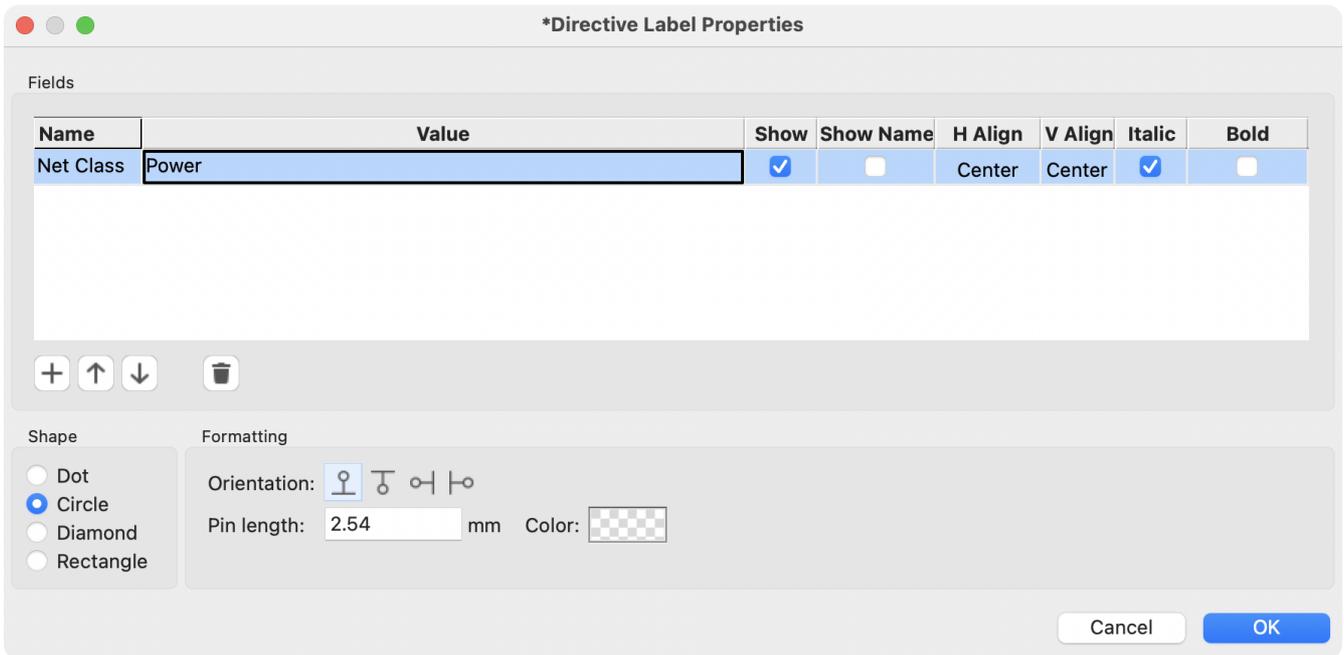
在下面的图片中，一个网络类指令被用来给 50R 网络类分配信号。



网络类指令是通过右边工具栏的 **QA** 按钮添加的。它们的行为类似于《标签，标签》，除了它们不能被用来命名一个网络。根据指令的 **网络类** 字段的值，附加的网络被分配了一个网络类。`网络类` 字段提供了一个设计中所有网络

类的下拉列表。

如果一个指令被附加到一个总线上，总线上的所有成员都被分配到指定的网络类。



除了相关的网络类，你还可以在指令的属性中编辑指令的**形状**（点、圆、钻石或矩形）、**方向**、**引脚长度**和**颜色**。

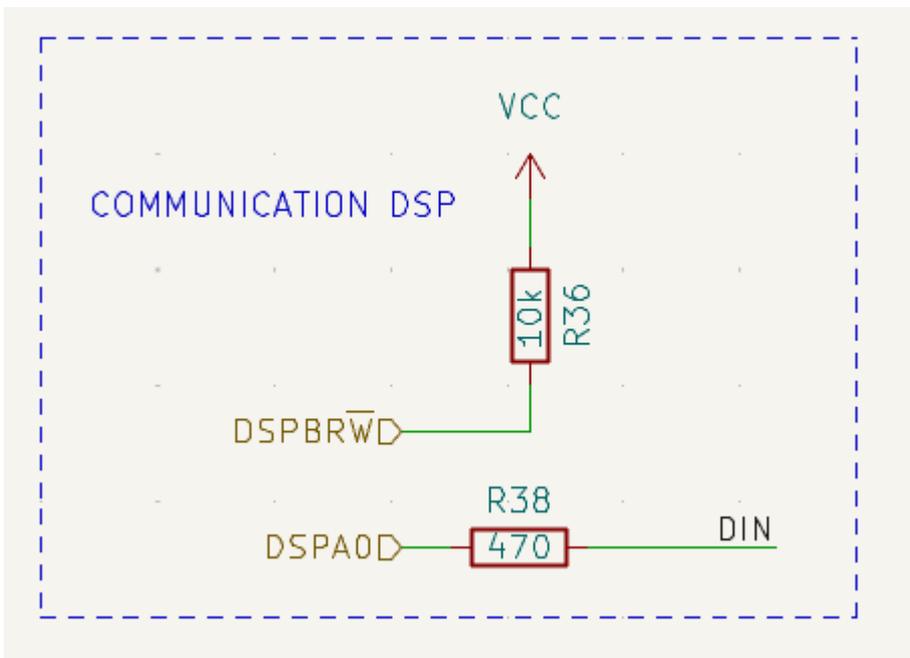
《标签网络类，网络标签也可以用来给网络分配网络类》，方法是在标签中添加一个 网络类 字段。

如果超过一个不同的网络类被图形化地分配给一个网络，《ERC 检查清单，ERC 将报告一个问题》。图形化的网络类分配优先于基于模式的分配：如果一个网络与网络类模式分配相匹配，并且也有一个图形化的网络类分配，那么将使用图形化的网络类分配。

图形项目

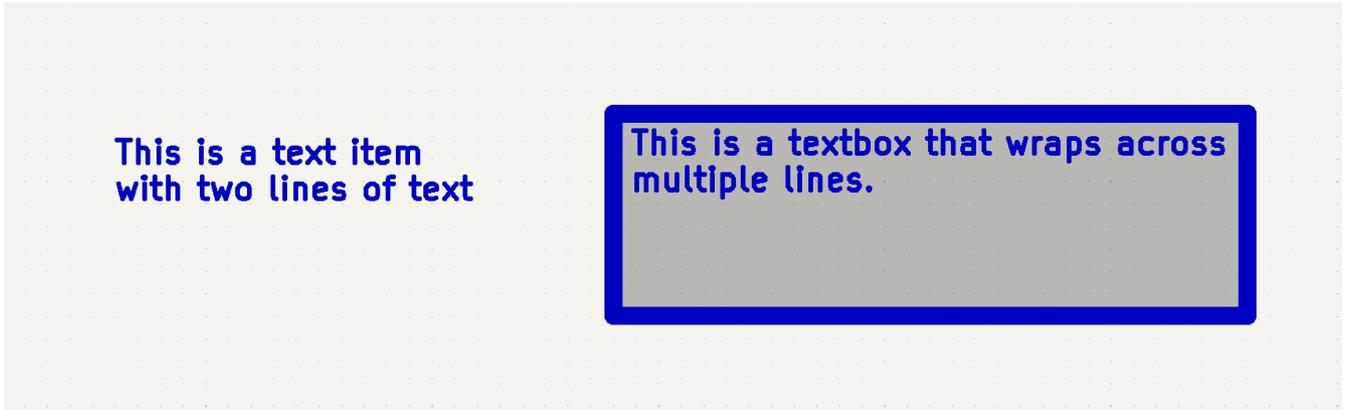
文字、图形和图像可以被添加到原理图中，用于记录。这些项目对原理图没有任何电气影响。

下面的图片除了符号和几种类型的标签外，还显示了图形线和文字（"COMMUNICATION DSP"）。



文本和文本框

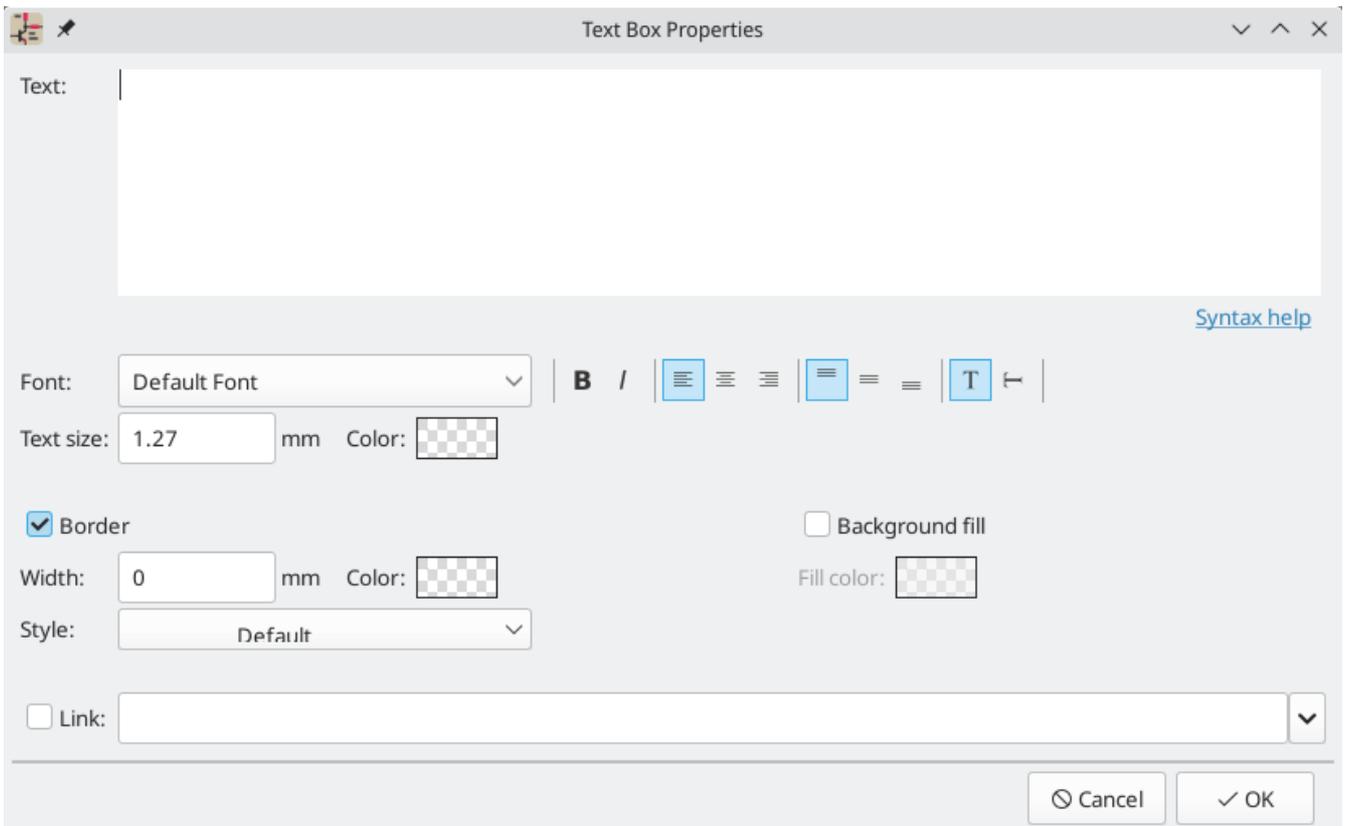
有两种文本可以被添加到原理图中，它们被称为文本（**T**）和文本框（**☐**）。两者都是使用右侧工具栏中各自的按钮添加的。



这两种文本项目都支持多行文本和基本的格式化功能，但文本框将文本包起来以适应大纲，并有额外的格式化选项。所有的文本都有可调整的字体、颜色、大小、黑体和斜体强调，左右对齐，以及垂直和水平方向。文本框还支持水平居中，垂直对齐选项，以及彩色边框和填充。

NOTE

The default text size can be set for a schematic in [Schematic Setup](#), and the default font can be set in [Preferences](#).



链接

文本和文本框可以通过在文本属性的 **链接** 框中输入一个目标而成为一个链接。链接目标可以是一个本地文件（使用 `file://` 协议前缀，后跟文件路径），到一个网站（使用 `http://` 或 `https://`，后跟其余的 URL），或到同一图示中的另一个页面（使用 `#` 后跟页码）。这些也可以使用链接目标框中的下拉菜单自动填写。

字体

文本和文本框支持自定义字体，可通过文本的属性对话框中的 **字体** 下拉选择。除了 KiCad 字体外，你还可以使用安装在你电脑上的任何 TTF 字体。

NOTE

用户字体不会被嵌入工程中。如果该工程在另一台没有安装所选字体的计算机上打开，将会有不同的字体被替换。为了获得最大的兼容性，请使用 KiCad 字体。

文本标记

Text supports markup for superscripts, subscripts, overbars, evaluating project variables, and accessing symbol field values.

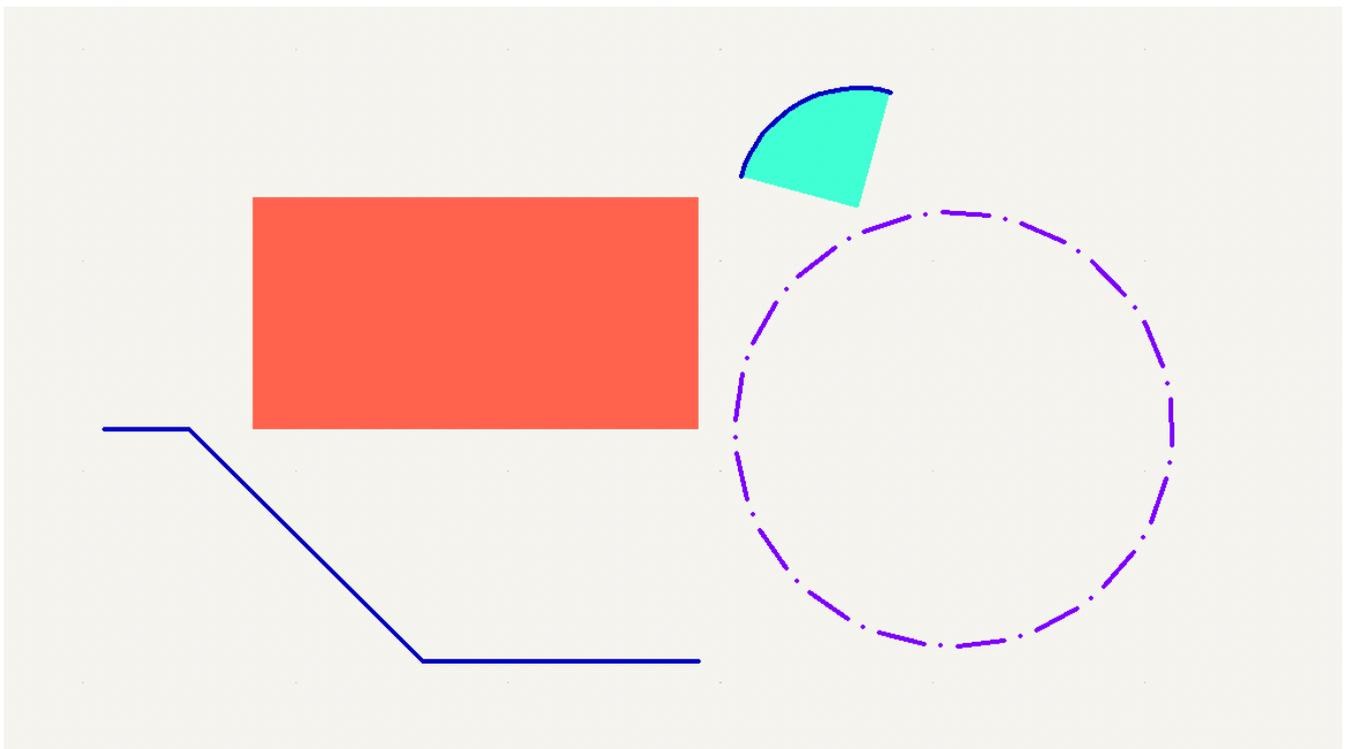
Feature	Markup Syntax	Result
Superscript	text ^{superscript}	text ^{superscript}
Subscript	text _{subscript}	text _{subscript}
Overbar	~{text}	<u>text</u>
Variables	\${variable}	variable_value
Symbol Fields	\${refdes:field}	field_value of symbol refdes

NOTE

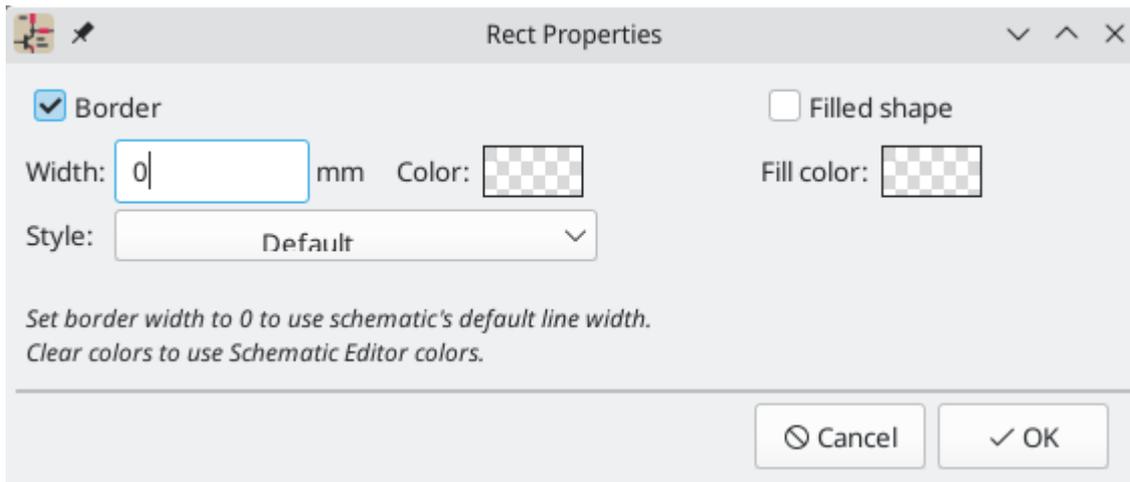
Variables must be defined in [Schematic Setup](#) before they can be used. There are also a number of [built-in system text variables](#).

图形形状

图形矩形 ()、圆形 ()、弧形 () 和直线 () 都可以使用右侧工具栏中各自的按钮添加。



线条的宽度、颜色和样式（实线、虚线或点线）可以在每个形状（E）的属性对话框中进行配置。矩形、圆形和弧线也可以设置填充颜色并去除其轮廓。



Setting a shape's line width to 0 uses the schematic default line width, which is configurable in [Schematic Setup](#). Spacing for line dashes is also configurable there. Removing a line or fill color uses the color theme's graphics color, which is configurable in [Preferences](#).

像《画和编辑导线，导线》一样，图形线服从线的绘制模式设置（90度，45度，或自由角度），你可以用左边工具栏上的切换按钮来设置（90度线图标，，和，分别）。`Shift + Space` 循环浏览这些模式。

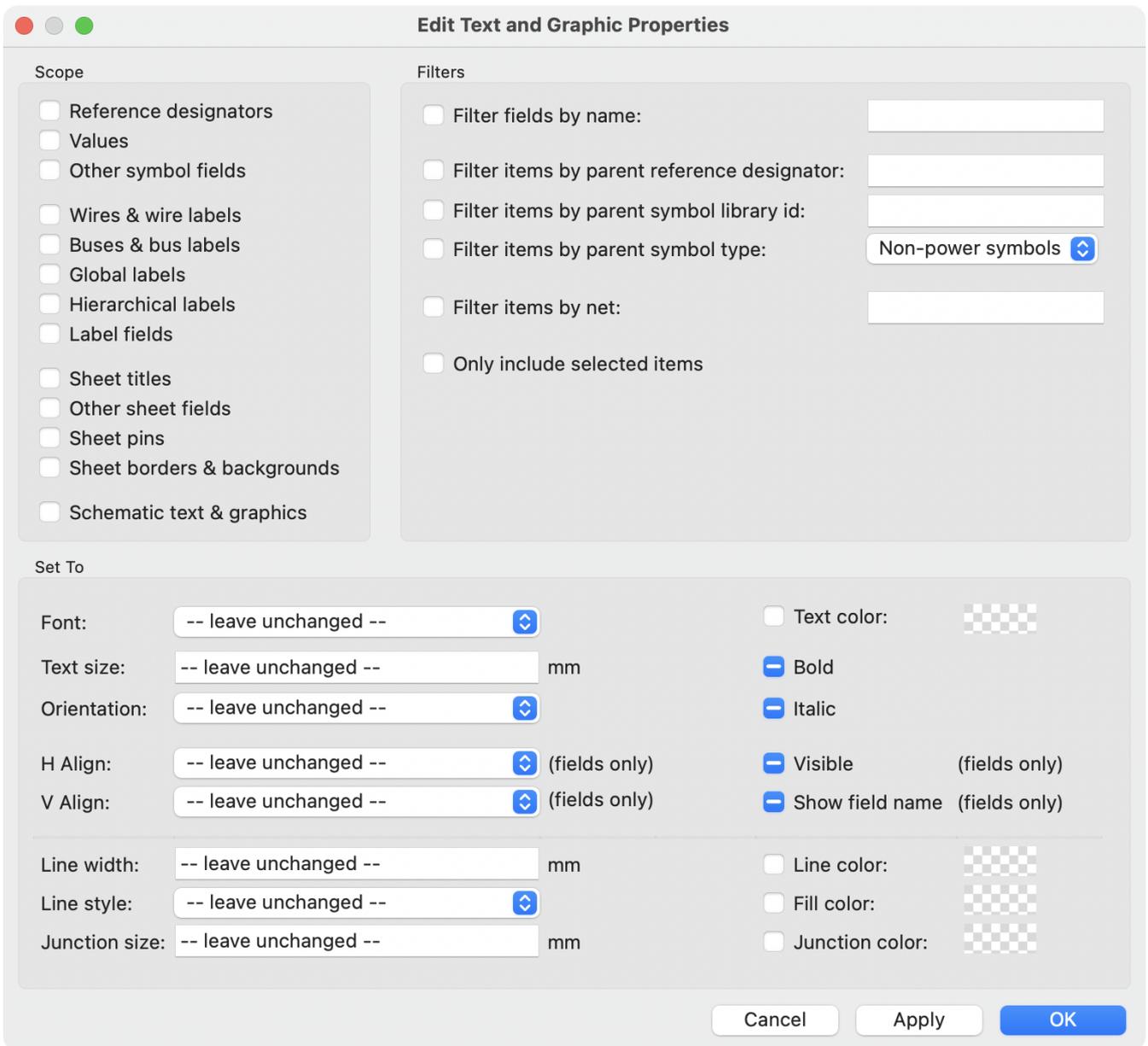
As with PCB tracks, the `/` hotkey switches line posture.

位图图像

位图图像可以通过  按钮添加到原理图中。原理图中的图像可以被移动和缩放。属性对话框允许设置位置和比例，以及将图像转换为灰度。

批量编辑文本和图形

文本和图形的属性可以通过 [编辑文本和图形属性](#) 对话框（工具 → [编辑文本和图形属性...](#)）进行批量编辑。该工具还可以修改导线和总线的视觉属性。



范围和筛选器

范围 设置限制了该工具只能编辑某些类型的对象。如果没有选择范围，就不会有任何东西被编辑。

Filters restrict the tool to editing particular objects in the selected scope. Objects will only be modified if they match all enabled and relevant filters (some filters do not apply to certain types of objects. For example, symbol field filters do not apply to wires and are ignored for the purpose of changing wire properties). If no filters are enabled, all objects in the selected scope will be modified. For filters with a text box, wildcards are supported: * matches any number of any characters, including none, and ? matches any single character.

按名称过滤字段 过滤到指定的符号、标签或页面字段。

按父级位号筛选项目 过滤到具有指定位号的符号中的字段。**通过父级符号库 ID** 筛选到具有指定库位号的符号中的字段。***通过父级符号类型*** 筛选到选定类型（电源或非电源）的符号中的字段。

通过网络筛选项目 筛选到指定网络上的导线和标签。

只包括选定的项目 筛选器到当前选择。

可编辑的属性

被筛选对象的属性可以在对话框的底部设置为新值。

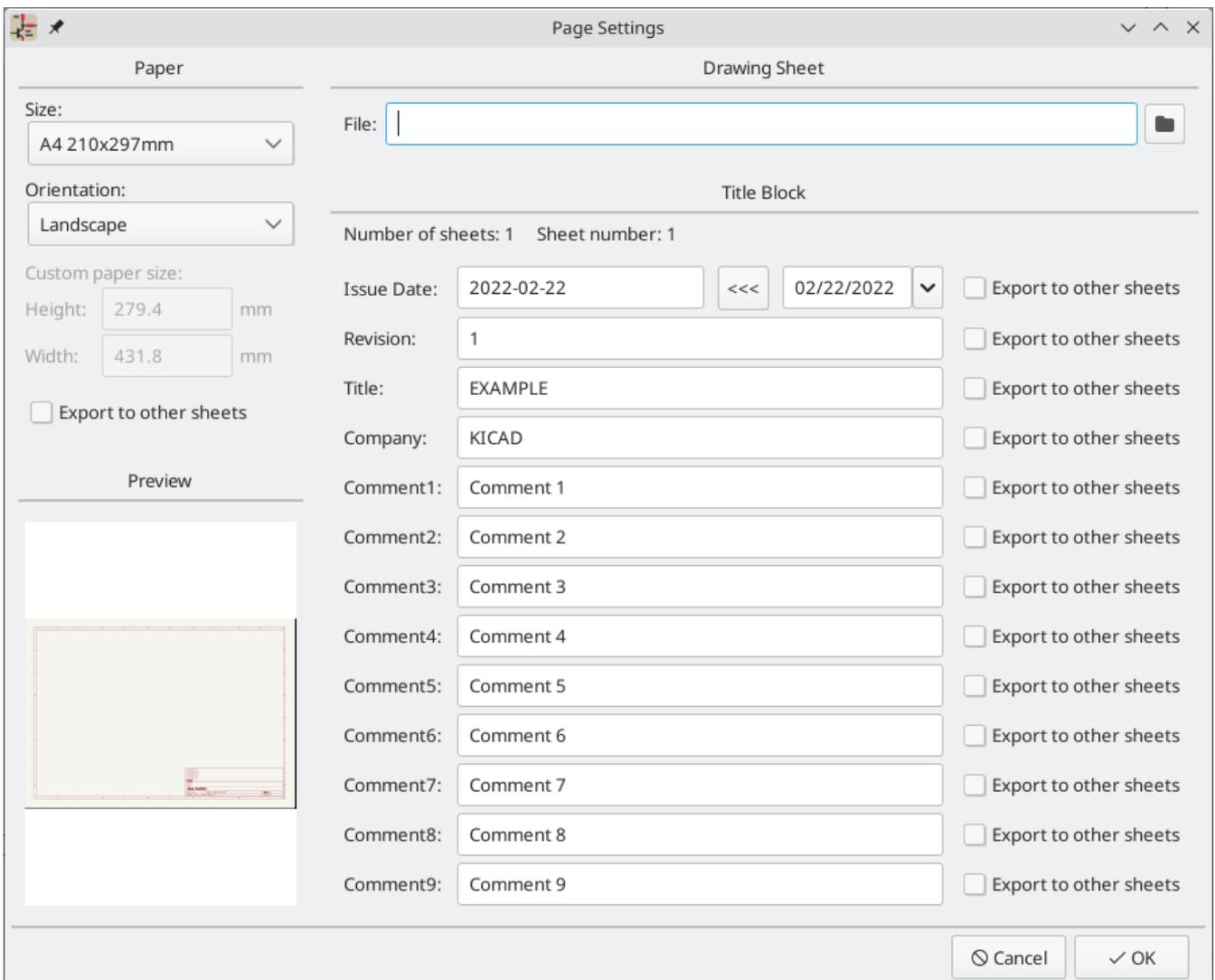
下拉列表和文本框可以被设置为 --保持不变-- 以保留现有值。复选框可以被选中或不被选中来启用或禁用一个变化，但也可以切换到第三个“保持不变”的状态。颜色属性必须被选中以改变值；棋盘式的色块表示颜色将从原理图设置或网络类属性的默认值中继承下来。

可以修改的文本属性有 **字体**、**文本大小**、**文本方向**（右/上/左/下）、**水平**和**垂直对齐**、**文本颜色**、**强调**（**粗体***和***大写**）以及字段和字段名的**可见度**。

可以修改的图形和导线属性有：**线宽**、**线样式**（实线、虚线和点线）、**线颜色**、图形的**填充颜色**，以及导线结点的**结点尺寸**和**结点颜色**。

原理图标题块

标题块是用页面设置工具（）编辑的。

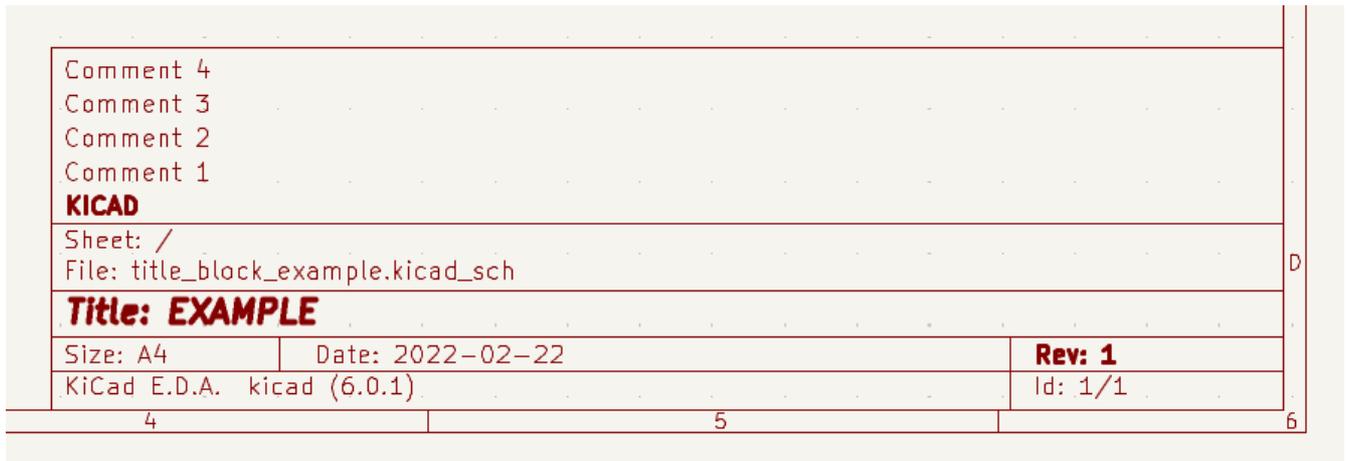


The screenshot shows the 'Page Settings' dialog box. It is divided into two main sections: 'Paper' and 'Drawing Sheet'.
The 'Paper' section includes:
- Size: A4 210x297mm (dropdown)
- Orientation: Landscape (dropdown)
- Custom paper size: Height: 279.4 mm, Width: 431.8 mm (input fields)
- An 'Export to other sheets' checkbox.
The 'Drawing Sheet' section includes:
- A 'File' input field.
- 'Number of sheets: 1' and 'Sheet number: 1'.
- A 'Title Block' section with the following fields:
 - Issue Date: 2022-02-22 (with a calendar icon and a dropdown showing 02/22/2022)
 - Revision: 1
 - Title: EXAMPLE
 - Company: KICAD
 - Comment1: Comment 1
 - Comment2: Comment 2
 - Comment3: Comment 3
 - Comment4: Comment 4
 - Comment5: Comment 5
 - Comment6: Comment 6
 - Comment7: Comment 7
 - Comment8: Comment 8
 - Comment9: Comment 9
 - Each field has an 'Export to other sheets' checkbox.
At the bottom right, there are 'Cancel' and 'OK' buttons.

Each field in the title block can be edited, as well as the paper size and orientation. If the **Export to other sheets** option is checked for a field, that field will be updated in the title block of all sheets, rather than only the current sheet.

You can set the date to today's or any other date by pressing the left arrow button next to **Issue Date**. Note that the date in the schematic will not be automatically updated.

也可以选择一个原理图模板文件。



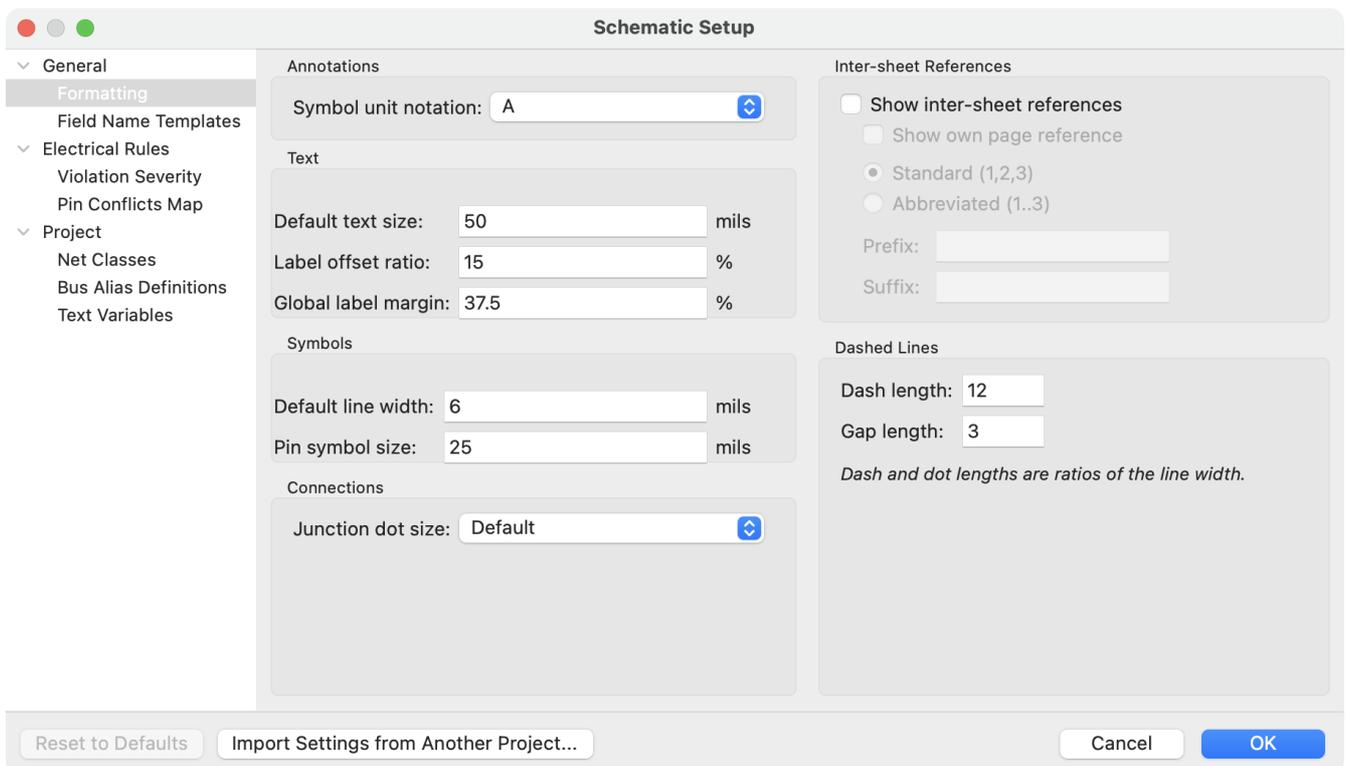
原理图编号 (X/Y纸) 会自动更新, 但原理图页码也可以用 **编辑** → **编辑原理图页码...** 手动设置。

原理图设置

原理图设置窗口用于设置特定于当前活动原理图的原理图选项。例如, 原理图设置窗口包含格式化选项、电气规则配置、网络类设置和原理图文本变量设置。

You can import schematic settings from an existing project using the **Import Settings from Another Project...** button. This allows you to choose a project to use as a template and select which settings to import (formatting preferences, field name templates, pin conflict map, violation severities, and net classes).

Schematic formatting



The formatting panel contains settings for the appearance of symbols, text, labels, graphics, and wires.

Symbol unit notation sets how each unit of a multi-unit symbol is referred to in its reference designator. By default, a different letter for each unit is appended to the reference designator with no separator, for example U1B for the second unit of symbol U1, but this can be changed. Numbers can be used instead of letters, and various separators can be used between the symbol designator and the unit identifier (. , - , _ , or none).

Default text size sets the default text height used by the text, text box, and label tools. **Label offset ratio** controls the vertical spacing between a local label's text and the attached wire, relative to the label's text size. This also affects the spacing between symbol pins and their pin number. **Global label margin** defines the size of the box around a global label, relative to the global label's text size. Increasing the margin may be useful to avoid overlapping text with overbars (~{ }) or letters with descenders, but this may cause closely packed global labels to overlap with each other.

Default line width sets the default line width for symbol graphics, if the symbol does not override the default line width. **Pin symbol size** scales symbol pin graphic style annotations, such as the bubble on an inverted pin.

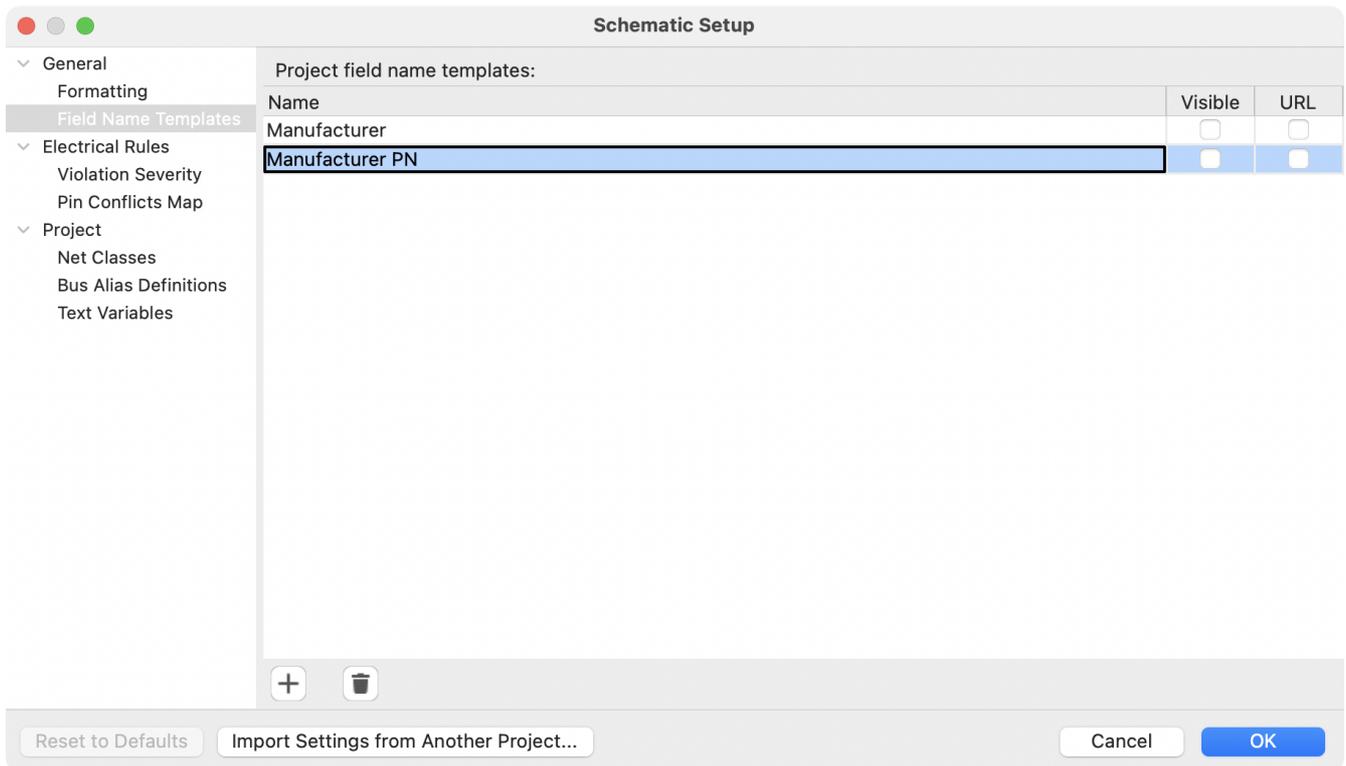
Junction dot size sets the schematic's default wire junction dot size. The default size can be overridden by editing an individual junction dot's properties.

Show inter-sheet references enables or disables the display of [inter-sheet references](#), which are a list of page numbers next to a global labels that link to other places in the schematic where the same global label appears. **Show own page reference** controls whether the current page is included in the list of page numbers. **Standard** and **abbreviated** determine whether to display the complete list of page numbers or only the first and last page numbers. The **prefix** and **suffix** fields add optional characters before and after the list of page numbers. In the image of an inter-sheet reference below, a prefix and suffix of [and], respectively, have been added.



Dashed line appearance is controlled in the Formatting section. **Dash length** controls the length of dashes, while **Gap length** controls the spacing between dashes and dots. The dash and gap lengths are relative to the line width: a gap length of 2 means twice the width of the line.

Field name templates



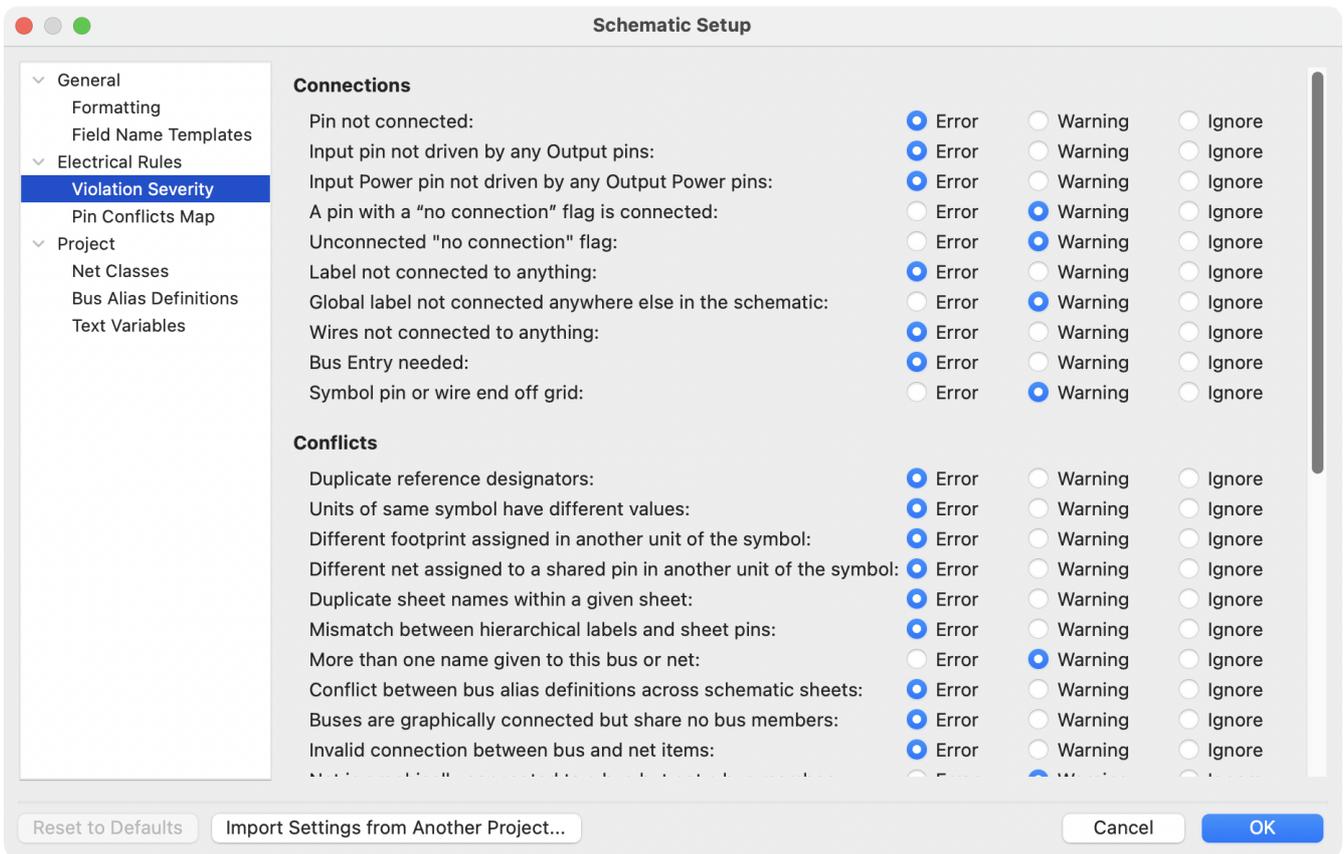
Field name templates are empty symbol fields that are automatically added to all symbols in the schematic. These can be useful when every symbol in the schematic needs additional fields beyond the fields that are defined in the library symbols, for example a field for the manufacturer's part number.

Template fields can be set as visible or invisible, and can also be set as URL fields.

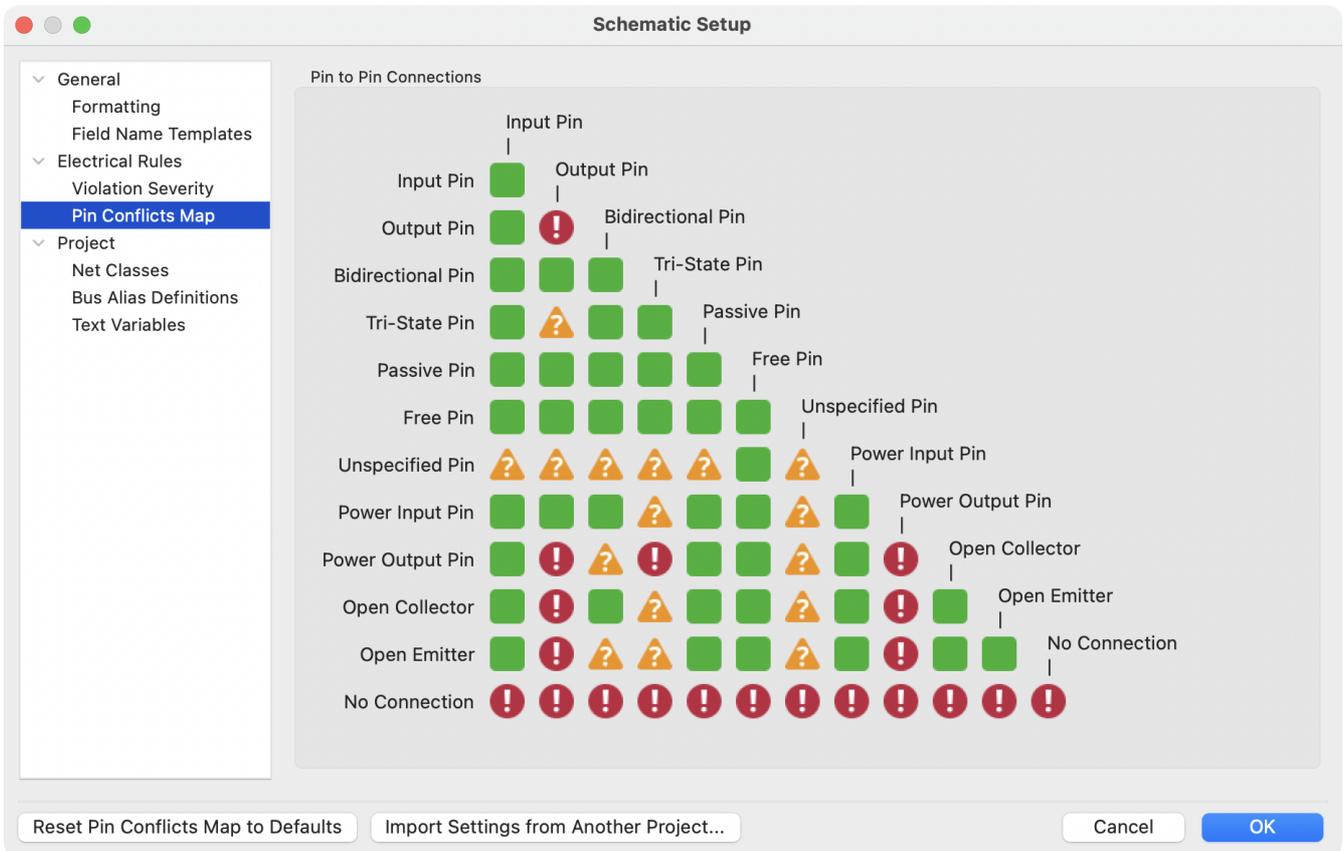
Field name templates that are defined in schematic setup apply only to the current project. Field name templates can also be defined in [Preferences](#), which apply to all projects edited on your computer.

ERC violation severity and pin conflicts map

The **Violation Severity** panel lets you configure what types of ERC messages should be reported as Errors, Warnings, or ignored.

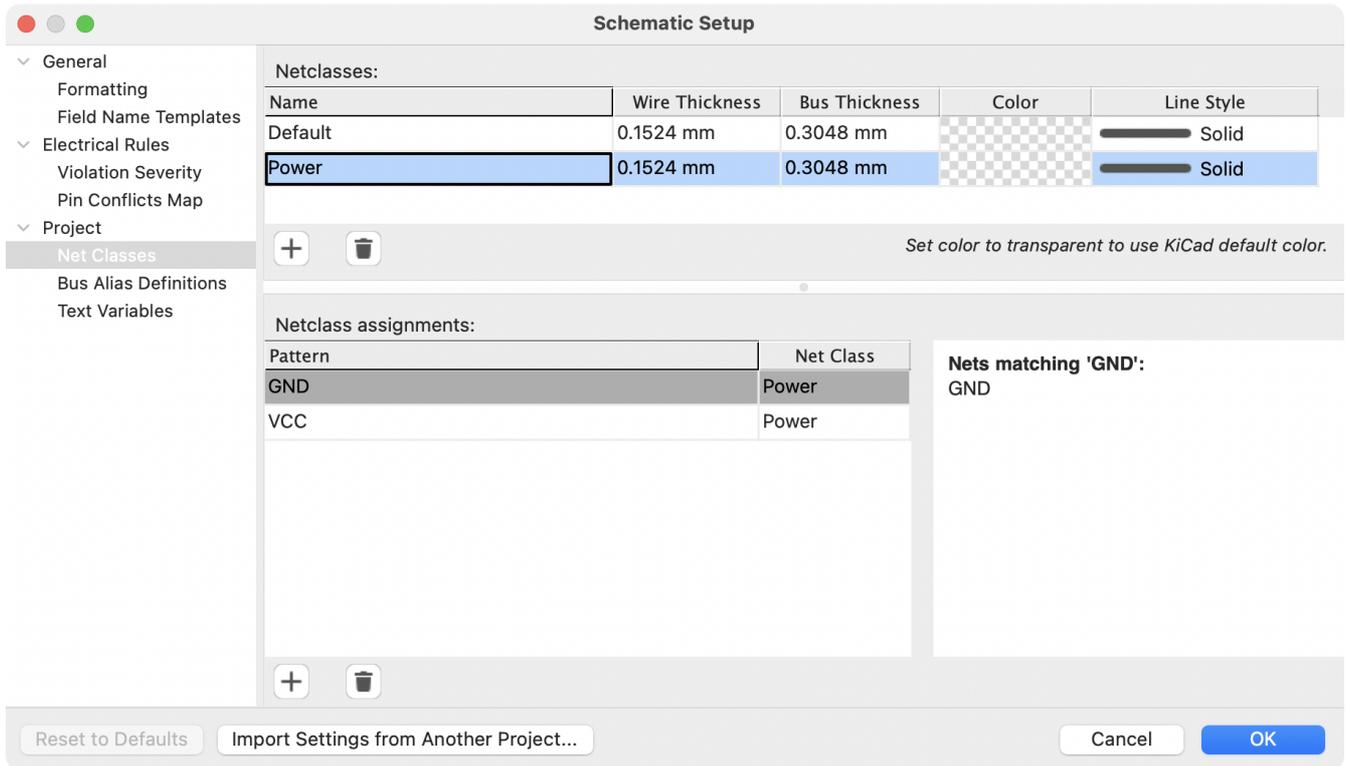


The **Pin Conflicts Map** allows you to configure connectivity rules to define electrical conditions for errors and warnings based on what types of pins are connected to each other. For example, by default an error is produced when an output pin is connected to another output pin.



These panels are explained in more detail in the [ERC section](#).

