KiCad 入门

October 31, 2021

Contents

掌握 KiCad 成功开发复杂电子印刷电路板的必要和简明指南。

Contributors

David Jahshan, Phil Hutchinson, Fabrizio Tappero, Christina Jarron, Melroy van den Berg.

翻译

taotieren <admin@taotieren.com>, 2019

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

反馈

请将任何错误报告、建议或新版本引导到此处:

- 关于 KiCad 文档: https://gitlab.com/kicad/services/kicad-doc/issues
- 关于 KiCad 软件: https://gitlab.com/kicad/code/kicad/issues
- 关于 KiCad 软件国际化 (i18n): https://gitlab.com/kicad/code/kicad-i18n/issues

发布日期

2015 年 5 月 16 日。

Chapter 1

KiCad 简介

KiCad 是一个开源软件工具，用于创建电子原理图和 PCB 图形。在其独特的表面下，KiCad 融合了以下独立软件工具的优雅集合：

程序名称	描述	文件扩展
KiCad	项目管理器	`*.pro`
Eeschema	原理图和元件库编辑器	`*.sch, *.lib, *.net`
Pcbnew	PCB 和封装编辑器	`*.kicad_pcb, *.kicad_mod`
GerbView	光绘文件及钻孔文件查看