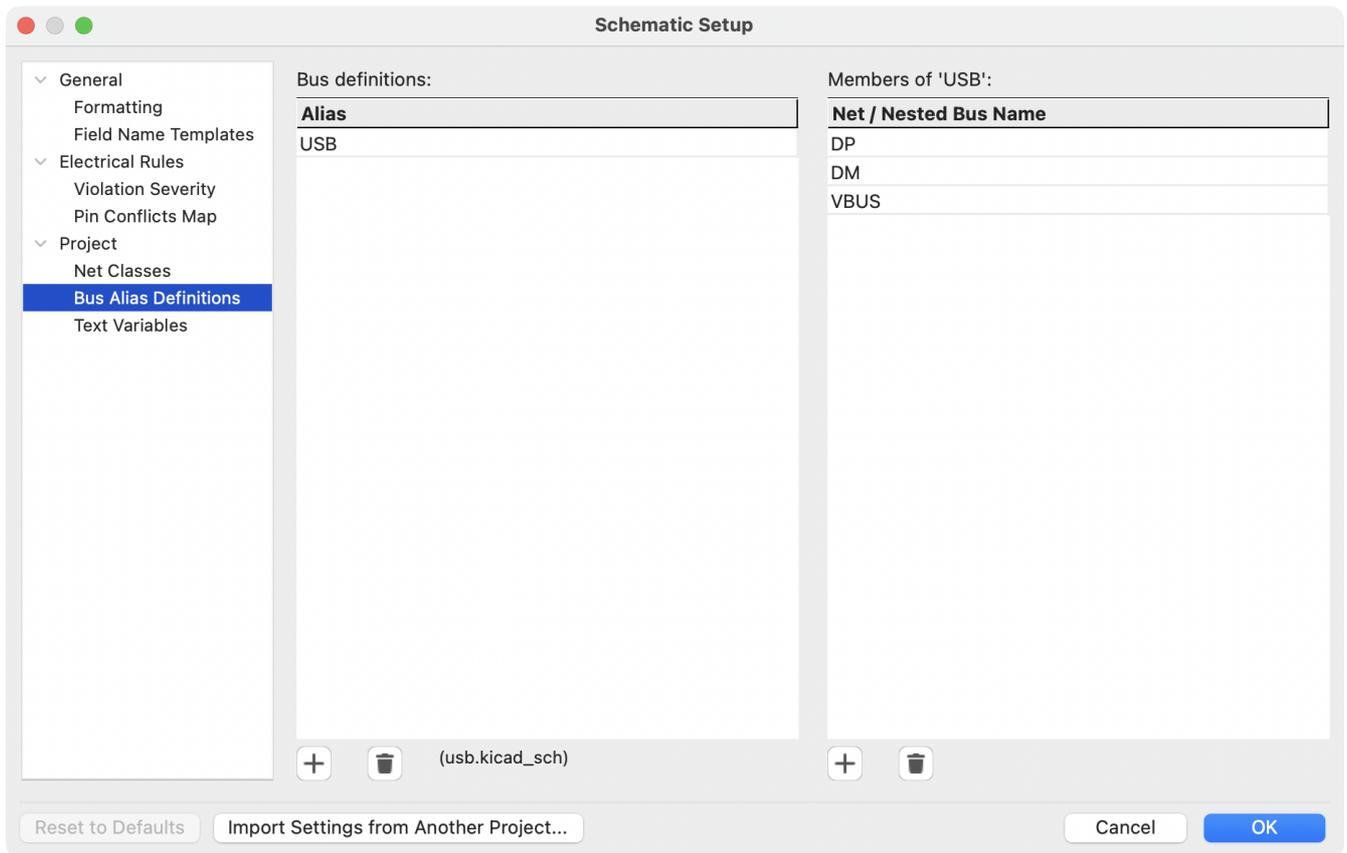
Net classes



The **Net Classes** panel allows you to manage netclasses for the project and assign nets to netclasses with patterns. Managing netclasses in this panel is equivalent to managing them in the [Board Setup dialog](#). Nets can also be assigned to netclasses in the schematic using graphical assignments with [net class directives](#) or [net labels](#).

Pattern-based netclass assignment is explained in more detail in the [net classes section](#).

Bus alias definitions



The **Bus Alias Definitions** panel allows you to create bus aliases, which are names for groups of signals in a bus. For more information about bus aliases, see the [bus alias documentation](#).

文本变量



Text replacement variables can be created in the Text Variables section. These variables allow you to substitute the variable name for any text string. This substitution happens anywhere the variable name is used inside the variable replacement syntax of `${VARIABLENAME}`.

For example, you could create a variable named `VERSION` and set the text substitution to `1.0`. Now, in any text object on the PCB, you can enter `${VERSION}` and KiCad will substitute `1.0`. If you change the substitution to `2.0`, every text object that includes `${VERSION}` will be updated automatically. You can also mix regular text and variables. For example, you can create a text object with the text `Version: ${VERSION}` which will be substituted as `Version: 1.0`.

Text variables can also be created in [Board Setup](#). Text variables are project-wide; variables created in the schematic editor are also available in the board editor, and vice versa.

There are also a number of [built-in system text variables](#).

Opening legacy schematics

Modern versions of KiCad can always open projects created in older versions of KiCad. However, schematics created in some older versions of KiCad have special considerations that must be observed when opening them in order to prevent any data loss.

Opening KiCad 5.0 and 5.1 schematics

Modern versions of KiCad can open schematics created in versions prior to KiCad 6.0, but the cache library file (`<projectname>-cache.lib`) must be present to load the schematic correctly.

Since version 6.0, KiCad stores all symbols used in a project in the schematic. This means that you can open a schematic made in KiCad 6.0 or later on any computer, even if the libraries used in the project are not installed or have changed. Modern KiCad schematic files use the `.kicad_sch` extension.

Prior to version 6.0, KiCad did not store symbols in the schematic. Instead, KiCad stored references to the symbols and their libraries. It also stored a copy of every symbol used by the project in a separate cache library file (`<projectname>-cache.lib`). As long as the cache library was included with the project, the project could be distributed without the system library files, because KiCad could load any needed symbols from the cache library as a fallback if the libraries referenced in the schematic were missing. Legacy KiCad schematic files use the `.sch` extension.

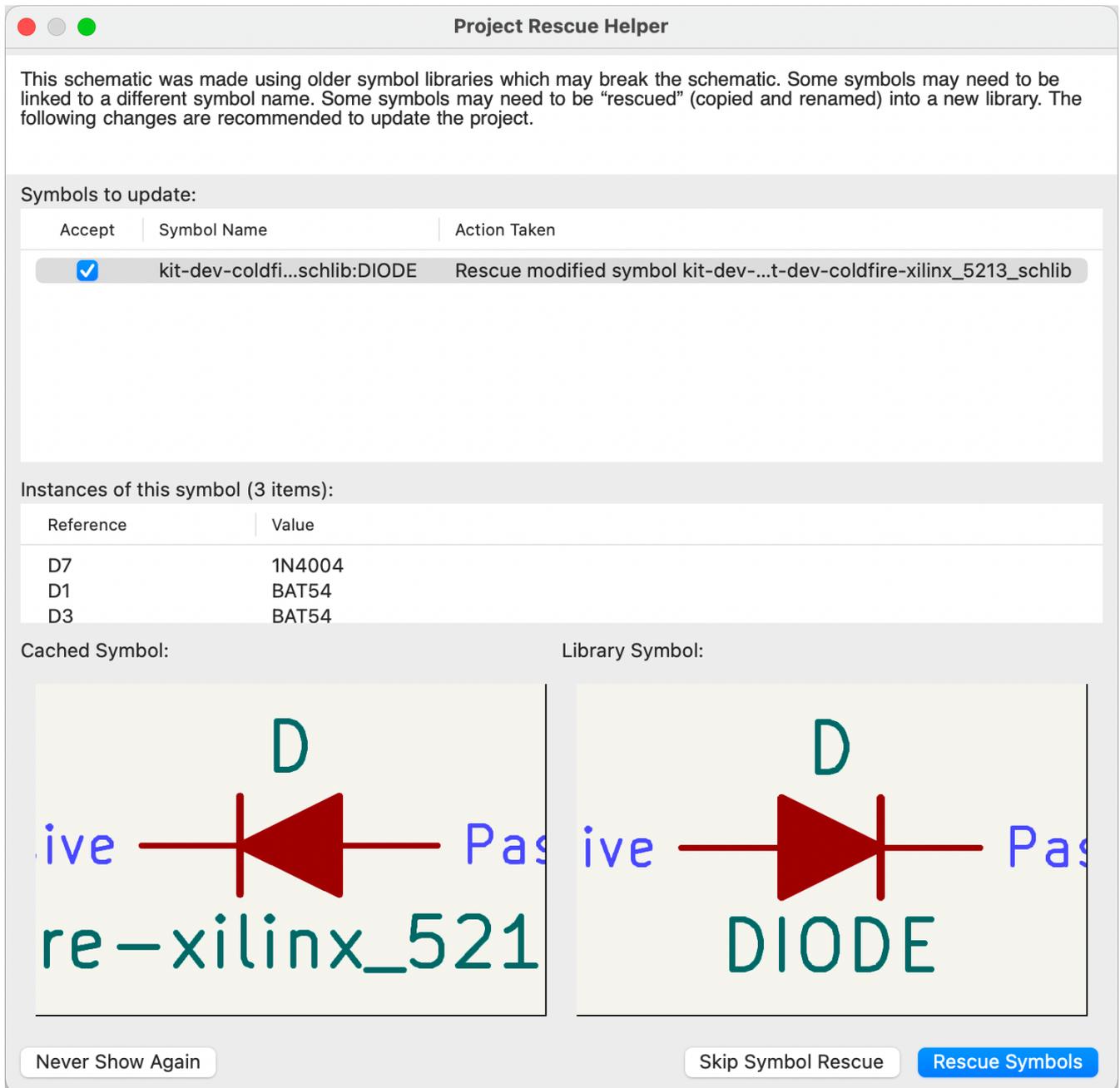
When you open a legacy schematic, KiCad will look in the cache library to find all of the symbols used in the schematic in the cache library. When you save the legacy schematic, KiCad will save it as a new file in the modern schematic format (`.kicad_sch`), with the necessary symbols embedded in the schematic itself. The original legacy schematic and the cache library will remain, unmodified, but they are no longer necessary once the schematic has been saved in the modern format.

NOTE

Projects created in KiCad prior to version 6.0 must have a cache library. If the cache library is missing, the schematic will lose symbol information if the system symbol libraries are modified, reorganized, moved, or deleted. The libraries included with legacy versions of KiCad are substantially different than the modern KiCad libraries, so in practice KiCad will almost always fail to open legacy projects unless the cache library is present.

When you open a legacy schematic, KiCad may display the **Project Rescue Helper** dialog. This means that one or more symbols in the cache library do not match the corresponding symbol in the external library. The

dialog helps you "rescue" symbols from the cache library into your schematic, if desired. You can also open the rescue dialog at any time using **Tools** → **Rescue Symbols....** The cache library file must be present in order to use the rescue tool.



The rescue dialog lists all symbols that don't match between the cache library and the external symbol library. The discrepancy can be because:

- the cached symbol or the library symbol has been modified, so the two symbols no longer match, or
- the cached symbol does not have a corresponding symbol in the symbol library, because the symbol or library was moved, renamed, deleted, or is not present on the current computer.

For each symbol in the list, selecting the symbol displays the reference designator and value for each instance of the symbol, and shows a visual preview of the symbol. If a corresponding symbol exists in the system symbol library, the dialog shows both copies of the symbol for comparison. If the symbol only exists in the cache library, the dialog only shows the cached symbol.

In this example, the project originally used a diode with the cathode facing left, but the library now contains one with the cathode facing right. This change would break the design, so it would be important to use the cached symbol as the original designer intended.

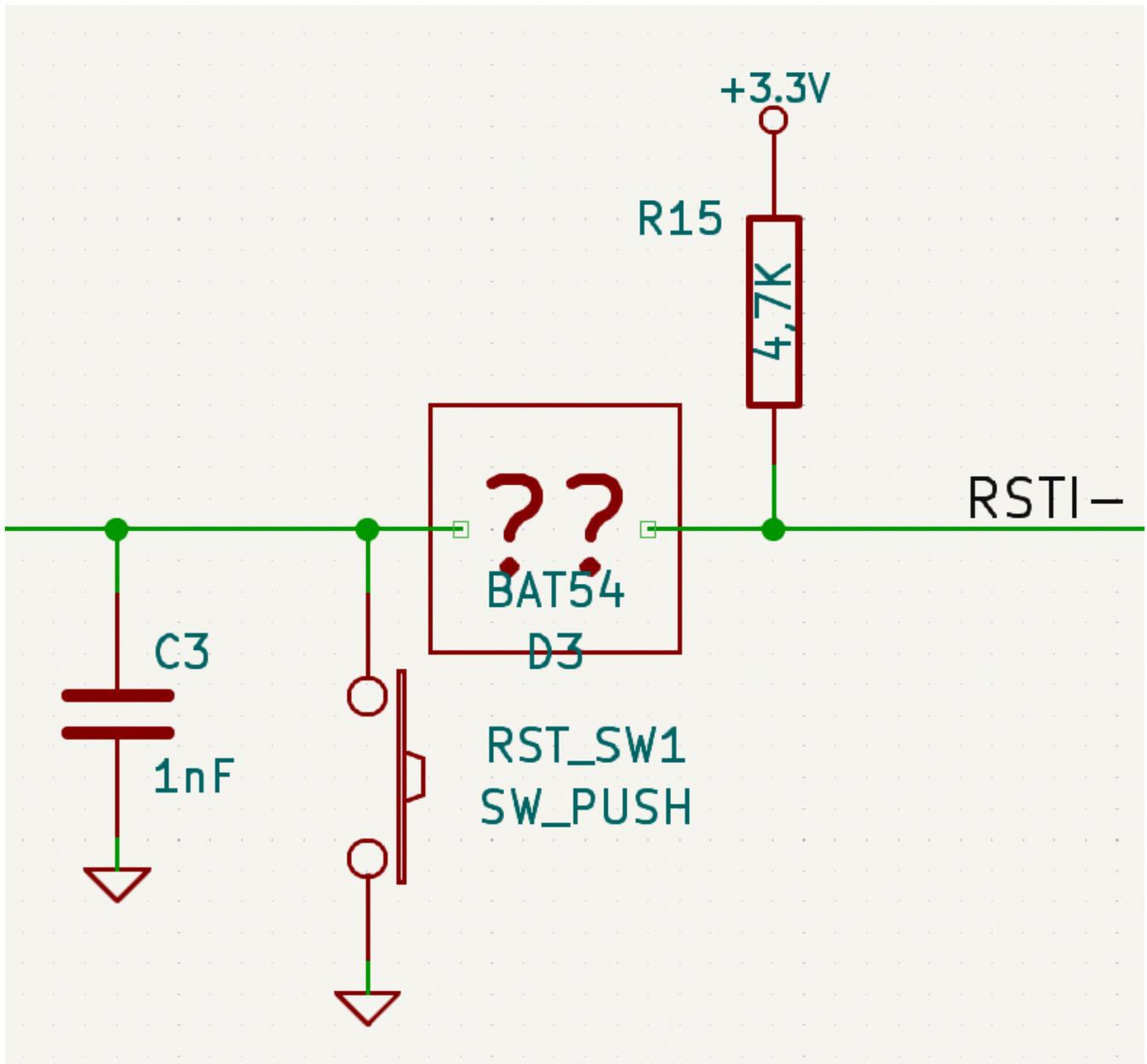
Pressing **Rescue Symbols** here will cause the selected symbols from the cache library to be saved into a special rescue library (`<projectname>-rescue.kicad_sym`). The corresponding symbols in the schematic will be updated to use the newly rescued symbols. Any unselected symbols will not be rescued, but their symbol linkage can be updated in the schematic later.

Alternatively, pressing **Skip Symbol Rescue** will exit the dialog without rescuing any symbols. KiCad will use the versions of the symbols found in the external libraries. You can run the rescue function again with **Tools** → **Rescue Symbols...**, or manually edit symbol linkage in the symbol's properties.

If you would prefer not to see this dialog, you can press **Never Show Again**. This has the same effect as pressing **Skip Symbol Rescue** for the current schematic and all future schematics.

If a symbol in a legacy schematic cannot be found in either the cache library or the external library, KiCad cannot rescue that symbol. A placeholder symbol is inserted into the schematic in its place, as shown below.

You can attempt to remap these orphaned symbols using the **Change Symbols** or **Edit Symbol Library Links** dialogs, but either option may require manual corrections to the schematic. These tools are explained in more detail in the [Updating and exchanging symbols](#) section.



Opening pre-5.0 schematics

Modern versions of KiCad can open schematics created in versions prior to KiCad 5.0, but you will need to go through a symbol remapping process to open the schematic without losing symbol information.

Since version 5.0, KiCad schematics refer to specific symbols using both the symbol and library name. Even if multiple libraries each contain a symbol with the same name, the designer's intended symbol is unambiguously specified.

Prior to version 5.0, KiCad schematics stored only the symbol name, not the library name. Symbols in the schematic were indirectly mapped back to the original library by searching through the project's library list for a matching symbol. When you open a pre-5.0 schematic, KiCad will attempt to automatically "remap" the symbols so that each bare symbol name is replaced with a fully-specified symbol library and symbol name pair. The original schematics will be backed up in a `rescue-backup` folder.

You can skip the automatic remapping, but you will need to remap the symbols yourself using the [Change Symbols dialog](#). You can also re-run the Remap Symbols tool using **Tools** → **Remap Legacy Library Symbols...**



Remap Symbols

This schematic currently uses the project symbol library list look up method for loading library symbols. KiCad will attempt to map the existing symbols to use the new symbol library table. Remapping will change some project files and schematics may not be compatible with older versions of KiCad. All files that are changed will be backed up to the "remap_backup" folder in the project folder should you need to revert any changes. If you choose to skip this step, you will be responsible for manually remapping the symbols.

Remap Symbols

Close

Output Messages

Show: All

Errors 0

Warnings 0

Actions

Infos

Save...

层次原理图

简介

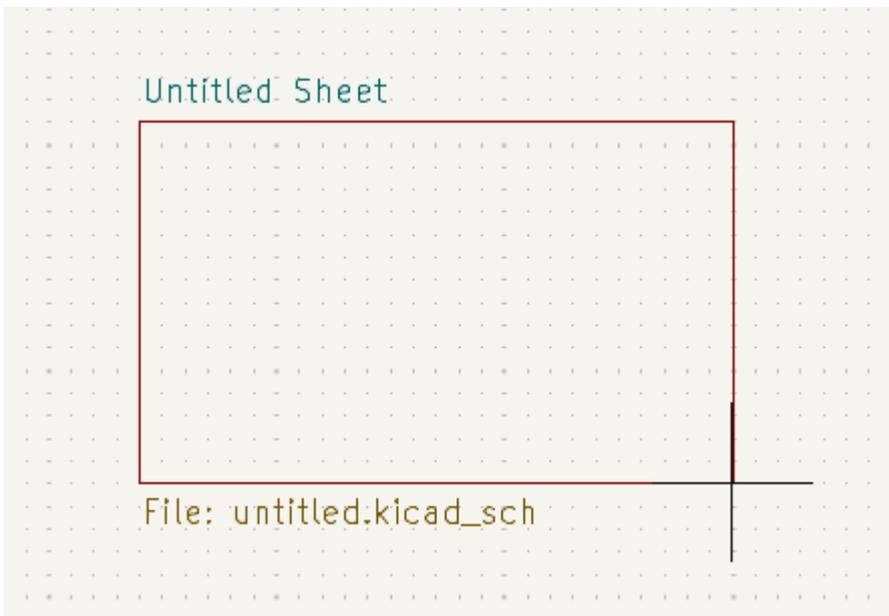
在 KiCad 中，多张原理图是层次的：有一个单一的根图，其他图作为根原理图或另一个子原理图的子原理图被创建。如果需要的话，原理图可以被多次包含在一个层次结构中。

仔细地将原理图绘制成层次设计，可以提高原理图的可读性，减少重复绘制。

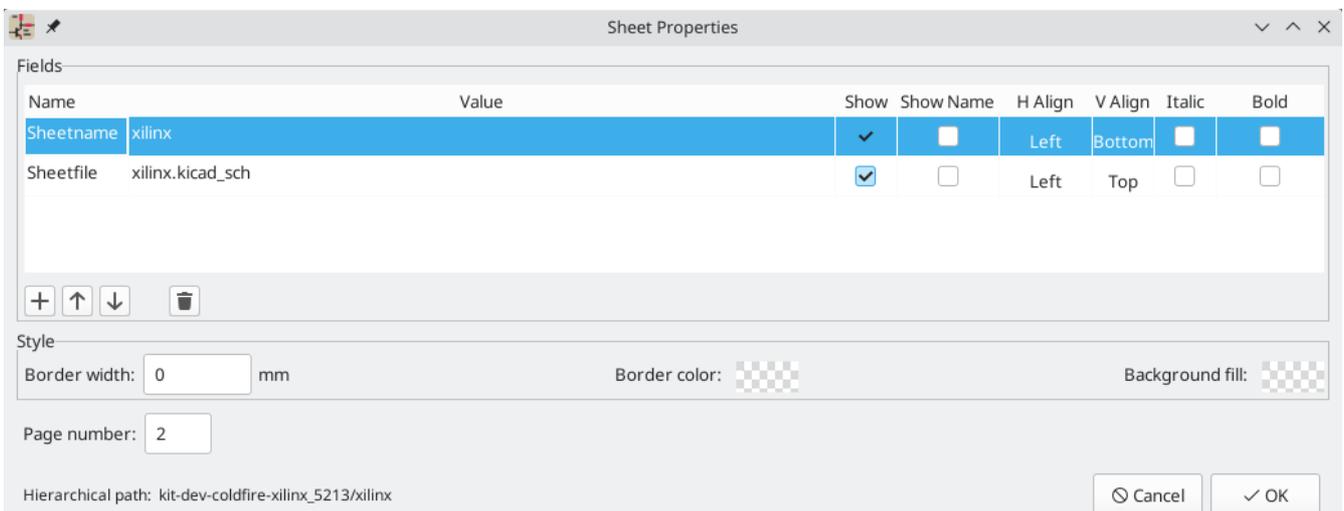
创建层次原理图是从根原理图开始的。其过程是创建一个子原理图，然后在子原理图中绘制电路，并在原理图之间进行必要的电气连接。可以使用层次页码和标签在子表和父表的网之间建立连接，也可以使用全局标签在层次中的任何两个网之间建立连接。

在设计中添加图框

你可以用“添加层次原理图”工具（**S** 快捷键，或右边工具栏上的  按钮）在设计中添加一个分表。启动该工具，然后在画布上点击两次，绘制子原理图符号的左上角和右下角。使原理图的轮廓足够大，以适应《层次工作表针，层次页码你将在后面添加》。



原理图属性对话框将出现，并提示你输入原理图名称和文件名。



原理图名称 必须是唯一的，因为它被用于子原理图中任何网络的全局网络名称中。例如，在原理图 sheet1 中具有局部标签 net1 的网络将具有 /sheet1/net1 的全局网络名称。原理图名称也被用来在图形用户界面的不同地方指代原理图，包括《原理图标题块》和《原理图之间的导航，层次导航器》。

The **sheet file** specifies the file that the new sheet will be saved to or loaded from. The path to the sheet file can be relative or absolute. It is usually preferable to save subsheet files in the project directory and use a relative path so that the project is portable.

通过为每个重复的原理图指定相同的文件名，一个原理图文件可以在一个工程中使用多次；原理图中绘制的电路将在每次使用时被实例化一次，一次实例中的任何编辑都将反映在其他实例中。

NOTE

原理图文件可以在多个工程之间共享，以允许在工程之间重复使用设计。然而，由于对路径可移植性的关联和在编辑共享原理图时无意中改变其他工程的风险，不建议这样做。

原理图的 **页数** 在这里是可以配置的。页码显示在原理图《原理图标题块，标题块》和《在原理图之间导航器，层次导航器》中，原理图在层次导航器和《生成输出，打印或绘制》中按页码排序。

也有几个图形选项可用。**边界宽度** 设置工作表形状周围的边界宽度。**边界颜色** 和 **背景填充** 分别设置工作表形状的边界和填充的颜色。如果没有设置颜色，就会显示一个棋盘式的色块，并使用颜色主题默认值。

Sheets support arbitrary custom fields, which can be added and removed with the **+** and **✖** buttons, respectively. Sheet fields can be optionally displayed on the schematic by checking their **Show** box, and they can be accessed from inside the sheet or in other sheet fields using [text variables](#).

通过选择一个工作表符号并使用 **E** 快捷键，或者通过右键单击工作表符号并选择 **属性...**，可以随时访问工作表属性对话框。

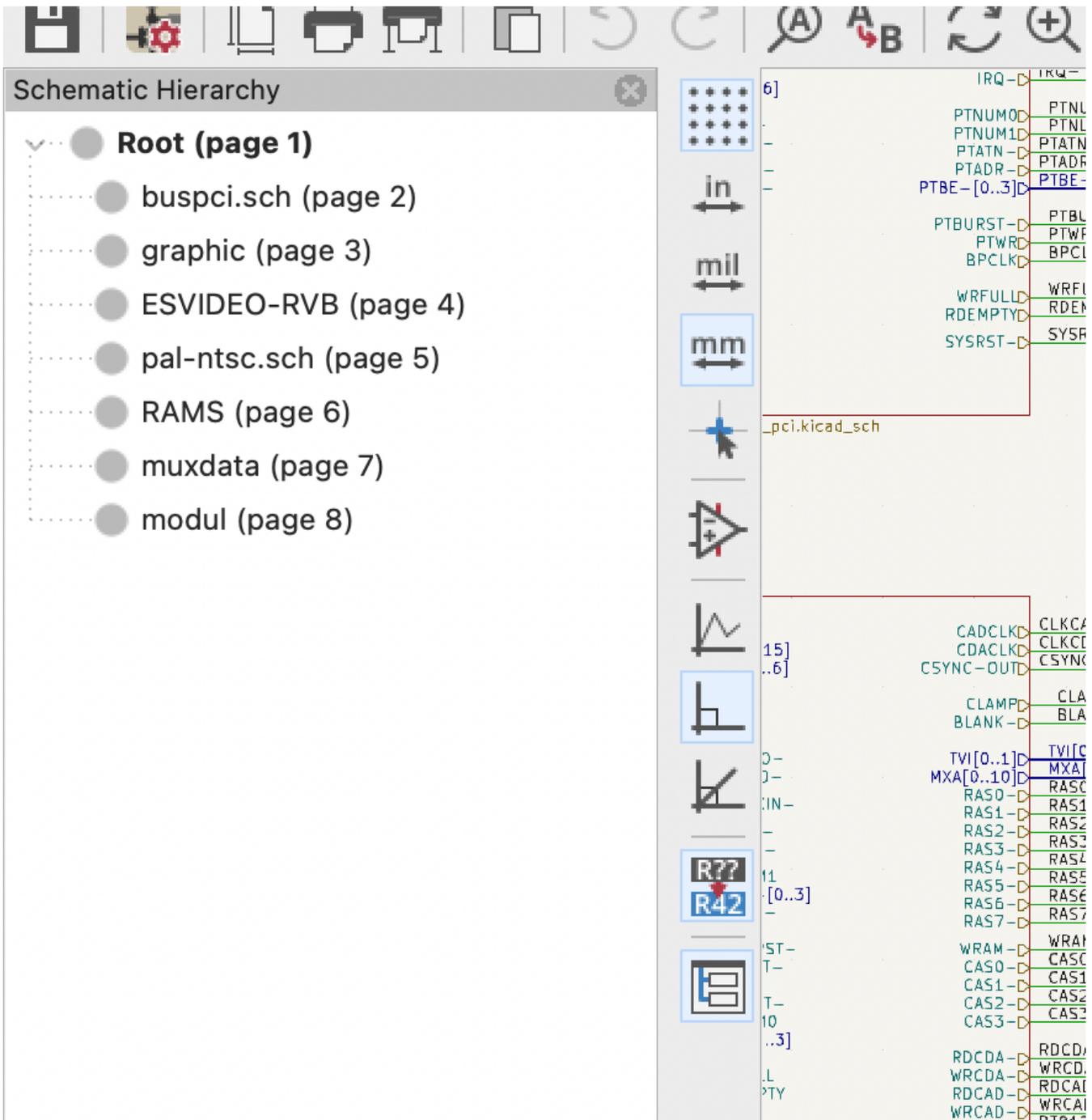
原理图之间导航器

你可以通过双击子原理图的形状，或右击子原理图并选择 **进入原理图**，从父原理图进入一个分层原理图。

通过使用顶部工具栏中的 **↑** 按钮，或者在原理图的空白部分点击右键，然后点击 **离开原理图**，返回到父原理图。

你可以用 **→** 按钮跳到下一个原理图，或者用 **←** 按钮跳到上一个原理图。

另外，你也可以通过层次结构导航器跳到任何原理图。要打开层次结构导航器，请点击左边工具栏上的 **☰** 按钮。层次结构导航器会停靠在屏幕的左边。设计中的每个原理图都显示为工作区中的一个工程。点击一个原理图的名字就可以在编辑画布上打开该原理图。



原理图之间的电气连接

标签概述

原理图之间的电气连接是通过《标签, 标签》进行的。在 KiCad 中, 有几种标签, 每种都有不同的连接范围。

- **局部标签** 只在一个原理图内进行连接。因此, 局部 标签不能用于原理图之间的连接。局部标签的添加方法是 **A** 按钮添加。
- **全局标签** 可以在原理图的任何地方进行连接, 而不考虑原理图。全局标签可以通过以下方式添加 **A** 按钮。
- **层次化标签** 连接到 **层次化原理图页码**, 可在父原理图中访问。父级原理图。分层设计依靠分层标签和页码来实现父原理图和子原理图之间的连接。在父原理图和子原理图之间建立联系; 你可以把分层的页码看成是定义父原理图的接口。你可以把分层的页码看作是定义原理图的接口。分层标签是用 **A** 按钮。

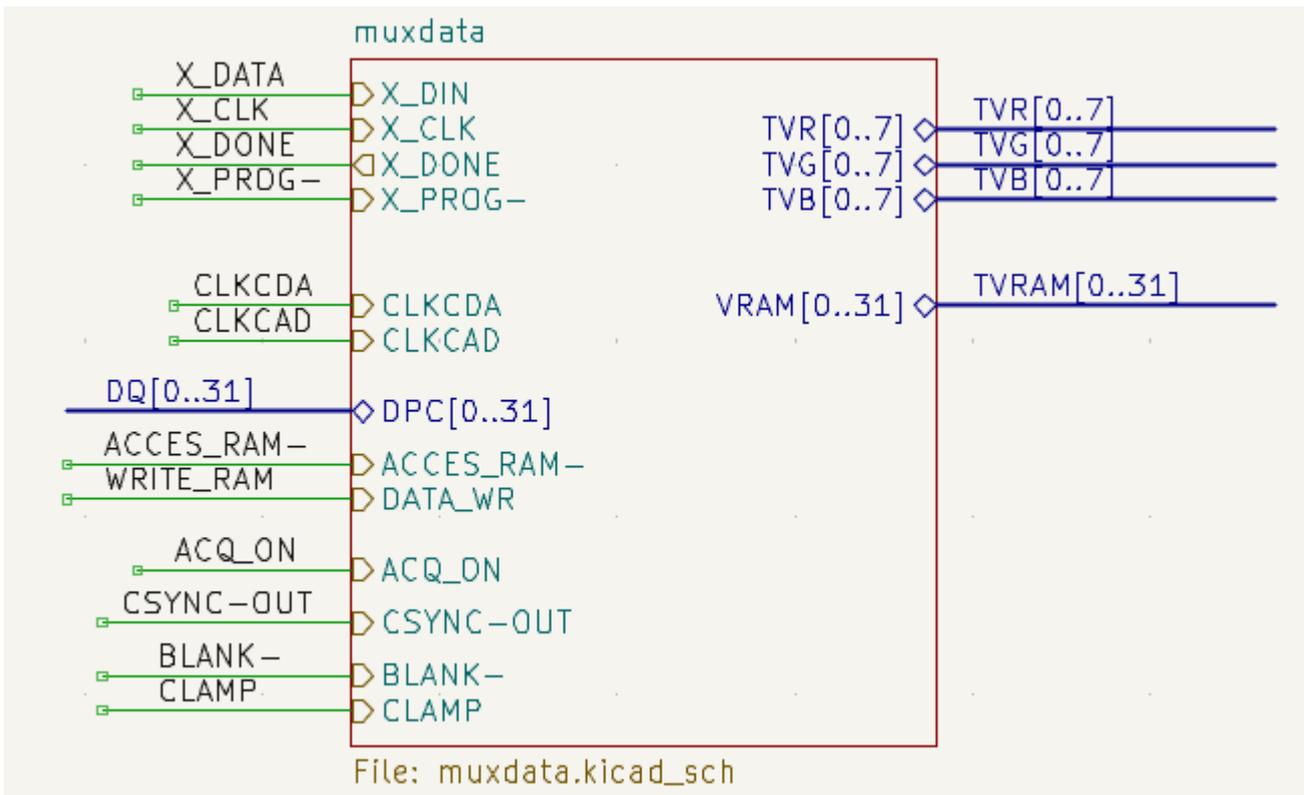
NOTE 具有相同名称的标签将被连接，无论标签类型如何。如果它们在同一张原理图上。

NOTE 《隐藏电源引脚，隐藏电源引脚》也可以被认为是全局标签，因为它们连接到原理图层次结构中的任何地方。

测试原理图页码

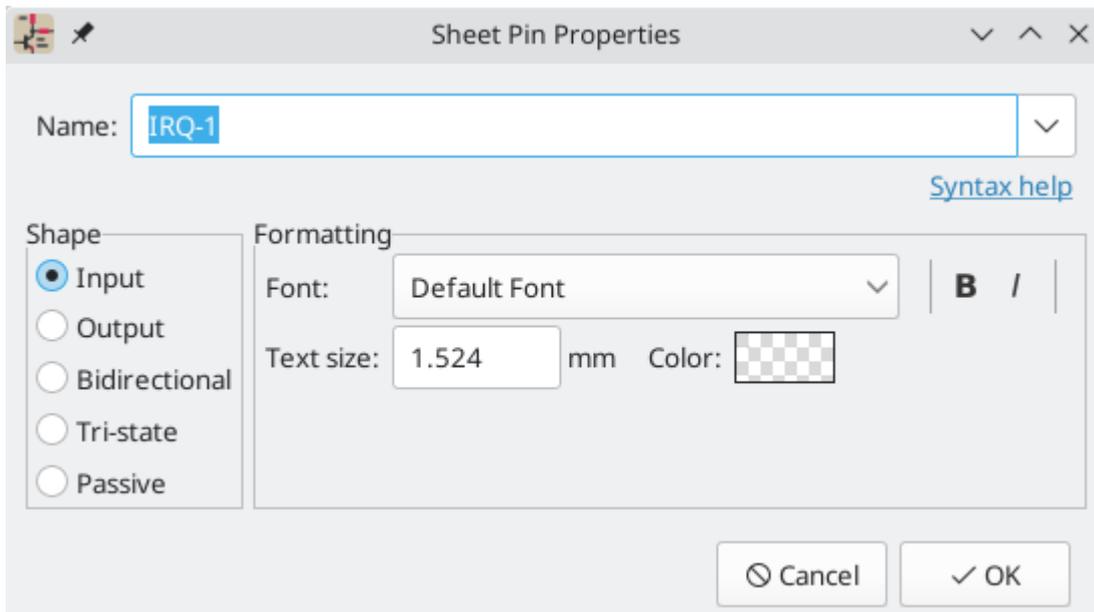
在子原理图内放置分层标签后，可以在父原理图内的子原理图符号上添加匹配的 **分层引脚**。然后，你可以用导线、标签和总线与分层引脚进行连接。子原理图符号中的分层引脚与子原理图本身的匹配分层标签相连。

NOTE 在原理图符号中导入相应的分层原理图页码之前，必须先在此子原理图中定义分层标签。在原理图符号中导入相应的分层原理图页码之前，必须先在此子原理图中定义分层标签。



对于子原理图中的每一个分层标签，通过点击右侧工具栏中的  按钮，然后点击原理图符号，将相应的分层标签导入原理图符号中。第一个不匹配的分层标签的原理图页码将附在光标上，它可以放在原理图符号边界的任何地方。再次点击该工具将继续导入更多的原理图页码，直到没有更多的分层标签需要从子原理图中导入。原理图页码也可以通过在原理图符号的右键菜单中选择 **导入原理图页码** 来导入。

你可以在原理图页码属性对话框中编辑原理图页码的属性。通过双击原理图页码，选择原理图页码并使用  热键，或者右击原理图页码并选择 **属性...** 来打开这个对话框。



原理图页码的 **名称** 可以在文本框中编辑，也可以从子原理图的分级标签的下拉列表中选择。原理图页码的名称必须与子原理图中相应的层次标签相匹配，所以如果页码的名称被改变，标签也必须改变。

Shape changes the shape of the sheet pin, and has no electrical effect. It can be set to Input, Output, Bidirectional, Tri-state, or Passive. The pin's **font**, **text size**, **color**, and emphasis (bold or italic) can also be changed.

层次设计实例

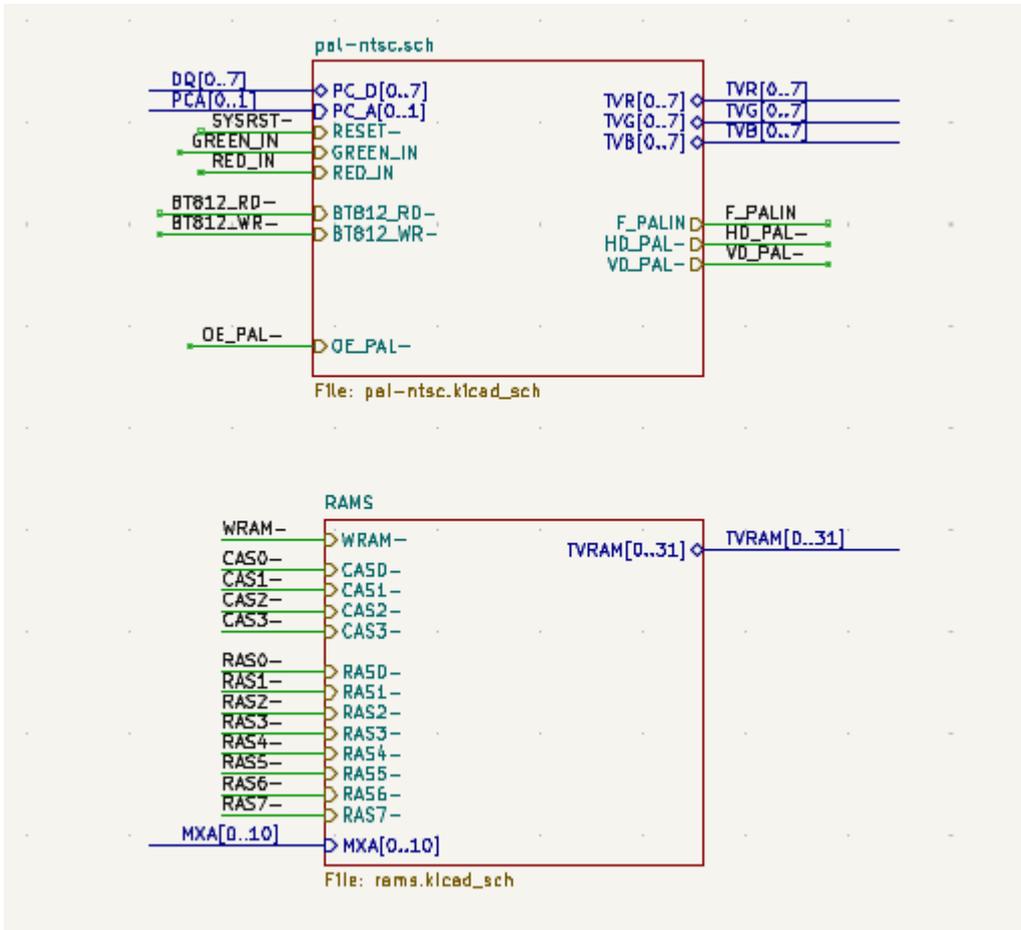
层次设计可以被归入几个类别中的一个：

- **简单：** 每张原理图只使用一次。
- **复杂：** 一些原理图被多次使用。
- **扁平化：** **简单** 层次结构的一个子案例，在子原理图和其父原理图之间没有连接。扁平的层次结构可以用来表示一种非层次结构设计。

每种层次结构模型都可能是有用的；最合适的模型取决于设计。

简单的层次结构

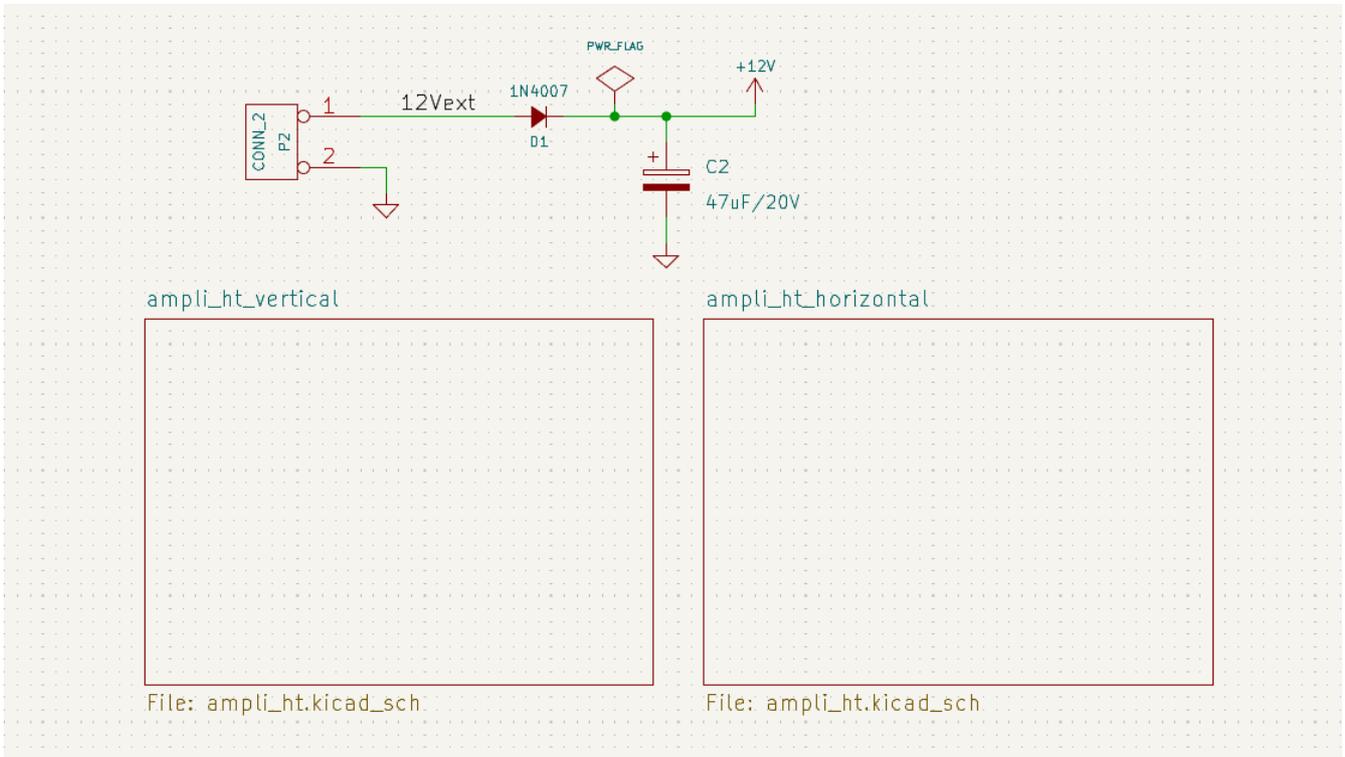
一个简单的层次结构的例子是 KiCad 包含的 `video` 演示工程。根原理图包含七个独特的子原理图，每个子原理图都有分层的标签和原理图页码，将原理图与根原理图相互连接。下面是其中两个子原理图的符号。



复杂的层次结构

复杂的层次结构 演示工程是一个复杂层次结构的例子。根原理图包含两个子原理图符号，它们都指向同一个原理图文件（`ampli_ht.kicad_sch`）。这使得设计中包含了同一个放大器电路的两个副本。尽管这两个原理图符号指向同一个文件名，但原理图的名称是唯一的（`ampli_ht_vertical` 和 `ampli_ht_horizontal`）。在每个子原理图内，除了位号外，其他的电路都是相同的，而位号也是唯一的。

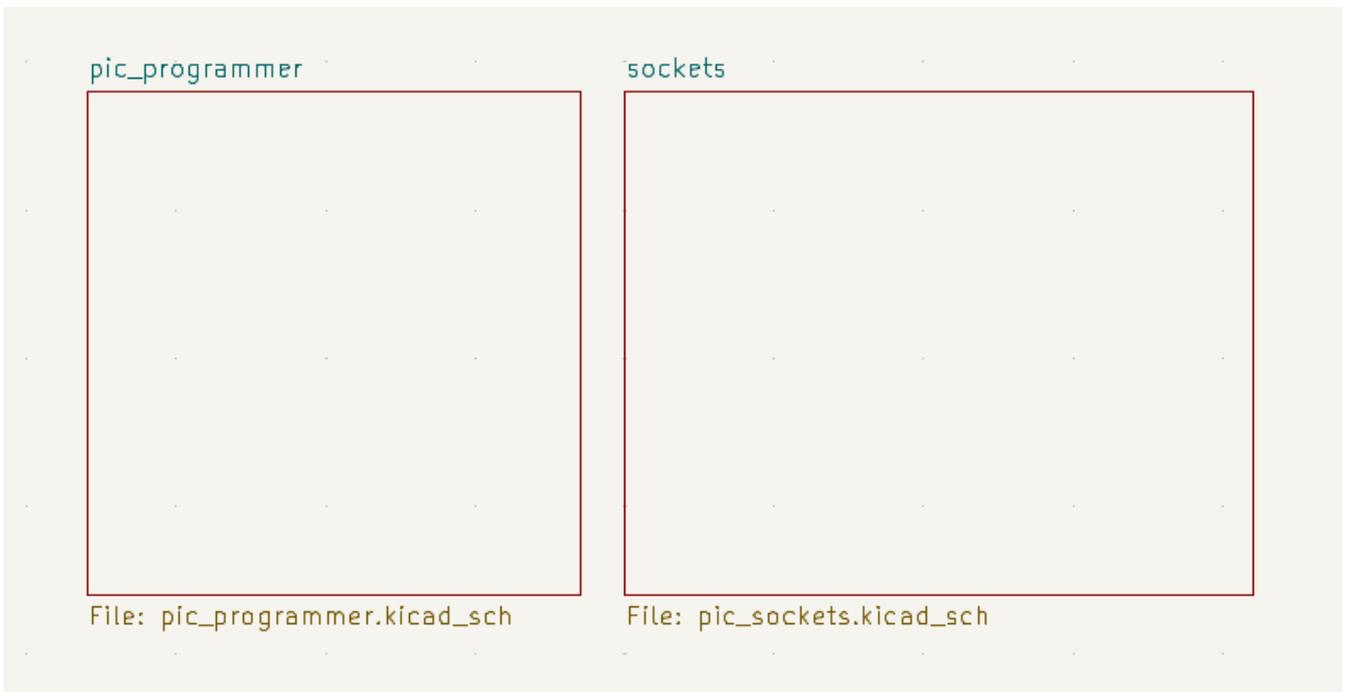
这个工程不包含原理图的引脚连接。根原理图和子原理图之间的唯一连接是用《电源符号，电源符号》进行的全局电源连接。然而，如果适合设计，复杂层次结构中的原理图可以包括原理图页码的连接。



扁平化层次结构

flat_hierarchy 演示工程是一个扁平化层次结构的例子。根原理图包含两个唯一的子原理图符号，没有分层的原理图页码。在这个工程中，根原理图除了容纳子原理图外没有任何作用，子原理图只是作为原理图中的附加页使用。

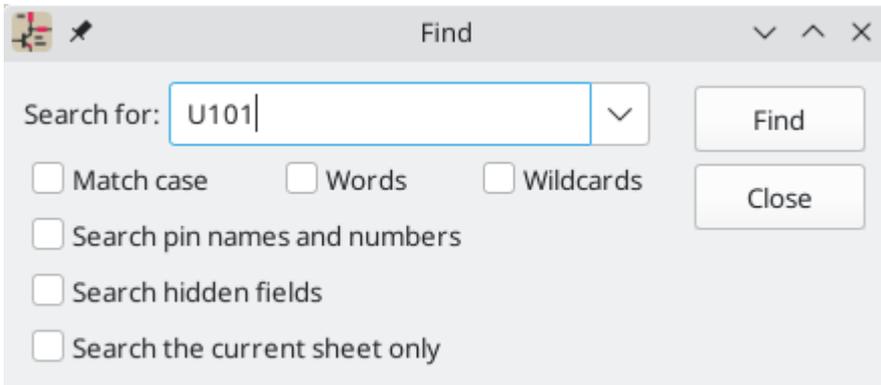
NOTE | 这是在 KiCad 中创建多页原理图的最简单方法。



检查原理图

筛选工具

筛选工具在原理图中搜索文本，包括位号、引脚名称、符号字段和图形文本。当该工具找到一个匹配的文本时，画布会被放大并居中显示该匹配的文本。使用顶部工具栏上的  按钮启动该工具。



筛选工具有几个选项：

匹配大小写： 选择搜索是否对大小写敏感。

关键词： 选择后，搜索将只与原理图中完整的搜索词相匹配。当未选择时，如果搜索词是原理图中一个较大的关键词的一部分，搜索将匹配。

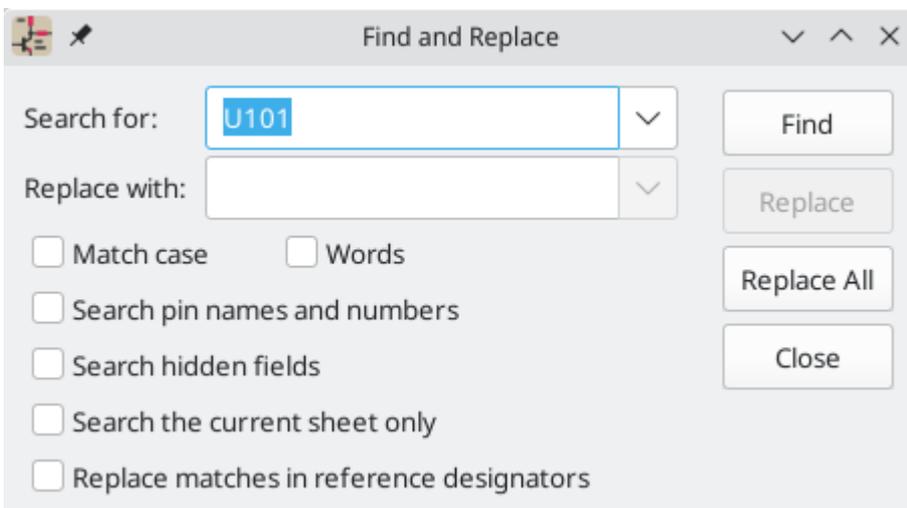
通配符： 当选择时，通配符可以在搜索词中使用。? 匹配任何单个字符，而 * 则匹配任何数量的字符。请注意，当选择这个选项时，不会返回部分匹配：搜索 abc* 将匹配字符串 abcd，但搜索 abc 不会。

搜索引脚名称和编号： 选择搜索是否应适用于引脚名称和编号。

搜索隐藏字段： 选择搜索是否只适用于可见字段，或是否应包括隐藏的符号字段。

只搜索当前原理图： 选择搜索是否应限于当前原理图或整个原理图。

还有一个查找和替换工具，可以通过顶部工具栏上的  按钮激活。这个工具的作用与筛选工具相同，但还可以用不同的文本替换部分或全部匹配的文本。



如果选中 **替换位号中的匹配内容** 选项，位号将被修改，如果它们包含匹配文本。否则，位号将不会受到影响。

网络高亮显示

电气网络可以在原理图编辑器中高亮显示它在原理图中出现的所有位置。电气网络高亮可以在原理图编辑器中激活，也可以在启用交叉探测高亮时在 PCB 编辑器中高亮相应的网（见下文）。当网络高亮激活时，高亮的网络将以不同的颜色显示。默认情况下，这种颜色是粉红色，但可以在 "偏好设置" 对话框的颜色部分进行配置。

Nets can be highlighted by clicking on a wire or pin using the Highlight Net tool in the right toolbar . Alternatively, the Highlight Net hotkey () highlights the net under the cursor.

Net highlighting can be cleared by using the Clear Net Highlight action (hotkey ) or by using the Highlight net tool on an empty region in the schematic. By default,  also clears net highlighting, but this can be disabled if desired in **Preferences** → **Schematic Editor** → **Editing Options**.

从 PCB 上交叉探测

KiCad 允许在原理图和 PCB 之间进行双向交叉探测。有几种不同类型的交叉探测。

选择交叉探测 允许你在原理图中选择一个符号或引脚来选择 PCB 中相应的封装或焊盘（如果存在的话），反之亦然。默认情况下，交叉探测将导致显示在交叉探测项目的中心并放大以适应。这种行为可以在偏好设置对话框的显示选项部分禁用这一行为。

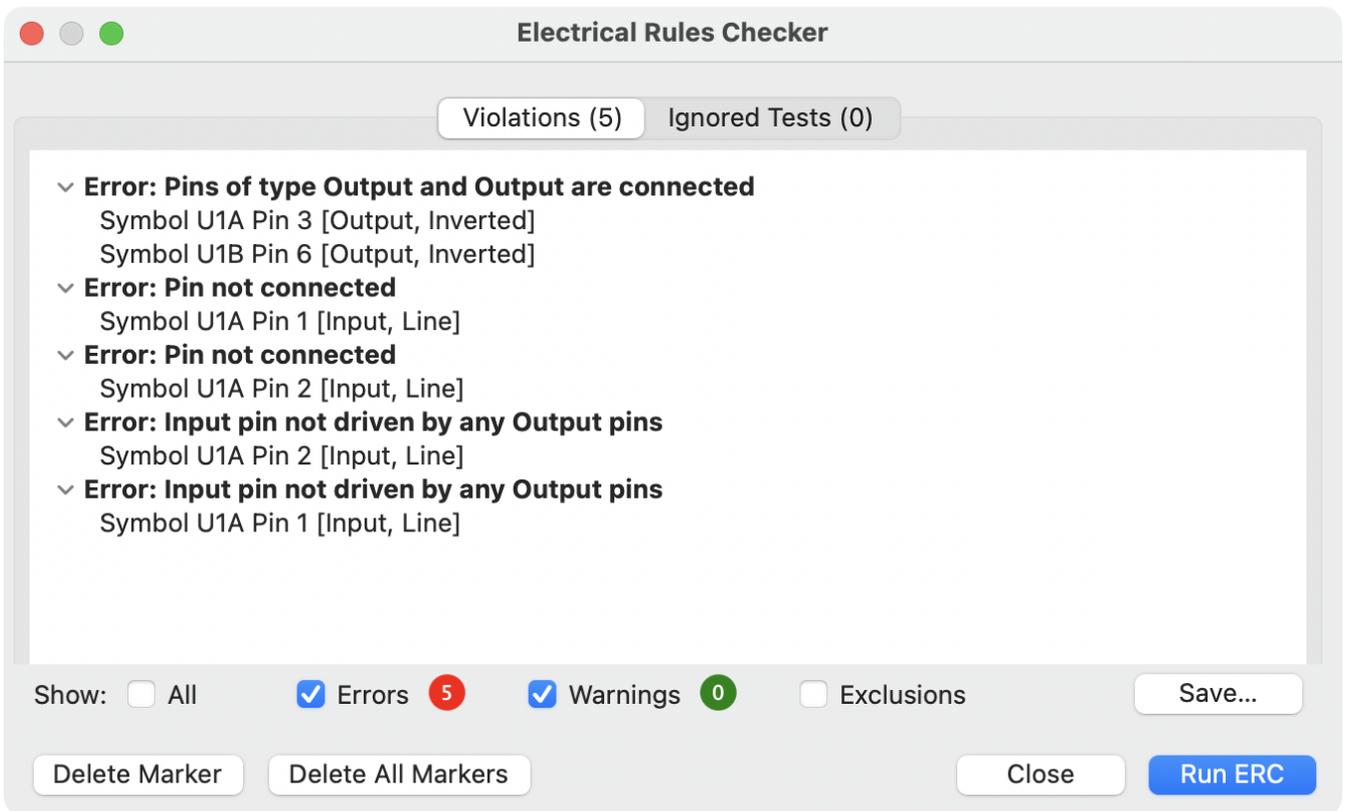
突出显示交叉探测 允许你在原理图和 PCB 中同时高亮显示一个网络。如果在 "偏好设置" 对话框的 "显示选项" 部分启用了 "高亮显示交叉探测的网络" 选项，那么在原理图编辑器中高亮显示一个网络或总线将导致 PCB 编辑器中相应的网络或网络被高亮显示，反之亦然。

电气规则检查

电气规则检查（ERC）工具检查你的原理图中的某些错误，如未连接的引脚、未连接的层次符号、短路的输出或其他非法连接，等等。违反 ERC 的情况会根据检测到的问题的严重程度，以错误或警告的形式报告。

ERC 是不完善的，不能检测所有的错误，但它可以检测到许多常见的问题和疏忽。所有检测到的问题都应该在继续之前进行检查和处理。ERC 的质量与符号创建过程中声明《引脚-电气-类型，电气引脚属性》的谨慎程度直接相关。如果符号设计不正确，ERC 将不会报告准确的信息。

ERC 可以通过点击顶部工具栏的  按钮并点击 **运行 ERC** 按钮来启动。

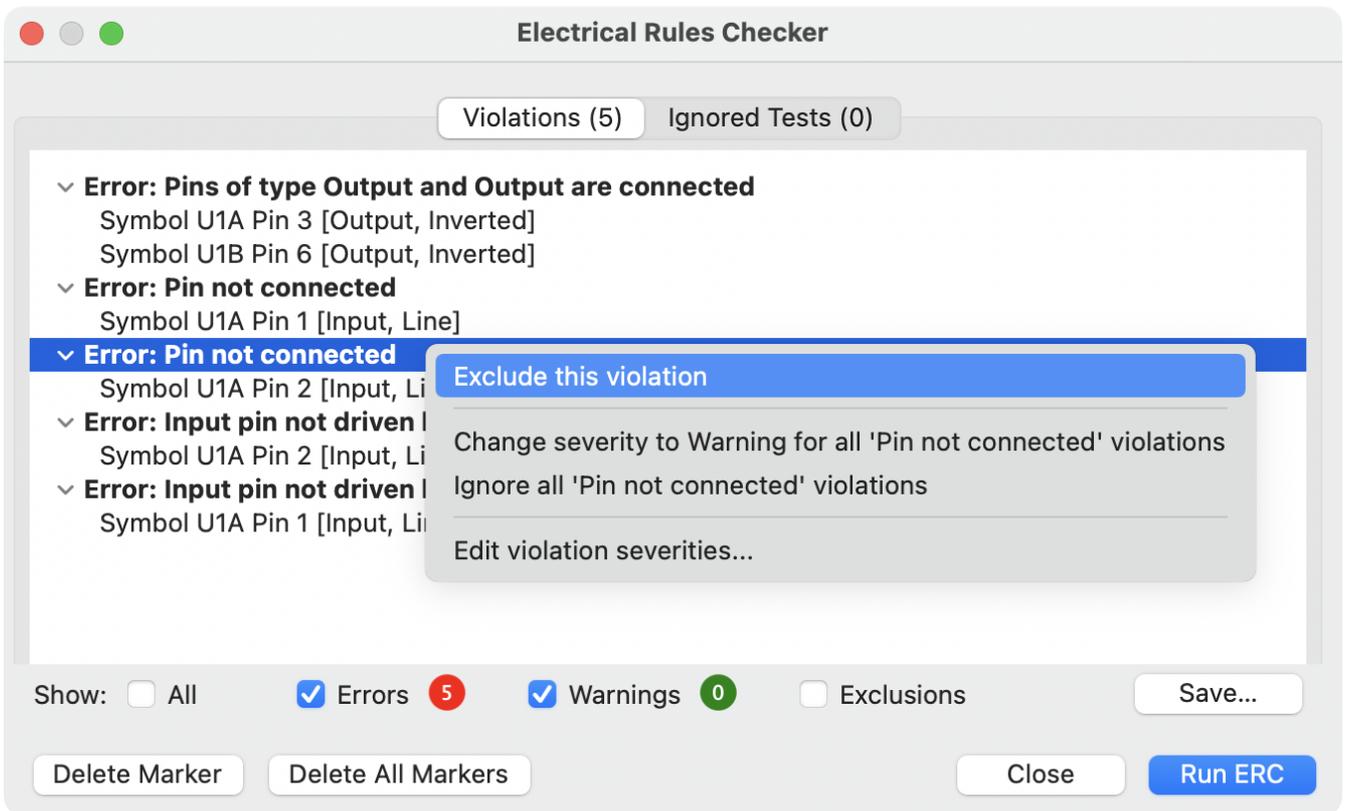


任何警告或错误都会在 **违规行为** 标签中报告，每个违规行为的标记都会放在原理图中，使其指向原理图的相关部分。警告用黄色箭头表示，错误用红色箭头表示。被排除的违规行为显示为绿色箭头。

NOTE | 在 ERC 窗口中选择一个违规，就会跳到原理图中所选择的违规标记。

窗口底部的数字显示错误、警告和排除的数量。每种类型的违规行为都可以用各自的复选框从列表中过滤出来。点击 **删除标记** 将清除所有违规行为，直到 ERC 再次运行。

可以在对话框中右键点击违规行为，以忽略它们或改变其严重程度：

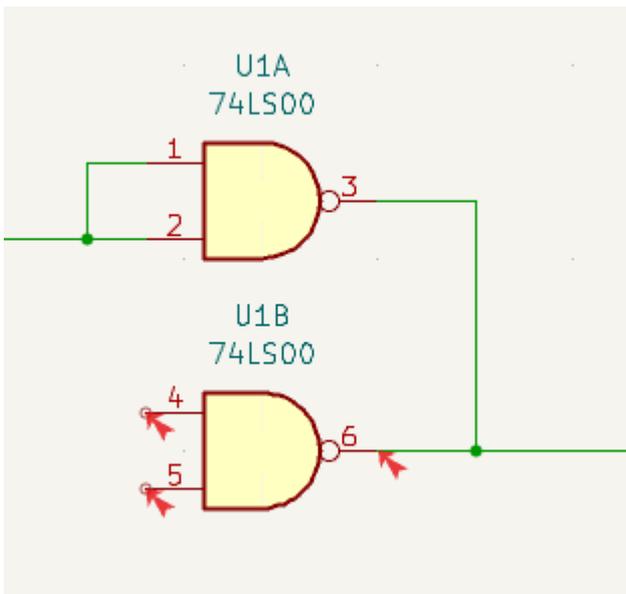


- **排除此违规行为：** 忽略此特定违规行为，但不影响其他任何违规行为。
- **改变严重程度：** 将一个违规类型从警告改为错误，或将错误改为警告。这影响到一个给定类型的所有违规行为。
- **忽略所有：** 忽略所有给定类型的违规行为。这个测试现在将出现在 **忽略的测试** 标签，而不是 **违规** 标签。

你也可以用 **检查** → **排除标记** 排除所选标记，用 **查看** 菜单显示或隐藏每一类标记（错误、警告和排除）。

在设计规则检查器运行期间，排除的和忽略的违规行为会被记住。

ERC 例子

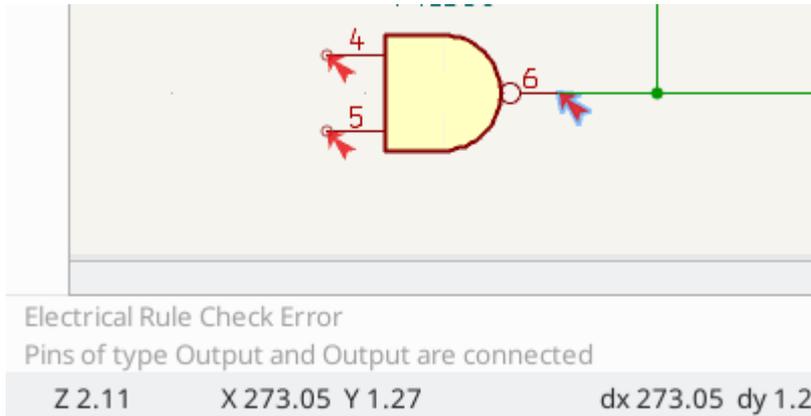


在上面的截图中，有三个错误。

- 两个输出被连接在一起（右边的红色箭头）。

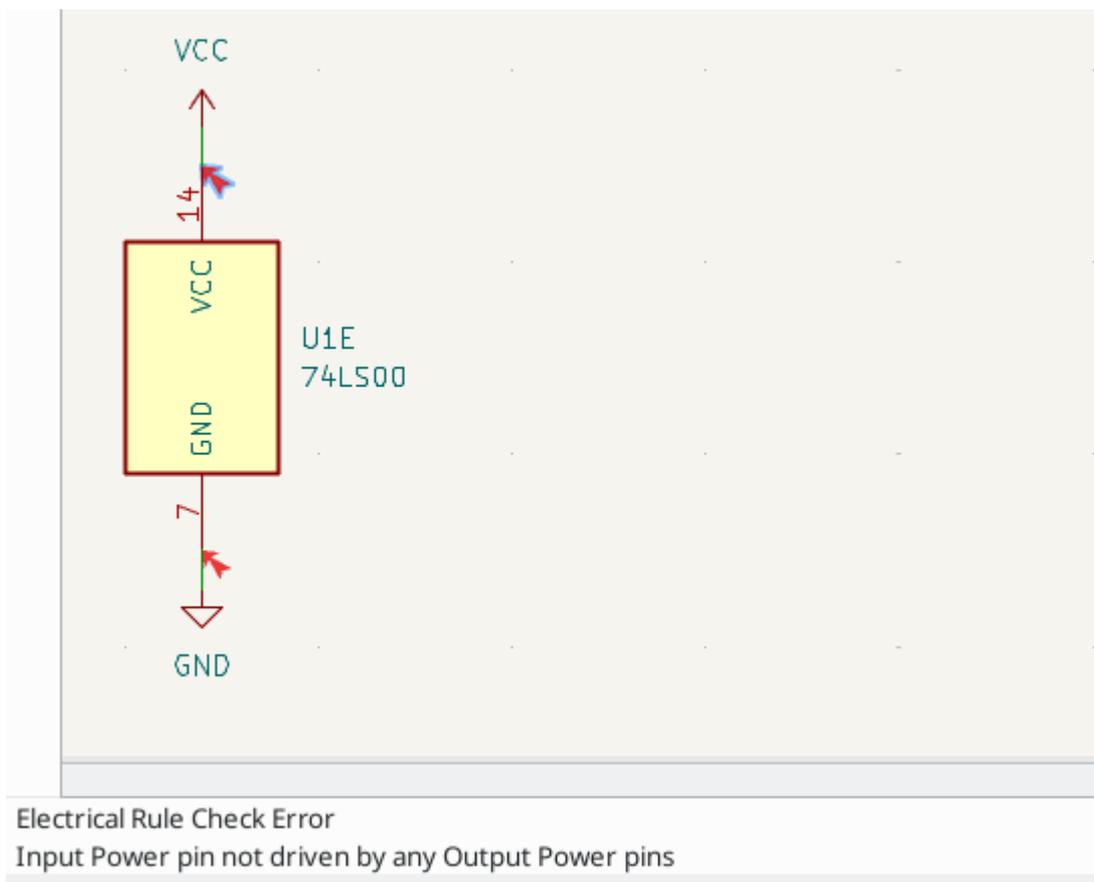
有两个输入没有连接（左边的红色箭头）。这实际上是每个引脚的两个错误：每个引脚都没有连接，而且每个引脚都是没有被输出引脚驱动的输出引脚。

选择一个 ERC 标记会在窗口底部的信息窗格中显示违规描述。

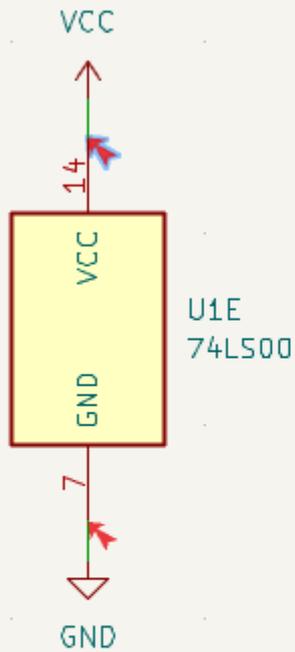


电源引脚和电源标志

在电源引脚上出现“输入电源引脚未被任何输出电源引脚驱动”的错误是很常见的，如下面的例子所示，即使电源引脚似乎正确地连接到电源导线。这种情况发生在通过连接器或其他元件提供电源的设计中，这些连接器或元件没有被标记为电源输出。在这些情况下，ERC 不会检测到任何连接到网络的输出电源引脚，而会判断输入电源引脚没有被电源驱动。



为了避免这个警告，请在这样的电源网络上将网络连接到 PWR_FLAG 符号上，如下例所示。PWR_FLAG 符号可以在 power 符号库中找到。或者，将任何电源输出引脚连接到该网；PWR_FLAG 只是一个具有单一电源输出引脚的符号。



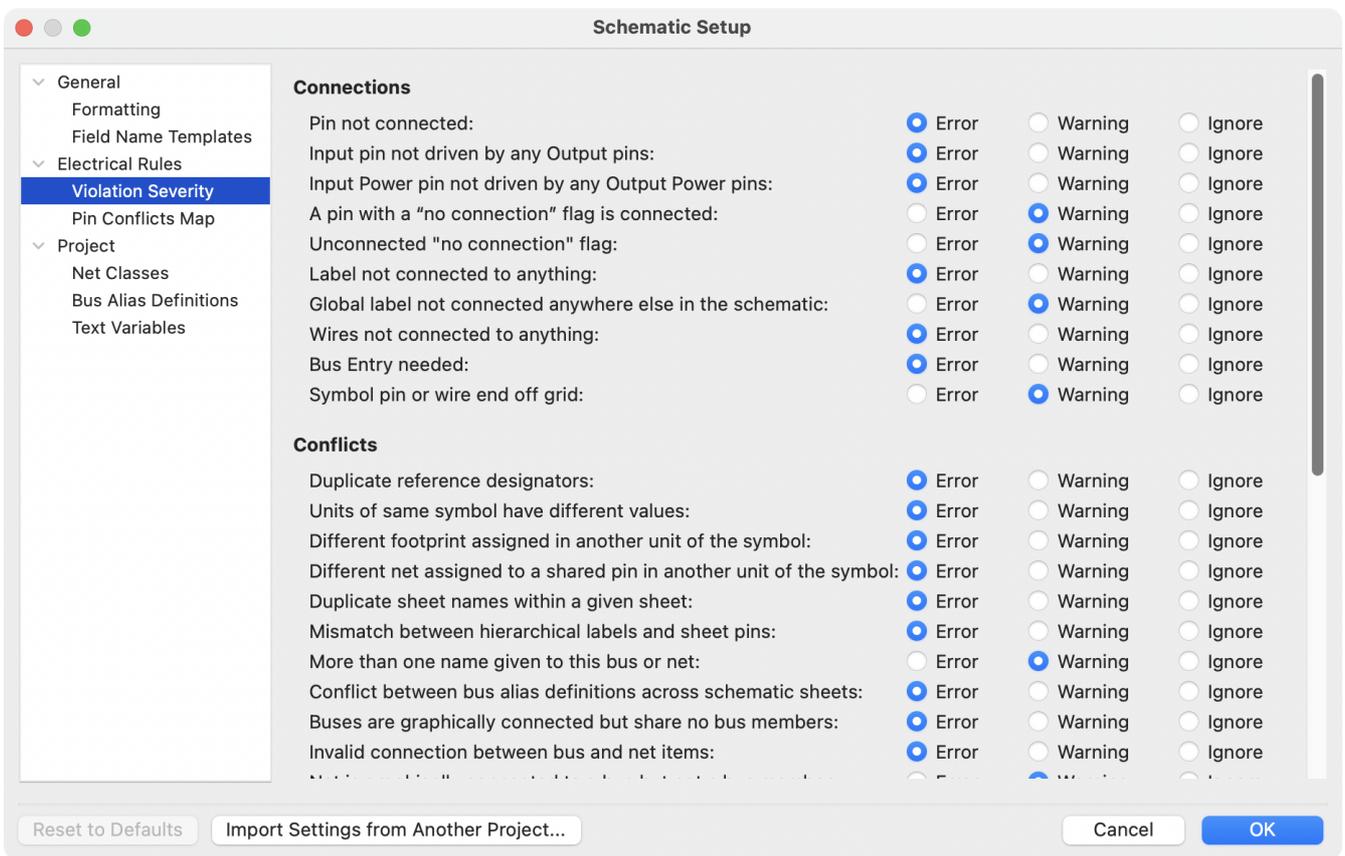
Electrical Rule Check Error
Input Power pin not driven by any Output Power pins

接地网络通常也需要一个 `PWR_FLAG`，因为电压调节器的输出被声明为电源输出，但其接地引脚通常被标记为电源输入。因此，除非使用 `PWR_FLAG` 符号，否则地线可能看起来没有与电源相连。

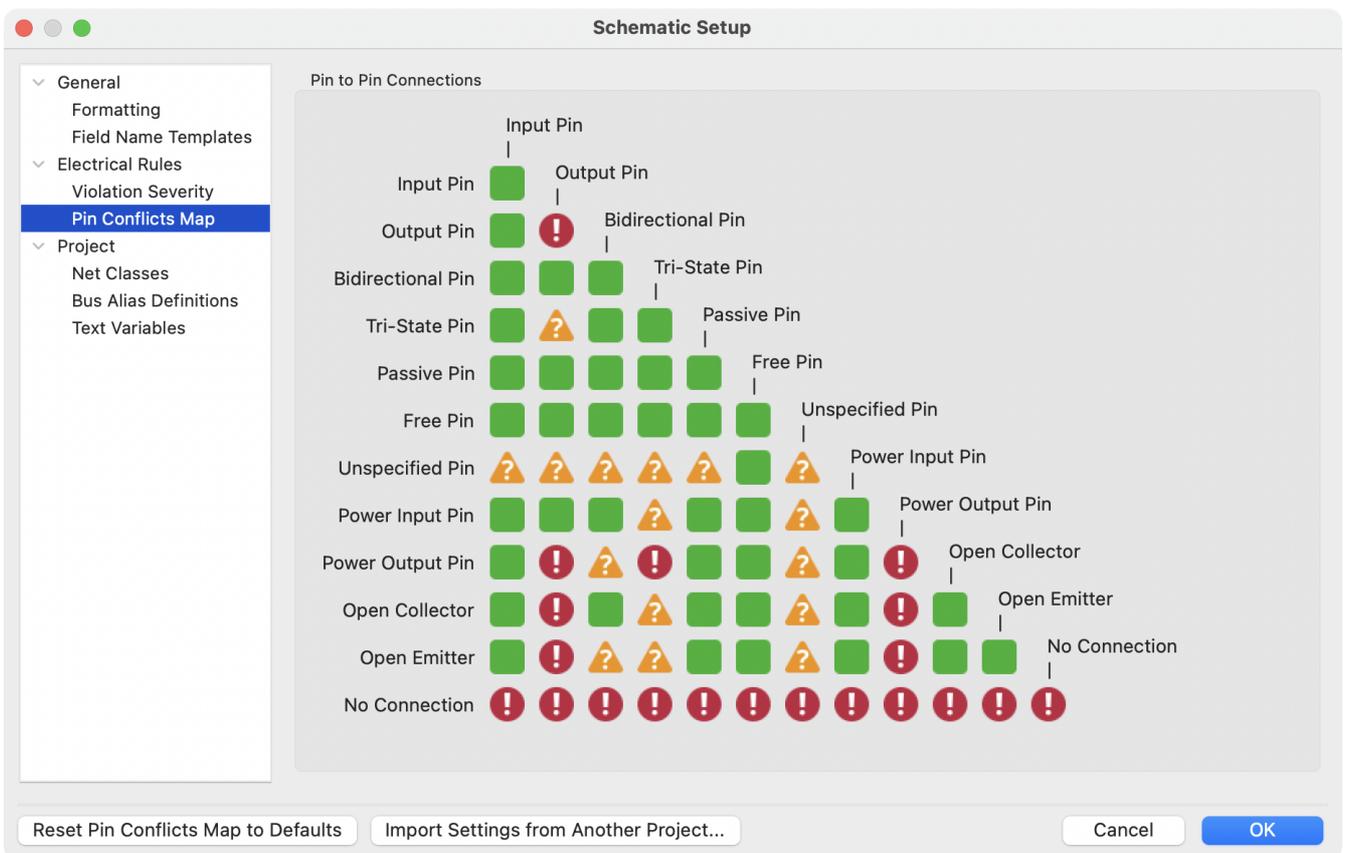
关于电源引脚和电源标志的更多信息，请参见《电源标记，`PWR_FLAG` 文档》。

ERC 配置

《原理图设置，原理图设置》中的 **违规严重性** 面板让你配置哪些类型的 ERC 信息应被报告为错误、警告或忽略。



《原理图设置，原理图设置》中的 **引脚冲突图** 面板允许你配置连接规则，根据哪些类型的引脚相互连接来定义错误和警告的电气条件。例如，默认情况下，当一个输出引脚与另一个输出引脚连接时，会产生错误。



可以通过点击矩阵中所需要的方块来改变规则，使其在各种选择中循环：允许、警告、错误。

ERC 检查清单

The table below lists the electrical rules that KiCad checks and the default violation severity for each check. All severities are configurable.

Violation	Description	Default Severity
Pin not connected	All symbol pins must be connected to a net, unless the pin has a no-connect flag or has electrical type Unconnected.	Error
Input pin not driven by any Output pins	All Input pins must be connected to an Output pin.	Error
Input Power pin not driven by any Output Power pins	All Input Power pins must be connected to an Output Power pin. A common cause of this violation is described above .	Error
A pin with a "no connection" flag is connected	Pins with no connection flags cannot be connected to anything else.	Warning
Unconnected "no connection" flag	No connection flags cannot be left unconnected. They must be connected to a pin or label.	Warning
Label not connected to anything	Global, hierarchical, and local labels must be connected to a pin or another label.	Error
Global label not connected anywhere else in the schematic	Global labels need at least two other pins on the same net; otherwise they are not connecting anything.	Warning
Wires not connected to anything	Wires must be connected to a pin or label.	Error
Bus Entry needed	This violation only applies to projects imported from EAGLE projects. It indicates places where the importer was unable to automatically add bus entries to the imported schematic, so you must add them by hand.	Error
Symbol pin or wire end off grid	Symbol pins and wire ends need to be aligned to the grid in order to connect to each other.	Warning
Duplicate reference designators	Two symbols cannot have the same reference designator.	Error
Units of same symbol have different values	All units of a single symbol must have the same value.	Error
Different footprint assigned in another unit of the symbol	All units of a single symbol must have the same footprint assigned.	Error

Violation	Description	Default Severity
Different net assigned to a shared pin in another unit of the symbol	Any pins that are shared between multiple units of a symbol must have the same net connected across all units.	Error
Duplicate sheet names within a given sheet	All hierarchical sheets within a given sheet must have unique names.	Error
Mismatch between hierarchical labels and sheet pins	All hierarchical labels must have a corresponding hierarchical sheet pin in the parent sheet, and all hierarchical sheet pins must have a matching hierarchical label in the child sheet.	Error
More than one name given given to this bus or net	Nets can only have a single name, so if multiple labels are attached to a net, one name will be selected and used as the canonical name.	Warning
Conflict between bus alias definitions across schematic sheets	If the same bus alias name is used in multiple sheets, the members of the alias must be the same for each sheet.	Error
Buses are graphically connected but share no bus members	Buses that are graphically connected must have bus members in common.	Error
Invalid connection between bus and net items	Buses cannot be connected to net items such as wires, labels referring to single nets, or sheet pins referring to single nets. Labels and sheet pins can only be connected to buses if they refer to buses rather than individual signals.	Error
Net is graphically connected to a bus but not a bus member	Nets can only be connected to a bus with a bus entry if they are a member of the bus.	Warning
Conflicting netclass assignments	Only one netclass can be assigned to a net.	Error
Symbol is not annotated	Symbols must be annotated with a unique reference designator .	Error
Unresolved text variable	Text variables (<code>{variable_name}</code>) must be defined in Schematic Setup before they can be used.	Error
SPICE model issue	SPICE models must not have syntax errors or other problems.	Error
Labels are similar (lower/upper case difference only)	If two labels are similar and differ only by the case of some letters, this may be a typo causing two labels to be disconnected when they are intended to be connected.	Warning

Violation	Description	Default Severity
Library symbol issue	Several symbol library issues are checked: <ul style="list-style-type: none"> • 每个符号的符号库都被包括在内，并在《管理符号库，库表》。 • 原理图中的每个符号都存在于其符号库中 • 原理图中的每个符号与它的符号库中的副本相匹配 	警告
符号的单元多于定义的单元	符号在原理图中放置的单元不能多于符号中的定义。原理图中的单元必须与符号定义完全对应。	错误
符号中有未被放置的单元	多单元符号中的所有单元都应放置在原理图中。	警告
符号有未放置的输入引脚	多单元符号中有单元的输入引脚没有被放置，所以这些输入引脚不会被连接到任何东西。所有带有输入引脚的单元都应该被放置在原理图中。	警告
符号中有未被放置的双向引脚	多单元符号有未被放置的双向引脚的单元，因此这些双向引脚不会连接到任何东西。所有具有双向引脚的单元都应该被放置在原理图中。	警告
符号中有未被放置的电源输入引脚	一个多单元符号有未被放置的电源输入引脚的单元，因此这些电源输入引脚将不会被连接到任何东西。所有带有电源输入引脚的单元都应放在原理图中。	错误
引脚之间的冲突问题	引脚之间的连接必须符合《ERC 配置，引脚冲突图》中的允许连接。	从引脚冲突图

ERC 报告文件

可以通过点击 ERC 对话框中的 **保存...** 按钮来生成和保存 ERC 报告文件。ERC 报告文件的扩展名为 `.rpt`。下面给出一个 ERC 报告文件的例子。

ERC report (Fri 21 Oct 2022 02:07:05 PM EDT, Encoding UTF8)

***** Sheet /

```
[pin_not_driven]: Input pin not driven by any Output pins
; Severity: error
@(149.86 mm, 60.96 mm): Symbol U1B [74LS00] Pin 4 [, Input, Line]
[pin_not_connected]: Pin not connected
; Severity: error
@(149.86 mm, 60.96 mm): Symbol U1B [74LS00] Pin 4 [, Input, Line]
[pin_not_connected]: Pin not connected
; Severity: error
@(149.86 mm, 66.04 mm): Symbol U1B [74LS00] Pin 5 [, Input, Line]
[pin_to_pin]: Pins of type Output and Output are connected
; Severity: error
@(165.10 mm, 63.50 mm): Symbol U1B [74LS00] Pin 6 [, Output, Inverted]
@(165.10 mm, 46.99 mm): Symbol U1A [74LS00] Pin 3 [, Output, Inverted]
[pin_not_driven]: Input pin not driven by any Output pins
; Severity: error
@(149.86 mm, 66.04 mm): Symbol U1B [74LS00] Pin 5 [, Input, Line]
```

** ERC messages: 5 Errors 5 Warnings 0

分配封装

在对 PCB 进行布线之前，需要为每一个将要装配在电路板上的元件选择封装。封装定义了物理元件和电路板上的布线之间的铜连接。

有些符号预留了封装，但对许多符号来说，有多种可能的封装，所以用户需要选择合适的封装。

KiCad 提供了几种分配封装的方法：

- 符号属性
 - 符号属性对话框
 - 符号字段表
- 在放置符号的同时
- 封装分配工具

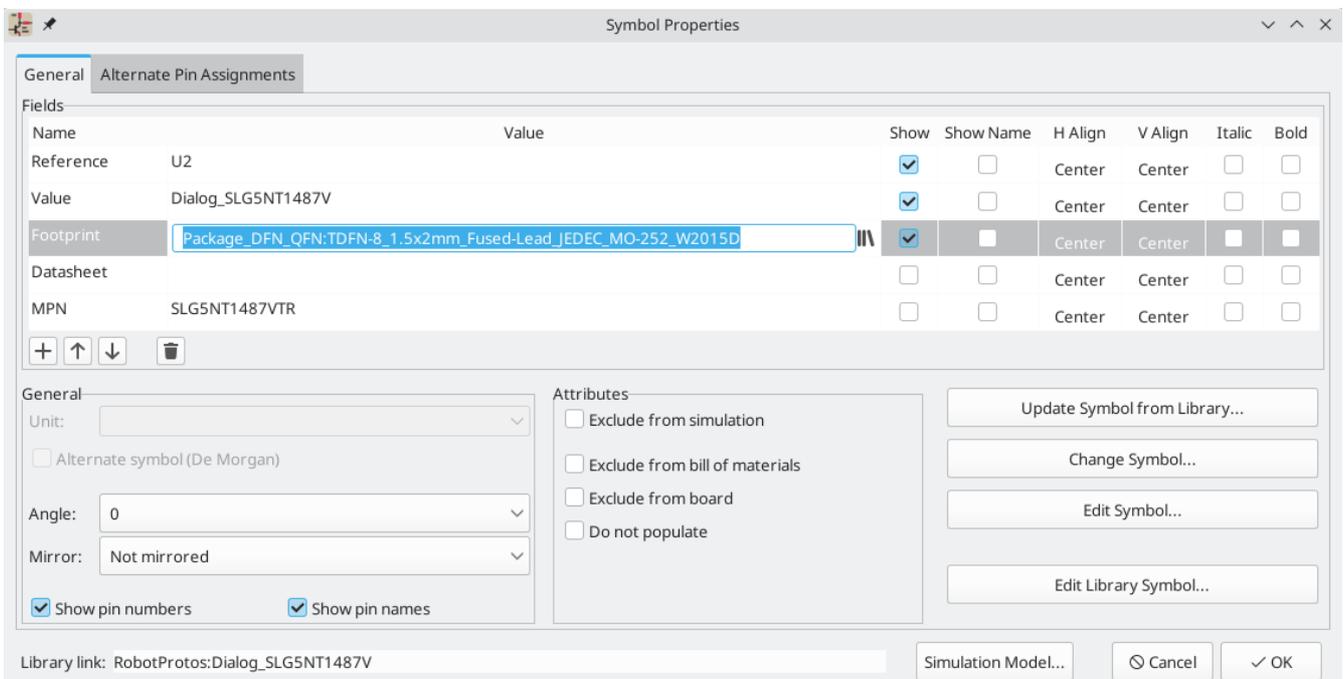
下面将对每种方法进行解释。使用哪种方法是一个偏好的问题；根据情况，一种方法可能更方便。所有这些方法都是等效的，它们在符号的 封装 字段中存储所选封装的名称。

NOTE

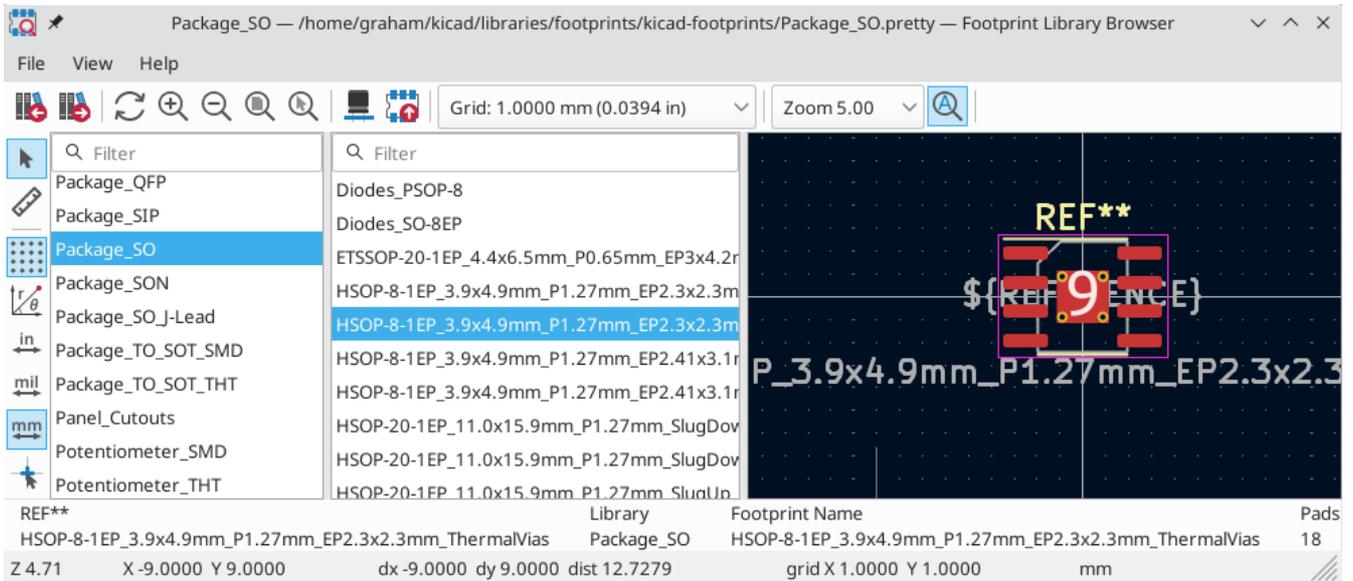
The Footprint Library Table needs to be configured before footprints can be assigned. For information on configuring the Footprint Library Table, please see the [PCB Editor manual](#).

在符号属性中分配封装

符号的 封装 字段可以直接在符号的属性窗口中编辑。



点击  按钮在 封装 字段打开封装库浏览器，显示可用的封装和封装库。单击一个封装名称可以选择该封装，并在右边的预览窗格中显示，而双击一个封装则可以关闭浏览器，并将符号的 封装 字段设置为所选的封装。

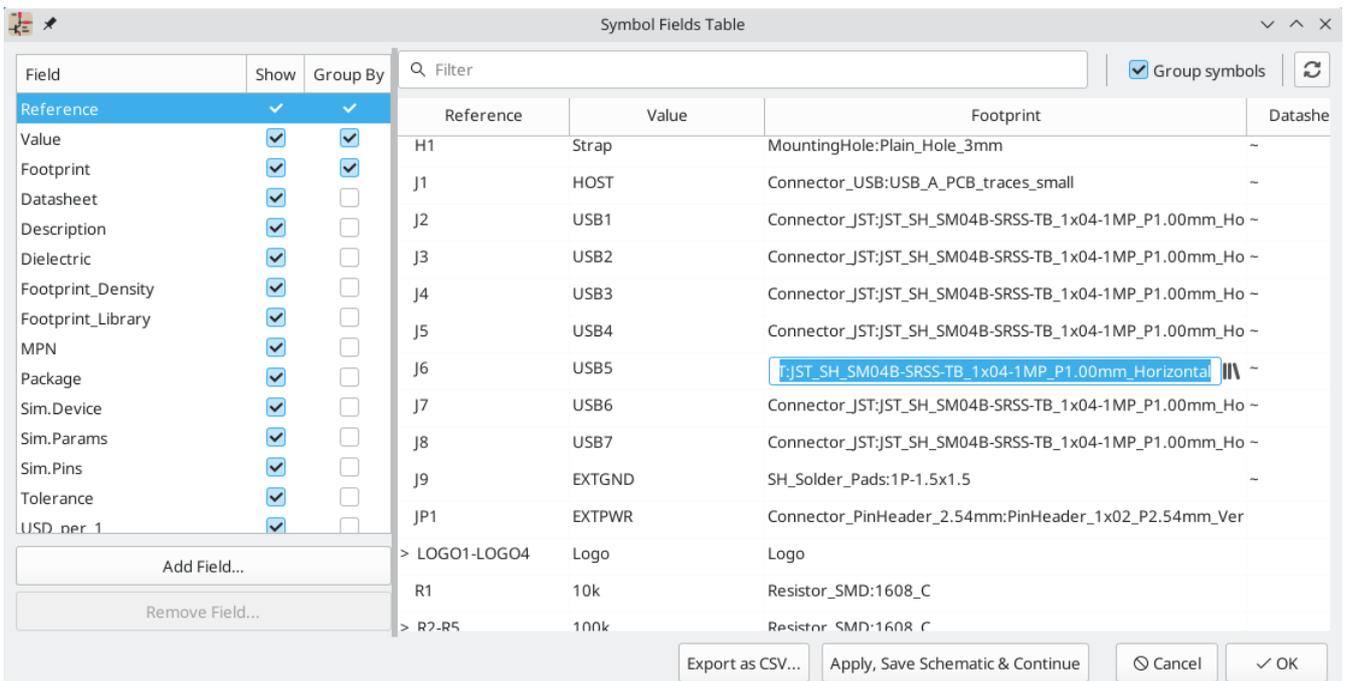


用符号字段表分配封装

与其单独编辑每个符号的属性，符号字段表可以用来在一个地方查看和编辑设计中所有符号的属性。这包括通过编辑每个符号的 `封装` 字段来分配封装。

符号字段表可以通过 **工具** → **编辑符号字段...**，或者通过顶部工具栏上的  按钮进入。

‘封装’字段在这里的作用与符号属性窗口的作用相同：可以直接编辑，也可以用封装库浏览器直观地选择封装。

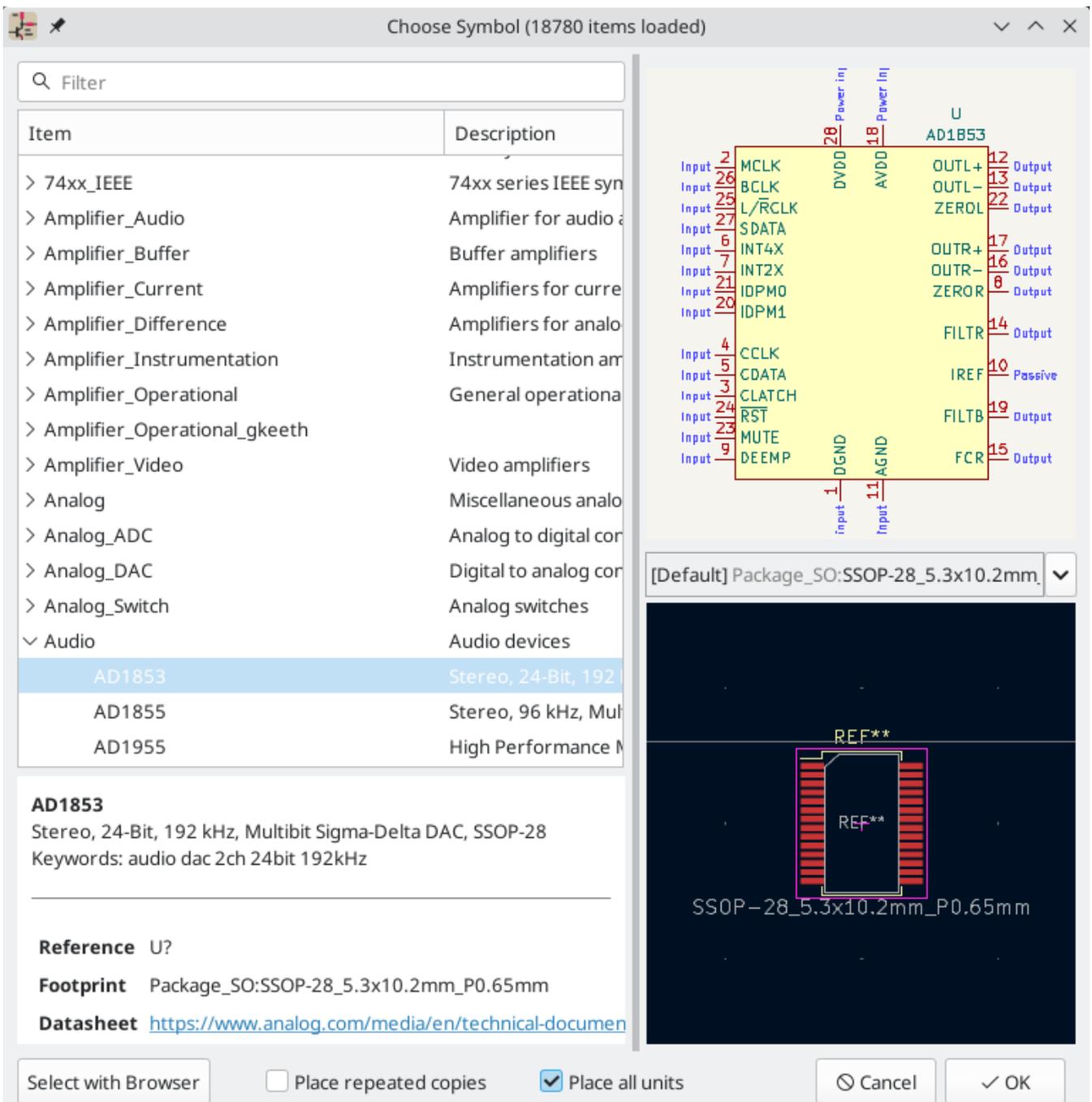


关于符号字段表的更多信息，请参见《符号字段表，关于编辑符号属性》一节。

放置符号时分配封装

当符号第一次被添加到原理图中时，可以为符号分配封装。

有些符号被定义为有一个默认的封装。当这些符号被添加到原理图中时，它们将预先被分配这个封装。默认的封装会显示在添加符号对话框中。对于没有定义默认符号的符号，封装下拉菜单会显示“没有默认封装”，而封装预览画布会显示“没有指定封装”。



符号可以有封装过滤器，指定哪些封装适合与该符号一起使用。如果为选定的符号定义了封装过滤器，那么所有符合封装过滤器的封装都会作为选项出现在封装下拉菜单中。选定的封装将显示在预览画布中，当符号被添加到原理图中时，将被分配给该符号。

NOTE

除非加载封装库，否则封装选项不会出现在封装下拉菜单中。在一个会话中第一次打开“封装编辑器”或“封装库浏览器”时，将加载封装库。

关于封装过滤器的更多信息，请参阅《封装过滤器，符号编辑器文档》。

用封装分配工具分配封装

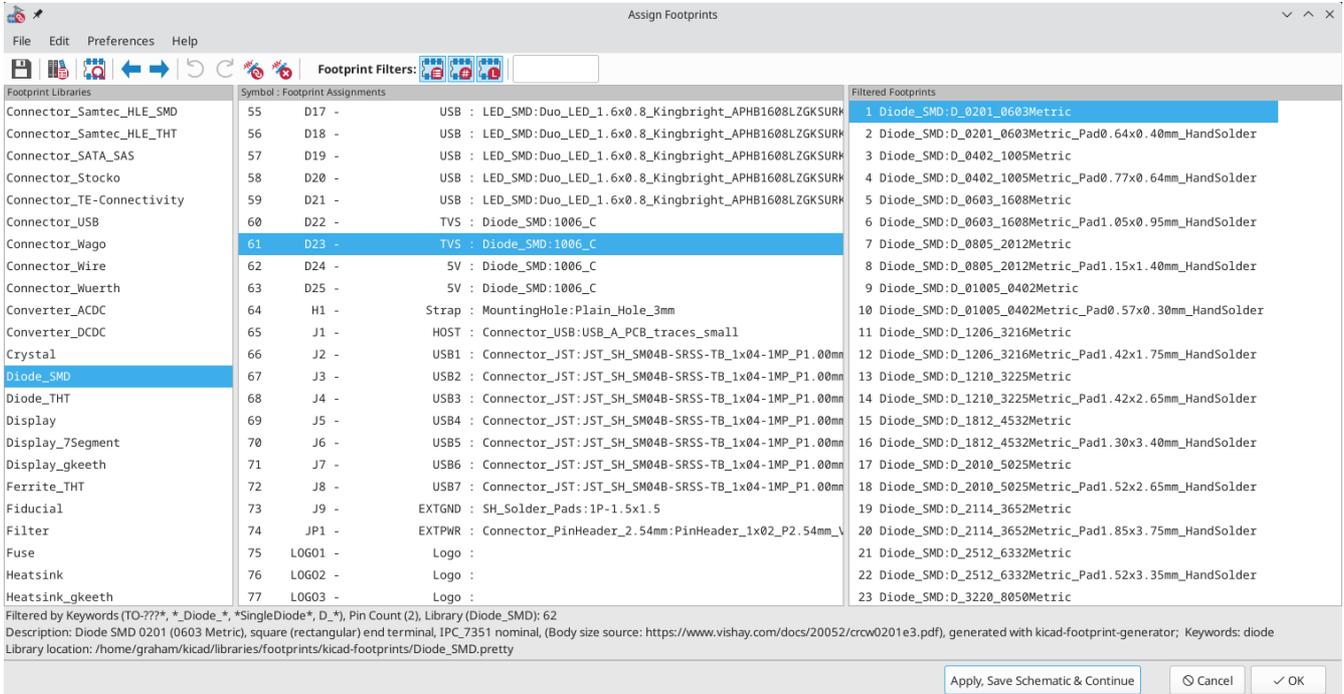
封装分配工具允许你将原理图中的符号与铺设印制电路板时使用的封装联系起来。它提供了封装列表过滤、封装查看和三维元件模型查看，以帮助确保将正确的封装与每个元件联系起来。

元件可以手动或通过创建等效文件（.equ 文件）自动分配到其相应的封装。等效文件是将每个元件与它的封装联系起来的查找表。

用 **工具** → **分配封装...** 来运行该工具，或者点击顶部工具栏上的  图标。

封装分配工具概述

下面的图片显示了封装分配工具的主窗口。



- 左边的窗格包含了与该工程相关的可用封装库的列表。
- 中心窗格包含原理图中的符号列表。
- 右边的窗格包含了从工程封装库加载的可用封装列表。
- 底部的窗格描述了应用于封装列表的过滤器，并打印了在最右边窗格中选择的封装的信息。

顶部的工具条包含以下命令：

	将当前的封装关联转移到原理图中。
	编辑全局和工程封装库表。
	在封装查看器中查看选定的封装。
	选择没有封装关联的前一个符号。
	选择下一个没有封装关联的符号。
	撤消上次的编辑。
	重做最后一次编辑。
	使用等效文件执行自动封装关联
	删除所有的封装分配。
	通过所选符号中定义的封装过滤器来过滤封装列表。
	按所选符号的引脚数过滤封装列表。
	按选定的库过滤封装列表。

下表列出了封装分配工具的键盘命令：

向右箭头/Tab	激活当前激活的窗格右侧的窗格。如果当前激活了最后一个窗格，则绕到第一个窗格。
向左箭头	激活当前激活的窗格左边的窗格。窗格的左边。如果当前激活的是第一个窗格，则绕到最后一个窗格。
向上箭头	选择当前所选列表中的前一个项目。
向下箭头	选择当前选择的列表中的下一个项目。
Page Up	选择当前所选项目向上翻一整页的项目。
Page Down	选择当前所选项目向下一整页的项目。
Home	选择当前选择的列表中的第一个项目。
End	选择当前所选列表的最后一个项目。

使用 "封装分配工具" 手动分配 "封装"

要手动将一个封装与一个元件联系起来，首先在元件（中间）窗格中选择一个元件。然后在封装（右）窗格中选择一个封装，双击所需封装的名称。封装将被分配给选定的元件，而下一个没有分配封装的元件将被自动选择。

NOTE

如果没有封装出现在封装窗格中，检查《过滤封装列表，封装过滤选项》是否正确应用。

当所有的元件都有封装时，点击 **OK** 按钮来保存分配并退出工具。或者，点击 **取消**，放弃更新的赋值，或者点击 **应用，保存原理图并继续**，保存新的赋值，而不退出工具。

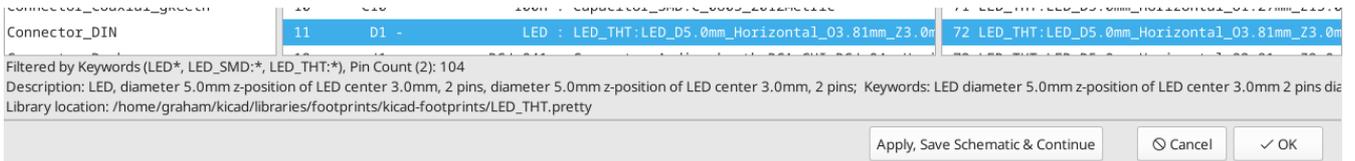
筛选 "封装" 列表

有四个过滤选项可以限制哪些封装显示在封装窗格中。过滤选项是通过顶部工具栏的三个按钮和一个文本框来启用和禁用的。

- ：激活《封装过滤器，过滤器可以在每个符号中定义》。例如，一个运算放大器符号可以定义过滤器，只显示 SOIC 和 DIP 封装。
- ：只显示符合所选符号引脚数的封装。
- ：只显示左侧窗格中选择的库的封装。
- 在文本框中输入文本，会隐藏与文本不匹配的封装。当文本框为空时，该过滤器被禁用。

当所有的过滤器被禁用时，会显示完整的封装列表。

窗口的底部窗格中描述了应用的过滤器，以及符合所选过滤器的封装数量。例如，当符号的封装过滤器和引脚数过滤器被启用时，底部窗格会打印封装过滤器和引脚数：



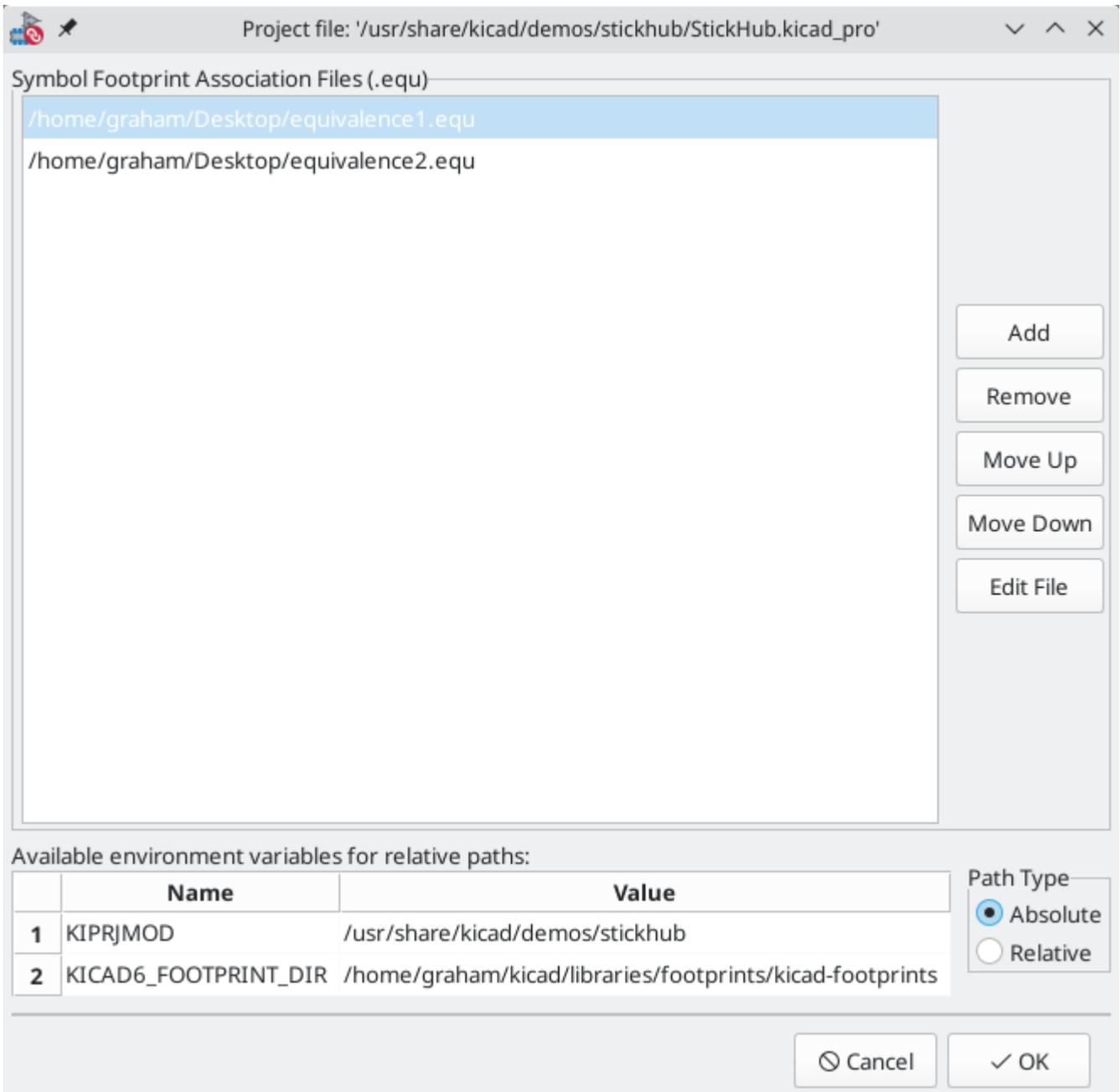
可以同时使用多个过滤器来帮助缩小封装窗格中可能合适的封装列表。KiCad 标准库中的符号定义了封装过滤器，旨在与引脚数过滤器结合使用。

使用封装分配工具自动分配封装

封装分配工具允许你在一个外部文件中存储封装分配，并在以后加载这些分配，甚至在不同的工程中。这使你能够自动将符号与适当的封装联系起来。

外部文件被称为等效文件，它存储了一个符号值到相应的封装的映射。等效文件通常使用 `.equ` 文件扩展名。等效文件是具有简单语法的纯文本文件，必须由用户用文本编辑器来创建。语法描述如下。

你可以在 "封装分配工具" 中点击 **偏好设置** → **管理 "封装关联文件"** 来选择使用哪些等值文件。



- 通过点击 **添加** 按钮添加新的等效文件。
- 点击 **删除** 按钮，删除所选的等效文件。
- 通过点击 **上移** 和 **下移** 按钮改变等效文件的优先级。如果一个符号的值在多个等效文件中被发现，最后一个匹配的等效文件的封装将覆盖早期的等效文件。
- 点击 **编辑文件** 按钮，打开所选的等值文件。

相关的环境变量显示在窗口的底部。当 **相对** 路径选项被选中时，这些环境变量将被自动用于使所选等效文件的路径与项目或封装库相对。

一旦所需的等效文件以正确的顺序载入，就可以通过点击封装分配工具顶部工具栏上的  按钮来进行自动封装分配。

所有在加载的等效文件中找到数值的符号都将自动分配其封装。然而，已经分配了封装的符号将不会被更新。

等效文件格式

等效文件由每个符号值的一行组成。每一行有以下结构：

```
'<symbol value>' '<footprint library>:<footprint name>'
```

每个名称/值必须用单引号 (') 包围，并由一个或多个空格隔开。以 # 开头的行是注释。

例如，如果你想让所有数值为 LM4562 的符号被分配到 Package_S0:SOIC-8_3.9x4.9_P1.27mm 的封装，等效文件中的一行应该是：

```
'LM4562' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
```

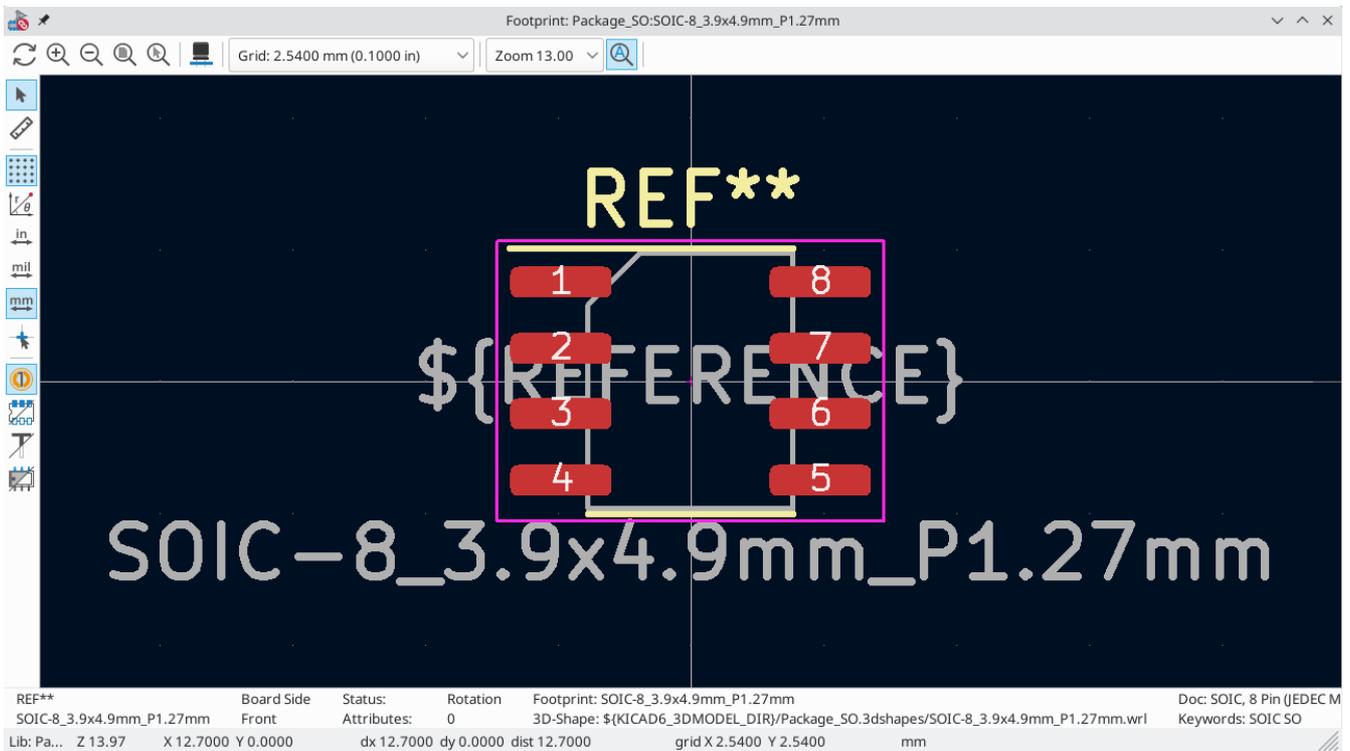
下面是一个等效文件的例子：

```
#integrated circuits (smd):
'74LV14' 'Package_S0:SOIC-14_3.9x8.7mm_P1.27mm'
'EL7242C' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'DS1302N' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LM324N' 'Package_S0:SOIC-14_3.9x8.7mm_P1.27mm'
'LM358' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LTC1878' 'Package_S0:MSOP-8_3x3mm_P0.65mm'
'24LC512I/SM' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LM2903M' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LT1129_S08' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LT1129CS8-3.3' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LT1129CS8' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'LM358M' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'TL7702BID' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'TL7702BCD' 'Package_S0:SOIC-8_3.9x4.9_P1.27mm'
'U2270B' 'Package_S0:SOIC-16_3.9x9.9_P1.27mm'

#regulators
'LP2985LV' 'Package_T0_S0T_SMD:SOT-23-5_HandSoldering'
```

查看当前封装

封装分配工具包含一个封装查看器。点击顶部工具栏上的  按钮可以启动封装查看器，并显示选定的封装。



顶部的工具条包含以下命令：

	刷新视图
	放大
	缩小
	放大到适合显示区域的图画
	显示 3D 查看器

左边的工具条包含以下命令：

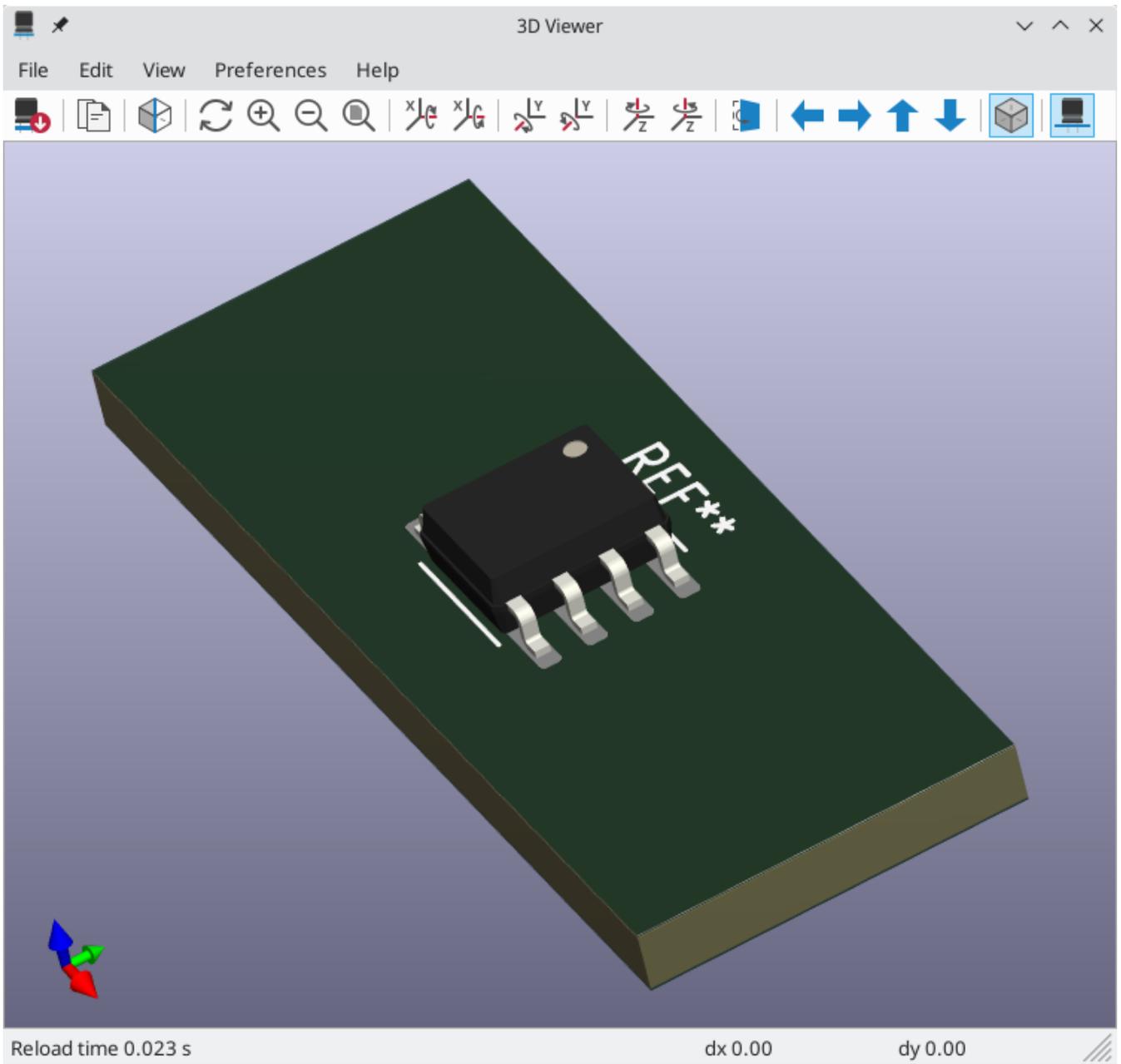
	使用选择工具
	在两点之间交互测量
	显示网格点或线
	在极地坐标系和车轴坐标系之间切换
	使用英寸
	以密尔（1/1000英寸）为单位显示坐标
	以毫米为单位显示坐标
	切换全窗口十字准线的显示
	在草图或正常模式下的绘图板之间进行切换
	在正常模式或轮廓模式下的绘图板之间进行切换
	在正常模式或大纲模式下绘制文本之间进行切换
	在正常模式或大纲模式下绘制图形线之间进行切换

查看当前 3D 模型

单击  按钮在 3D 模型查看器中打开封装。

NOTE

如果当前的封装不存在 3D 模型，那么在 3D 浏览器中只显示封装本身。



The 3D Viewer is described in the [PCB Editor manual](#).

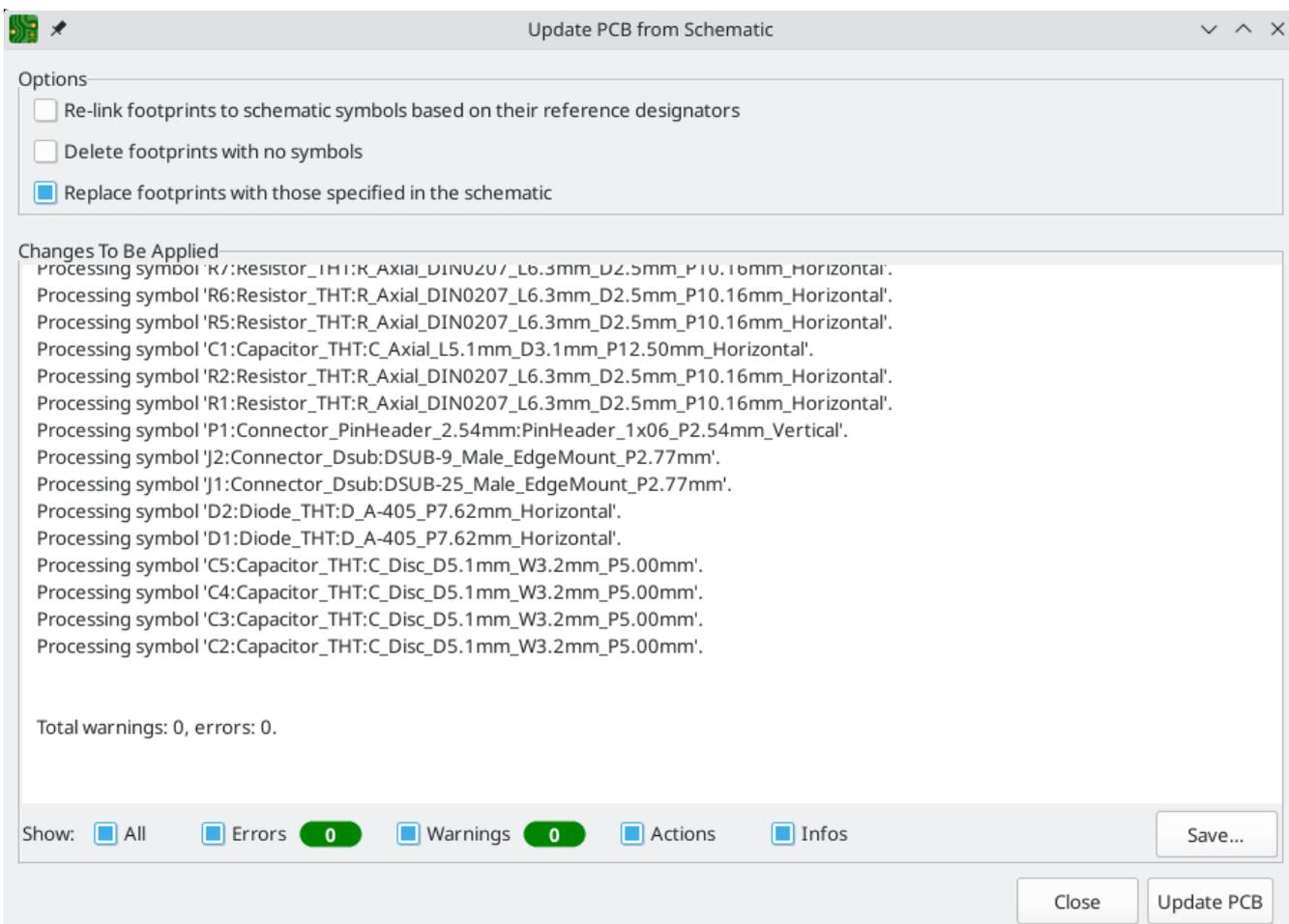
正向和反向批注

从原理图更新 PCB（正向批注）

使用“从原理图更新 PCB”工具将设计信息从原理图编辑器同步到电路板编辑器。在原理图编辑器和电路板编辑器中都可以用 **工具** → **从原理图更新 PCB** (F8) 来访问该工具。你也可以使用电路板编辑器顶部工具栏上的  图标。这个过程通常被称为正向批注。

NOTE

从原理图更新 PCB 是将设计信息从原理图转移到 PCB 的首选方法。在旧版本的 KiCad 中，相应的过程是将网表从原理图编辑器中导出并导入到电路板编辑器中。现在已经没有必要使用网表文件了。



该工具将每个符号的封装添加到电路板上，并将更新的原理图信息传输到电路板上。特别是，电路板的网络连接会被更新以符合原理图的要求。带有《编辑符号属性，排除在电路板属性之外》的符号不会被转移到 PCB 上。

将对 PCB 进行的修改列在 *待应用的修改* 窗格中。在你点击 **更新 PCB** 按钮之前，PCB 不会被修改。

你可以使用窗口底部的复选框来显示或隐藏不同类型的信息。可以使用 **保存...** 按钮将更改的报告保存到文件中。

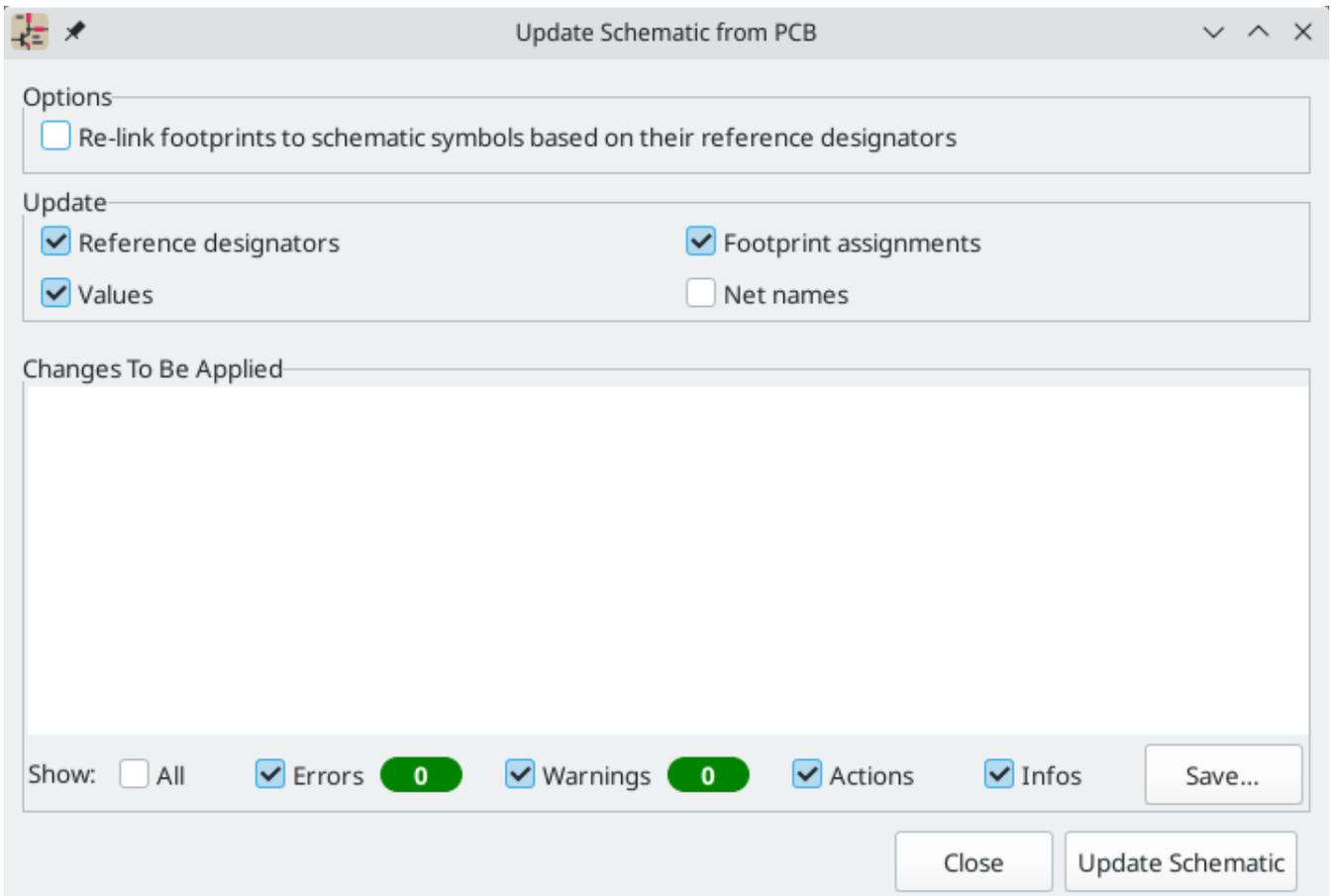
选项

该工具有几个选项可以控制其行为。

选项	说明
根据位号将封装重新链接到原理图符号上	<p>封装通常是通过符号添加到原理图中时创建的唯一标识符与原理图符号相连。符号的唯一标识符不能被改变。</p> <p>如果选中，PCB 中的每个封装将被重新链接到与该封装具有相同位号的符号上。</p> <p>如果不勾选，封装和符号将像往常一样通过唯一的标识符连接，而不是通过位号。每个封装的位号将被更新以匹配其链接符号的位号。</p> <p>这个选项一般不应该被选中。它对于依赖改变原理图符号和封装之间的联系的具体工作流程是很有用的，例如重构原理图以方便布局，或者在设计中的相同通道之间复制布局。</p>
删除没有符号的封装	<p>如果选中，PCB 中任何在原理图中没有相应符号的封装都将从 PCB 中删除。带有 "不在原理图中" 属性的封装将不受影响。</p> <p>如果不勾选，没有相应符号的封装将不会被删除。</p>
用原理图中指定的封装替换封装	<p>如果选中，PCB 中的封装将被替换成相应原理图符号中指定的封装。</p> <p>如果不勾选，即使原理图符号被更新以指定不同的封装，PCB 中已经存在的封装也不会被改变。</p>

从 PCB 上更新原理图（反向批注）

KiCad 的典型工作流程是在原理图中进行修改，然后使用 "从原理图更新 PCB" 工具将修改内容同步到电路板上。然而，相反的过程也是可行的：可以在电路板上进行设计修改，然后在原理图或电路板编辑器中使用 **工具** → **从 PCB 更新原理图** 同步回原理图。这个过程也被称为 "反向批注"。



该工具将位号、数值、封装分配和网络名称的变化从电路板同步到原理图。每种类型的变化都可以单独启用或禁用。

将对原理图进行的修改列在 *待应用的修改* 窗格中。在您点击 **更新原理图** 按钮之前，原理图不会被修改。

您可以使用窗口底部的复选框来显示或隐藏不同类型的信息。可以使用 **保存...** 按钮将更改的报告保存到文件中。

选项

该工具有几个选项可以控制其行为。

选项	说明
根据位号将封装重新链接到原理图符号上	<p>如果选中，PCB 中的每个封装将被重新链接到与该封装具有相同位号的符号上。这个选项与更新符号位号不兼容。</p> <p>如果不勾选，封装和符号将像往常一样通过唯一的标识符连接，而不是通过位号。</p>
位号	<p>如果选中，符号位号将被更新，以匹配链接封装的位号。</p> <p>如果不勾选，符号位号将不会被更新。</p>
值	<p>如果选中，符号值将被更新以匹配链接封装的值。</p> <p>如果不勾选，符号值不会被更新。</p>
封装分配	<p>如果选中，符号的封装分配将被更新，这些符号的封装在电路板上被改变或替换。</p> <p>如果不勾选，符号的封装分配将不会被更新。</p>
网络名	<p>如果选中，原理图将根据电路板上的任何网络名变化进行更新。如有必要，网络标签将被更新或添加到原理图中，以符合电路板的要求。</p> <p>如果不勾选，网络名将不会在原理图中更新。</p>

NOTE

The [Geographical Reannotation](#) feature can be used in combination with backannotating reference designators to reannotate all components in the design based on their location in the layout.

用 CMP 文件进行反向批注

通过从 PCB 编辑器中导出 CMP 文件（**文件** → **导出** → **封装关联 (.cmp) 文件...**）并在原理图编辑器中导入（**文件** → **导入** → **封装分配...**），也可以将选择变化从 PCB 同步到原理图。

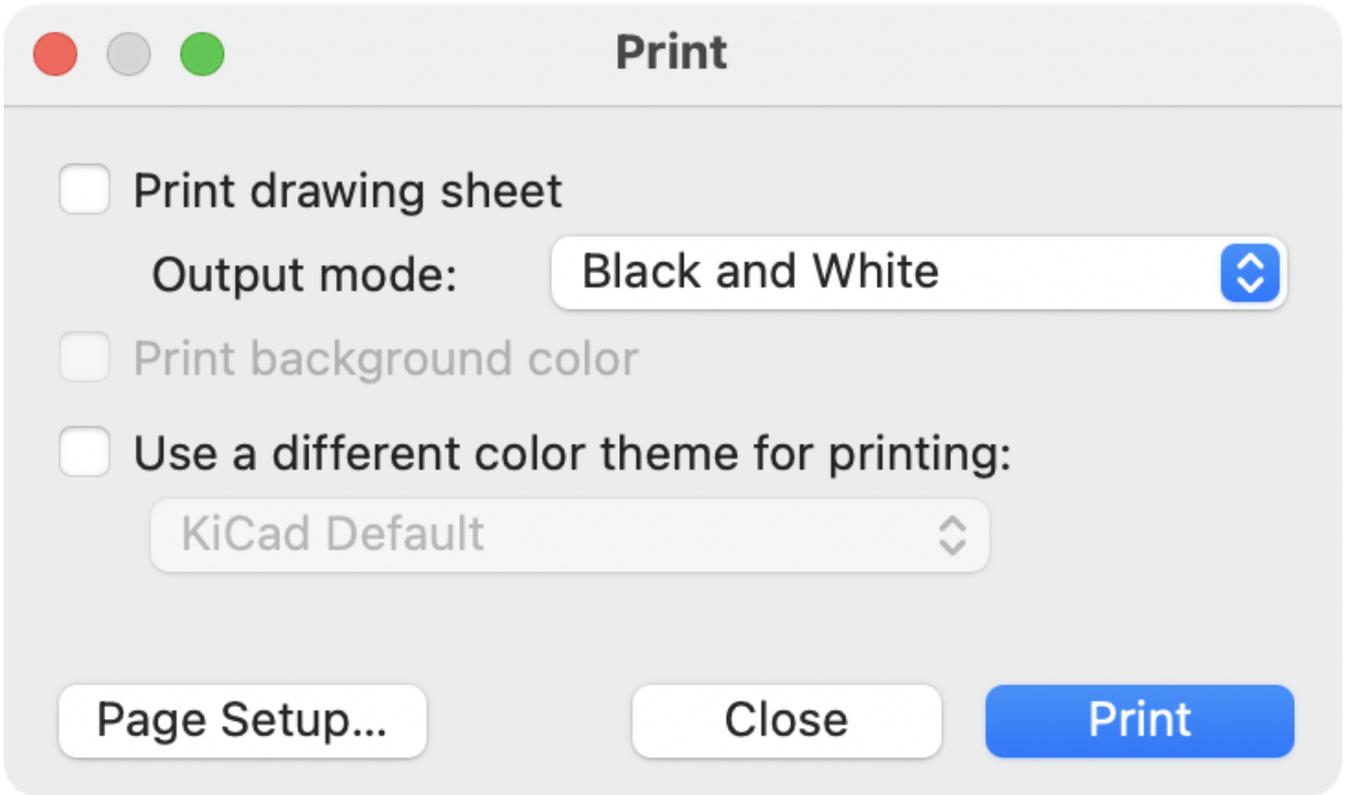
NOTE

这种方法只能同步对封装分配和封装字段的修改。建议使用“从 PCB 更新原理图”工具来代替。

生成输出

打印

KiCad 可以使用 **文件** → **打印...** 将原理图打印到标准打印机上。



打印设置

打印图样： 在打印的原理图中包括图样的边框和标题块。

输出模式： 以彩色或黑白方式打印原理图。

打印背景色： 包括打印的原理图中的背景色。包括在打印的原理图中的背景颜色。这个选项只有在 **只以黑白打印** 被禁用时才会启用。禁用此选项。

使用不同的颜色主题进行打印： 选择不同的颜色方案进行打印，与在原理图编辑器中选择的显示方案不同。与在 "原理图编辑器" 中显示时选择的颜色方案不同，用于打印。

页面设置...：打开一个页面设置对话框，用于设置纸张大小和方向。

预览： 打开一个打印预览对话框。

关闭： 关闭对话框，不打印。

打印： 打开系统打印对话框。

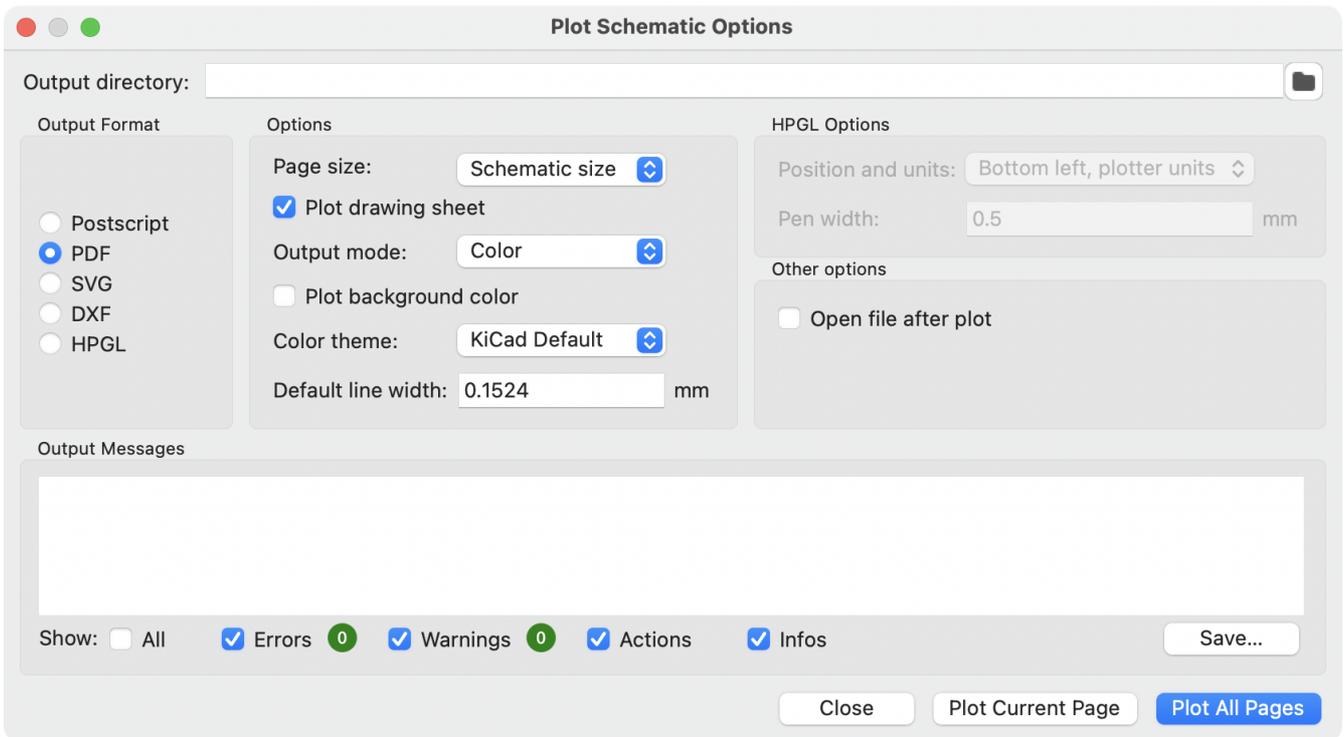
NOTE

打印使用平台和打印机特定的驱动程序，可能会产生意想不到的结果。当打印到一个文件时，建议使用 **绘图** 而不是 **打印**。

绘制

KiCad 可以使用 **文件** → **绘图...** 将原理图绘制到文件中。

支持的输出格式是 Postscript、PDF、SVG、DXF 和 HPGL。



输出信息 窗格显示有关生成文件的信息。可以用复选框显示或隐藏不同类型的信息，还可以用 **保存...** 按钮将信息保存到文件中。

图示当前页 按钮绘制原理图的当前页。**绘制全部页面** 按钮可以绘制原理图的所有页面。每一页都会生成一个文件，但 PDF 输出除外，它将原理图的每一页作为单独的一页绘制在一个 PDF 文件中。

绘制选项

输出目录：指定保存绘图文件的位置。如果这是一个相对路径，它是相对于项目目录创建的。

输出格式：选择要绘制的格式。有些格式有不同的选项。

页面大小：设置用于绘制输出的页面大小。这可以被设置为匹配原理图的尺寸或其他页面尺寸。

绘制图纸：在打印的原理图中包括图样表边框和标题块。

输出模式：设置输出为彩色或黑白。不是所有的输出格式都支持彩色。

绘图背景色：包括绘图输出中的原理图背景色。绘制的输出中。如果输出格式不支持彩色，或者输出模式是黑白的，那么背景色将不会被绘制出来。不支持彩色，或输出模式是黑白的，则不会绘制背景色。

颜色主题：选择用于绘制输出的颜色主题。

默认线宽：选择没有指定厚度的线的默认宽度。厚度的线的默认宽度（厚度设置为 0 的线）。有设定厚度的线的线条将以该厚度绘制。

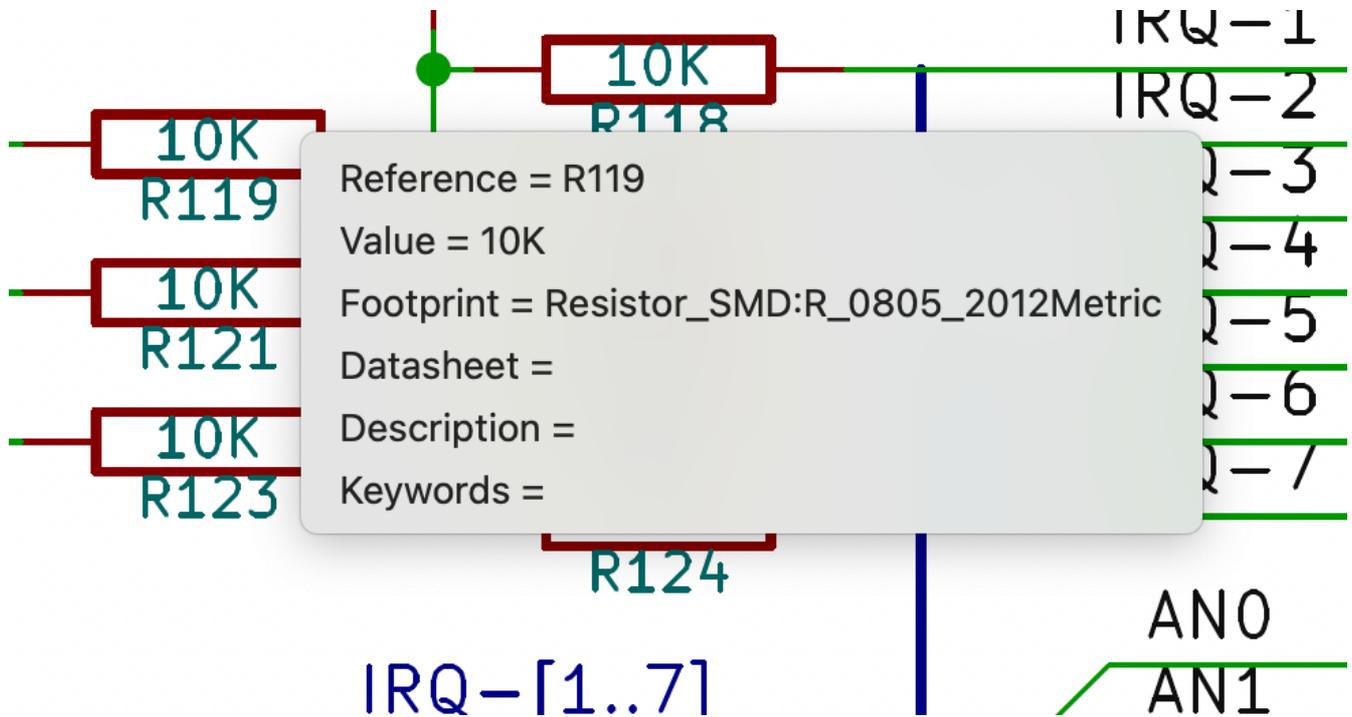
位置和单位：设置绘图仪原点和单位。这个选项只适用于 HPGL 输出。

笔宽： 设置绘图仪的笔宽。这个选项只适用于 HPGL 输出。

绘制后打开文件： 在绘制完成后自动打开绘制的输出文件。

互动式 PDF 功能

绘图的 PDF 有几个互动功能。

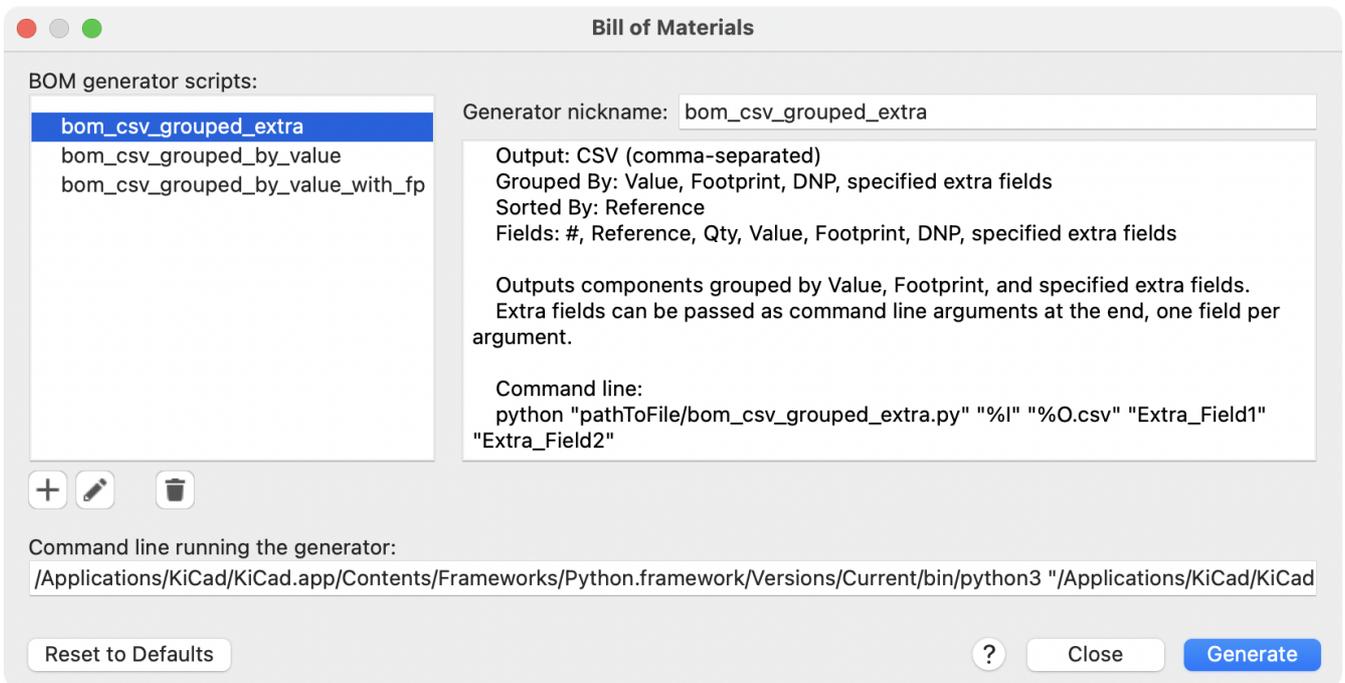


- 可以点击超链接。
- 目录中填充了原理图页以及每张页面中的符号和层次标签。
- 点击许多原理图元素会显示一个包含相关信息的弹出菜单。
 - 符号显示其符号字段。
 - 层次的子页显示它们的原理图页面名称和文件名，以及一个输入页面本身的选项。
 - 标签显示已解决的网络和网络类。
 - 总线显示其成员。

NOTE | 其中一些功能并非所有 PDF 阅读器都支持。

生成物料清单

KiCad 可以使用 **工具** → **生成 BOM...** 或顶部工具栏上的  按钮生成材料清单。生成的 BOM 列出了设计中的所有元件，除了设置了《编辑符号属性，从物料清单中排除属性》的元件。



BOM 工具使用一个外部脚本将设计信息处理成所需的输出格式。KiCad 中包含几个 BOM 生成器脚本，用户也可以创建自己的脚本。BOM 生成器脚本一般使用 Python 或 XSLT，但也可以使用其他工具，只要你能指定一个《生成器命令行格式，命令行》让 KiCad 在运行生成器时执行。

您可以在 **BOM 发生器脚本** 列表中选择要使用的 BOM 发生器。对话框的其余部分显示所选生成器的信息。您可以用 **生成器昵称** 文本框来改变生成器的显示名称。

右边的窗格显示所选脚本的信息。当生成器被执行时，右边的窗格则显示脚本的输出。

底部的文本框包含了 KiCad 用来执行生成器的命令。当选择脚本时，该文本框会自动填充，但对于某些生成器来说，该命令可能需要手工编辑。当关闭 BOM 工具时，KiCad 会保存每个生成器的命令行，所以命令行的定制会被保留下来。关于命令行的更多细节，请参阅《生成器命令行格式，高级文档》。

在 Windows 下，BOM 生成器对话框有一个额外的选项 **显示控制台窗口**。当这个选项未被选中时，BOM 生成器在一个隐藏的控制台窗口中运行，任何输出都会被重定向并打印在对话框中。当该选项被选中时，BOM 生成器在一个可见的控制台窗口中运行，如果生成器插件提供了一个图形用户界面，这可能是必要的。

BOM 生成器脚本

默认情况下，BOM 工具呈现三个输出脚本选项。

- `bom_csv_grouped_extra` 输出一个 CSV 文件，其中包含设计中的每个元件。元件按值、封装、DNP（不使用）和命令行中指定的任何额外字段分组。要指定额外的字段，在命令行的末尾添加所需的字段名作为引号字符串。例如，要包括 MPN 字段，命令行的结尾应该是 `<脚本路径>/bom_csv_grouped_extra.py "%I" "0.csv" "MPN"`。BOM 中的列是：
 - 项目编号
 - 位号
 - 数量
 - 值
 - 封装

DNP

- 指定的额外字段
- `bom_csv_grouped_by_value` 输出一个有两个部分的 CSV。第一部分包含了设计中的每个元件，每行都有一个元件。第二部分也包含每个元件，但元件是按符号名称、值、封装和 DNP（不使用）分组的。BOM 中的列是：
 - 项目编号
 - 数量
 - 位号
 - 值
 - 符号库和符号名称
 - 封装
 - 数据手册
 - DNP
 - 任何其他符号字段
- `bom_csv_grouped_by_value_with_fp` 输出一个 CSV，其中有一个单独的部分，包含设计中的每个元件。元件按值、封装和 DNP（不使用）进行分组。BOM 中的列是：
 - 位号
 - 数量
 - 值
 - 符号名称
 - 封装
 - 符号描述
 - 供应商
 - DNP

额外的生成器脚本与 KiCad 一起安装，但默认情况下不会在生成器脚本列表中填入。这些脚本的位置取决于操作系统，并可能根据安装位置而有所不同。

运行系统	位置
Windows	C:\Program Files\KiCad\6.0\bin\scripting\plugins\
Linux	/usr/share/kicad/plugins/
macOS	/Applications/KiCad/KiCad.app/Contents/SharedSupport/plugins/

通过点击  按钮，可以在 BOM 生成器脚本列表中添加其他脚本。点击  按钮，可以删除脚本。 按钮在文本编辑器中打开所选的脚本。

关于创建和使用自定义 BOM 生成器的更多信息，请参见《自定义网表和 BOM 格式，高级文档》。

生成 BOM 的其他方法

KiCad 提供了几种额外的方法来生成 BOM，尽管推荐使用 BOM 工具。

- 《符号字段表，符号字段表》可以导出一个 CSV，可以作为 BOM 使用。这个选项包括所有的符号信息，但对输出的控制有限。
- PCB 编辑器可以通过 **文件** → **制造输出** → **BOM...** 输出 BOM。这种方法不提供对输出格式的控制，也不包括所有符号信息。

生成网表

网表是一个描述符号引脚之间电气连接的文件。这些连接被称为网路。网表文件包含：

- 符号和它们的引脚的列表。
- 符号引脚之间的连接（网络）列表。

存在许多不同的网表格式。有时符号表和网表是两个独立的文件。这个网表是使用原理图设计软件的基础，因为网表是与其他电子 CAD 软件，如 PCB 布局软件、仿真器和可编程逻辑编译器的联系。

KiCad 支持几种网表格式：

- KiCad 格式，可由 KiCad PCB 编辑器导入。然而，应该使用《eeschema_schematic_to_pcb.adoc#原理图到 PCB，"从原理图更新 PCB"》工具而不是将 KiCad 网表导入 PCB 编辑器。
- OrCAD PCB2 格式，用于用 OrCAD 设计 PCB。
- CADSTAR 格式，用于用 CADSTAR 设计 PCB。
- Spice 格式，用于各种外部电路仿真器。

NOTE

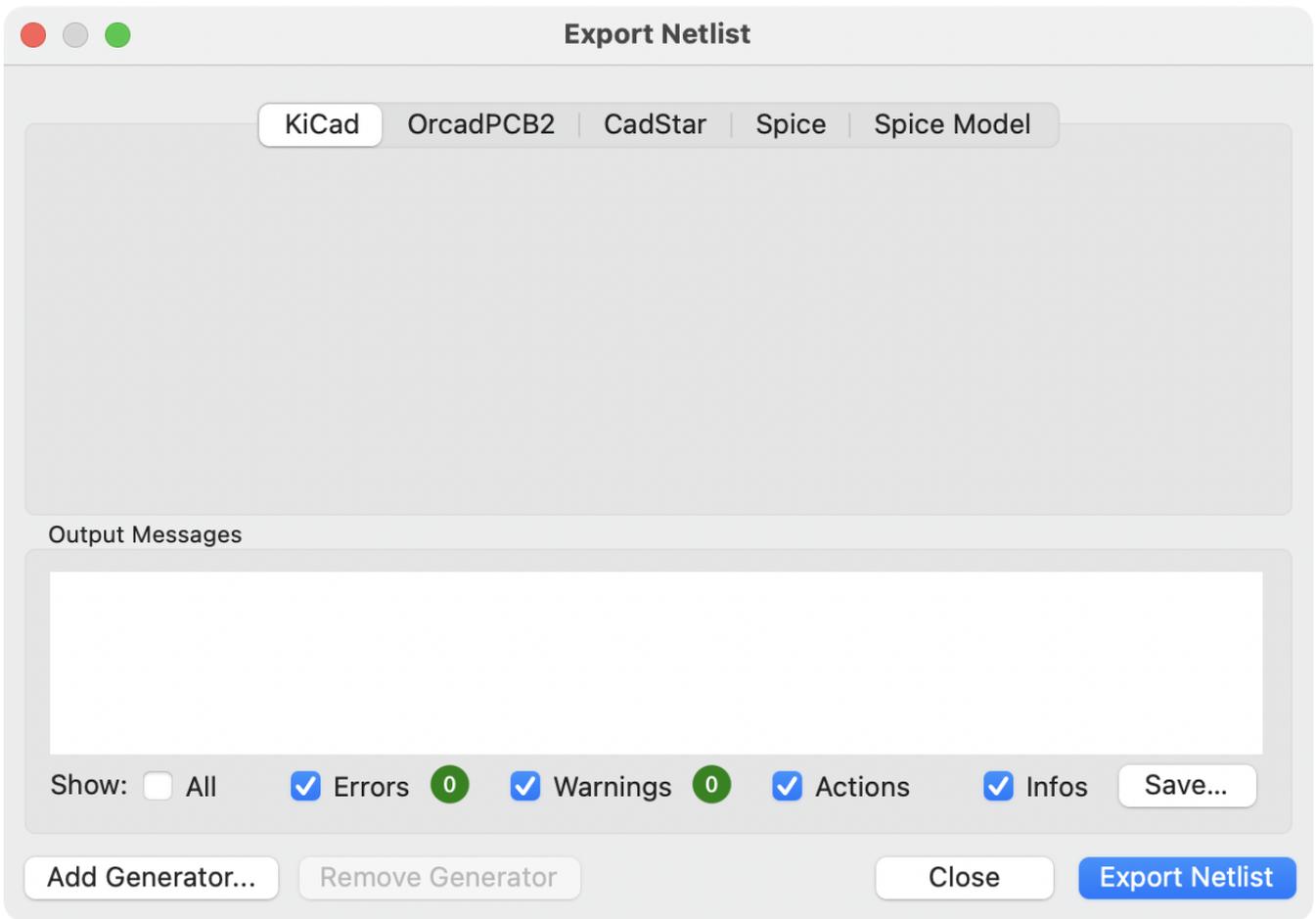
在 KiCad 5.0 及以后的版本中，将设计从原理图编辑器转移到 PCB 编辑器时，没有必要创建网表。相反，使用《eeschema_schematic_to_pcb.adoc#schematic-to-pcb，"根据原理图更新 PCB"》工具。

NOTE

其他使用网表的软件工具可能对元件名称中的空格和特殊字符有限制。在元件名称、引脚、网络和其他字段中的空格和特殊字符的限制。为了兼容性，请注意你计划使用的其他工具中的这种限制，并相应地命名元件、网络等。

网表格式

网表是通过导出网表对话框（**文件** → **导出** → **网表...**）导出的。



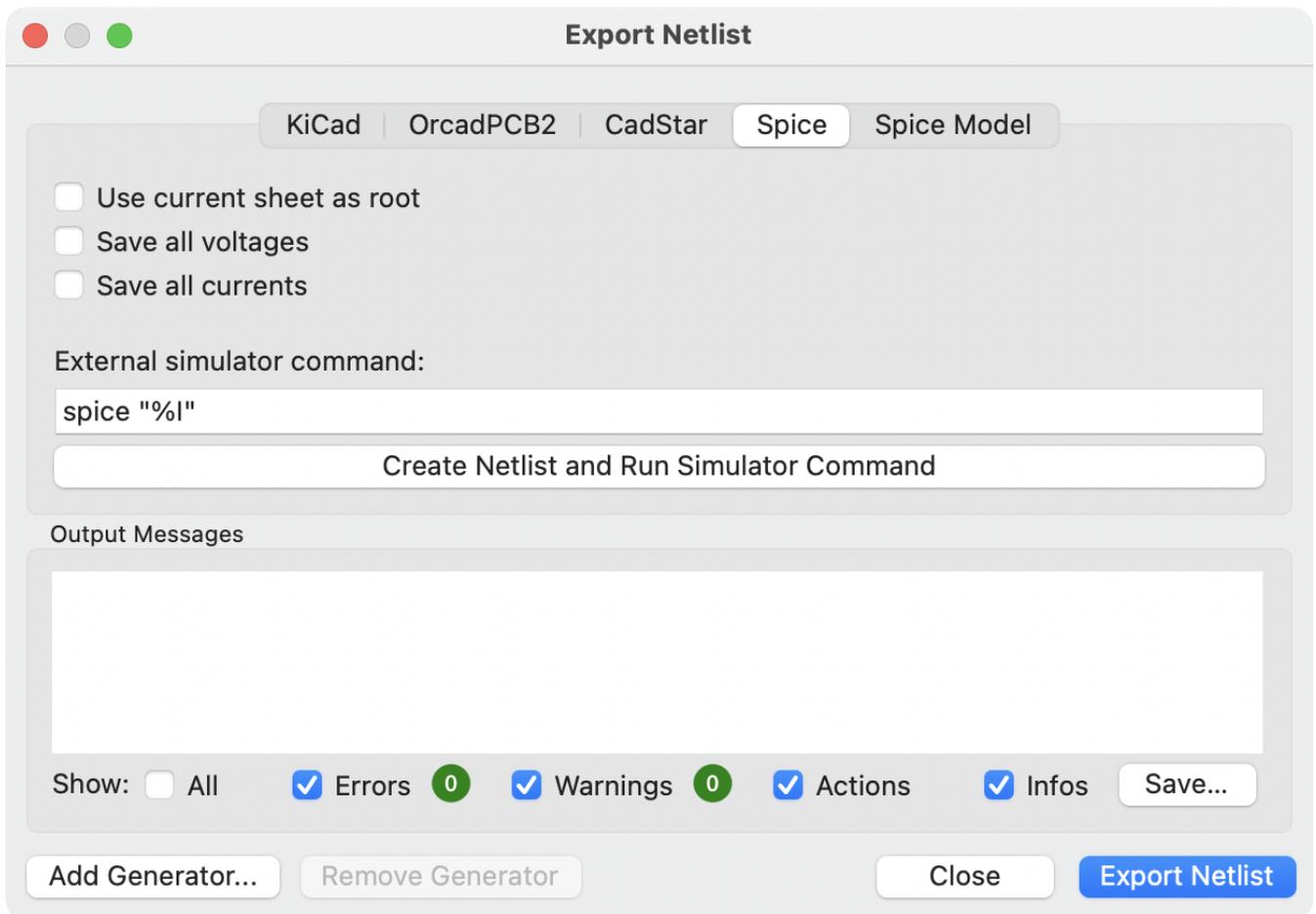
KiCad 支持以多种格式导出网表。KiCad、OrcadPCB2、CADSTAR、Spice 和 Spice Model。每种格式都可以通过选择窗口顶部的相应标签来选择。一些网表格式有额外的选项。

点击 **导出网表** 按钮，会提示输入网表文件名并保存网表。

NOTE | 对于大型原理图，网表的生成可能需要数分钟。

其他网表格式的自定义生成器可以通过点击 **添加生成器...** 按钮添加。自定义生成器是由 KiCad 调用的外部工具，例如 Python 脚本或 XSLT 样式表。关于自定义网表生成器的更多信息，请参见《自定义网表和 BOM 格式，添加自定义网表生成器》部分。

Spice 网表格式



Spice 网表格式提供了几个选项。

当选择 **使用当前页面作为根页面** 时，只有当前页面被导出到子电路模型中。否则，整个原理图页面将被导出。

保存所有电压 选项为网表添加了一个 `.save all` 命令，这使得模拟器保存所有节点的电压。**保存所有电流** 选项在网表中增加了一个 `.probe all` 命令，使模拟器保存所有节点的电流。

NOTE | 确切的行为在不同的仿真工具之间可能有所不同。

无源符号值会自动调整，以便与各种 Spice 仿真器兼容。具体而言：

- μ 和 M 作为单位前缀，分别被替换成 u 和 Meg
- 单位被删除（例如，4.7k Ω 被改为 4.7k）
- RKM 格式的数值被改写为与 Spice 兼容（例如，4u7 被改成 4.7u）

Spice 网表导出器还提供了一种简单的方法，可以用外部仿真器对生成的网表进行仿真。这对于在不使用《仿真器，KiCad 的内部 ngspice 仿真器》的情况下运行仿真，或者使用 KiCad 的仿真器工具不支持的选项运行 ngspice 仿真是很有用的。

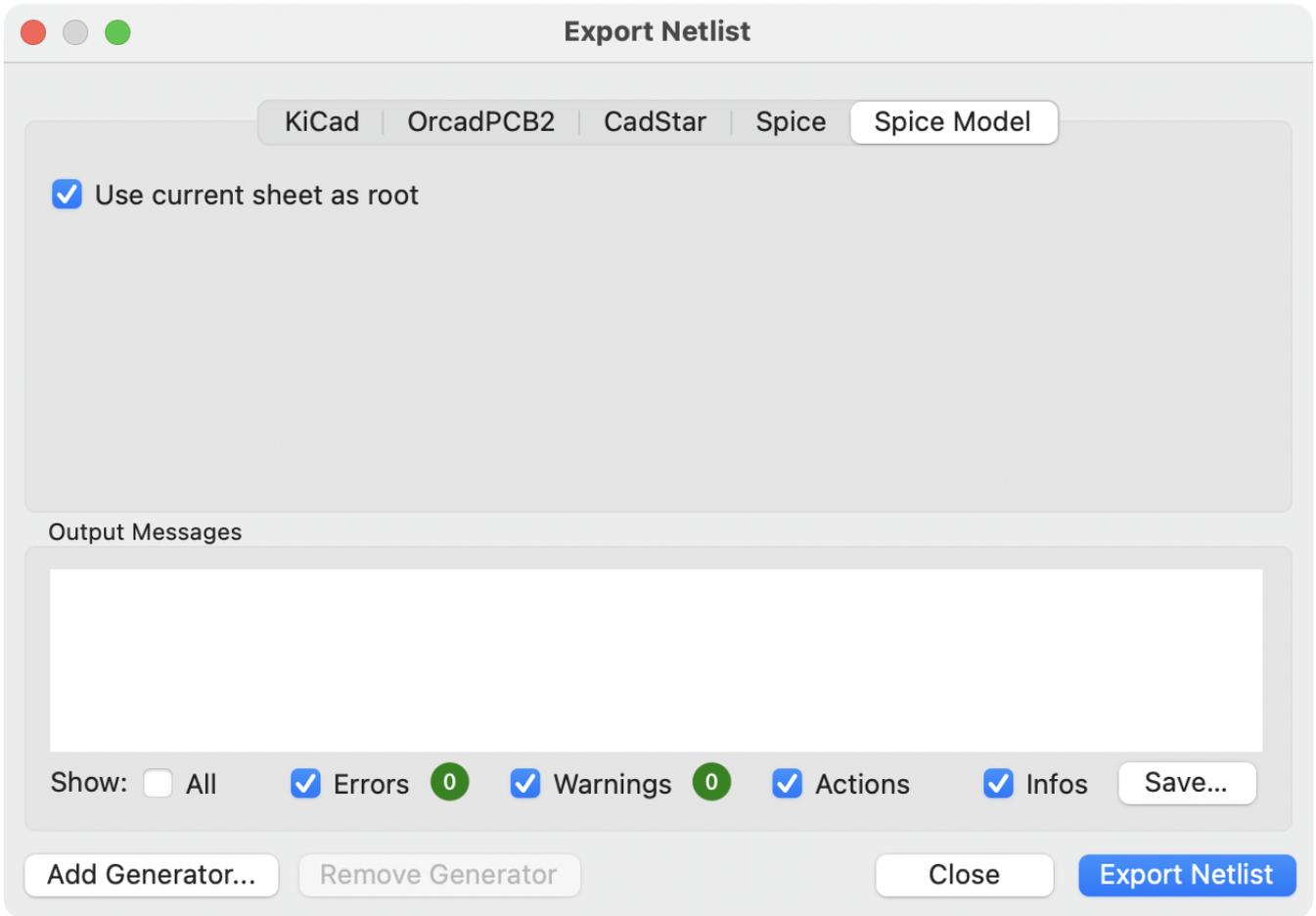
在文本框中输入外部模拟器的路径，其中 %I 代表生成的网表。点击 **创建网表和运行仿真器命令** 按钮，生成网表并自动运行仿真器。

NOTE | 默认的仿真器命令（`spice "%I"`）必须调整为指向你系统上安装的仿真器。

Spice 仿真器希望仿真命令（`.PROBE`，`.AC`，`.TRAN``等）包含在网表中。原理图中包含的任何以句号（`.`）开始的文本行都将被包含在网表中。如果一个文本对象包含多行，只有以句号开头的行将被包括在内。

根据原理图中符号的 Spice 模型设置，用于包括模型库文件的 `.include` 指令被自动添加到网表。

Spice 模型网表格式



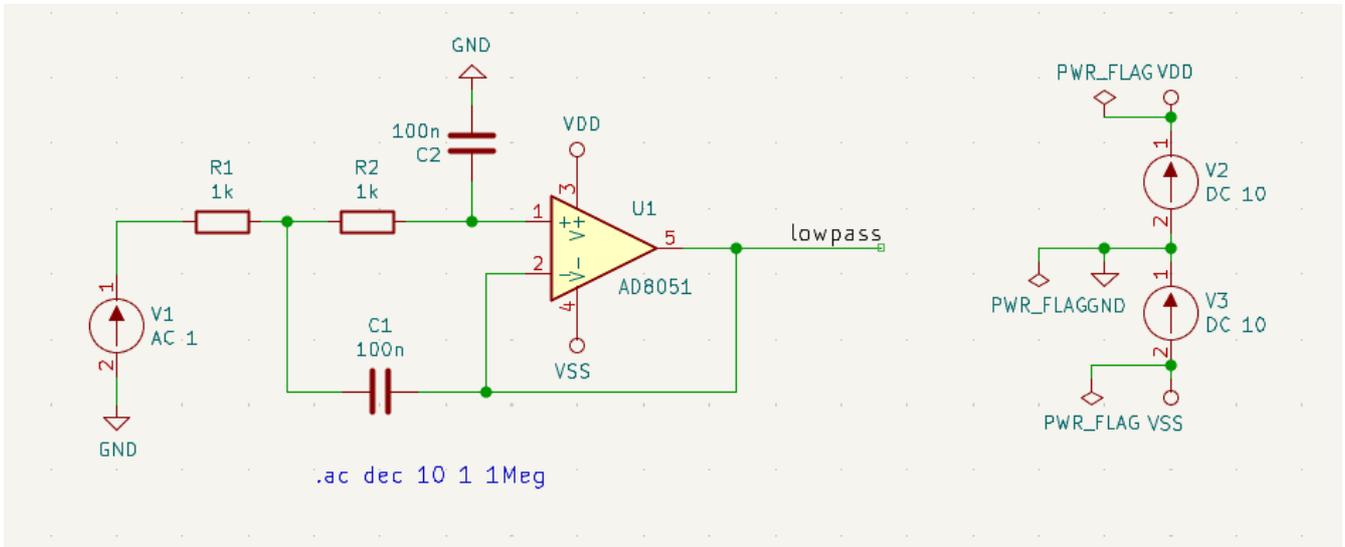
KiCad 还可以将原理图的网表导出为 Spice 子电路模型，这可以包含在一个单独的 Spice 仿真中。原理图中的任何层次标签都被用作子电路模型的引脚。模型中的每个引脚都有批注，描述该引脚的电气方向：

- `输入` 的层次标签被映射到一个 `输入` 的批注上
- `输出` 的层次化标签被映射到 `输出` 批注上
- `双向` 层次标签被映射到 `输入输出` 批注上
- `三态` 的层次标签被映射到 `三态` 批注上
- `被动` 层次标签被映射到 `被动` 批注上

当选择 **使用当前页面作为根页面** 时，只有当前页面被导出到子电路模型中。否则，整个原理图页面将被导出。

网表示例

下面是 KiCad 模拟演示中的 `sallen_key` 工程的原理图。



该原理图的 KiCad 格式网表如下：

```

(export (version "E")
  (design
    (source "/usr/share/kicad/demos/simulation/sallen_key/sallen_key.kicad_sch")
    (date "Sun 01 May 2022 03:14:05 PM EDT")
    (tool "Eeschema (6.0.4)")
    (sheet (number "1") (name "/") (tstamps "/"))
    (title_block
      (title)
      (company)
      (rev)
      (date)
      (source "sallen_key.kicad_sch")
      (comment (number "1") (value ""))
      (comment (number "2") (value ""))
      (comment (number "3") (value ""))
      (comment (number "4") (value ""))
      (comment (number "5") (value ""))
      (comment (number "6") (value ""))
      (comment (number "7") (value ""))
      (comment (number "8") (value ""))
      (comment (number "9") (value ""))))
    (components
      (comp (ref "C1")
        (value "100n")
        (libsource (lib "sallen_key_schlib") (part "C") (description ""))
        (property (name "Sheetname") (value ""))
        (property (name "Sheetfile") (value "sallen_key.kicad_sch"))
        (sheetpath (names "/") (tstamps "/"))
        (tstamps "00000000-0000-0000-0000-00005789077d"))
      (comp (ref "C2")
        (value "100n")
        (fields
          (field (name "Fieldname") "Value")
          (field (name "SpiceMapping") "1 2")
          (field (name "Spice_Primitive") "C"))
        (libsource (lib "sallen_key_schlib") (part "C") (description ""))
        (property (name "Fieldname") (value "Value"))
        (property (name "Spice_Primitive") (value "C"))
        (property (name "SpiceMapping") (value "1 2"))
        (property (name "Sheetname") (value ""))
        (property (name "Sheetfile") (value "sallen_key.kicad_sch"))
        (sheetpath (names "/") (tstamps "/"))
        (tstamps "00000000-0000-0000-0000-00005789085b"))
      (comp (ref "R1")
        (value "1k")
        (fields
          (field (name "Fieldname") "Value")
          (field (name "SpiceMapping") "1 2")
          (field (name "Spice_Primitive") "R"))
        (libsource (lib "sallen_key_schlib") (part "R") (description ""))
        (property (name "Fieldname") (value "Value"))
        (property (name "SpiceMapping") (value "1 2"))
        (property (name "Spice_Primitive") (value "R"))
        (property (name "Sheetname") (value ""))
        (property (name "Sheetfile") (value "sallen_key.kicad_sch"))
        (sheetpath (names "/") (tstamps "/"))
        (tstamps "00000000-0000-0000-0000-0000578906ff"))
      (comp (ref "R2")
        (value "1k")
        (fields

```

在 Spice 格式中, 网表如下 :

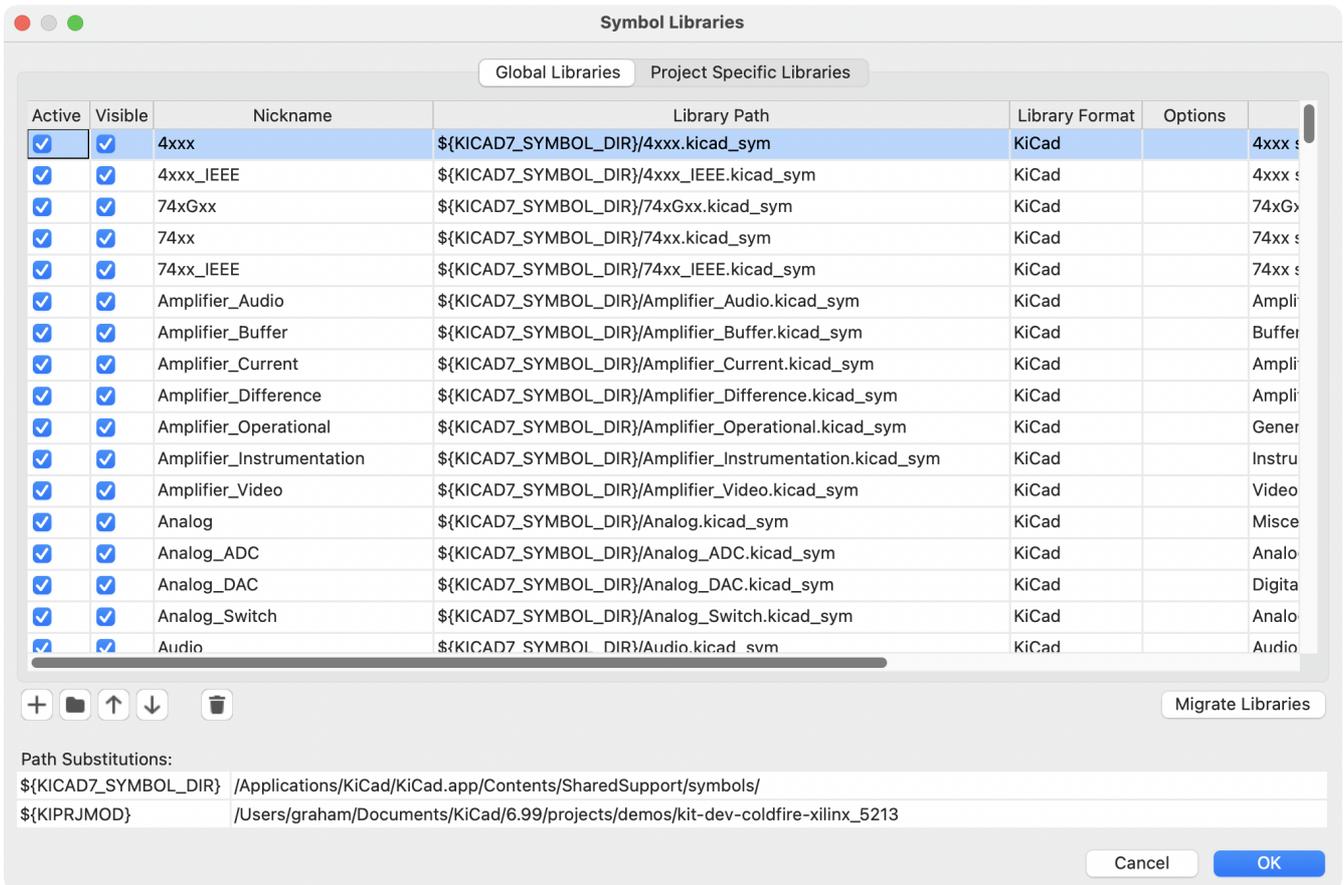
```
.title KiCad schematic
.include "ad8051.lib"
XU1 Net-_C2-Pad1_ /lowpass VDD VSS /lowpass AD8051
C2 Net-_C2-Pad1_ GND 100n
C1 /lowpass Net-_C1-Pad2_ 100n
R2 Net-_C2-Pad1_ Net-_C1-Pad2_ 1k
R1 Net-_C1-Pad2_ Net-_R1-Pad2_ 1k
V1 Net-_R1-Pad2_ GND AC 1
V2 VDD GND DC 10
V3 GND VSS DC 10
.ac dec 10 1 1Meg
.end
```

符号和符号库

KiCad 将符号组织到符号库中，而符号库中则保存着符号的集合。原理图中的每个符号都由一个全称来唯一标识，该全称由库的名称和符号名称组成。例如，标识符 Audio:AD1853 是指 Audio 库中的 AD1853 符号。

管理符号库

KiCad 使用一个符号库表将符号库名称映射到磁盘上的底层符号库。KiCad 使用一个全局的符号库表，以及一个针对每个工程的表。要编辑任何一个符号库表，请使用 **偏好设置** → **管理符号库...**。



全局符号库表包含了无论当前加载的工程是什么，都可以使用的库列表。该表保存在 KiCad 配置文件夹下的 `sym-lib-table` 文件中。[该文件夹的位置](#) 取决于使用的操作系统。

工程专用的符号库表包含了专门为当前加载的工程提供的库的列表。如果有任何工程专用的符号库，该表将保存在工程文件夹下的 ``sym-lib-table`` 文件中。

初始配置

第一次运行 KiCad 原理图编辑器时，如果在 KiCad 配置文件夹中没有找到全局符号表文件 `sym-lib-table`，KiCad 将引导用户设置一个新的符号库表。这个过程将在《初始配置》中描述。

管理列表条目

符号库只有在它们被添加到全局或工程专用的符号库表中时才能使用。

通过点击 **+** 按钮并选择一个库或点击 **+** 按钮并输入库文件的路径来添加一个库。选定的库将被添加到当前打开的库表中（全局或特定工程）。可以通过选择所需的库条目并点击 **🗑** 按钮来删除库。

这个 **↑** 和 **↓** 按钮在库表中上下移动所选库。这并不影响符号库浏览器、符号编辑器或添加符号工具中库的显示顺序。

通过取消第一列中的 **活动** 复选框，可以使库不活动。不活动的库仍然在库表中，但不会出现在任何库的浏览器中，也不会从磁盘加载，这可以减少加载时间。

通过点击范围内的第一个库，然后 **Shift** - 点击范围内的最后一个库，就可以选择一个库的范围。

每个库必须有一个独特的昵称：在同一个表中不允许有重复的库昵称。然而，昵称可以在全局和工程库表中重复。工程表中的库比全局表中的同名库更有优先权。

库的昵称不一定要与库的文件名或路径有关。冒号字符 (**:**) 不能在库的昵称或符号名称中使用，因为它被用作昵称和符号之间的分隔符。

每个库条目必须有一个有效的路径。路径可以定义为绝对的、相对的，或者通过《环境变量替换,环境变量替换》来定义。

必须选择适当的库格式，以便正确读取库。"KiCad" 格式用于 KiCad 6+ 版库 (**.kicad_sym** 文件)，而 "Legacy" 格式用于 KiCad 旧版本的库 (**.lib** 文件)。遗留库是只读的，但可以通过 **迁移库** 按钮迁移到 KiCad 格式的库 (见《迁移库，迁移遗留库》一节)。

有一个可选的描述字段，用于添加库条目的描述。选项字段目前不使用，所以添加选项在加载库时不会有任何影响。

环境变量的替代

符号库表支持环境变量的替换，这允许你定义包含自定义路径的环境变量来存储你的库。通过在符号库路径中使用 **\${ENV_VAR_NAME}** 的语法，支持环境变量的替换。

默认情况下，KiCad 定义了几个环境变量，这些变量在 [工程管理文档](#) 中描述。环境变量可以在 **偏好设置** → **配置路径...** 对话框中进行配置。

在符号库表中使用环境变量，可以在不破坏符号库表的情况下重新定位库，只要在库的位置改变时更新环境变量即可。

\${KIPRJMOD} 是一个特殊的环境变量，总是展开为当前工程目录的绝对路径。**\${KIPRJMOD}** 允许将库存储在工程文件夹中，而不必在工程库表中使用绝对路径。这使得重新定位工程而不破坏其工程库表成为可能。

使用模式

符号库可以全局定义，也可以专门为当前加载的项目定义。在用户的全局表中定义的符号库总是可用的，并存储在用户的 KiCad 配置文件夹中的 **sym-lib-table** 文件中。工程专用的符号库表只对当前打开的工程文件有效。

每种方法都有其优点和缺点。在全局表中定义所有的库意味着它们在需要时总是可用的。这样做的缺点是，加载时间会增加。

在工程的基础上定义所有的符号库，意味着你只有工程所需的库，这减少了符号库的加载时间。缺点是你总是要记住添加每个工程所需的每个符号库。

一种使用模式是在全局范围内定义常用的库，并在工程专用的库表中定义工程所需的库。对于如何定义库没有任何限制。

迁移遗留库

遗留库（.lib 文件）是只读的，但它们可以被迁移到 KiCad 6 版库（.kicad_sym）。6.0.0 以上的 KiCad 版本不能查看或编辑 KiCad 6 版本的库。

通过在符号库表中选择传统库并点击 **迁移库** 按钮，可以将其转换为 KiCad 6 库。通过 **Ctrl** - 点击或 **shift** - 点击，可以一次选择和迁移多个库。

库也可以通过在符号编辑器中打开它们并将其保存为一个新的库来一次转换。

遗留工程重新映射

当加载一个在符号库表实施之前创建的原理图时，KiCad 会尝试将原理图中的符号库链接重新映射到适当的库表符号。这个过程的成功与否取决于几个因素：

- 原理图中使用的原始库仍然可用，并且与符号被添加到原理图中时相比没有变化。
- 所有恢复操作都是在检测到创建恢复库或保持现有恢复库最新时进行的。
- 工程符号缓存库的完整性没有被破坏。

WARNING

重新映射将在工程文件夹中的 `rescue-backup` 文件夹中备份所有在重新映射期间被改变的文件。在重新映射之前，一定要对你的工程做一个备份，以防出错。

WARNING

即使恢复操作已被禁用，也会被执行，以确保正确的符号可用于重新映射。不要取消这个操作，否则重新映射将不能正确地重新映射原理图符号。任何错误的符号链接将不得不手动修复。

NOTE

如果原来的库已经被删除，并且没有进行恢复，作为最后的手段，缓存库可以作为恢复库使用。将缓存库复制到一个新的文件名，并使用符号库表实现之前的 KiCad 版本将新的库文件添加到库列表的顶部。

创建和编辑符号

符号和符号库的概述

符号是一个元件的原理图。一个符号由以下部分组成：

- 图形项目（直线、圆形、圆弧、文字等），决定了符号在原理图中的样子。
- 引脚，具有图形属性（直线、时钟、反相、低电平有效等）和电气属性（输入、输出、双向等），由电气规则检查（ERC）工具使用。
- 字段，如位号、值、PCB 设计的对应封装名称等。

符号库是由一个或多个符号组成的。一般来说，这些符号按功能、类型和/或制造商进行逻辑分组。

符号可以从同一库中的另一个符号衍生出来。派生符号共享基础符号的图形形状和引脚定义，但可以覆盖基础符号的属性字段（值、封装、封装筛选器、数据手册、描述等）。派生符号可用于定义与基本部件相似的符号。例如，74LS00、74HC00 和 7437 符号都可以从 7400 符号衍生出来。在以前的 KiCad 版本中，派生符号被称为别名。

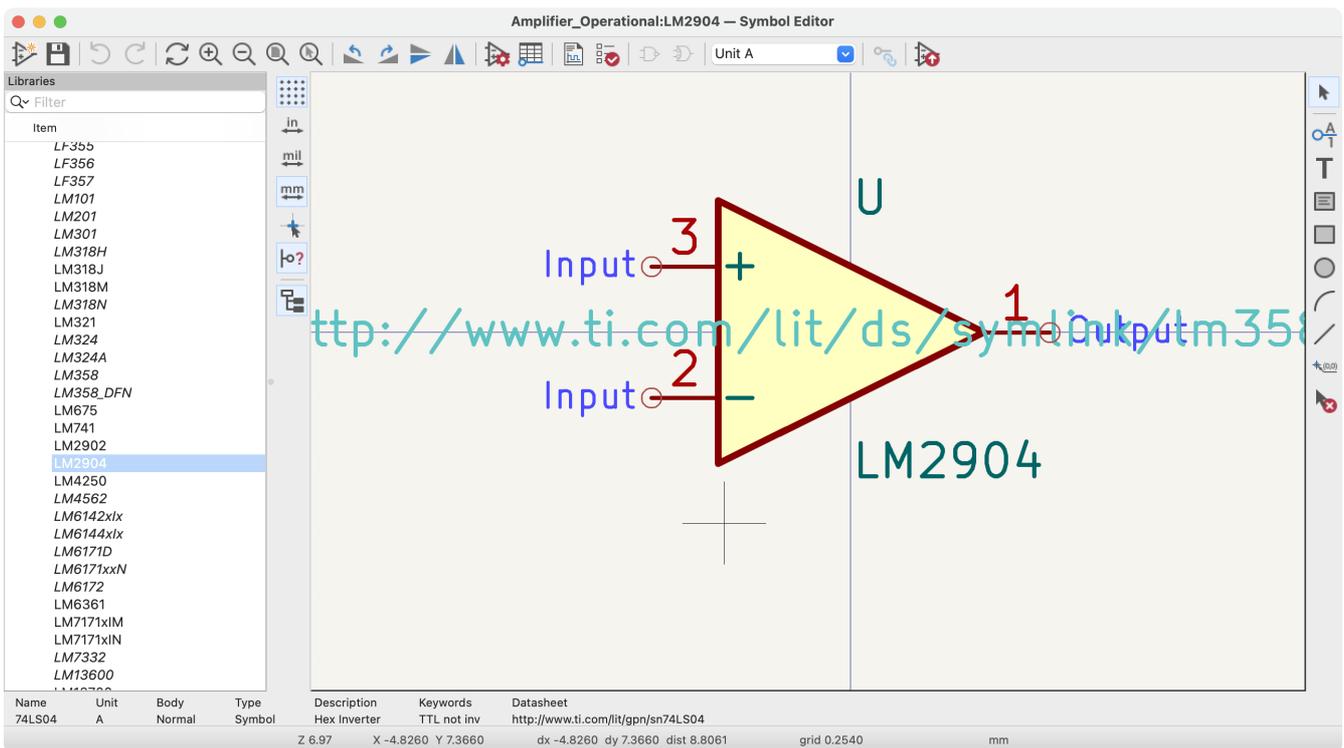
符号编辑器概述

KiCad 提供了一个符号编辑工具，允许你创建库，在库之间添加、删除或转移符号，将符号导出到文件，并从文件中导入符号。

一般来说，设计一个符号的流程包括：

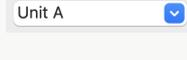
- 定义符号是否由一个或多个单元组成。
- 定义符号是否有替代的主体风格（也称为德摩根代号）。
- 使用直线、矩形、圆形、多边形和文字设计其符号表示。
- 通过仔细定义每个引脚的图形元素、名称、编号和电气属性（输入、输出、三态、电源输出等）来添加引脚。
- 确定该符号是否应从另一个具有相同图形设计和引脚定义的符号中衍生出来。
- 添加可选的字段，如 PCB 设计软件使用的封装名称和/或定义其可见性。
- 通过添加描述字符串和数据手册链接来记录该符号，等等。
- 将其保存在所需的库中。

符号编辑器的主窗口如下所示。它包括三个用于快速访问常用功能的工具条和一个符号查看/编辑区。不是所有的命令都可以在工具栏上使用，但所有的命令都可以在菜单中使用。



主工具栏

主工具条位于主窗口的顶部。它有撤销/重做命令、缩放命令、符号属性对话框和单位/代表管理控制等按钮。

	在选定的库中创建一个新的符号。
	保存当前选定的库。库中所有修改过的符号都将被保存。
	撤销上次的编辑。
	重做最后一次撤消。
	刷新显示。
	放大。
	缩小。
	放大到适合显示的符号。
	放大到适合选择。
	逆时针旋转。
	顺时针旋转。
	水平镜像。
	垂直镜像。
	编辑当前的符号属性。
	在一个表格界面中编辑符号的引脚。
	打开符号的数据手册，如果它被定义的话。
	测试当前符号是否有设计错误。
	选择正常的主体风格。如果当前符号没有备选的主体样式，该按钮将被禁用。
	选择备用的主体风格。如果当前的符号没有备用的主体风格，该按钮将被禁用。
	选择要显示的多单元符号的单位。如果当前的符号不是来自于多单元的符号，则下拉控件将被禁用。
	启用同步引脚编辑模式。启用该模式后，任何引脚的修改都会传播到所有其他符号单元。引脚编号的变化不会被传播。这种模式对有多个可互换单元的符号自动启用，对只有一个单元的符号不能启用。

元素工具栏

位于主窗口右侧的垂直工具条允许你放置设计一个符号所需的所有元素。

	选择工具。用选择工具右键单击可以打开光标下的对象的上下文菜单。用选择工具左键单击，在主窗口底部的信息面板中显示光标下的对象的属性。用选择工具左键双击将打开光标下对象的属性对话框。
	引脚工具。左键单击可以添加一个新的引脚。
	图形文字工具。左键单击以添加一个新的图形文本项目。
	矩形工具。左键单击开始绘制图形矩形的第一个角。再次左键单击以放置矩形的另一个角。
	圆形工具。左键单击，开始从中心绘制一个新的圆形。再次左键单击来定义圆的半径。
	弧形工具。左键单击，从第一个弧形端点开始绘制一个新的图形弧项目。再次左键单击以定义第二个弧形端点。通过拖动弧形中心点来调整半径。
	连接线工具。左键单击开始在当前符号中绘制一个新的图形线项。每增加一条连接线都要左击。双击左键来完成该线。
	锚点工具。左键单击以设置符号的锚定位置。
	删除工具。左键单击，从当前符号中删除一个对象。

选项工具栏

位于主窗口左侧的垂直工具条允许你设置一些编辑器的绘图选项。

	切换网格的可见性和关闭。
	设置单位为英寸。
	设置单位为密尔（0.001英寸）。
	设置单位为毫米。
	切换全屏光标的开启和关闭。
	切换显示引脚的电气类型。
	切换显示库和符号工作区。

将符号保存到库中

这个  按钮显示或隐藏可用库的列表，可以选择一个活动库。当一个符号被保存时，它将被放在这个库中。

点击左侧工具栏上的  图标，可以切换库和符号的树状视图。点击一个符号就可以打开该符号。

NOTE

有些符号是由其他符号衍生出来的。派生的符号名称在树状图中以 **大写字母** 显示。如果一个派生符号被打开，它的符号图形将不能被编辑。它的符号字段将可以正常编辑。要编辑一个基本符号及其所有衍生符号的图形，请打开基本符号。

修改后的符号可以保存在当前库或不同的库中。

要在当前库中保存修改后的符号，请点击  图标。修改的内容将被写入现有的符号中。

NOTE

保存一个修改过的符号也会保存同一库中所有其他修改过的符号。

要把符号的变化保存为一个新的符号，请点击 **文件** → **另存为...**。符号可以保存在当前库或不同的库中。可以为该符号设置一个新的名称。

要创建一个只包含当前符号的新文件，请点击 **文件** → **导出** → **符号...**。这个文件将是一个标准库文件，只包含一个符号。

创建符号

创建一个新符号

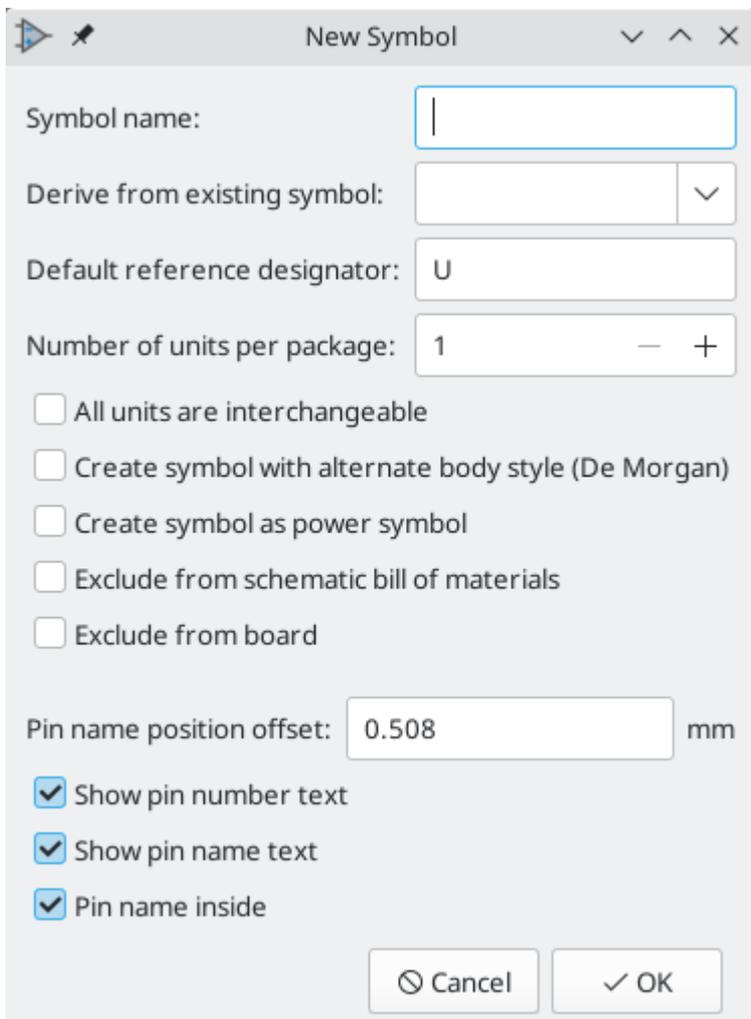
点击  图标就可以创建一个新的符号。你会被要求提供一些符号的属性。

- 符号名称
- 一个可选的基础符号，用于派生新的符号。新符号将使用基础符号的图形形状和引脚配置，但其他符号信息可以在派生符号中修改。基础符号必须与新派生符号在同一个库中。
- 位号前缀（U, C, R）。
- 每个封装的单元数，以及这些单元是否可以互换（例如，一个 7400 四路 NAND 符号可以有 4 个单元，每个门一个）。
- 如果需要另一种主体样式（有时被称为“德-摩根等价”）。
- 该符号是否为电源符号。电源符号出现在原理图编辑器的 **添加电源符号** 对话框中，它们的“值”字段在原理图中不可编辑，不能为它们分配一个封装，它们不会被添加到 PCB 上，也不包括在物料清单中。
- 该符号是否应从材料清单中排除。
- 该符号是否应从 PCB 中排除。

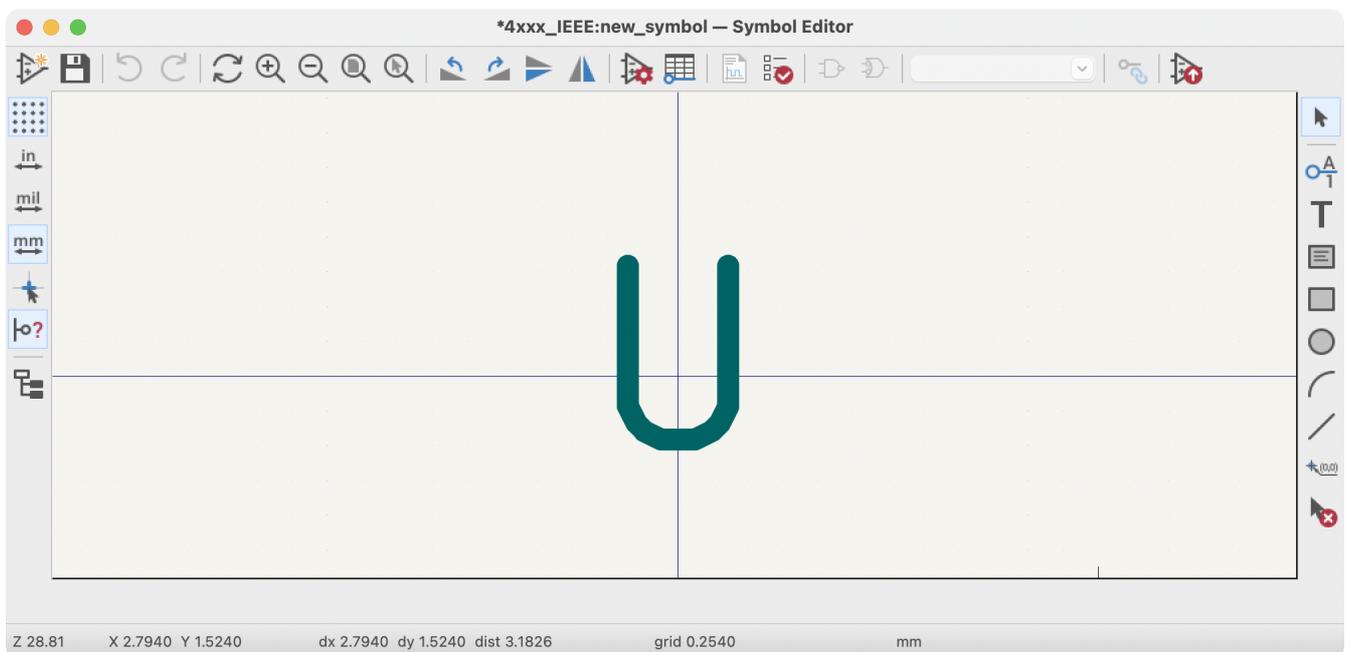
还有几种图形选项。

- 每个引脚的末端和它的引脚名称之间的偏移。
- 是否应显示引脚编号和引脚名称。
- 引脚名称是否应显示在引脚旁边，或显示在符号主体内的引脚末端。

这些属性以后也可以在《符号属性，符号属性窗口》中修改。



一个新的符号将使用上述属性被创建，并将出现在编辑器中，如下图所示。



中间的蓝色十字是符号的锚，它指定了符号的原点，即坐标 (0, 0)。锚点可以通过选择  图标并点击新的所需锚点位置来重新定位。

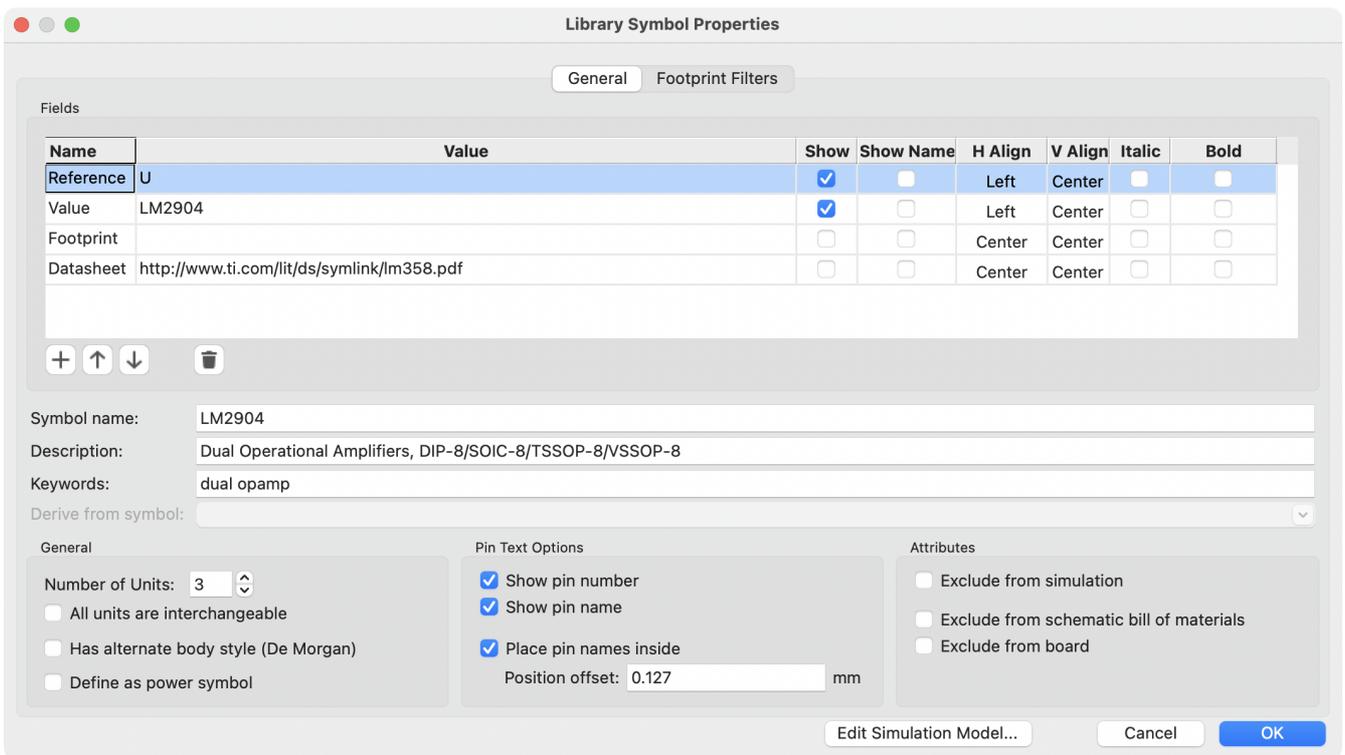
从另一个符号创建一个符号

通常，你想制作的符号与符号库中已有的符号相似。在这种情况下，加载和修改一个现有的符号是很容易的。

- 加载将被作为起点的符号。
- 使用 **文件** → **另存为...** 保存一个新的符号副本。"另存为"对话框将提示新符号的名称和要保存在哪个库中。
- 根据需要编辑新的符号。
- 保存修改后的符号。

符号属性

符号属性是在创建符号时设置的，但它们可以在任何时候被修改。要改变符号属性，请点击  图标，显示符号属性对话框。你也可以双击编辑画布中的一个空点。

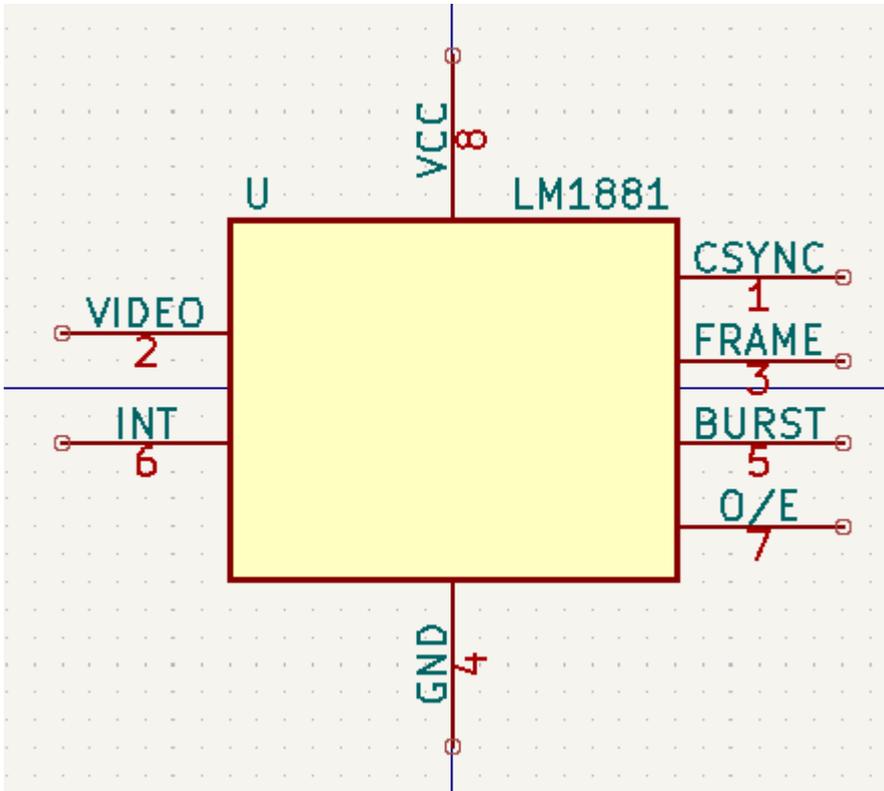


设置 **单元编号**，并检查 **所有单元都可互换** 和 **具有交替的主体风格**（如适用）是很重要的，因为这些设置会影响引脚和图形被添加到每个符号单元的方式。

如果你在向符号添加引脚后改变了每个包的单元编号，你将需要做额外的工作来为额外的单元添加引脚和图形。如果最初正确设置了这些属性，引脚和图形就会自动添加到每个单元。尽管如此，还是可以在任何时候修改这些属性。

图形选项 **显示引脚编号** 和 **显示引脚名称** 定义引脚编号和引脚名称文本的可见性。选项 **将引脚名称置于内部** 定义了引脚名称相对于引脚主体的位置。如果该选项被选中，引脚被显示在符号的轮廓内。在这种情况下，**引脚名称位置偏移** 属性定义了文字远离引脚末端的位置。通常，0.02 到 0.05 英寸的数值是合理的。

下面的例子显示了一个没有勾选 **将引脚名称放在里面** 选项的符号。注意名称和引脚编号的位置。

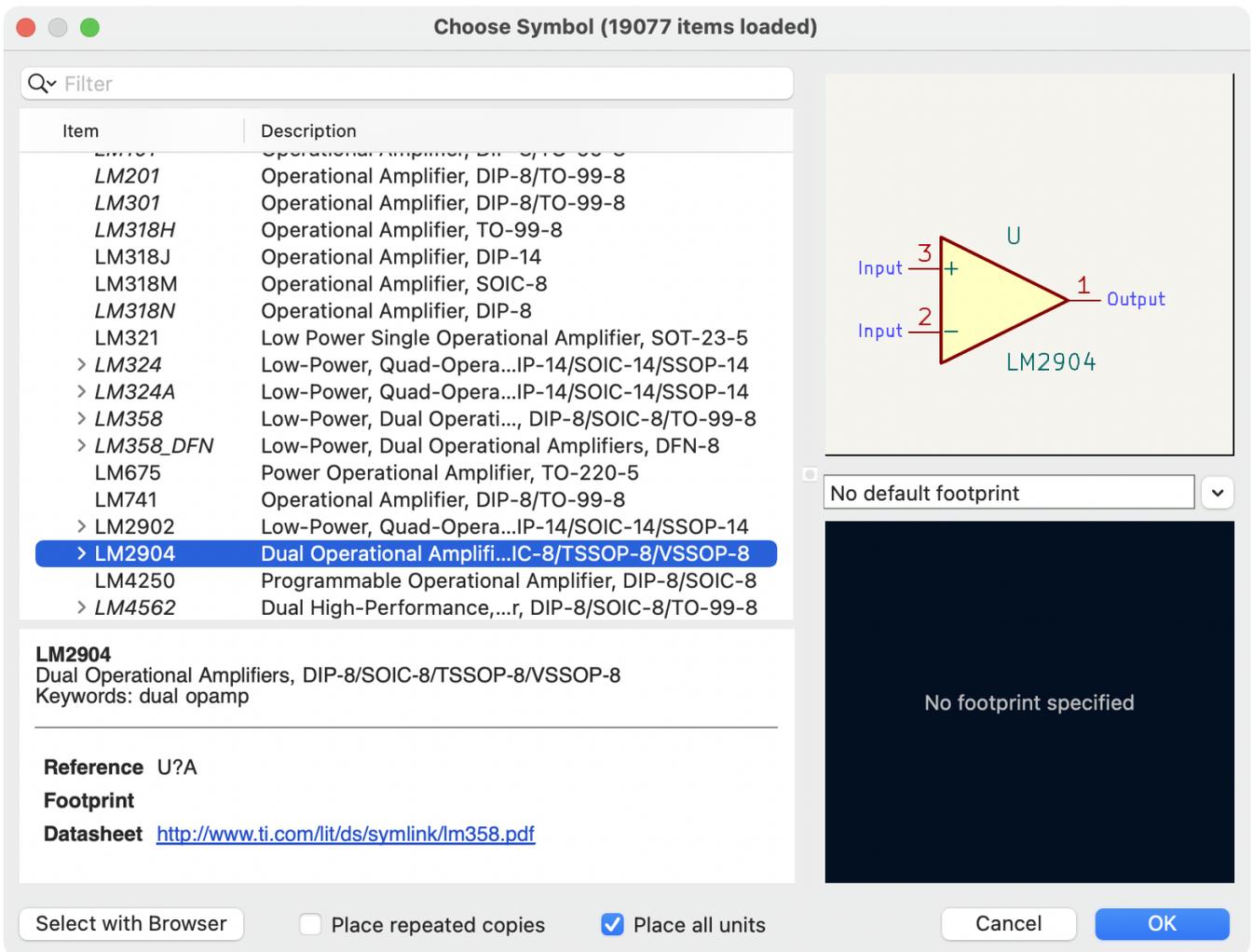


符号名称、描述和关键词

符号名称 是符号在库中的名称。符号是由库和符号名称的组合来识别的。

在以前的 KiCad 版本中，符号名称与 值 字段相连。在 KiCad 7.0 及以后的版本中，这种联系被删除。

符号描述应包含对组件的简要描述，如元件的功能、突出的特点和包装选项。关键字应包含与该元件相关的其他术语。关键词主要是用来帮助搜索符号的。



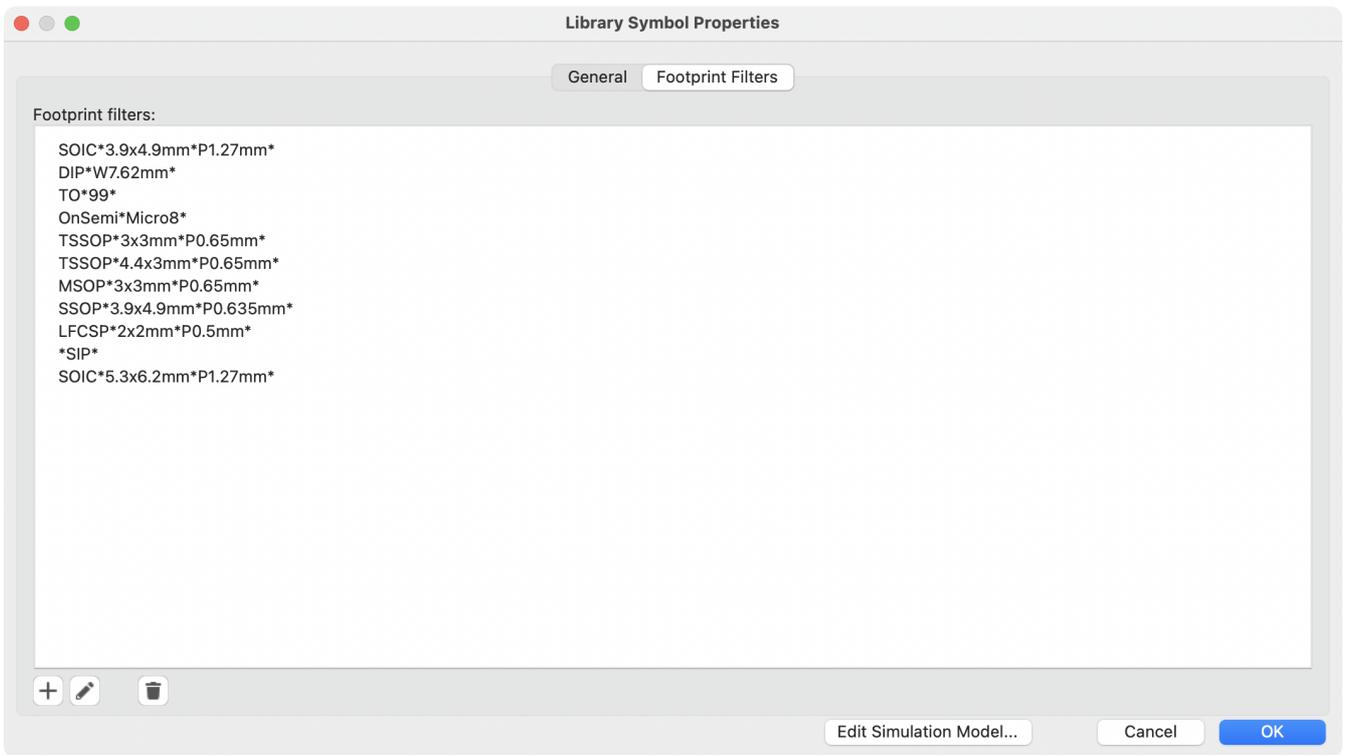
符号的名称、描述和关键词都是在符号编辑器和添加符号对话框中搜索符号时使用的。描述和关键词会显示在符号库浏览器和添加符号对话框中。

封装筛选器

封装筛选器标签用于定义哪些封装适合与符号一起使用。这些筛选器可以在封装分配工具中应用，以便每个符号只显示合适的封装。

可以定义多个封装筛选器。符合任何一个筛选器的封装将被显示；如果没有定义筛选器，那么所有的封装都将被显示。

筛选器可以使用通配符。* 匹配任何数量的字符，包括零，而 ? 匹配零或一个字符。例如，SOIC-* 将匹配 SOIC-8_3.9x4.9mm_P1.27mm 封装以及任何其他以 SOIC- 开头的封装。筛选器 SOT?23 匹配 SOT23 以及 SOT-23。



具有替代性符号表示的符号

如果该符号定义了一个备用的主体样式，每次必须选择一个主体样式进行编辑。要编辑正常表示，请点击  图标。

要编辑备用替换，请点击  图标。使用下面显示的



单，选择你想编辑的单位。

图形元素

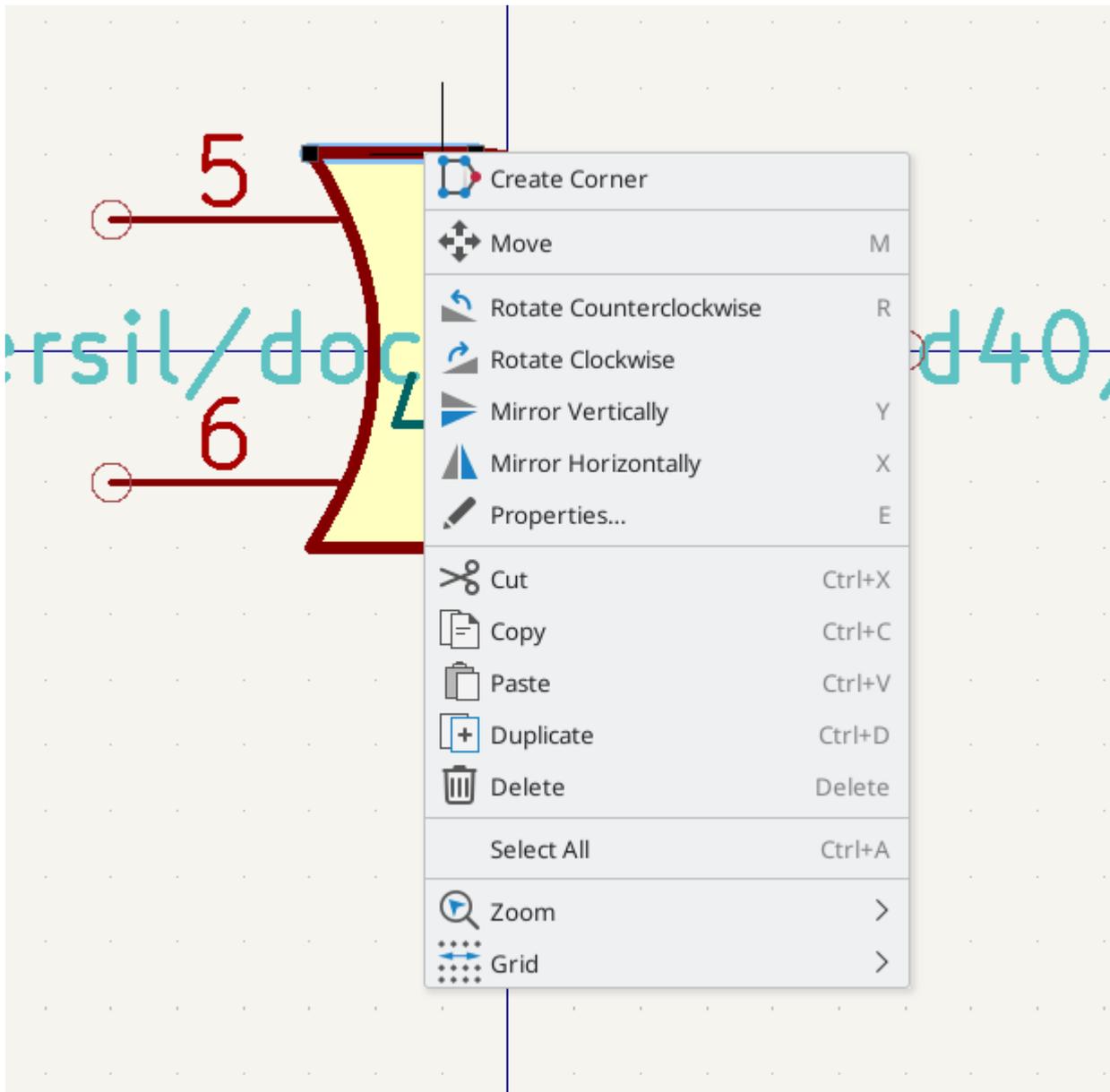
图形元素创建了一个符号的视觉表现，不包含电气连接信息。图形元素是用以下工具创建的：

- 由起点和终点定义的直线和多边形。
- 由两个对角线角定义的矩形。
- 由中心和半径定义的圆形。
- 弧线由弧线的起点和终点以及它的中心定义。一个弧线从 0° 到 180°。

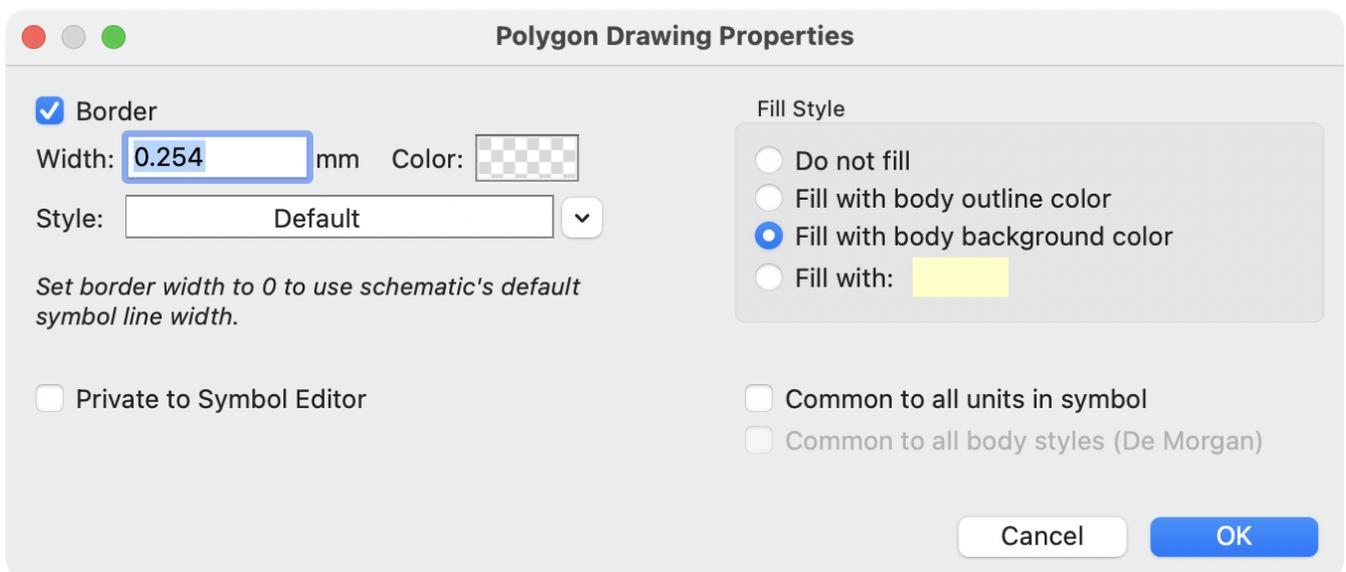
主窗口右侧的垂直工具条允许你放置设计一个符号的表现形式所需的所有图形元素。

图形元素成员

每个图形元素（直线、圆弧、圆形等）都可以被定义为所有单元和/或主体样式所共有的，或特定单元和/或主体样式所特有的。元素选项可以通过右键点击元素来快速访问，显示所选元素的上下文菜单。下面是一个线条元素的上下文菜单。



你也可以在一个元素上双击左键来修改其属性。下面是一个多边形元素的属性对话框。



图形元素的属性是：

边界 决定是否应该绘制形状的边框。

- **宽度** 和 **颜色** 定义了边框的线宽和颜色。边界宽度为 0 时，使用原理图的默认符号线宽。**风格** 决定了边框的线条样式（实线、虚线、点线等）。
- **填充样式** 决定了图形元素所定义的形状是不填充还是填充。填充颜色可以是颜色主题的主体边框颜色、主体背景颜色或自定义颜色。
- **符号中所有单元的共有** 决定了图形元素是否为符号中的每个单元绘制，每个包有一个以上的单元，或者图形元素只为当前单元绘制。
- **对所有主体风格通用（德摩根）** 决定图形元素是否为具有替代主体风格的符号中的每个符号表示绘制，或者图形元素是否只为当前的主体风格绘制。
- **私有化的符号编辑器** 导致形状仅在符号编辑器中编辑符号时可见。当符号被添加到原理图中时，该形状将被隐藏。

图形化文本元素

这个 **T** 图标允许创建图形文本。图形文本会自动调整方向，以便可读，即使符号被镜像化。请注意，图形文本项目与符号字段不一样。

每个符号的多个单元和替代的主体样式

符号最多可以有两种主体样式（一个标准符号和一个备用符号，通常被称为“德摩根等同”）和/或每个封装有一个以上的单元（例如逻辑门）。有些符号每个封装可以有一个以上的单元，每个单元有不同的符号和引脚配置。

例如，考虑一个有两个开关的继电器，它可以被设计成有三个不同单元的符号：线圈、开关 1 和开关 2。设计一个每个封装有多个单元的符号和/或备用的主体样式是非常灵活的。一个引脚或一个主体符号项目可以是所有单元所共有的，也可以是特定单元所特有的，或者它们可以是两个符号表示所共有的，所以是特定的符号表示。

默认情况下，引脚是特定于一个单元和主体样式的。当一个引脚是所有单位或所有主体款式所共有的，它只需要被创建一次。主体样式的图形和文字也是如此，它们可能是每个单元所共有的，但通常是每个主体样式所特有的）。

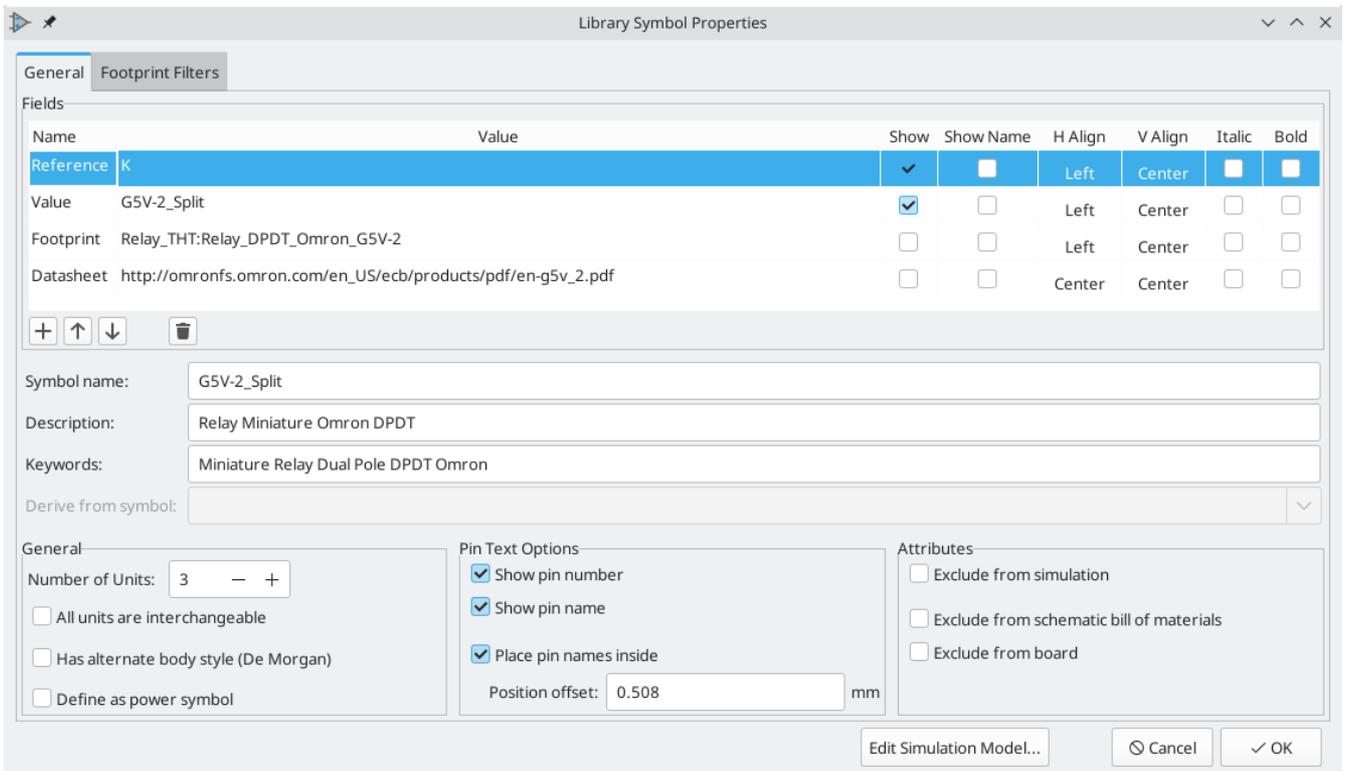
要向一个符号添加额外的单元，在符号属性对话框中将 **单元编号** 属性设置为适当的数字。默认情况下，符号单元被命名为 **单元 A**、**单元 B** 等，但你可以使用 **编辑** → **设置单元显示名称...** 为当前单元设置一个任意的名称。

要添加一个备用的主体样式，请在符号属性对话框中设置 **具有备用的主体样式（德摩根）** 属性。

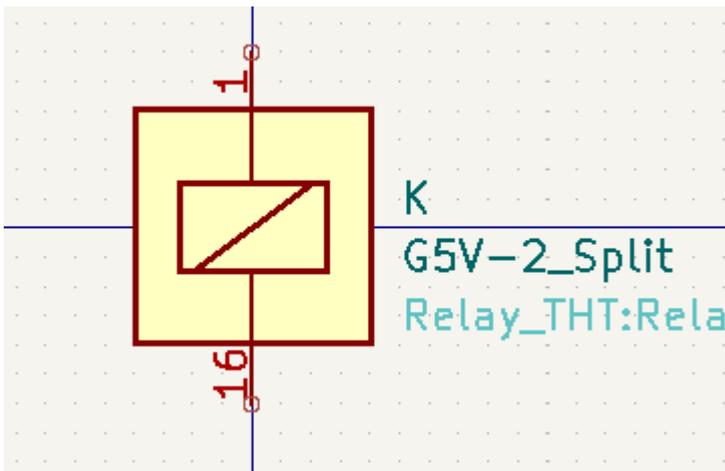
具有多个不可互换单元的符号示例

对于一个有多个不能互换的单元的符号的例子，考虑一个每包有 3 个单元的继电器：一个线圈、开关 1 和开关 2。

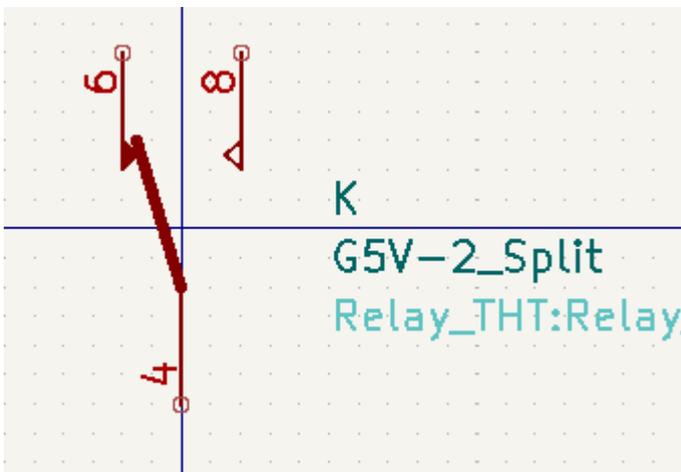
这三个单元并不都是一样的，所以在符号属性对话框中应取消选择 **所有单元都可以互换**。另外，这个选项可以在最初创建符号时指定。



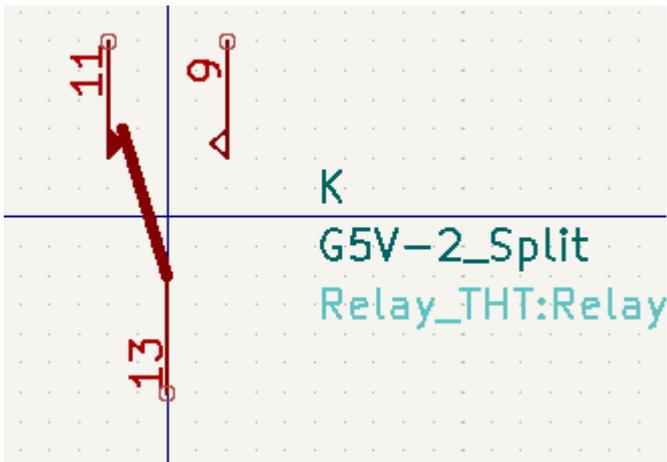
单元 A



单元 B



单元 C



单元 A 与单元 B 和单元 C 的符号和引脚布局不一样，所以这些单元不能互换。

NOTE

同步引脚编辑模式 可以通过点击  图标启用。在这种模式下，引脚的修改会在符号单元之间同步；一个单元中的修改也会反映在其他单元中。当这种模式被禁用时，一个单元中的引脚修改不会影响其他单元。当 **所有单元可互换** 被选中时，该模式会自动启用，但它可以被禁用。当 **所有单元可互换** 未被选中或符号只有一个单元时，该模式不能被启用。

引脚创建和编辑

你可以通过点击  按钮来创建和插入一个引脚。可以通过双击引脚来编辑引脚的属性。你也可以删除或移动你已经添加的引脚。必须小心地创建引脚，因为任何错误都会对 PCB 设计产生影响。

引脚概述

一个引脚是由它的图形表示、它的名字和它的编号定义的。引脚的名称和编号可以包含字母、数字和符号，但不能包含空格。为了使电气规则检查（ERC）工具发挥作用，引脚的电气类型（输入、输出、三态...）也必须被正确定义。如果这个类型定义不正确，原理图上的 ERC 检查结果可能是无效的。

重要笔记：

- 符号的引脚按编号与封装的焊盘相匹配。符号中的引脚编号必须与封装中相应的焊盘编号相匹配。
- 不要在引脚名称和数字中使用空格。空格将被自动替换成下划线（`_`）。
- To define a pin name with an inverted signal (overbar) use the `~` (tilde) character followed by the text to invert in braces. For example `~{FO}O` would display $\overline{FO}O$.
- 如果引脚名称为空，则认为该引脚未被命名。
- 引脚名称可以在一个符号中重复。
- 引脚编号在一个符号中必须是唯一的。

引脚属性

Pin name:

Pin number:

Electrical type:

Graphic style:

X position: in

Y position: in

Orientation:

Pin length: in

Name text size: in

Number text size: in

Common to all units in symbol

Common to all body styles (De Morgan)

Visible

Preview:

> Alternate pin definitions

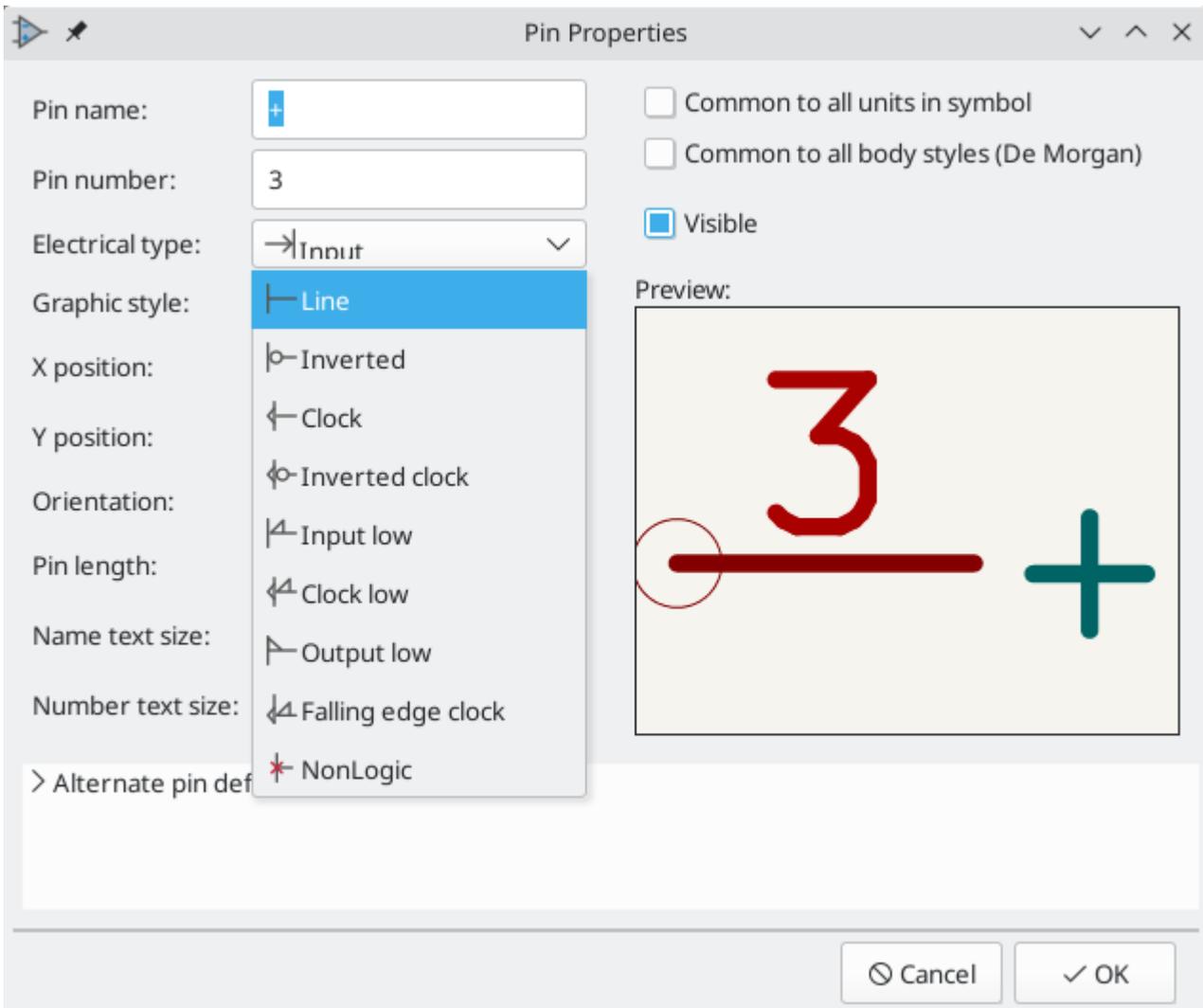
Cancel OK

引脚属性对话框允许你编辑引脚的所有特性。当你创建一个引脚或双击一个现有的引脚时，这个对话框会自动弹出。这个对话框允许你修改：

- 引脚名称和文字大小。
- 引脚编号和文字大小。
- 引脚的长度。
- 引脚的电气类型和图形风格。
- 单元和备用成员。
- 引脚可见性。
- 《备用引脚定义，备用引脚定义》。

引脚图形样式

不同的引脚图形风格如下图所示。这些样式是纯粹的图形，不影响引脚的电气类型。



引脚电气类型

Each pin in a symbol has an electrical type, such as input, output, or tri-state.

选择正确的电气类型对原理图 ERC 工具很重要。ERC 将检查引脚的连接是否恰当，例如，确保输入引脚被驱动，电源输入从适当的来源接收电源。

You can use the **Pin Conflicts Map** in the schematic editor to configure which pin types are allowed to connect and which will conflict. The default Pin Conflicts settings are briefly explained below. For more information, see the [ERC documentation](#).

Additionally, some pin types have special behavior outside of ERC: in the router, pads corresponding to a **free** pin can be connected to copper of any other net without causing a DRC error, and pads corresponding to stacked **unconnected** pins do not need to be connected to each other.

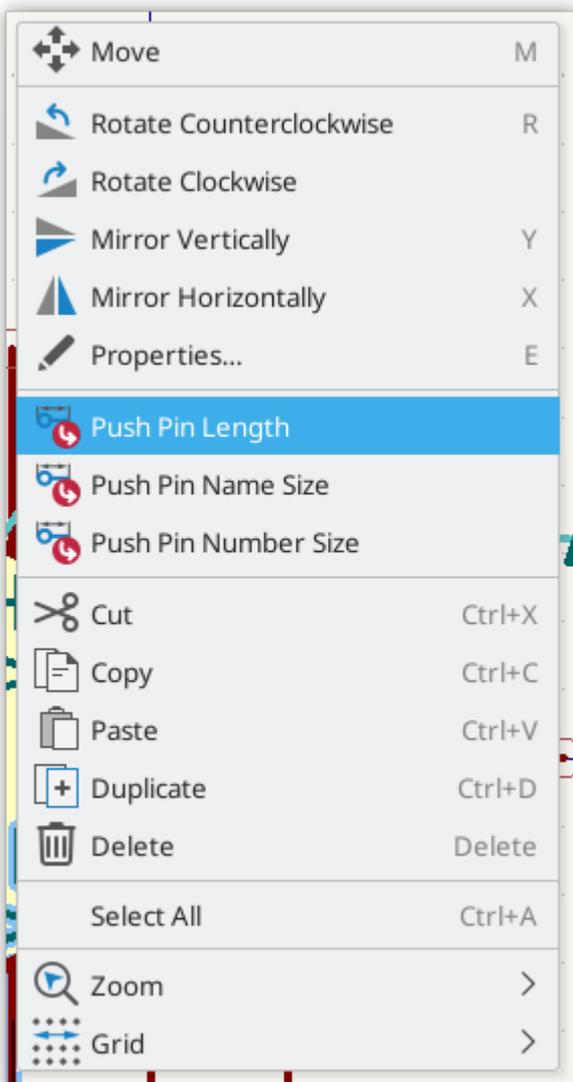
Pin Type	Description
Input	A pin which is exclusively an input. The default Pin Conflicts settings allow input pins to connect to most other types of pin.
Output	A pin which is exclusively an output. The default Pin Conflicts settings allow output pins to connect to most types of pin that aren't also outputs.

Bidirectional	A pin that can be either an input or an output, such as a microcontroller data bus pin. The default Pin Conflicts settings allow bidirectional pins to connect to most other types of pins, though there are a few more restrictions than with input pins.
Tri-state	A three state output pin (high, low, or high impedance). The default Pin Conflicts settings allow tri-state pins to connect to most other types of pins, but warnings are generated when they are connected to most types of output or power pins.
Passive	A pin that is not connected to active electronics, for example pins on a resistor or connector. The default Pin Conflicts settings allow passive pins to connect to most other types of pin.
Free	<p>A pin that does not electrically affect the operation of the device. These pins typically represent package leads that are not internally connected to the chip. The default Pin Conflicts settings allow free pins to connect to most other types of pin.</p> <p>In the PCB editor, pads corresponding to free pins can be connected to copper of any other net without causing a DRC error.</p>
Unspecified	A pin which has an unspecified type. With the default Pin Conflicts settings, ERC generates warnings when unspecified pins are connected to most other types of pins.
Power input	<p>A pin that powers the device. The default Pin Conflicts settings allow power input pins to connect to most other pin types. However, power input pins that are not connected to a power output pin generate an ERC violation.</p> <p>Additionally, power input pins that are marked invisible are automatically connected to the net with the same name as the pin. See the Power Symbols section for more information.</p>
Power output	A pin that provides power to other pins, such as a regulator output. The default Pin Conflicts settings allow power output pins to connect to most types of output pins, but not input pins.
Open collector	An open collector logic output. The default Pin Conflicts settings allow open collector pins to connect to most input pins and other open collector pins, but not to most other types of outputs.
Open emitter	An open emitter logic output. The default Pin Conflicts settings allow open collector pins to connect to most input pins and other open emitter pins, but not to most other types of outputs.

<p>Unconnected</p>	<p>A pin that should not be connected to anything. The default Pin Conflict settings do not allow pins of type unconnected to connect to any other type of pin, and ERC will not generate an "unconnected pin" violation when pins of this type are left unconnected.</p> <p>If a footprint has multiple pads corresponding to a single unconnected pin, the pads do not need to be connected to each other in the board.</p> <p>When multiple pins of type unconnected are stacked in a symbol, they are connected to separate nets, whereas stacked pins of other types are connected to the same net.</p> <p>Note that this pin type is different than placing a no connect flag on a pin in the schematic. The unconnected pin type indicates that the pin should never be connected in any schematic, while a no connect flag indicates that the pin is intentionally unconnected in the current schematic.</p>
--------------------	--

将引脚属性推送到其他引脚

你可以通过右击一个引脚，并分别选择 **推送引脚长度**、**推送引脚名称大小** 或 **推送引脚数字大小**，将该引脚的长度、名称大小或数字大小应用于符号中的其他引脚。符号中的所有其他引脚将被更新。



为多个单元和替代符号表示法定义引脚

具有多个单元和/或图形表示的符号在创建和编辑引脚时，问题特别多。大多数引脚都是针对每个符号单元（因为每个单元都有一组不同的引脚）和每个主体样式（因为正常主体样式和备用形式之间的形式和位置不同）。

符号库编辑器允许同时创建引脚。默认情况下，对引脚所做的修改是针对一个多单元符号的所有单元，以及对具有替代符号表示法的符号的两种表示法。唯一的例外是引脚的图形类型和名称，它们在符号单元和主体样式之间保持无关联。建立这种依赖关系是为了在大多数情况下更容易创建和编辑引脚。这个依赖关系可以通过切换主工具栏上的  图标来禁用。这将允许你完全独立地为每个单元和代表创建引脚。

引脚可以是共同的，也可以是不同单位所特有的。引脚也可以是两个符号表示法都有的，也可以是每个符号表示法都有的。当一个引脚对所有单位都通用时，它只需要绘制一次。引脚可以在引脚属性对话框中设置为通用或特定。

一个例子是 7400 四路双输入 NAND 门的输出引脚。由于有四个单元和两个符号表示，在符号定义中有八个独立的输出引脚。当创建一个新的 7400 符号时，正常符号表示法的单元 A 将显示在库编辑器中。要编辑备用符号表示法中的引脚样式，必须首先通过点击工具栏上的  按钮来启用它。要编辑每个单元的引脚编号，请使用



下拉控件选择合适的单元。

引脚表

另一种编辑引脚的方法是使用引脚表，可以通过  图标访问。引脚表以表格的形式显示符号中的所有引脚及其属性，所以它对批量修改引脚很有用。

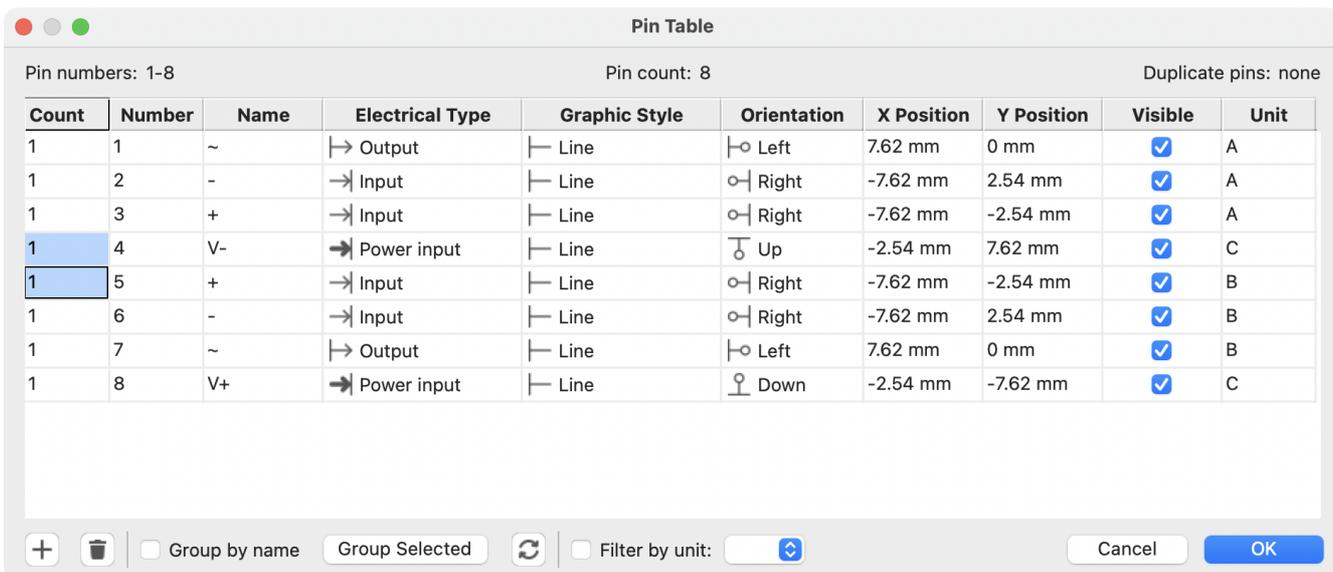
任何引脚属性都可以通过点击相应的单元格进行编辑。可以用  和  图标分别添加和删除引脚。

你可以通过分组来同时编辑多个引脚的同一属性。引脚可以按名称自动分组，或者你通过选择几个引脚并点击 **选择的分组** 来手动分组。点击  按钮来清除手动分组。你也可以过滤表格，只显示某些单元的引脚。

NOTE

通过右键单击标题行并勾选或取消附加列，可以显示或隐藏引脚表的各列。有些列是默认隐藏的。

下面的截图显示了一个四路运算放大器的引脚表。

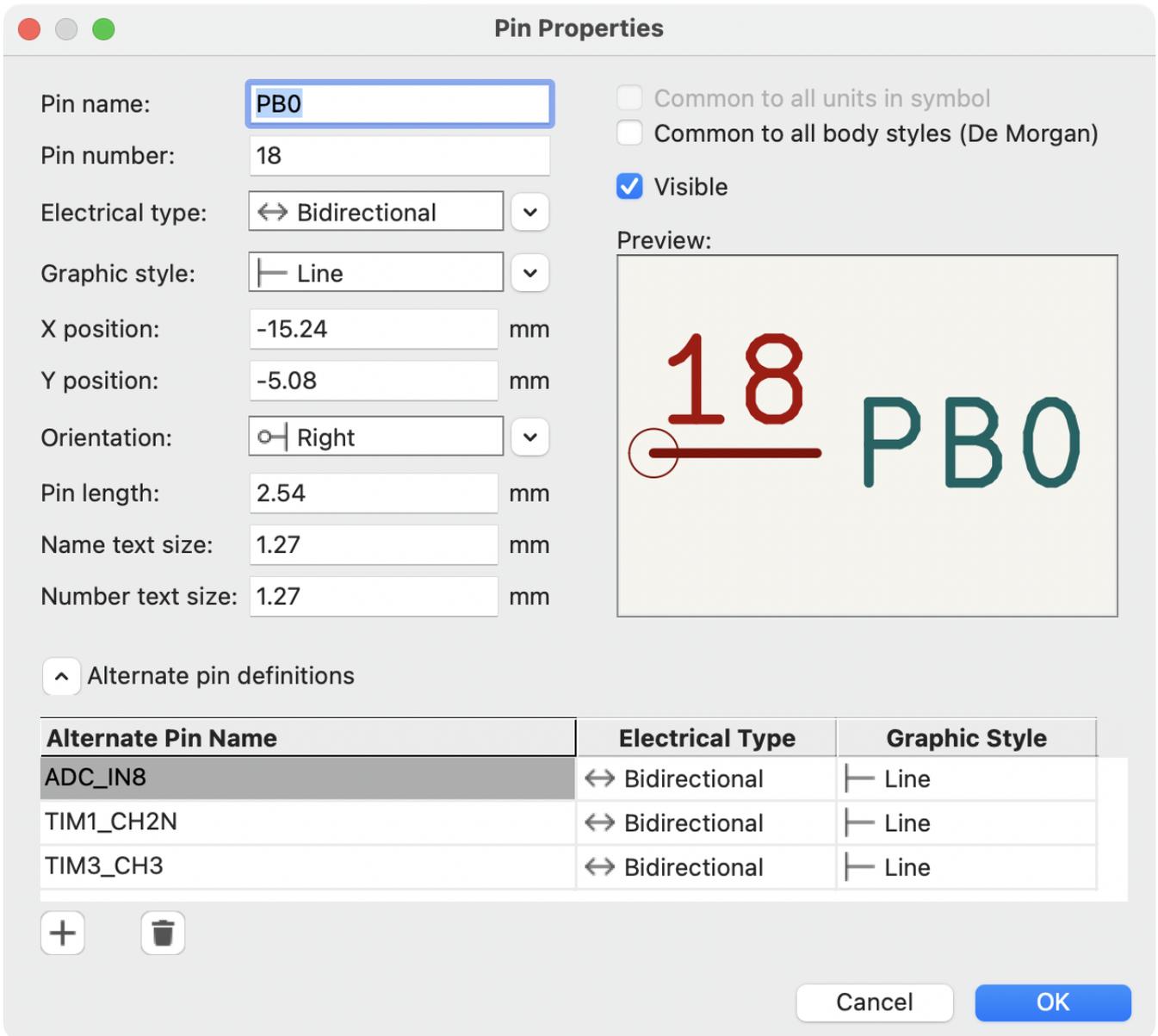


Count	Number	Name	Electrical Type	Graphic Style	Orientation	X Position	Y Position	Visible	Unit
1	1	~	↳ Output	├ Line	↳ Left	7.62 mm	0 mm	<input checked="" type="checkbox"/>	A
1	2	-	→ Input	├ Line	⇨ Right	-7.62 mm	2.54 mm	<input checked="" type="checkbox"/>	A
1	3	+	→ Input	├ Line	⇨ Right	-7.62 mm	-2.54 mm	<input checked="" type="checkbox"/>	A
1	4	V-	→ Power input	├ Line	↑ Up	-2.54 mm	7.62 mm	<input checked="" type="checkbox"/>	C
1	5	+	→ Input	├ Line	⇨ Right	-7.62 mm	-2.54 mm	<input checked="" type="checkbox"/>	B
1	6	-	→ Input	├ Line	⇨ Right	-7.62 mm	2.54 mm	<input checked="" type="checkbox"/>	B
1	7	~	↳ Output	├ Line	↳ Left	7.62 mm	0 mm	<input checked="" type="checkbox"/>	B
1	8	V+	→ Power input	├ Line	↓ Down	-2.54 mm	-7.62 mm	<input checked="" type="checkbox"/>	C

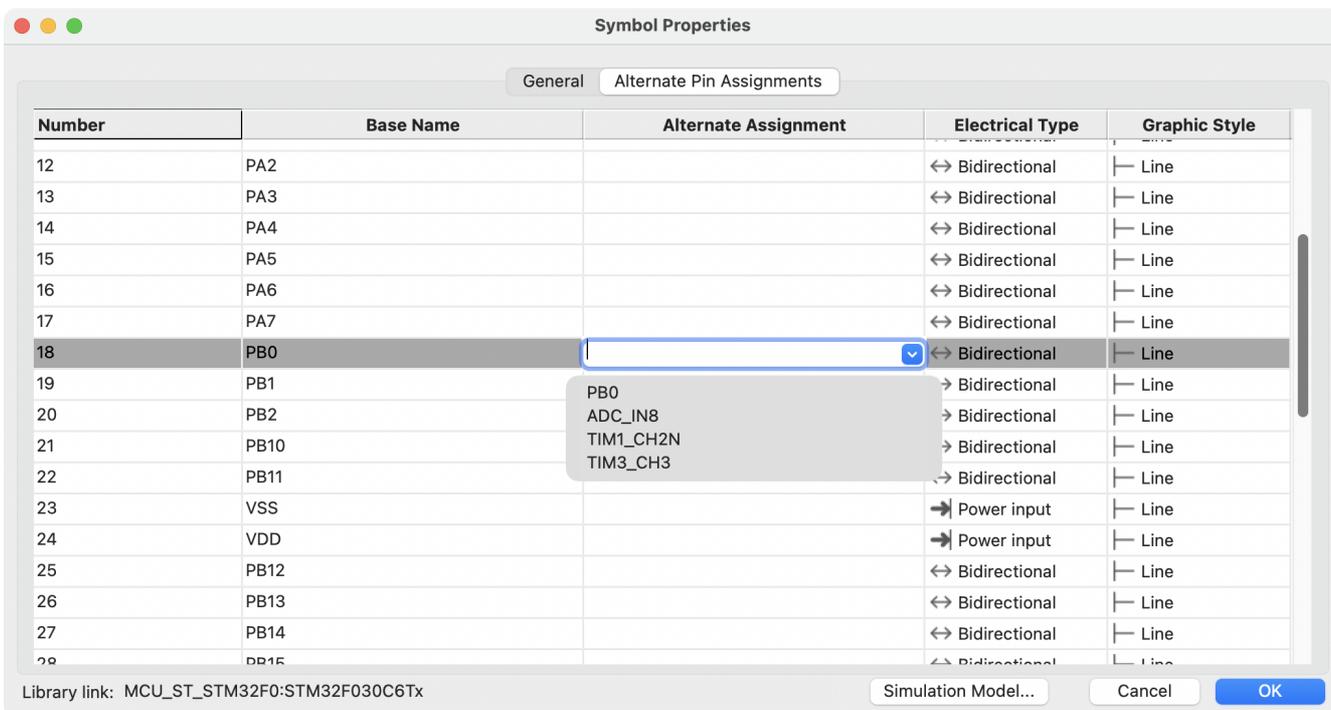
备用引脚定义

备用引脚定义引脚可以添加备用引脚定义。替代引脚定义允许用户在原理图中放置符号时为引脚选择不同的名称、电气类型和图形风格。这可用于具有多种功能的引脚，如微控制器引脚。

替代引脚定义是在引脚属性对话框中添加的，如下图所示。每个备用定义包含一个引脚名称、电气类型和图形风格。这个微控制器引脚的所有外围功能都在符号中定义为备用引脚名称。



一旦符号被放置在原理图中，就可以在原理图编辑器中选择备用引脚的定义。备用引脚是在符号属性对话框的备用引脚分配选项卡中分配的。备用定义可在备用分配栏的下拉菜单中选择。



符号字段

所有库中的符号都定义有四个默认字段。每当创建或复制一个符号时，就会创建位号、值、封装分配和数据表链接字段。只有位号是必需的。

The `Footprint` field, if used, contains a reference to a footprint for the symbol. The format is `LIBNAME:FOOTPRINTNAME`, where `LIBNAME` is the name of the footprint library in the footprint library table (see the [Footprint Library Table](#) section in the PCB Editor manual) and `FOOTPRINTNAME` is the name of the footprint in the library `LIBNAME`.

在库中定义的符号通常只有这四个默认字段。额外的字段，如供应商、零件编号、单位成本等，可以添加到库中的符号中，但一般是在原理图编辑器中完成，这样额外的字段就可以应用到原理图中的所有符号。

NOTE

创建额外的空符号字段的一个方便方法是使用定义字段名模板。字段名模板定义空字段，当每个符号插入原理图时，这些字段会被添加到每个符号中。字段名模板可以在原理图编辑器首选项中全局定义（针对所有原理图），也可以在原理图设置对话框中局部定义（针对每个工程）。

NOTE

如果你想在符号字段中管理大量的元件数据，可以考虑使用《数据库，数据库库》。

编辑符号字段

要编辑一个现有的符号字段，请双击该字段，选择它或悬停并按 **E**，或右击字段文本并选择 **属性...**。

要添加新字段、删除可选字段或编辑现有字段，请使用主工具栏上的  图标，打开《符号属性,符号属性对话框》。

字段是与符号相关的文本信息。不要将它们与符号的图形表示中的文字混淆。

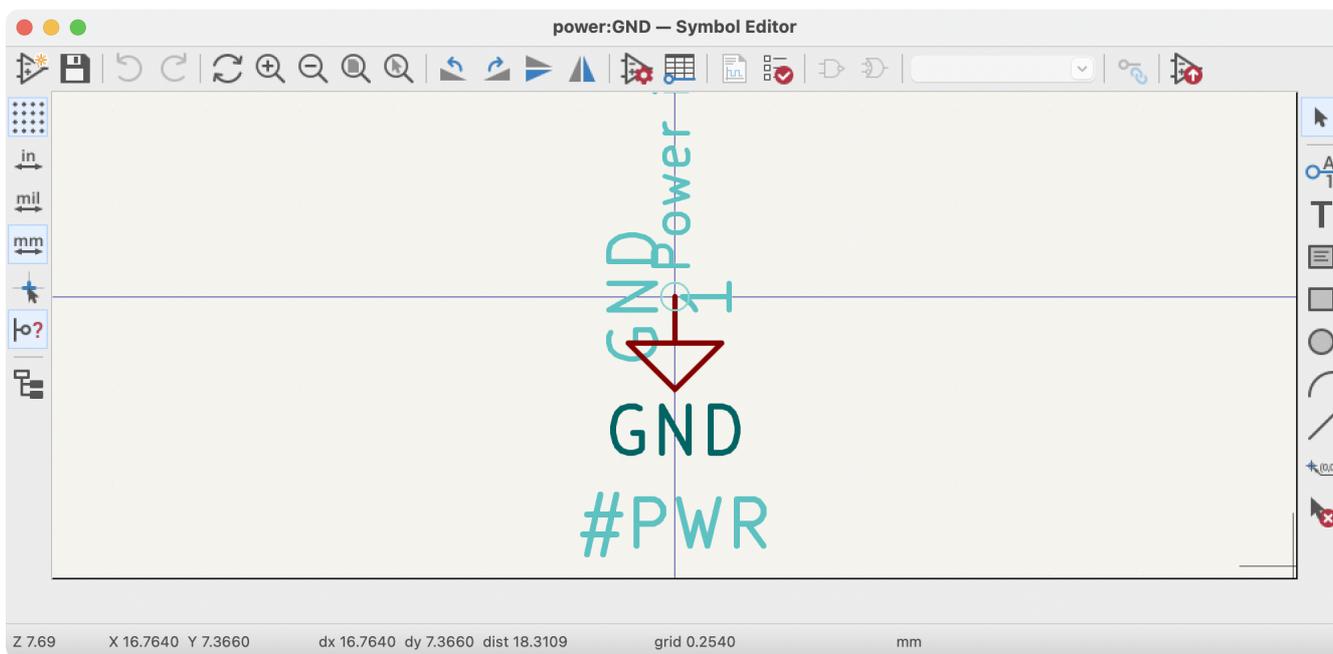
创建电源符号

Power symbols are symbols that are used to label a wire as part of a global power net, like `VCC` or `GND`. The behavior of power symbols is described in the [electrical connections section](#). Power symbols are handled

and created the same way as normal symbols, but there are several additional considerations described below.

将电源符号放在一个专门的库中可能是有用的。KiCad 的符号库将电源符号放在 `power` 库中，用户可以创建库来存储自己的电源符号。如果在符号的属性中勾选了 **定义为电源符号** 框，该符号就会出现在原理图编辑器的 **添加电源符号** 对话框中，以便于访问。

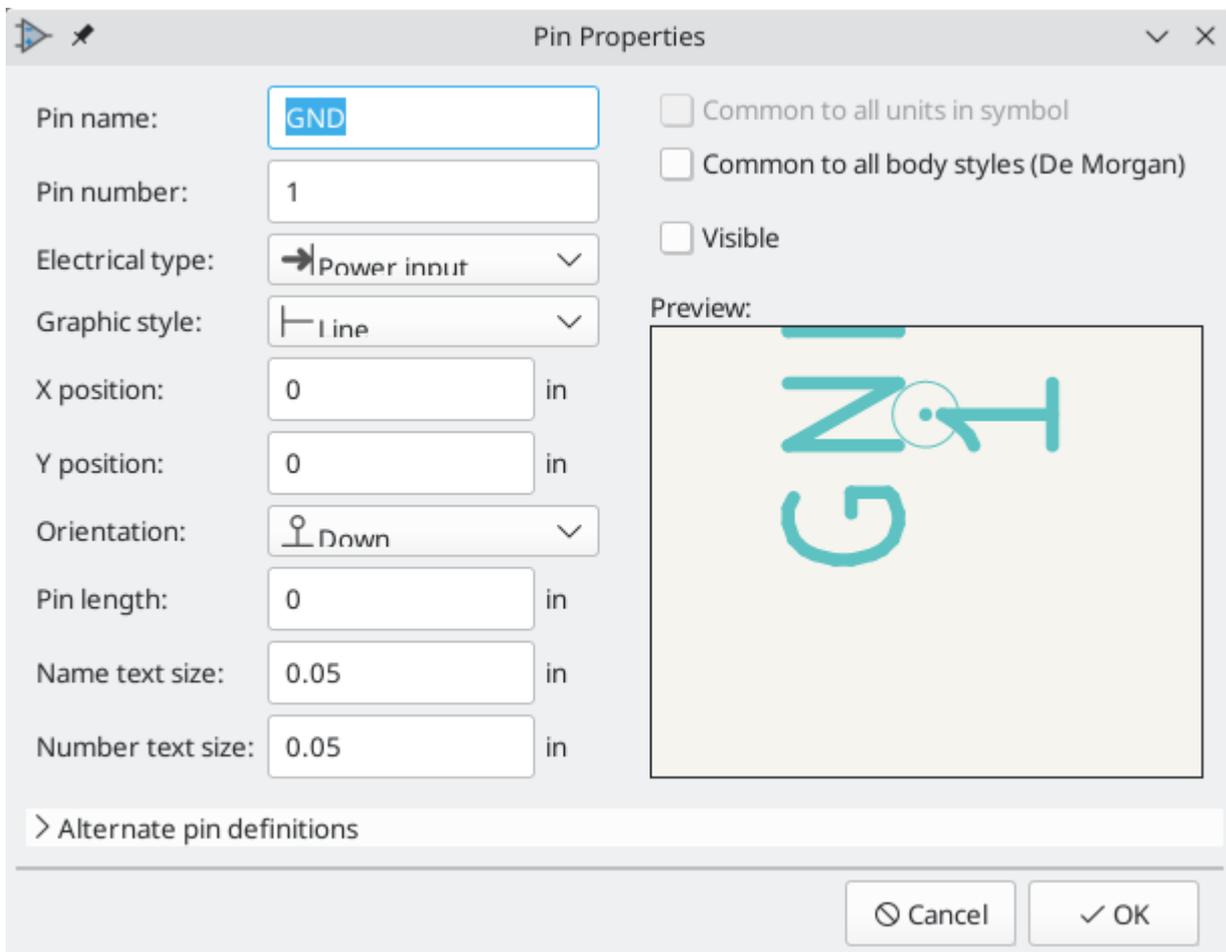
下面是一个 `GND` 电源符号的例子。



电源符号包括一个标记为不可见的 "电源输入" 类型的引脚。它们还必须检查 **定义为电源符号** 的属性。《隐藏的电源引脚,隐形的电源输入引脚》有一个特殊的属性，即根据引脚名称进行隐性的全局连接。

NOTE

如果电源符号勾选了 **定义为电源符号** 属性，那么电源输入引脚就不需要被标记为不可见。然而，惯例是无论如何都要使这些引脚不可见。



要创建一个电源符号，请使用以下步骤：

- 添加一个 **电源输入** 类型的引脚，**可见** 不被选中，并根据所需的网络命名引脚。使引脚的编号为 1 长度为 0，并将图形样式设置为 **直线**。引脚名称建立了与网络的连接；在这种情况下，引脚将自动连接到网络 **GND**。引脚编号、长度和线条样式在电气上并不重要。
- 将引脚放在符号锚点上。
- 使用形状工具来绘制符号图形。
- 设置符号值。符号值在电气上并不重要，但它会显示在原理图中。为了避免混淆，它应该与引脚名称相匹配（这决定了连接的网络名称）。
- 勾选符号属性窗口中的 **定义为电源符号** 框。这使得该符号出现在 **添加电源符号** 对话框中，使原理图中的“值”字段成为只读，防止该符号被分配一个封装，并将该符号从电路板、BOM 和网表中排除。
- 设置符号参考并取消勾选 **显示** 框。除了第一个字符外，参考文本并不重要，它应该是 `#`。对于上面所示的电源符号，位号可以是 `#GND`。位号以 `#` 开头的符号不会被添加到 PCB 上，不包括在材料清单的输出或网表中，也不能在封装分配工具中分配一个封装。如果一个电源符号的参考文献不是以 `#` 开头，当运行批注或封装分配工具时，该字符将被自动插入。

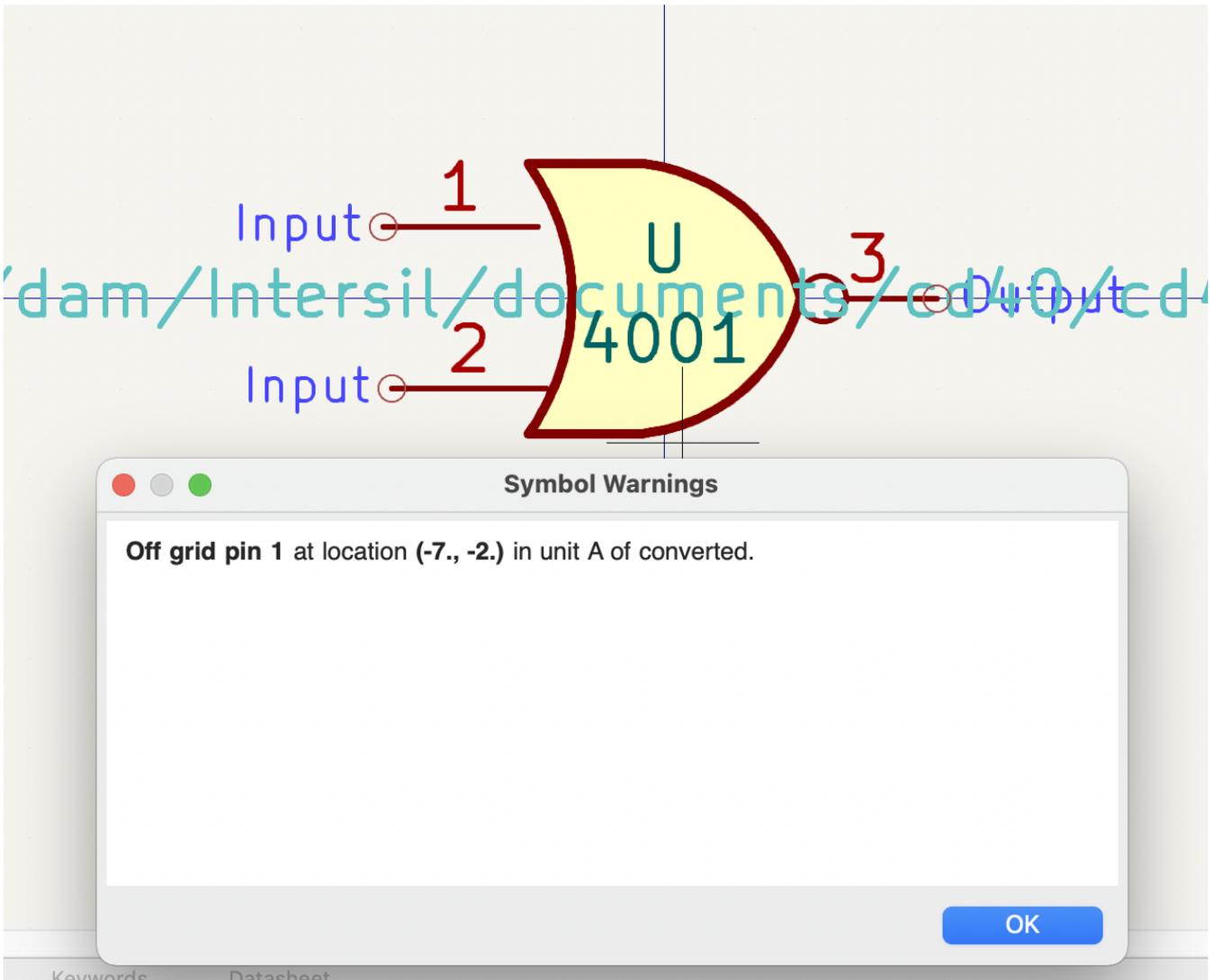
创建一个新的电源符号的一个更简单的方法是使用另一个符号作为起点，《从另一个符号创建一个符号，如前面所述》。

NOTE

连接的网名是由电源符号的 **引脚名称** 决定的，而不是符号的名称或值。在修改现有的电源符号时，请确保重新命名引脚，以便新符号连接到适当的电源网络。这意味着电源符号网络名称只能在符号编辑器中改变，而不能在原理图中改变。

检查符号

符号编辑器可以检查你的符号中的常见问题。使用顶部工具栏上的  按钮运行符号检查器。



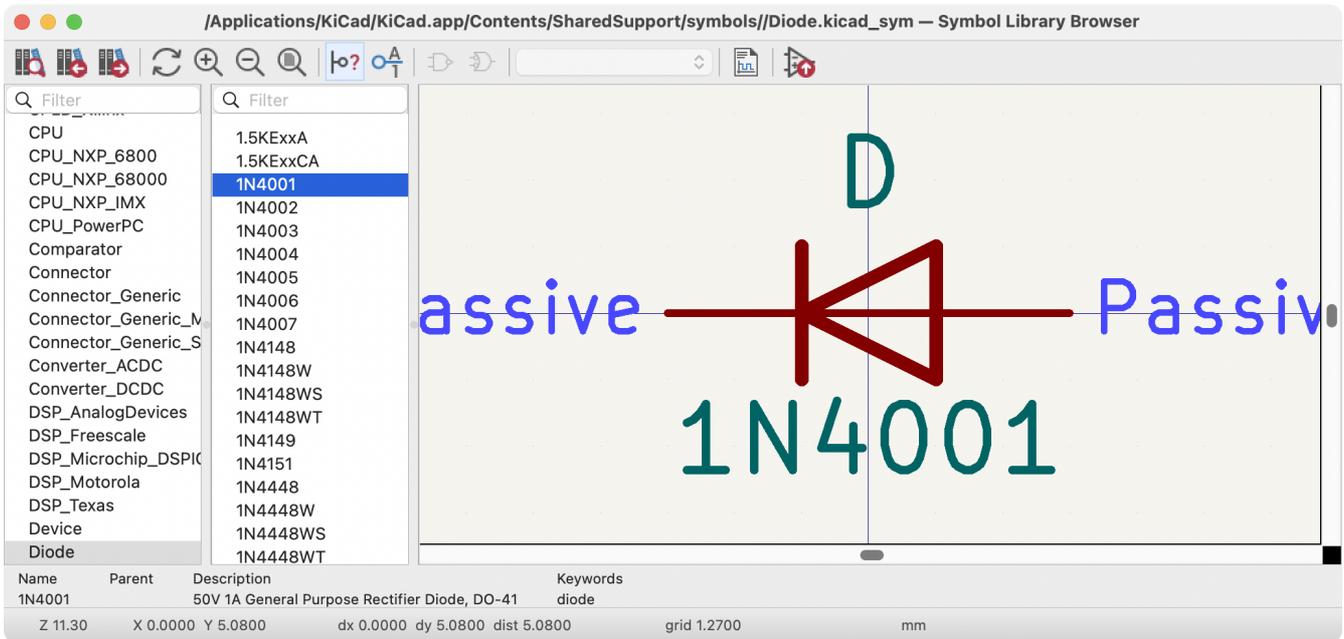
符号检查器检查的是：

- 错位的引脚
- 符号单元之间重复的引脚
- 设计不正确的《电源符号，电源符号》。电源符号应该有：
 - 一个单位
 - 没有备用的主体样式
 - 一个单一的引脚，它要么是电源输出类型（见《电源标记，电源标记》），要么是不可见的电源输入类型（见《电源符号，电源符号》）。
- 非法的位号前缀：位号前缀不应该以数字或 ? 结尾
- 《隐藏的电源引脚,隐藏的电源输入引脚》在非电源符号中：这些创建了隐含的连接，不推荐使用

浏览符号库

符号库浏览器允许你快速检查符号库的内容。符号库浏览器可以通过点击主工具栏上的  图标，查看 → 符号库浏览器，或点击 **选择符号** 窗口中的 **用浏览器** 选择来访问。

要检查一个库的内容，从左侧窗格的列表中选择一个库。所选库中的所有符号将出现在第二个窗格中。选择一个符号名称来查看该符号。



双击符号名称或使用  按钮将符号添加到原理图中。

顶部的工具条包含以下命令：

	在 选择符号 窗口中打开选定的符号。
	在库中选择前一个符号。
	选择库中的下一个符号。
	缩放工具。
	选择标准或备用的德摩根符号表示，如果适用的话。
	选择多单元符号的单元。
	打开符号的数据手册，如果它被定义。
	将当前符号插入原理图中。

仿真器

KiCad provides an embedded electrical circuit simulator using [ngspice](#) as the simulation engine. All but the simplest simulations will require you to add simulation models to your symbols before you can simulate your schematic.

When working with the simulator, the official `Simulation_SPICE` symbol library may be useful. It contains common symbols used for simulation, like voltage/current sources and transistors with pins numbered to match the ngspice node order specification.

KiCad includes a few demo projects to illustrate the simulator's capabilities. They can be found in the `demos/simulation` directory.

Value notation

The simulator supports several notations for writing numerical values:

- Plain notation: `10100`, `0.003`,
- Scientific notation: `1.01e4`, `3e-3`,
- Prefix notation: `10.1k`, `3m`.
- RKM notation: `4k7`, `10R`.

You can mix prefix and scientific notations. As such, `3e-4k` is a valid input and is equivalent to `0.3`. The list of valid prefixes is shown below. The prefixes are case sensitive.

Prefix	Meaning	Value
a	atto	10^{-18}
f	femto	10^{-15}
p	pico	10^{-12}
n	nano	10^{-9}
u	micro	10^{-6}
m	milli	10^{-3}
k	kilo	10^3
M	mega	10^6
G	giga	10^9
T	tera	10^{12}
P	peta	10^{15}
E	exa	10^{15}

NOTE

Raw SPICE Element models and directives are passed to ngspice directly, without KiCad reformatting the values for ngspice to consume. Ngspice uses a different, case-insensitive notation: 1 mega (10^6) is denoted there as 1Meg, while 1M is 1 milli (10^{-3}). Depending on the compatibility mode selected, ngspice may not support the same value notations as KiCad, so care should be taken when using raw SPICE elements and simulation directives.

分配模型

You need to assign simulation models to symbols before you can simulate your circuit.

Each symbol can have only one model assigned, even if the symbol consists of multiple units. For symbols with multiple units, you should assign the model to the first unit.

SPICE model information is stored as text in symbol fields. Therefore you may define it in either the symbol editor or the schematic editor. To assign a simulation model to a symbol, open the Symbol Properties dialog and click the **Simulation Model...** button, which opens the Simulation Model Editor dialog.

You can exclude a symbol from simulation entirely by checking the **exclude from simulation** checkbox in the Symbol Properties dialog.

Inferred models

Resistor, inductor, and capacitor models can be inferred, which means that KiCad will detect that they are passives and automatically assign an appropriate simulation model. Therefore they do not require any special settings; users only need to set the Value field of the symbol.

KiCad infers simulation models for symbols based on the following criteria:

- The symbol has exactly two pins,
- The reference designator begins with R, L or C.

Inferred models are ideal models. If the simulation requires a non-ideal model, you must explicitly assign a model.

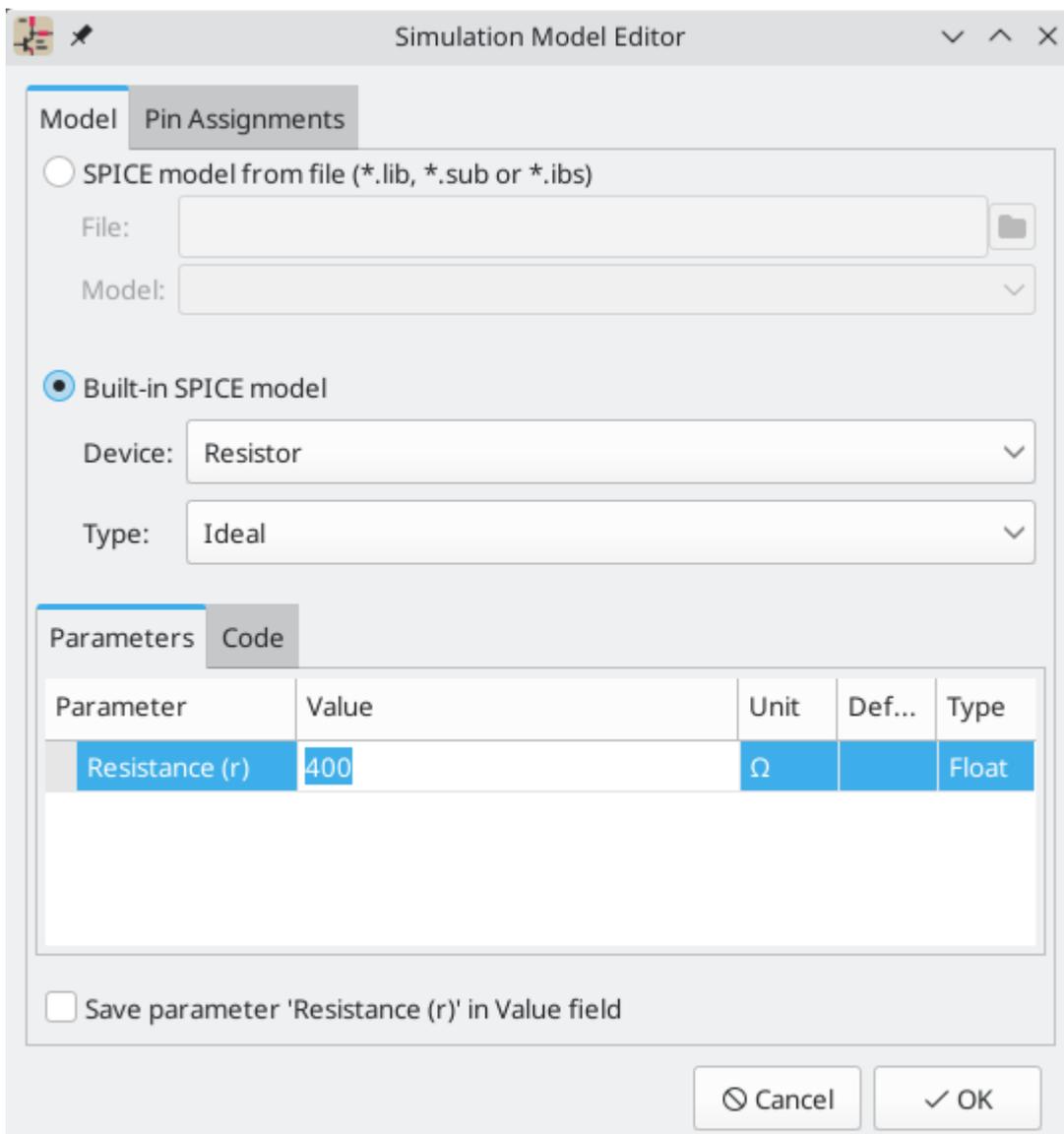
Built-in models

KiCad offers several standard simulation models. They do not require an external model file. The following devices are available:

- Resistors (including potentiometers), capacitors, inductors
- Transmission lines
- Switches
- Voltage and current sources
- Diodes
- Transistors (BJT, MOSFET, MESFET, and JFET)
- Raw SPICE elements

To add a built-in model to a symbol, open the Simulation Model Editor dialog (**Symbol Properties** → **Simulation Model...**) and select **Built-in SPICE model**.

Refer to the [ngspice documentation](#) for more details about these models.



Device sets the type of device to simulate: a resistor, BJT, voltage source, etc. This value is stored in the symbol's `Sim.Device` field.

Type selects the type of model to use for the device. Most devices have several types of models to choose from, which may vary in their degree of accuracy, which characteristics they are optimized for, and what parameters they have available. Refer to the [ngspice documentation](#) for details. This value is stored in the symbol's `Sim.Type` field.

The **parameters** tab displays the parameters of the model and lets you edit them. For example, a resistor's resistance, a voltage source's waveform, a MOSFET's width and length, etc. Any parameters that differ from the model's defaults are stored in the symbol's `Sim.Params` field.

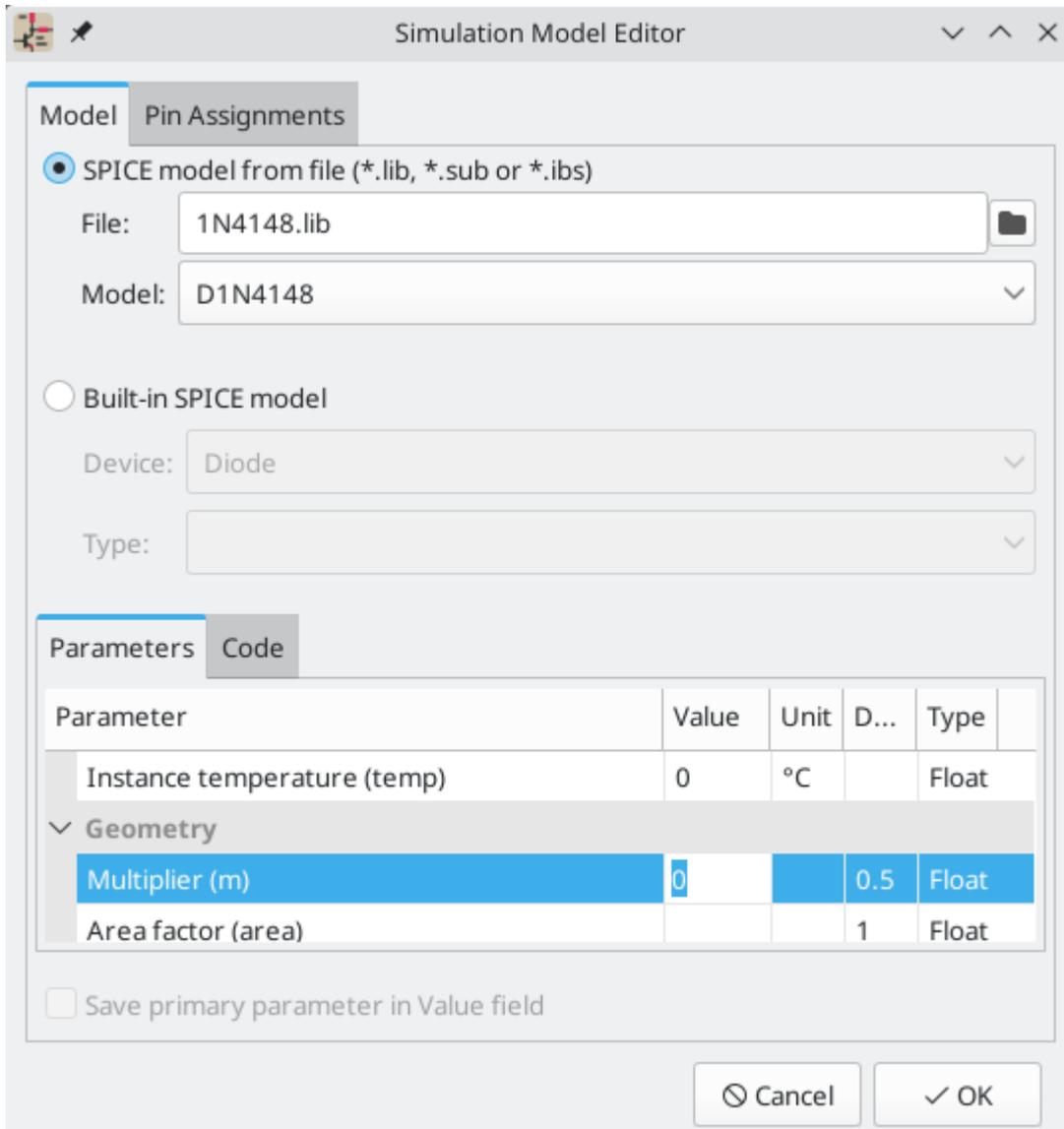
The **code** tab displays the generated SPICE model as it will be written to the SPICE netlist for simulation.

The **Save parameter '<parameter name>' in Value field** checkbox uses the symbol's `Value` field for storing parameters instead of the `Sim.Params` field. This may make it easier to edit simple models from the schematic, without opening the Simulation Model Editor. This option is only available for ideal passive models (R, L, C) and DC sources. If the field `Sim.Params` exists in the symbol, it will take priority over the `Value` field.

Library models

KiCad can load SPICE models from external files. The models must be in a standard SPICE format and not encrypted.

To load a model from an external file, open the Simulation Model Editor dialog (**Symbol Properties** → **Simulation Model...**) and select **SPICE model from file**.



The screenshot shows the 'Simulation Model Editor' dialog box. It has two tabs: 'Model' and 'Pin Assignments'. The 'Model' tab is selected and contains two radio buttons. The first radio button, 'SPICE model from file (*.lib, *.sub or *.ibs)', is selected. Below it are a 'File:' text box containing '1N4148.lib' and a 'Model:' dropdown menu containing 'D1N4148'. The second radio button, 'Built-in SPICE model', is unselected. Below it are a 'Device:' dropdown menu containing 'Diode' and a 'Type:' dropdown menu. Below these are two tabs: 'Parameters' and 'Code'. The 'Parameters' tab is selected and contains a table with the following data:

Parameter	Value	Unit	D...	Type
Instance temperature (temp)	0	°C		Float
▼ Geometry				
Multiplier (m)	0		0.5	Float
Area factor (area)			1	Float

At the bottom of the dialog is a checkbox labeled 'Save primary parameter in Value field' which is unchecked. At the very bottom are 'Cancel' and 'OK' buttons.

File is the path to the model file to use. The path can be absolute or relative to the project folder. The path can also be relative to the value of `SPICE_LIB_DIR` if you have [defined that path variable](#). The library filename is saved in the symbol's `Sim.Library` field.

Model is the name of the desired model in the model file. If the file contains multiple models, they will all be listed in the dropdown. The selected model is listed in the symbol's `Sim.Name` field.

Parameters can be overridden (or additional parameters specified) using the **parameters** tab. All parameters specified in the selected model or the **parameters** tab are stored in the symbol's `Sim.Params` field.

The **code** tab displays the generated SPICE model as it will be written to the SPICE netlist for simulation.

NOTE

KiCad does not ship with SPICE models for symbols. SPICE models are usually available from device manufacturers.

Library models (IBIS)

IBIS (I/O Buffer Information Specification) files are an alternative to SPICE models for modeling the behavior of input/output buffers on digital parts. In order to load an IBIS file, users should follow the procedure for SPICE library models, but provide a `.ibs` file.

The screenshot shows the 'Simulation Model Editor' dialog box. The 'Pin Assignments' tab is selected, and the 'SPICE model from file (*.lib, *.sub or *.ibs)' radio button is chosen. The 'File' field contains 'ibis_v1_1.ibs', 'Component' is 'Virtual', 'Pin' is '4 - Y', and 'Model' is 'Output'. Below this, the 'Built-in SPICE model' radio button is unselected, with 'Device' set to 'IBIS Model' and 'Type' set to 'Rectangular wave driver'.

The 'Parameters' tab is also visible, showing a table of parameters:

Parameter	Value	Unit	Def...	Type
Power supply (vcc)	typ		typ	String
Parasitic pin resistance (rpin)	typ		typ	String
Parasitic pin inductance (lpin)	typ		typ	String
Parasitic pin capacitance (cpin)	typ		typ	String
Waveform				
ON time (ton)	100n	s		Float
OFF time (toff)	100n	s		Float
Delay (td)	0	s	0	Float
Number of cycles (n)	50		1	Int

At the bottom of the dialog, there is a checkbox for 'Save primary parameter in Value field' which is unchecked, and 'Cancel' and 'OK' buttons.

File is the path to the model file to use. The path can be absolute or relative to the project folder. The path can also be relative to the value of `SPICE_LIB_DIR` if you have [defined that path variable](#). The library filename is saved in the symbol's `Sim.Library` field. If an IBIS model file is loaded, the remaining fields in the dialog will relate to the IBIS model.

Component selects which component from the IBIS file to use, as IBIS files can contain multiple components. The component name is saved in the symbol's `Sim.Name` field.

Pin selects which pin in the IBIS model to simulate. The selected pin must be mapped to a symbol pin in the **Pin Assignments** tab. The chosen pin's number is saved in the symbol's `Sim.Ibis.Pin` field.

Model is the list of models available for the selected pin, for example an input or an output. The chosen model name is saved in the symbol's `Sim.Ibis.Model` field.

Type selects what the pin should do in the simulation. A pin can be a passive **device** that doesn't drive any value; it can be a **DC driver** that drives high, low, or high-impedance; or it can be a **rectangular wave** or **PRBS driver**. This value is stored in the symbol's `Sim.Type` field.

The **Parameters** tab lets you see and edit the parameters of the model. For each parameter, you can switch between a minimum, typical, or maximum value, as defined in the IBIS file. You can also choose the parameters of the driven waveform, depending on the pin's chosen **type**. Any parameters that differ from the defaults are stored in the symbol's `Sim.Params` field.

NOTE

KiCad does not ship with IBIS models for symbols. IBIS models are usually available from device manufacturers.

NOTE

KiCad's `Simulation_SPICE` symbol library provides several symbols that may be useful for IBIS simulations. `IBIS_DEVICE` can be used for device (input) pins, while `IBIS_DRIVER` can be used for simulating driver pins. There are also variants of each for differential pins.

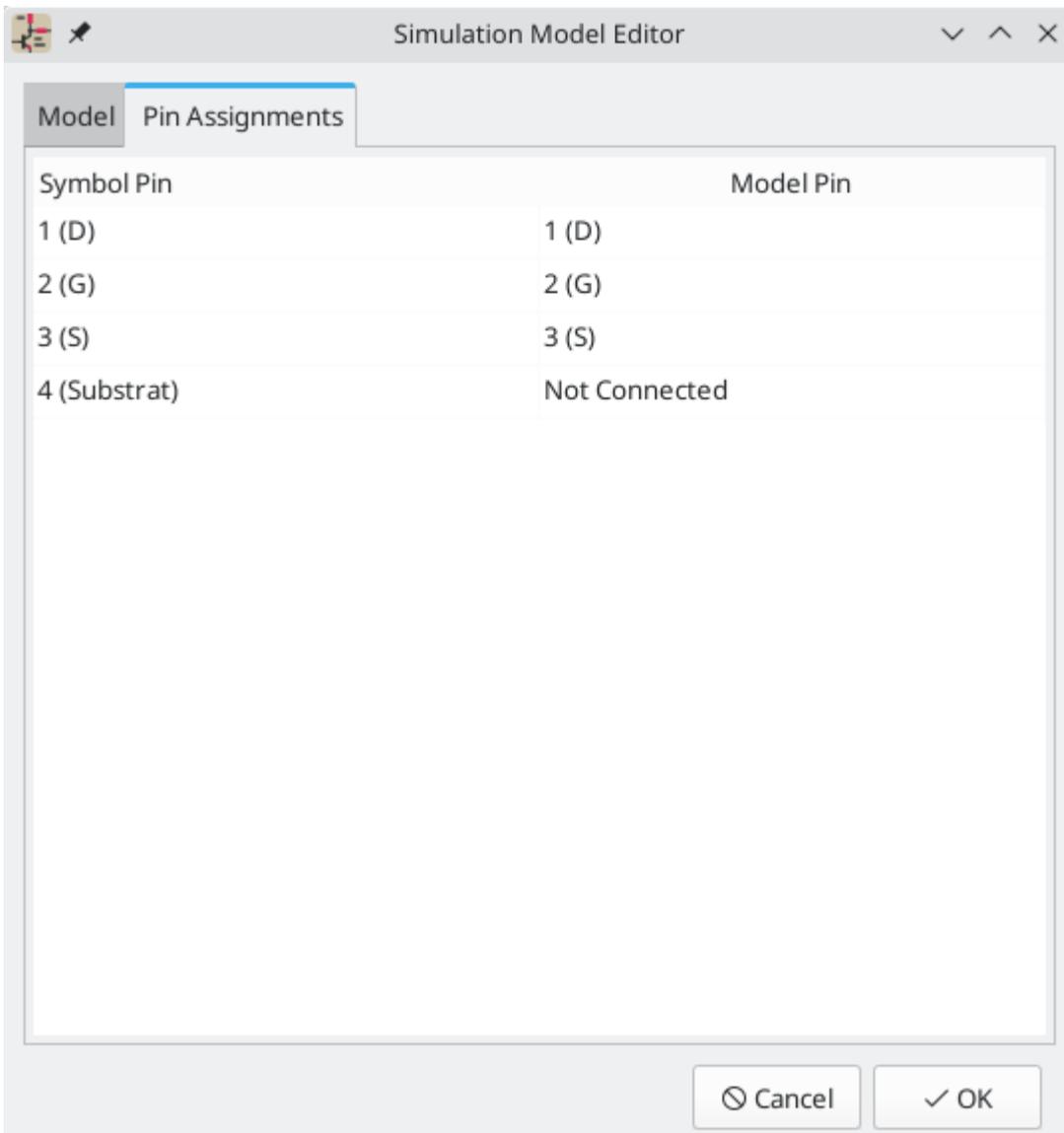
Pin Assignment

Simulation models may have their pins numbered differently than the corresponding symbol. For example, SPICE models for diodes usually consider pin 1 to be the anode, while schematic symbols are usually drawn with pin 1 as the cathode.

You can use the Simulation Model Editor's **Pin Assignments** tab to map the symbol's pins to the simulation model pins.

NOTE

Always make sure symbol pins are correctly mapped to simulation model pins.



The left column displays the name and number of each symbol pin. For each symbol pin, you can select the corresponding pin from the simulation model in the dropdown in the right column.

When you use a subcircuit model, the dialog displays the model's code under the pin assignments for use as a reference while assigning pins.

SPICE directives

It is possible to add SPICE directives by placing them in text fields on a schematic sheet. This approach is convenient for defining the default simulation type. The list of supported directives in text fields is:

- Directives starting with a dot (e.g. `.tran 10n 1m`)
- Coupling coefficients for inductors (e.g. `K1 L1 L2 0.89`)

It is not possible to place additional components using text fields.

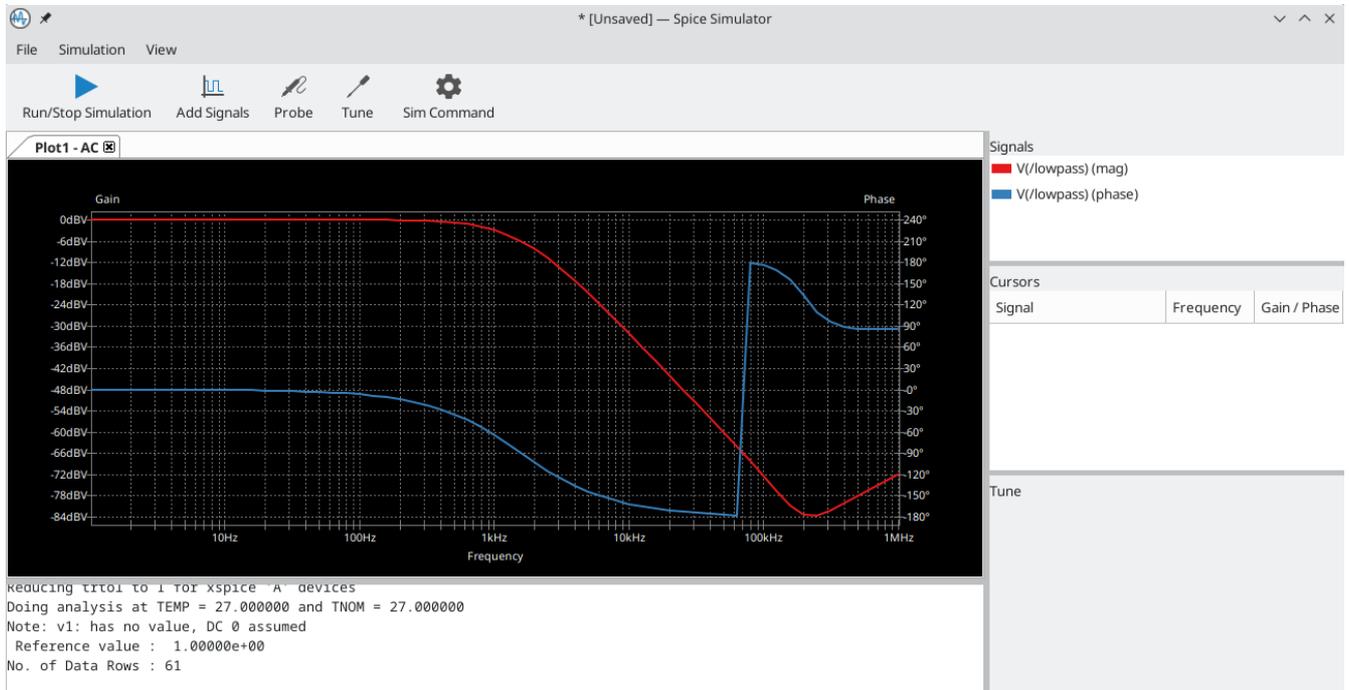
If a simulation command is included in schematic text, the simulator will use it as the simulation command when you open the simulator. However, you can override it in the Simulation Command dialog.

Refer to the [ngspice documentation](#) for more details about SPICE directives.

Running simulations

User Interface

To run a simulation, open the SPICE simulator dialog by clicking **Inspect** → **Simulator** in the schematics editor window or using the  button in the top toolbar.



该对话框分为几个部分：

- The top of the window has a toolbar with buttons for commonly used actions.
- The main part of the window graphically shows the simulation results. Signals need to be probed before they are displayed in the plot.
- Below the plot panel, the output console shows logs from the ngspice simulation engine.
- The right side of the window displays a list of plotted signals, a list of active cursors, and a tuning tool for adjusting component values based on simulation results.

Workbooks

Workbooks are files that store information about the simulation environment, including simulation setup parameters and the list of displayed signals. They can be used to store the setup for a set of analyses, which can then be reloaded and rerun at a later time.

You can save and load a workbook using **File** → **Save Workbook** and **File** → **Open Workbook**.

NOTE

Workbooks store simulation setup information, but they do not store simulation results. You can export simulation results to PNG (graphics) or CSV (data points) with **File** → **Export Current Plot as PNG...** and **File** → **Export Current Plot as CSV...**

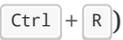
Running a Simulation

Before running a simulation, you need to select a simulation type and choose the simulation parameters. This can be done using **Simulation** → **Settings...** or the **Sim Command** button () and then selecting one of the available analysis types:

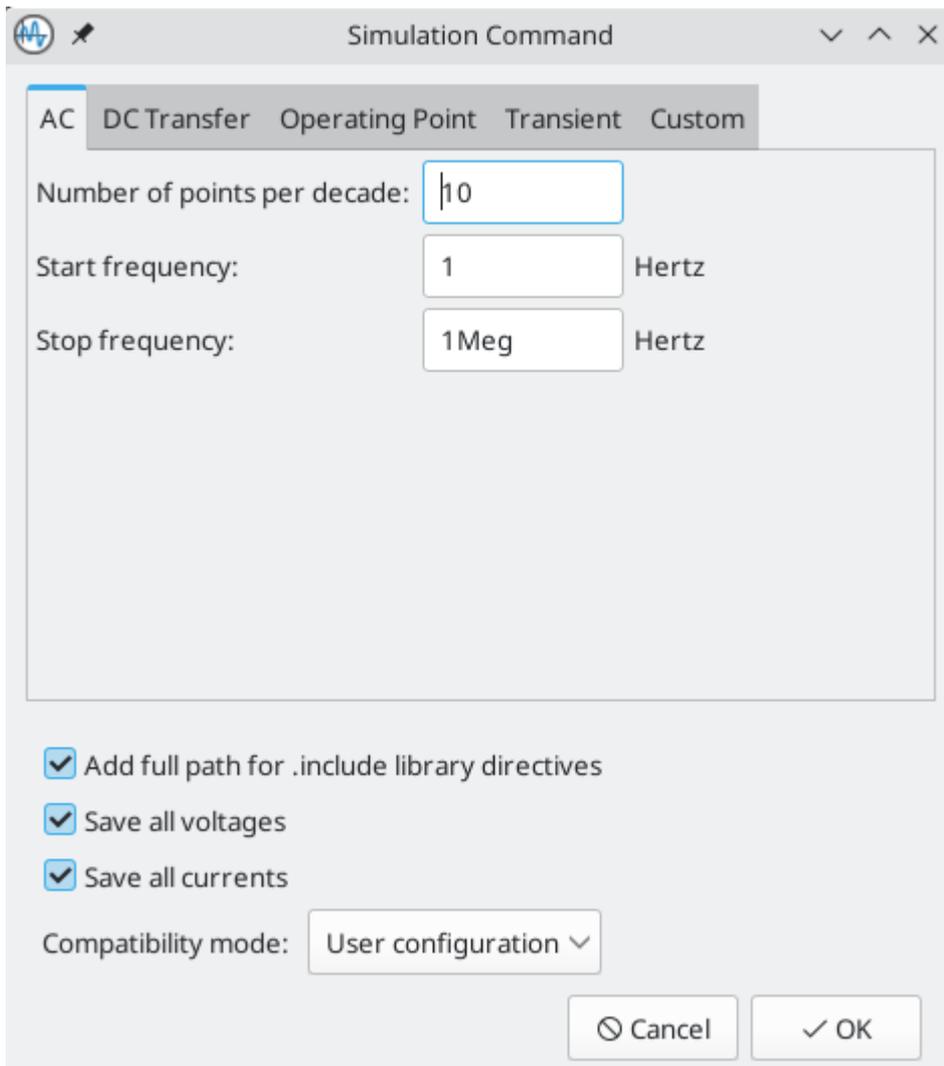
- 交流
- 直流传输
- Operating Point
- 瞬态
- 自定义

Analysis types are further explained in the [ngspice documentation](#).

An alternative way to configure a simulation is to type [SPICE directives](#) into text fields on schematics. Any text field directives related to a simulation command are overridden by the settings selected in the dialog. This means that once a simulation has run, the dialog overrides the schematic directives until the simulator is reopened.

Once the simulation command is set, users can start a simulation with **Simulation** → **Start Simulation** () or the **Run/Stop Simulation** button ()

AC analysis



The image shows a software dialog box titled "Simulation Command" with a standard window control bar (minimize, maximize, close). The dialog has five tabs: "AC", "DC Transfer", "Operating Point", "Transient", and "Custom". The "AC" tab is selected. Inside the dialog, there are three input fields: "Number of points per decade:" with the value "10", "Start frequency:" with the value "1" and the unit "Hertz", and "Stop frequency:" with the value "1Meg" and the unit "Hertz". Below these fields are three checked checkboxes: "Add full path for .include library directives", "Save all voltages", and "Save all currents". At the bottom left, there is a "Compatibility mode:" label followed by a dropdown menu showing "User configuration". At the bottom right, there are two buttons: "Cancel" and "OK".

Calculates the small signal AC behavior of the circuit in response to a stimulus. Performs a decade sweep of stimulus frequency.

To run an AC analysis, you must choose a number of points to measure per decade and the start and end frequencies for the decade sweep.

The output is displayed as a Bode plot (output magnitude and phase vs. a decade sweep of stimulus frequency).

DC transfer analysis

The image shows a software dialog box titled "Simulation Command" with a window icon, a pencil icon, and standard window controls (minimize, maximize, close). The dialog has five tabs: "AC", "DC Transfer" (which is selected and highlighted in blue), "Operating Point", "Transient", and "Custom".

Inside the "DC Transfer" tab, there are two columns for "Source 1" and "Source 2". The "Source 2" column has a checkbox that is currently unchecked. Each source column contains the following controls:

- Sweep type:** A dropdown menu with "V" selected.
- Source:** A dropdown menu.
- Starting value:** A text input field followed by the unit "Volts".
- Final value:** A text input field followed by the unit "Volts".
- Increment step:** A text input field followed by the unit "Volts".

Below these source settings is a button labeled "Swap sources".

At the bottom of the dialog, there are three checked checkboxes:

- Add full path for .include library directives
- Save all voltages
- Save all currents

Below the checkboxes is a "Compatibility mode:" label followed by a dropdown menu showing "User configuration".

At the bottom right, there are two buttons: "Cancel" and "OK".

Calculates the DC behavior of the circuit while sweeping one or two sources (voltage or current), resistors, or the simulation temperature.

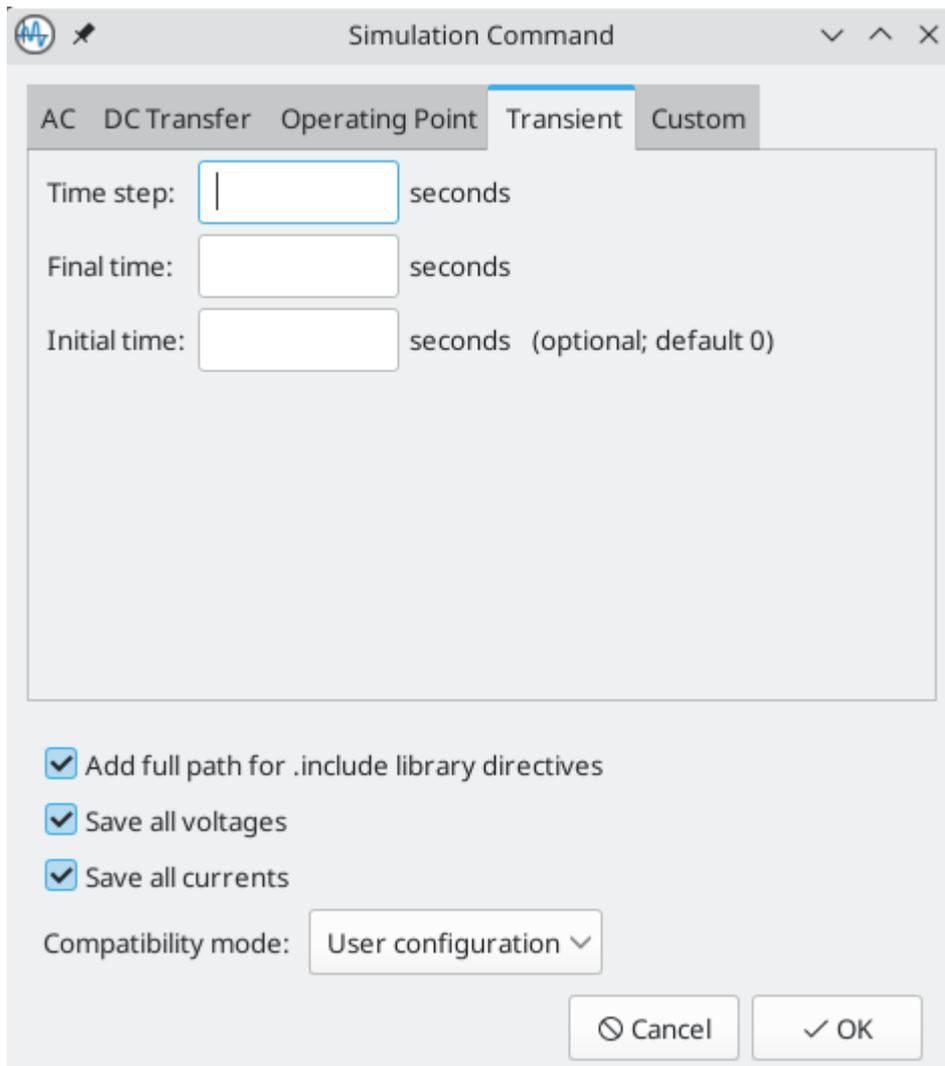
To run a DC analysis, you must choose what type of sweep(s) to perform, which source, resistor, or temperature value(s) to sweep, and what the sweep range(s) and step(s) should be.

The output is displayed as a plot.

Operating point analysis

Calculates the DC operating point of the circuit. This analysis has no options, and results are printed to the SPICE log.

Transient analysis



The image shows a software dialog box titled "Simulation Command" with a tabbed interface. The "Transient" tab is selected. It contains three input fields for time settings: "Time step:" (with a vertical cursor), "Final time:", and "Initial time:". Below these are three checked checkboxes: "Add full path for .include library directives", "Save all voltages", and "Save all currents". A "Compatibility mode:" dropdown menu is set to "User configuration". At the bottom right are "Cancel" and "OK" buttons.

Simulation Command

AC DC Transfer Operating Point **Transient** Custom

Time step: seconds

Final time: seconds

Initial time: seconds (optional; default 0)

Add full path for .include library directives

Save all voltages

Save all currents

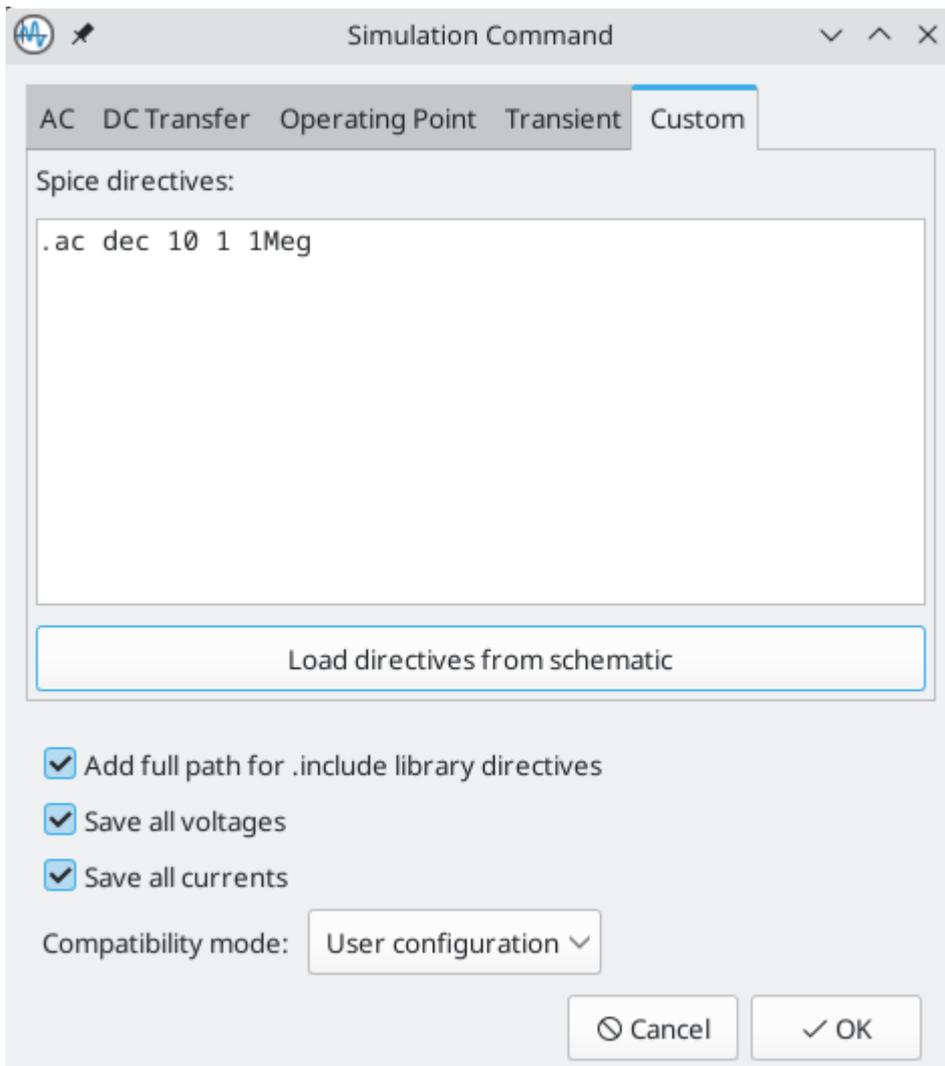
Compatibility mode:

Calculates the time-varying behavior of the circuit.

To run a transient analysis, you need to specify a suggested time step and a final simulation time. You can optionally specify a starting time to override the default of 0.

The output is displayed as a plot.

Custom analysis



A custom analysis lets you write SPICE commands to set up the analysis. Press the **Load directives from schematic** button to copy any SPICE directives in schematic text into the custom analysis textbox.

Refer to the [ngspice documentation](#) for more information about analysis commands.

Additional simulation settings

There are several simulation options that apply to all types of simulations.

Add full path for .include library directives controls whether to convert relative paths to absolute in `.include` directives in schematic text.

Save all voltages and **Save all currents** controls whether the simulator saves voltages and currents, respectively, for internal nodes of devices. When unchecked, the simulator only saves voltages and currents for external nodes.

The **Compatibility mode** dropdown selects the compatibility mode that the simulator uses to load models. The **User configuration** option refers to the user's `.spiceinit` ngspice configuration file. Compatibility modes are described in the [ngspice documentation](#).

Probing signals

Using the signal list

You can display a list of available signals with **Simulation** → **Add signals** (`Ctrl` + `A` hotkey or  button). Double clicking an element in the list will add it to list of probed signals. Several signals can be selected at once using the `Ctrl` / `Shift` keys.

Using the probe tool

A probe tool is available in the toolbar of the simulator window () , which provides a user-friendly way of selecting signals for plotting. When activated, users can click elements in the schematic editor. To probe a voltage, click a wire with the probe tool. When hovering over wires with the tool, the corresponding net is highlighted. To probe a current, click on a symbol pin. When hovering over a pin, the cursor will change from a voltage probe symbol to a current clamp symbol. It is not possible to probe power signals with this tool.

Using a SPICE directive

While ngspice supports the `.plot` directive, it cannot be used in current versions of KiCad.

Differential voltage

Probing differential voltages usually requires a simulation symbol. One is available in `Simulation_SPICE:VOLTMETER_DIFF` in the official libraries. This symbol has two terminals for differential voltage sensing, and one the user can probe.

NOTE

A SPICE model called `kicad_builtin_vdiff` is preassigned to the built-in `VOLTMETER_DIFF` symbol. Users should not define a SPICE model with the same name.

It is also possible to probe across two-terminal devices such as a resistor by placing the `.probe vd(X)` directive, where `X` is the name of the device (such as `R1`). Then in order to probe it, enter `vd_X` when adding a signal using the list of signals. Note that the signal will not appear in the list of signals, it should be typed in by the user. This method has the advantage of not adding extra symbols to the schematic sheet.

Remove a signal probe

In order to remove a signal probe, double click on it in the signal list, or right-click it and select **Remove Signal**.

Tuning components

It is possible to change passive (R, L, C) values using sliders to graphically adjust them. Whenever the slider is set to a new position, the simulation is run with the new parameters and plots are updated. In order to add a slider for a component, use **Simulation** → **Tune Component Value** or the  button in the toolbar, and then click on the component to tune.

- 顶部的文本字段设置了最大的元件值。
- 中间的文本字段设置实际的元件值。
- 底部的文本字段设置了最小元件值。
- The slider allows the user to modify the component value.

The **Save** button modifies component value on the schematic to the value selected with the slider.

- The  button removes the component from the Tune panel and restores its original value.

Visualizing results

Plotted quantities

This section applies to simulation commands that output vectors such as `.ac` and `.tran`.

Simulation results are visualized as plots. There can be multiple plots opened in separate tabs, but only the active one is updated when a simulation is executed. This way it is possible to compare simulation results for different runs.

Plots can be customized by toggling grid and legend visibility using the **View** menu. When a legend is visible, it can be dragged to change its position. You can toggle the plot background from dark to light with **View** → **White Background** and switch current and phase plots to use dotted lines instead of solid with **View** → **Dotted Current/Phase**.

For precise measurement, it is possible to add a cursor to a signal. To do so, right click on a signal name in the **Signals** pane and select **Show Cursor**. It is then possible to move the cursor to the point of interest. The values are then shown in the **Cursor** section of the window. To hide a cursor, right click on a signal and select **Hide Cursor**. Each signal may have one cursor displayed.

The following interactions are possible with the plot panel:

- 滚动鼠标滚轮来放大/缩小
- 右击打开上下文菜单，以调整视图
- 绘制一个选择矩形，以放大所选区域
- 拖动光标以改变其坐标

Numerical values

Some analysis types, such as the DC operating point analysis (`.op`), do not have any graphical output to plot. Instead, their output is printed in the SPICE console.

Exporting results

KiCad's simulator offers two ways to export results:

- As an image using **File** → **Export Current Plot as PNG...**
- As a `.csv` file containing the datapoints using **File** → **Export Current Plot as CSV...**

Troubleshooting

Incorrect netlist

It is possible to inspect the SPICE netlist with **Simulation** → **Show SPICE netlist...** This method of troubleshooting requires some SPICE knowledge, but spotting errors in the netlist can help determine the cause of simulation problems.

Simulation error

The output console displays messages from the simulator. It is advised to check the console output to verify there are no errors or warnings.

Convergence problems

In case the simulation does not converge in a reasonable amount of time (or not at all), it is possible to add the following SPICE directives.

WARNING

Changing convergence options can lead to erroneous results. These options should be used with caution.

```
.options gmin=1e-10  
.options abstol=1e-10  
.options reltol=0.003  
.options cshunt=1e-15
```

- `gmin` is the minimum conductance allowed by the program. The default value is `1e-12`.
- `abstol` is the absolute current error tolerance of the program. The default value is 1 pA.
- `reltol` is the relative error tolerance of the program. The default value is `0.001` (0.1%).
- `cshunt` adds a capacitor from each voltage node in the circuit to ground.

高级主题

配置和定制

NOTE

This section of the KiCad documentation has not yet been written. We appreciate your patience as our small team of volunteer documentation writers work to update and expand the documentation.

文本变量

KiCad supports text variables, which allow you to substitute the variable name with a defined text string. This substitution happens anywhere the variable name is used inside the variable replacement syntax of `${VARIABLENAME}`.

You can define your own text variables in the [schematic](#) or [board setup](#) dialogs, but there are also a number of built-in system text variables. System text variables may be available in some contexts and not others.

The following variables can be used in schematic text, label names, label fields, hierarchical sheet fields, symbol text, and symbol fields. There are also a number of [variables that can be used in the PCB Editor](#).

Variables used in hierarchical sheet fields refer to the properties of the hierarchical sheet, not the parent, unless otherwise noted. For example, `${#}` returns the subsheet's page number when used in a hierarchical sheet field, but the parent sheet's page number when used in graphic text in the parent sheet.

Variable name	Description
#	Sheet number.
##	Total number of schematic sheets.
SHEETPATH	Sheet path of the current sheet.
FILENAME	Filename of the root schematic sheet , with a file extension.
PROJECTNAME	Project name, without a file extension.
ISSUE_DATE	Contents of drawing sheet's Issue Date field.
CURRENT_DATE	Today's date, in ISO format.
REVISION	Contents of drawing sheet's Revision field.
TITLE	Contents of drawing sheet's Title field.
COMPANY	Contents of drawing sheet's Company field.
COMMENT1 - COMMENT9	Contents of drawing sheet's Comment<n> field.
<variablename>	Contents of project text variable <variablename> .
<fieldname>	<p>Contents of symbol field, hierarchical sheet field, or label field <fieldname>. Fields can only be accessed from within of their parent object, so symbol fields can be accessed from other text or fields within the symbol, and hierarchical sheet fields can be accessed within the sheet or in other sheet fields of the sheet.</p> <p>Both built-in and user-defined fields are available. Built-in fields use all uppercase letters: for example, to access a symbol's value, use <code>#{VALUE}</code>.</p> <p>Built-in symbol fields are DATASHEET , DNP , EXCLUDE_FROM_BOARD , EXCLUDE_FROM_BOM , FOOTPRINT , FOOTPRINT_LIBRARY , FOOTPRINT_NAME , REFERENCE , SYMBOL_DESCRIPTION , SYMBOL_KEYWORDS , SYMBOL_LIBRARY , SYMBOL_NAME , UNIT , VALUE , NET_NAME(<pin_number>) , SHORT_NET_NAME(<pin_number>) , NET_CLASS(<pin_number>) , PIN_NAME(<pin_number>) .</p> <p>Built-in sheet fields are SHEETNAME and SHEETFILE .</p> <p>Built-in label fields are CONNECTION_TYPE , SHORT_NET_NAME , NET_NAME , NET_CLASS , and INTERSHEET_REFS (global labels only).</p>
<refdes>: <fieldname>	<p>Contents of field <fieldname> in symbol <refdes> .</p> <p>Both built-in and user-defined fields are available. Built-in fields use all uppercase letters: for example, to access the value of U1 , use <code>#{U1:VALUE}</code> .</p> <p>Built-in symbol fields are DATASHEET , DNP , EXCLUDE_FROM_BOARD , EXCLUDE_FROM_BOM , FOOTPRINT , FOOTPRINT_LIBRARY , FOOTPRINT_NAME , REFERENCE , SYMBOL_DESCRIPTION , SYMBOL_KEYWORDS , SYMBOL_LIBRARY , SYMBOL_NAME , UNIT , VALUE .</p>

数据基础库

数据基础库是 KiCad 符号库的一种类型，它在外部 SQL 数据库中保存有关零件的数据。数据基础库本身不包含任何符号或封装的定义。相反，它们 **位号** 其他 KiCad 库中的符号和封装。每个数据基础库条目都将一个 KiCad 符号（来自其他库）映射到一组属性（字段），通常还有一个 KiCad 封装（来自封装库）。

使用数据基础库可以从 KiCad 符号和封装中创建完全定义的元件（有时称为 **最小元件**），而不需要在符号库中存储所有元件属性。外部数据库可以链接到第三方工具，用于管理元件数据和生命周期。数据基础库的工作流程通常比标准 KiCad 库的工作流程更复杂，因此这种类型的库通常只用于使管理大型完全定义的零件库更有效的情况下（如在组织或团队设置中）。

KiCad 不提供编辑 SQL 数据库或定义数据库库的 GUI。这取决于用户是否能找到最合适的工作流程和工具链来创建和更新数据库本身。有些用户可能希望通过第三方数据库客户端直接编辑数据库，有些用户可能使用其他第三方软件，如元件生命周期管理 (PLM) 工具来创建和编辑数据。

在数据基础库中，有一个或多个 **表**，通常代表单一类型的元件（如电阻或电容）。每个表可以有一个独立的模式，这意味着不同类型的元件可以有不同的属性，这些属性在 KiCad 中被转化为符号字段。每个表必须有一个唯一的 ID 列，作为从该表放置的符号的标识符。这个唯一的 ID 通常是一个元件号（制造商的元件号，或内部组织的元件号）。每张表还必须有一列包含 KiCad 符号的映射，形式为 `LibraryNickname:SymbolName`。LibraryNickname 必须与 KiCad 库表中的符号库相匹配。表也可以包含一个包含 KiCad 封装的列，形式为 ``LibraryNickname:FootprintName``。如果这一列存在，从表中放置的符号将包括一个封装映射。

表也可以包含任意的额外列，这些列可以选择性地映射到 KiCad 中的符号字段。KiCad 数据基础库配置文件控制这些字段的命名方式，是否使字段可见，以及是否将字段包含在符号选择器中显示的数据。

数据基础库配置文件

要创建一个数据基础库，你必须创建一个配置文件，其中包含 KiCad 连接到你的数据库并从表中获取数据的必要信息。将下面的模板复制到一个新文件中，并以 `kicad_dbl` 为扩展名保存。然后你可以使用配置符号库对话框将此文件添加到你的全局符号库表中。

```

{
  "meta": {
    "version": 0
  },
  "name": "My Database Library",
  "description": "A database of components",
  "source": {
    "type": "odbc",
    "dsn": "",
    "username": "",
    "password": "",
    "timeout_seconds": 2,
    "connection_string": ""
  },
  "libraries": [
    {
      "name": "Resistors",
      "table": "Resistors",
      "key": "Part ID",
      "symbols": "Symbols",
      "footprints": "Footprints",
      "fields": [
        {
          "column": "MPN",
          "name": "MPN",
          "visible_on_add": false,
          "visible_in_chooser": true,
          "show_name": true,
          "inherit_properties": true
        },
        {
          "column": "Value",
          "name": "Value",
          "visible_on_add": true,
          "visible_in_chooser": true,
          "show_name": false
        }
      ],
      "properties": {
        "description": "Description",
        "footprint_filters": "Footprint Filters",
        "keywords": "Keywords",
        "exclude_from_bom": "No BOM",
        "exclude_from_board": "Schematic Only"
      }
    }
  ]
}

```

配置源

KiCad 目前只支持 ODBC 连接到 SQL 数据库。你可以用 DSN 或连接字符串进行连接。如果提供了 DSN 名称，可选的 `username` 和 `password` 字段将被用来连接到 DSN。如果提供了一个连接字符串，`dsn`，`username`，和

password 字段将被忽略。连接字符串将直接传递给 ODBC 驱动程序，所以你可以包括你的 ODBC 驱动程序支持的任何参数。

KiCad 不推荐或认可任何特定的 ODBC 驱动程序或数据库服务器，但经测试可与 Sqlite、MySQL、MariaDB 和 PostgreSQL 兼容。

NOTE

Flatpak 用户。由于 Flatpak 的沙盒功能，连接到本地机器上运行的数据库服务器的唯一方法是通过 TCP/IP。使用默认的 UNIX 域套接字连接，即 MySQL、MariaDB 或 PostgreSQL 是不可能的。首先，确保你的数据库服务器允许 TCP/IP 连接，然后在你的连接字符串中添加所需的 Port 参数。例如，为 MySQL/MariaDB 的默认 TCP 端口添加 Port=3306;，或 Server=localhost;Port=5432;，强制 PostgreSQL 使用 TCP 连接到本地服务器。

配置库

每个数据库基础库可以包含映射到单个数据库表的 "子库"。配置文件中的 libraries 条目包含一个对象列表，每个对象定义一个库。每个库必须存在以下设置：

name: 将显示在 KiCad 用户界面中的表的名称。

table: 数据库中的表的名称。

key: 包含唯一键的列名，将用于识别表中的部分。

symbols: 包含 KiCad 符号位号的列名。

footprints: 包含 KiCad 封装位号的列名。

fields: 一个字段定义的列表。在这里定义的每个字段都会在符号被放置在原理图上时被添加到符号中。如果源符号中已经定义了一个名称相同的字段，那么数据库表中的值将覆盖源符号中定义的任何值。每个字段定义都可以包含：

column: 应该被映射到一个字段的数据库表列的名称。

name: 要从数据库中填充的 KiCad 字段的名称。

visible_on_add: 如果是 true，当符号被添加时，这个字段将在原理图中可见。如果没有指定这个设置，它将默认为 false。

visible_in_chooser: 如果 true，这个字段将在符号选择器中显示为一个列。如果没有指定这个设置，它将默认为 false。

show_name: 如果是 true，该字段的名称将在原理图中显示其数值之外。如果没有指定这个设置，将默认为 false。

inherit_properties: 如果 true，并且源符号中已经存在一个给定的 name 字段，那么只有字段内容将从数据库中更新，其他属性（visible_on_add, show_name 等）将保持源符号中的设置。如果给定的字段名在源符号中不存在，这个设置将被忽略。如果没有指定这个设置，它将默认为 false。

properties: 符号属性与数据库列的映射。所有的属性都是可选的；任何没有在数据库库配置中指定的属性都将从源符号设置的值中继承下来。支持以下属性：

description: 符号的描述属性。

footprint_filters: 保留给未来的扩展。

`keywords`: 符号的关键词属性。

`exclude_from_bom`: 符号的 "不包括在 BOM 中" 设置。这里命名的列必须是数字类型, 并将被视为布尔值 (0 代表假, 1 代表真)。

`exclude_from_board`: 符号的 "排除在 PCB 之外" 设置。此处命名的列必须是数字类型, 并将被当作布尔值 (0 代表假, 1 代表真)。

数据库列可以被映射到自定义 (用户定义的) 字段, 或某些内置的 KiCad 字段, 包括 `Value` 和 `Datasheet`。

使用数据基础库

创建配置文件并将其添加到符号库表中后, 您可以使用符号选择器从数据库表中放置元件。从数据基础库中放置的元件可以使用 "从库中更新符号" 功能进行更新, 该功能将更新数据库中被改变的任何字段, 以及更新源库中被改变的基本符号。

请注意, 数据库表所引用的任何源库也必须存在于符号库表中, 这样数据基础库才能发挥作用。如果你想把一个库只作为数据基础库的符号源, 你可以通过清除 "管理符号库" 对话框中的 "可见" 复选框将其从符号选择器中隐藏起来。

自定义网表和 BOM 格式

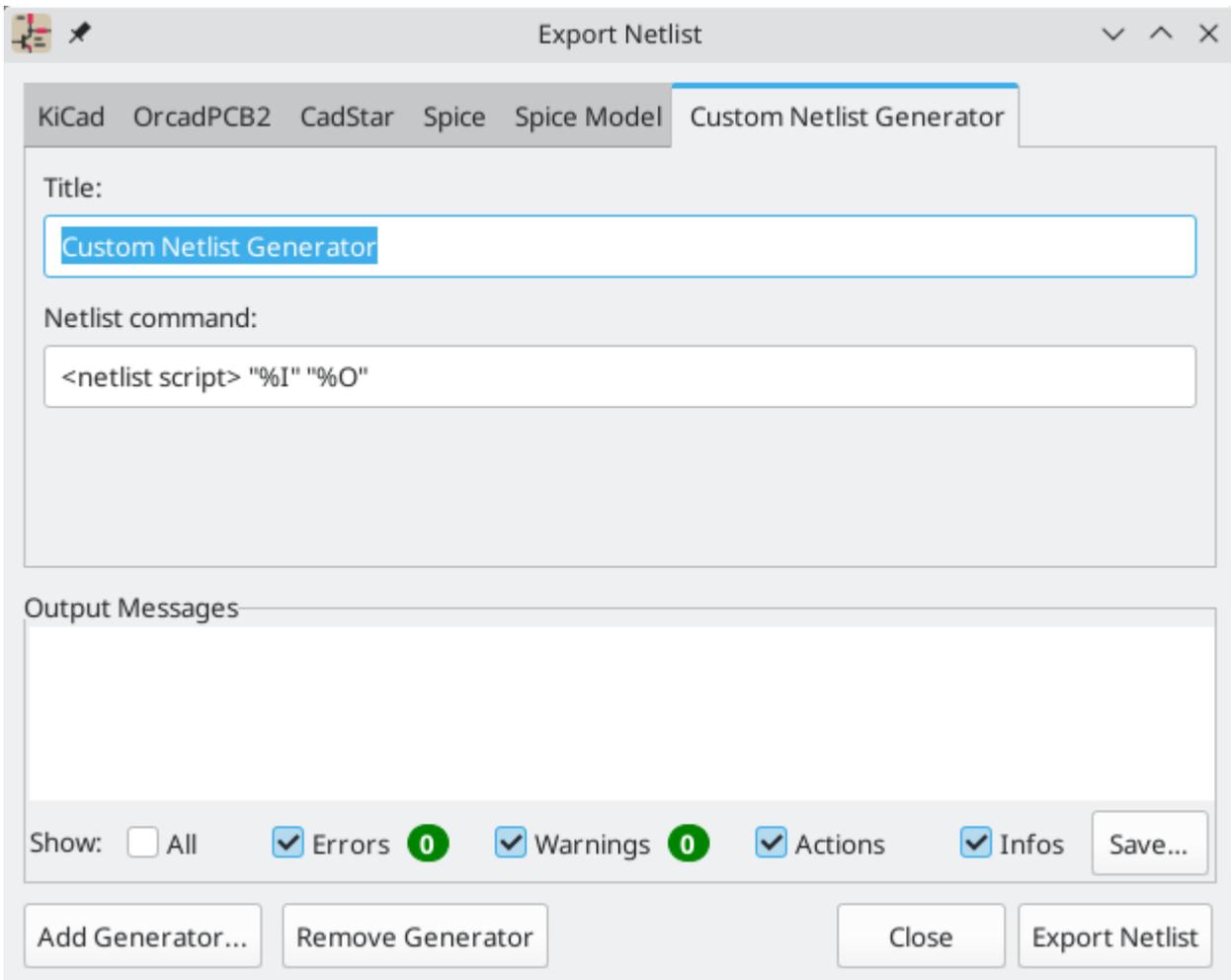
KiCad 可以导出各种格式的网表和 BOM, 如果需要, 用户可以定义新的格式。

导出网表的过程在《网表导出, 网表导出部分》中描述。BOM 导出在《BOM 导出, BOM 导出部分》中描述。

下文描述了如何为一个新的导出格式创建一个导出器。

添加新的网表生成器

新的网表生成器可以通过点击 **添加生成器...** 按钮添加到 **导出网表** 对话框中。



新的生成器需要一个名称和一个命令。名称显示在标签中，只要点击 **导出网表** 按钮，就会运行该命令。

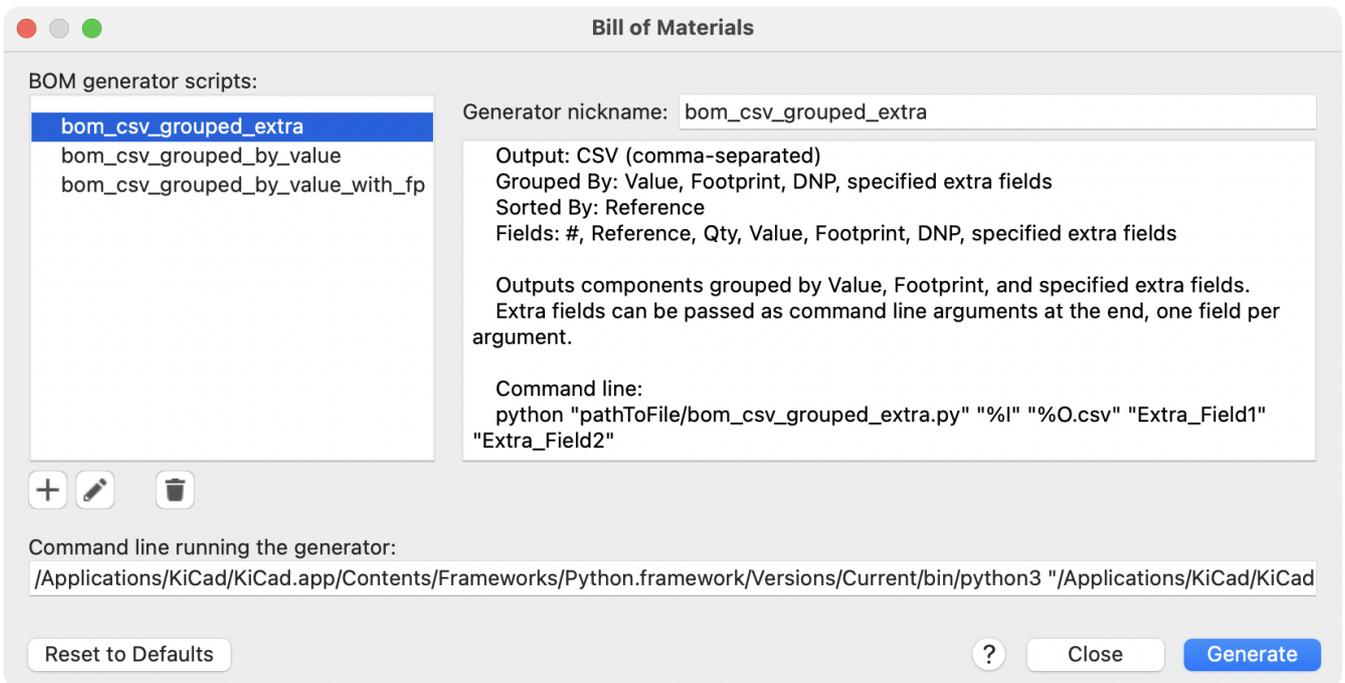
当网表生成时，KiCad 会创建一个中间的 XML 文件，其中包含原理图中所有的网表信息。然后运行生成器命令，以将中间网表转换为所需的网表格式。

网表命令必须正确设置，以便网表生成器脚本将中间网表文件作为输入并输出所需的网表文件。确切的网表命令将取决于所使用的生成器脚本。下面描述了《生成器命令行格式，命令格式》。

Python 和 XSLT 是创建自定义网表生成器的常用工具。

添加新的 BOM 生成器

KiCad 还使用中间网表文件，通过《BOM 导出，生成 BOM 工具》生成 BOM。



通过点击 **+** 按钮，可以在 BOM 生成器脚本列表中添加其他脚本。点击 **🗑️** 按钮，可以删除脚本。**📝** 按钮在文本编辑器中打开所选的脚本。

用 Python 和 XSLT 编写的生成器脚本可以包含一个头注释，描述生成器的功能和用法。这个头注释会作为每个生成器的描述显示在 BOM 对话框中。头部注释必须包含字符串 `@package`。该字符串之后直到注释结束的所有内容都被用作生成器的描述。

当添加新的生成器脚本时，KiCad 会自动填充命令行字段，但根据生成器脚本的情况，可能需要手工调整命令行。KiCad 试图从生成器脚本标题中的示例命令行自动确定输出文件扩展名。

生成器命令行格式

网表或 BOM 导出器的命令行定义了 KiCad 将运行以生成所选输出文件的命令。

对于使用 `xsltproc` 的网表导出器，例如：

```
xsltproc -o %O.net /usr/share/kicad/plugins/netlist_form_pads-pcb.asc.xsl %I
```

对于使用 Python 的 BOM 导出器，一个例子是：

```
/usr/bin/python3 /usr/share/kicad/plugins/bom_csv_grouped_by_value.py "%I" "%O.csv"
```

NOTE | 建议在命令行中用引号 (") 将参数括起来，以防它们包含空格或其他特殊字符。

某些字符序列（如 `%I`` 和 `%O`）在命令行中具有特殊含义，因为 KiCad 在执行命令之前将它们替换为文件名或路径。

参数	替换为...	描述
%I	<工程路径>/<工程名称>.xml	中间网表文件的绝对路径和文件名，它是 BOM 或网表生成器插件的输入。BOM 或网表生成器插件的输入。
%O	<工程路径>/<工程名称>	输出 BOM 或网表文件的绝对路径和文件名（没有文件扩展名）。适当的文件扩展名可能需要在 %O 序列后指定。
%B	<工程名称>	输出 BOM 或网表文件的基本文件名（没有路径或文件扩展名）。适当的文件扩展名可能需要在 `%B` 序列后指定。
%P	<工程路径>	工程目录的绝对路径，没有尾部斜线

中间网表文件格式

当输出 BOM 文件和网表时，KiCad 会创建一个中间网表文件，然后运行一个单独的工具，将中间网表后期处理成所需的网表或 BOM 格式。

中间网表使用 XML 语法。它包含有关设计的大量数据。根据输出（BOM 或网表），完整中间网表文件的不同子集将包含在最终输出文件中。

中间网表文件的结构详述《中间网表结构，如下》。

由于从中间网表文件到输出网表或 BOM 的转换是文本到文本的转换，因此可以使用 Python、XSLT 或任何其他能够将 XML 作为输入的工具编写后处理过滤器。

NOTE

不建议新的网表或 BOM 导出器使用 XSLT;应该改用 Python 或其他工具。从 KiCad 7 开始，xsltproc 不再随 KiCad 一起安装，尽管它可以单独安装。不过，下面列出了使用 XSLT 的网表导出器的几个示例。

中间网表结构

此示例提供了网表文件格式的概念。

```

<?xml version="1.0" encoding="utf-8"?>
<export version="D">
  <design>
    <source>F:\kicad_aux\netlist_test\netlist_test.sch</source>
    <date>29/08/2010 21:07:51</date>
    <tool>eeschema (2010-08-28 BZR 2458)-unstable</tool>
  </design>
  <components>
    <comp ref="P1">
      <value>CONN_4</value>
      <libsource lib="conn" part="CONN_4"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamps>4C6E2141</tstamps>
    </comp>
    <comp ref="U2">
      <value>74LS74</value>
      <libsource lib="74xx" part="74LS74"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamps>4C6E20BA</tstamps>
    </comp>
    <comp ref="U1">
      <value>74LS04</value>
      <libsource lib="74xx" part="74LS04"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamps>4C6E20A6</tstamps>
    </comp>
    <comp ref="C1">
      <value>CP</value>
      <libsource lib="device" part="CP"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamps>4C6E2094</tstamps>
    <comp ref="R1">
      <value>R</value>
      <libsource lib="device" part="R"/>
      <sheetpath names="/" tstamps="/"/>
      <tstamps>4C6E208A</tstamps>
    </comp>
  </components>
  <libparts/>
  <libraries/>
  <nets>
    <net code="1" name="GND">
      <node ref="U1" pin="7"/>
      <node ref="C1" pin="2"/>
      <node ref="U2" pin="7"/>
      <node ref="P1" pin="4"/>
    </net>
    <net code="2" name="VCC">
      <node ref="R1" pin="1"/>
      <node ref="U1" pin="14"/>
      <node ref="U2" pin="4"/>
      <node ref="U2" pin="1"/>
      <node ref="U2" pin="14"/>
      <node ref="P1" pin="1"/>
    </net>
    <net code="3" name="">
      <node ref="U2" pin="6"/>
    </net>
    <net code="4" name="">
      <node ref="U1" pin="2"/>
    </net>
  </nets>

```

一般网表文件结构

中间网表占五个部分。

- “标题”部分。
- “元件”部分。
- “库元件”部分。
- “库”部分。
- “网络”部分。

文件内容具有分隔符 `<export>`

```
<export version="D">
...
</export>
```

“标题”部分

标题具有分隔符 `<design>`

```
<design>
<source>F:\kicad_aux\netlist_test\netlist_test.sch</source>
<date>21/08/2010 08:12:08</date>
<tool>eeschema (2010-08-09 BZR 2439)-unstable</tool>
</design>
```

此部分可视为评论部分。

“元件”部分

元件部分具有分隔符 `<components>`

```
<components>
<comp ref="P1">
<value>CONN_4</value>
<libsource lib="conn" part="CONN_4"/>
<sheetpath names="/" tstamps="/">
<tstamps>4C6E2141</tstamps>
</comp>
</components>
```

本节包含原理图中的元件列表。每个元件都是这样描述的：

```

<comp ref="P1">
<value>CONN_4</value>
<libsource lib="conn" part="CONN_4"/>
<sheetpath names="/" tstamps="/">
<tstamps>4C6E2141</tstamps>
</comp>

```

元素名称	元素描述
libsource	找到该元件的库的名称。
part	该库中的组件名称。
sheetpath	层次结构中的工作表的路径：在整个原理图层次结构中识别工作表。
tstamps	该元件的时间戳。

关于元件的时间戳的注意事项

为了识别网表中的元件以及电路板上的元件，时间戳参考被用来作为每个元件的唯一参考。然而，KiCad 提供了一种辅助的方式来识别一个元件，即电路板上的对应封装。这允许在原理图工程中重新标注元件，并且不会失去元件和其封装之间的联系。

时间戳是原理图工程中每个元件或工作表的唯一标识符。但是，在复杂的层次结构中，同一个工作表会被多次使用，因此此工作表包含具有相同时间戳的元件。

在一个复杂的层次结构中，一个给定的图纸有一个唯一的标识符：它的图纸路径。一个给定的元件（在一个复杂的层次结构内）有一个唯一的标识符：图纸路径和它的时间戳。

“库部件”部分

这个 libparts 部分有分隔符 <libparts> 该部分的内容在原理图库中定义。

```

<libparts>
<libpart lib="device" part="CP">
  <description>Condensateur polarise</description>
  <footprints>
    <fp>CP*</fp>
    <fp>SM*</fp>
  </footprints>
  <fields>
    <field name="Reference">C</field>
    <field name="Valeur">CP</field>
  </fields>
  <pins>
    <pin num="1" name="1" type="passive"/>
    <pin num="2" name="2" type="passive"/>
  </pins>
</libpart>
</libparts>

```

元素名称	元素描述
<footprints>	符号的引脚过滤器。每个引脚过滤器都在一个单独的 <fp> 标签中。
<fields>	符号的字段。每个字段的名称和值都在一个单独的 <field name="字段名称">...</field> 标签中给出
<pins>	符号的引脚。每个引脚都在一个单独的 <pin num="引脚编号" type="pintype"/> 标签中给出。可能的引脚类型描述如下。

可能的电气引脚类型有：

引脚类型	描述
Input	通用输入引脚
Output	通用输出引脚
Bidirectional	输入或输出（双向）
Tri-state	总线输入/输出（三态）
Passive	无源元件的通用端头
Unspecified	未知的电气类型
Power input	元件电源输入引脚
Power output	电源输出引脚作为稳压器输出
Open collector	模拟比较器中常见的开路集电极输出
Open emitter	有时出现在逻辑器件中的开放发射器输出
Not connected	必须在原理图中保持未连接状态

“库”部分

库部分的分隔符为 <libraries>。该部分包含工程中使用的原理图库的列表。

```
<libraries>
  <library logical="device">
    <uri>F:\kicad\share\library\device.lib</uri>
  </library>
  <library logical="conn">
    <uri>F:\kicad\share\library\conn.lib</uri>
  </library>
</libraries>
```

“网络”部分

网络部分的分隔符为 <nets>。这一部分通过列出所有的网络和连接到每个网络的引脚来描述原理图的连接性。

```

<nets>
  <net code="1" name="GND">
    <node ref="U1" pin="7"/>
    <node ref="C1" pin="2"/>
    <node ref="U2" pin="7"/>
    <node ref="P1" pin="4"/>
  </net>
  <net code="2" name="VCC">
    <node ref="R1" pin="1"/>
    <node ref="U1" pin="14"/>
    <node ref="U2" pin="4"/>
    <node ref="U2" pin="1"/>
    <node ref="U2" pin="14"/>
    <node ref="P1" pin="1"/>
  </net>
</nets>

```

一个可能的网包含以下内容。

```

<net code="1" name="GND">
  <node ref="U1" pin="7"/>
  <node ref="C1" pin="2"/>
  <node ref="U2" pin="7"/>
  <node ref="P1" pin="4"/>
</net>

```

元素名称	元素描述
net code	该网络的内部标识符
name	网络的名称
node	符号（由 ref 标识）的引脚（由 pin 标识），该引脚与网络连接。

网表导出器实例

下面包括一些使用 XSLT 的网表导出器的例子。

XSLT 本身是一种 XML 语言，非常适用于 XML 转换。<http://xmlsoft.org/XSLT/xsltproc.html>[xsltproc 程序] 可以用来读取中间的 XML 网表输入文件，应用样式表来转换输入，并将结果保存在输出文件中。使用 xsltproc 需要一个使用 XSLT 惯例的样式表文件。整个转换过程由 KiCad 处理，在它被配置为以特定方式运行 xsltproc 之后。

描述 XSL 变换（XSLT）的文件可在此获得：<http://www.w3.org/TR/xslt>

NOTE

当编写一个新的网表导出器时，考虑使用 Python 或其他工具而不是 XSLT。

使用 XSLT 的 PADS 网表实例

下面的例子显示了如何使用 xlstproc 为 PADS 网表格式创建一个输出器。

PADS 网表格式由两部分组成：

- 封装的清单
- 一个网络的列表，以及与每个网络相连的焊盘。

下面是一个 XSL 样式表，将中间网表文件转换为 PADS 网表格式。

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to Eeschema Generic Netlist Format to PADS netlist format
Copyright (C) 2010, SoftPLC Corporation.
GPL v2.

How to use:
https://lists.launchpad.net/kicad-developers/msg05157.html
-->

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl "&#xd;&#xa;"> <!--new line CR, LF -->
]>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<xsl:template match="/export">
  <xsl:text>*PADS-PCB*&nl;*PART*&nl;</xsl:text>
  <xsl:apply-templates select="components/comp"/>
  <xsl:text>&nl;*NET*&nl;</xsl:text>
  <xsl:apply-templates select="nets/net"/>
  <xsl:text>*END*&nl;</xsl:text>
</xsl:template>

<!-- for each component -->
<xsl:template match="comp">
  <xsl:text> </xsl:text>
  <xsl:value-of select="@ref"/>
  <xsl:text> </xsl:text>
  <xsl:choose>
    <xsl:when test = "footprint != ' ' ">
      <xsl:apply-templates select="footprint"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:text>unknown</xsl:text>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>&nl;</xsl:text>
</xsl:template>

<!-- for each net -->
<xsl:template match="net">
  <!-- nets are output only if there is more than one pin in net -->
  <xsl:if test="count(node)>1">
    <xsl:text>*SIGNAL* </xsl:text>
    <xsl:choose>
      <xsl:when test = "@name != ' ' ">
        <xsl:value-of select="@name"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text>N-</xsl:text>
        <xsl:value-of select="@code"/>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:text>&nl;</xsl:text>
    <xsl:apply-templates select="node"/>
  </xsl:if>
</xsl:template>

<!-- for each node -->

```

这是运行 `xsltproc` 后的 PADS 网表输出文件：

```
*PADS-PCB*
*PART*
P1 unknown
U2 unknown
U1 unknown
C1 unknown
R1 unknown
*NET*
*SIGNAL* GND
U1.7
C1.2
U2.7
P1.4
*SIGNAL* VCC
R1.1
U1.14
U2.4
U2.1
U2.14
P1.1
*SIGNAL* N-4
U1.2
U2.3
*SIGNAL* /SIG_OUT
P1.2
U2.5
U2.2
*SIGNAL* /CLOCK_IN
R1.2
C1.1
U1.1
P1.3

*END*
```

进行这种转换的命令是：

```
kicad\bin\xsltproc.exe -o test.net kicad\bin\plugins\netlist_form_pads-pcb.xsl test.tmp
```

使用 XSLT 的 Cadstar 网表例子

下面的例子显示了如何使用 `xlstproc` 为 Cadstar 网表格式创建一个输出器。

Cadstar 的格式由两部分组成：

- 封装列表
- 网络列表：按网络对焊盘参考资料进行分组

下面是一个 XSL 样式表，将中间网表文件转换为 Cadstar 网表格式。

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to Eeschema Generic Netlist Format to CADSTAR netlist format
  Copyright (C) 2010, Jean-Pierre Charras.
  Copyright (C) 2010, SoftPLC Corporation.
  GPL v2. -->

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl "&#xd;&#xa;"> <!--new line CR, LF -->
]>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<!-- Netlist header -->
<xsl:template match="/export">
  <xsl:text>.HEA&nl;</xsl:text>
  <xsl:apply-templates select="design/date"/> <!-- Generate line .TIM <time> -->
  <xsl:apply-templates select="design/tool"/> <!-- Generate line .APP <eeschema version>
-->
  <xsl:apply-templates select="components/comp"/> <!-- Generate list of components -->
  <xsl:text>&nl;&nl;</xsl:text>
  <xsl:apply-templates select="nets/net"/> <!-- Generate list of nets and
connections -->
  <xsl:text>&nl;.END&nl;</xsl:text>
</xsl:template>

  <!-- Generate line .TIM 20/08/2010 10:45:33 -->
<xsl:template match="tool">
  <xsl:text>.APP "</xsl:text>
  <xsl:apply-templates/>
  <xsl:text>"&nl;</xsl:text>
</xsl:template>

  <!-- Generate line .APP "eeschema (2010-08-17 BZR 2450)-unstable" -->
<xsl:template match="date">
  <xsl:text>.TIM </xsl:text>
  <xsl:apply-templates/>
  <xsl:text>&nl;</xsl:text>
</xsl:template>

<!-- for each component -->
<xsl:template match="comp">
  <xsl:text>.ADD_COM </xsl:text>
  <xsl:value-of select="@ref"/>
  <xsl:text> </xsl:text>
  <xsl:choose>
    <xsl:when test = "value != '' ">
      <xsl:text>"</xsl:text> <xsl:apply-templates select="value"/> <xsl:text>"
</xsl:text>
    </xsl:when>
    <xsl:otherwise>
      <xsl:text>""</xsl:text>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>&nl;</xsl:text>
</xsl:template>

<!-- for each net -->
<xsl:template match="net">
  <!-- nets are output only if there is more than one pin in net -->

```

这是 Cadstar 输出文件。

```
.HEA
.TIM 21/08/2010 08:12:08
.APP "eeschema (2010-08-09 BZR 2439)-unstable"
.ADD_COM P1 "CONN_4"
.ADD_COM U2 "74LS74"
.ADD_COM U1 "74LS04"
.ADD_COM C1 "CP"
.ADD_COM R1 "R"

.ADD_TER U1.7 "GND"
.TER      C1.2
          U2.7
          P1.4
.ADD_TER R1.1 "VCC"
.TER      U1.14
          U2.4
          U2.1
          U2.14
          P1.1
.ADD_TER U1.2 "N-4"
.TER      U2.3
.ADD_TER P1.2 "/SIG_OUT"
.TER      U2.5
          U2.2
.ADD_TER R1.2 "/CLOCK_IN"
.TER      C1.1
          U1.1
          P1.3

.END
```

使用 XSLT 的 OrcadPCB2 网表示例

这种格式只有一个部分，就是封装列表。每个封装都包括其焊盘的列表，并参考一个网络。

下面是一个 XSL 样式表，将中间网表文件转换为 Orcad 网表格式。

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!--XSL style sheet to Eeschema Generic Netlist Format to CADSTAR netlist format
Copyright (C) 2010, SoftPLC Corporation.
GPL v2.

How to use:
https://lists.launchpad.net/kicad-developers/msg05157.html
-->

<!DOCTYPE xsl:stylesheet [
  <!ENTITY nl  "&#xd;&#xa;"> <!--new line CR, LF -->
]>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" omit-xml-declaration="yes" indent="no"/>

<!--
Netlist header
Creates the entire netlist
(can be seen as equivalent to main function in C
-->
<xsl:template match="/export">
  <xsl:text>( { Eeschema Netlist Version 1.1  </xsl:text>
  <!-- Generate line .TIM <time> -->
<xsl:apply-templates select="design/date"/>
<!-- Generate line eeschema version ... -->
<xsl:apply-templates select="design/tool"/>
<xsl:text>}&nl;</xsl:text>

<!-- Generate the list of components -->
<xsl:apply-templates select="components/comp"/> <!-- Generate list of components -->

<!-- end of file -->
<xsl:text>)&nl;*&nl;</xsl:text>
</xsl:template>

<!--
Generate id in header like "eeschema (2010-08-17 BZR 2450)-unstable"
-->
<xsl:template match="tool">
  <xsl:apply-templates/>
</xsl:template>

<!--
Generate date in header like "20/08/2010 10:45:33"
-->
<xsl:template match="date">
  <xsl:apply-templates/>
  <xsl:text>&nl;</xsl:text>
</xsl:template>

<!--
This template read each component
(path = /export/components/comp)
creates lines:
( 3EBF7DBD $oname U1 74LS125
... pin list ...
)
and calls "create_pin_list" template to build the pin list
-->

```

这是 OrcadPCB2 输出文件。

```
( { Eeschema Netlist Version 1.1 29/08/2010 21:07:51
eeschema (2010-08-28 BZR 2458)-unstable}
( 4C6E2141 $noname P1 CONN_4
( 1 VCC )
( 2 /SIG_OUT )
( 3 /CLOCK_IN )
( 4 GND )
)
( 4C6E20BA $noname U2 74LS74
( 1 VCC )
( 2 /SIG_OUT )
( 3 N-04 )
( 4 VCC )
( 5 /SIG_OUT )
( 6 ? )
( 7 GND )
( 14 VCC )
)
( 4C6E20A6 $noname U1 74LS04
( 1 /CLOCK_IN )
( 2 N-04 )
( 7 GND )
( 14 VCC )
)
( 4C6E2094 $noname C1 CP
( 1 /CLOCK_IN )
( 2 GND )
)
( 4C6E208A $noname R1 R
( 1 VCC )
( 2 /CLOCK_IN )
)
)
*
```

操作参考

以下是 KiCad 原理图编辑器中每个可用 **操作** 的列表：可以分配给热键的命令。

原理图编辑器

以下操作在原理图编辑器中可用。可以在偏好设置的 **快捷键** 部分中将快捷键分配给这些操作中的任何一个。

Action	Default Hotkey	Description
Align Elements to Grid		
Annotate Schematic...		Fill in schematic symbol reference designators
Annotate Automatically		Toggle automatic annotation of new parts symbols
Assign Footprints...		Run footprint assignment tool
Clear Net Highlighting		Clear any existing net highlighting
Export Drawing to Clipboard		Export drawing of current sheet to clipboard
Edit Library Symbol...	 +  + 	Open the library symbol in the Symbol Editor
Edit Sheet Page Number...		Edit the page number of the current or selected sheet
Edit Symbol Fields...		Bulk-edit fields of all symbols in schematic
Edit Symbol Library Links...		Edit links between schematic and library symbols
Edit with Symbol Editor	 + 	Open the selected symbol in the Symbol Editor
Export Netlist...		Export file containing netlist in one of several formats
Export Symbols to Library...		Add symbols used in schematic to an existing symbol library (does not remove other symbols from this library)
Export Symbols to New Library...		Create a new symbol library using the symbols used in the schematic (if the library already exists it will be replaced)
Generate BOM...		Generate a bill of materials for the current schematic

Action	Default Hotkey	Description
Highlight Net		Highlight net under cursor
Highlight Nets		Highlight wires and pins of a net
Import Footprint Assignments...		Import symbol footprint assignments from .cmp file created by board editor
Line Mode for Wires and Buses		Constrain drawing and dragging to horizontal, vertical, or 45-degree angle motions
Line Mode for Wires and Buses		Draw and drag at any angle
Line Mode for Wires and Buses	 + 	Switch to next line mode
Line Mode for Wires and Buses		Constrain drawing and dragging to horizontal or vertical motions
Remap Legacy Library Symbols...		Remap library symbol references in legacy schematics to the symbol library table
Repair Schematic		Run various diagnostics and attempt to repair schematic
Rescue Symbols...		Find old symbols in project and rename/rescue them
Simulator...		Simulate circuit in SPICE
Save Current Sheet Copy As...		Save a copy of the current sheet to another location or name
Schematic Setup...		Edit schematic setup including annotation styles and electrical rules
Select on PCB		Select corresponding items in PCB editor
Show ERC Errors		Show markers for electrical rules checker errors
Show ERC Exclusions		Show markers for excluded electrical rules checker violations
Show ERC Warnings		Show markers for electrical rules checker warnings
Show Hidden Fields		Toggle display of hidden text fields
Show Hidden Pins		Toggle display of hidden pins
Switch to PCB Editor		Open PCB in board editor
Scripting Console		Show the Python scripting console

Action	Default Hotkey	Description
Show Datasheet	D	Opens the datasheet in a browser
Add Arc		Add an arc
Add Circle		Add a circle
Add Rectangle		Add a rectangle
Add Sheet	S	Add a hierarchical sheet
Add Text Box		Add a text box
Finish Sheet		Finish drawing sheet
Import Sheet Pin		Import a hierarchical sheet pin
Add Wire to Bus Entry	Z	Add a wire entry to a bus
Add Net Class Directive		Add a net class directive label
Add Global Label	Ctrl + L	Add a global label
Add Hierarchical Label	H	Add a hierarchical label
Add Image		Add bitmap image
Add Junction	J	Add a junction
Add Label	L	Add a net label
Add No Connect Flag	Q	Add a no-connection flag
Add Power	P	Add a power symbol
Add Text	T	Add text
Add Symbol	A	Add a symbol
Add Bus	B	Add a bus
Add Lines	I	Add connected graphic lines
Add Wire	W	Add a wire
Finish Wire or Bus	K	Complete drawing at current segment
Finish Bus		Complete bus with current segment
Finish Lines		Complete connected lines with current segment

Action	Default Hotkey	Description
Undo Last Segment		Walks the current line back one segment.
Unfold from Bus		Break a wire out of a bus
Assign Netclass...		Assign a netclass to nets matching a pattern
Autoplace Fields		Runs the automatic placement algorithm on the symbol or sheet's fields
Break		Divide into connected segments
Change Symbol...		Assign a different symbol from the library
Change Symbols...		Assign different symbols from the library
Cleanup Sheet Pins		Delete unreferenced sheet pins
Edit Footprint...		Displays footprint field dialog
Edit Reference Designator...		Displays reference designator dialog
Edit Text & Graphics Properties...		Edit text and graphics properties globally across schematic
Edit Value...		Displays value field dialog
Mirror Horizontally		Flips selected item(s) from left to right
Mirror Vertically		Flips selected item(s) from top to bottom
Pin Table...		Displays pin table for bulk editing of pins
Properties...		Displays item properties dialog
Repeat Last Item		Duplicates the last drawn item
Rotate Counterclockwise		Rotates selected item(s) counter-clockwise
Rotate Clockwise		Rotates selected item(s) clockwise
De Morgan Alternate		Switch to alternate De Morgan representation
De Morgan Standard		Switch to standard De Morgan representation
Slice		Divide into unconnected segments

Action	Default Hotkey	Description
Change to Label		Change existing item to a label
Change to Text		Change existing item to a text comment
Change to Text Box		Change existing item to a text box
De Morgan Conversion		Switch between De Morgan representations
Update Symbol...		Update symbol to include any changes from the library
Update Symbols from Library...		Update symbols to include any changes from the library
Move Activate		
Drag	G	Drags the selected item(s)
Move	M	Moves the selected item(s)
Select Connection	Alt + 4	Select a complete connection
Select Node	Alt + 3	Select a connection item under the cursor
Navigate Back	Alt + Left	Move backward in sheet navigation history
Change Sheet		Change to provided sheet's contents in the schematic editor
Enter Sheet		Display the selected sheet's contents in the schematic editor
Navigate Forward	Alt + Right	Move forward in sheet navigation history
Leave Sheet	Alt + Back	Display the parent sheet in the schematic editor
Next Sheet	PgDn	Move to next sheet by number
Previous Sheet	PgUp	Move to previous sheet by number
Navigate Up	Alt + Up	Navigate up one sheet in the hierarchy
Push Pin Length		Copy pin length to other pins in symbol
Push Pin Name Size		Copy pin name size to other pins in symbol
Push Pin Number Size		Copy pin number size to other pins in symbol
Create Corner		Create a corner
Remove Corner		Remove corner

Action	Default Hotkey	Description
Add Lines		Add connected graphic lines
Add Text Box		Add a text box item
Finish Drawing		Finish drawing shape
Move Symbol Anchor		Specify a new location for the symbol anchor
Add Pin		Add a pin
Add Text		Add a text item
Add Symbol to Schematic		Add Symbol to Schematic
Copy		
Cut		
Delete Symbol		Remove the selected symbol from its library
Duplicate Symbol		Make a copy of the selected symbol
Edit Symbol		Show selected symbol on editor canvas
Export...		Export a symbol to a new library file
Export Symbol as SVG...		Create SVG file from the current symbol
Export View as PNG...		Create PNG file from the current view
Hide Symbol Tree		
Import Symbol...		Import a symbol to the current library
New Symbol...		Create a new symbol
Paste Symbol		
Rename Symbol...		Rename the selected symbol

Action	Default Hotkey	Description
Save Library As...		Save the current library to a new file.
Save As...		Save the current symbol to a different library.
Set Unit Display Name...		Set the display name for a unit
Show Pin Electrical Types		Annotate pins with their electrical types
Show Pin Numbers		Annotate pins with their numbers
Show Symbol Tree		
Synchronized Pins Mode		Synchronized Pins Mode When enabled propagates all changes (except pin numbers) to other units. Enabled by default for multiunit parts with interchangeable units.
Update Symbol Fields...		Update symbol to match changes made in parent symbol
Symbol Move Activate		

通用

以下操作可在 KiCad 中使用，包括在原理图编辑器中。可以在偏好设置的 **快捷键** 部分中将热键分配给这些操作中的任何一个。

Action	Default Hotkey	Description
Exclude Marker		Mark current violation in Checker window as an exclusion
Next Marker		Go to next marker in Checker window
Previous Marker		Go to previous marker in Checker window
Add Library...		Add an existing library folder
Click		Performs left mouse button click
Double-click		Performs left mouse button double-click
Cursor Down		

Action	Default Hotkey	Description
Cursor Down Fast	Ctrl + Down	
Cursor Left	Left	
Cursor Left Fast	Ctrl + Left	
Cursor Right	Right	
Cursor Right Fast	Ctrl + Right	
Cursor Up	Up	
Cursor Up Fast	Ctrl + Up	
Switch to Fast Grid 1	Alt + 1	
Switch to Fast Grid 2	Alt + 2	
Switch to Next Grid	N	
Switch to Previous Grid	Shift + N	
Grid Properties...		Set grid dimensions
Reset Grid Origin	Z	
Grid Origin	S	Set the grid origin point
Inactive Layer View Mode		Toggle inactive layers between normal and dimmed
Inactive Layer View Mode (3-state)	H	Cycle inactive layers between normal, dimmed, and hidden
Inches		Use inches
Millimeters		Use millimeters
Mils		Use mils
New...	Ctrl + N	Create a new document in the editor
New Library...		Create a new library folder
Open...	Ctrl + O	Open existing document
Page Settings...		Settings for paper size and title block info

Action	Default Hotkey	Description
Plot...		Plot
Print...	Ctrl + P	Print
Quit		Close the current editor
Reset Local Coordinates	Space	
Revert		Throw away changes
Save	Ctrl + S	Save changes
Save All		Save all changes
Save As...	Ctrl + Shift + S	Save current document to another location
Save a Copy...		Save a copy of the current document to another location
Select Columns		
3D Viewer	Alt + 3	Show 3D viewer window
Show Context Menu		Perform the right-mouse-button action
Footprint Library Browser		Browse footprint libraries
Footprint Editor		Create, delete and edit footprints
Symbol Library Browser		Browse symbol libraries
Symbol Editor		Create, delete and edit symbols
Draw Bounding Boxes		Draw Bounding Boxes
Always Show Cursor	Ctrl + Shift + X	Display crosshairs even in selection tool
Full-Window Crosshairs		Switch display of full-window crosshairs
Show Grid		Display background grid in the edit window
Polar Coordinates		Switch between polar and cartesian coordinate systems
Switch units	Ctrl + U	Switch between imperial and metric units

Action	Default Hotkey	Description
Center on Cursor	F4	Center on Cursor
Zoom to Objects	Ctrl + Home	Zoom to Objects
Zoom to Fit	Home	Zoom to Fit
Zoom In at Cursor	F1	Zoom In at Cursor
Zoom In		Zoom In
Zoom Out at Cursor	F2	Zoom Out at Cursor
Zoom Out		Zoom Out
Refresh	F5	Refresh
Zoom to Selection	Ctrl + F5	Zoom to Selection
Cancel		Cancel current tool
Copy	Ctrl + C	Copy selected item(s) to clipboard
Cut	Ctrl + X	Cut selected item(s) to clipboard
Cycle arc editing mode	Ctrl + Space	Switch to a different method of editing arcs
Delete	Del	Deletes selected item(s)
Interactive Delete Tool		Delete clicked items
Duplicate	Ctrl + D	Duplicates the selected item(s)
Find	Ctrl + F	Find text
Find and Replace	Ctrl + Alt + F	Find and replace text
Find Next	F3	Find next match
Find Next Marker	Shift + F3	
Paste	Ctrl + V	Paste item(s) from clipboard
Paste Special...		Paste item(s) from clipboard with annotation options
Redo	Ctrl + Y	Redo last edit
Replace All		Replace all matches

Action	Default Hotkey	Description
Measure Tool	Ctrl + Shift + M	Interactively measure distance between points
Select item(s)		Select item(s)
Configure Paths...		Edit path configuration environment variables
Donate		Open "Donate to KiCad" in a web browser
Get Involved		Open "Contribute to KiCad" in a web browser
Getting Started with KiCad		Open "Getting Started in KiCad" guide for beginners
Help		Open product documentation in a web browser
List Hotkeys...	Ctrl + F1	Displays current hotkeys table and corresponding commands
Preferences...	Ctrl + ,	Show preferences for all open tools
Report Bug		Report a problem with KiCad
Manage Footprint Libraries...		Edit the global and project footprint library lists
Manage Symbol Libraries...		Edit the global and project symbol library lists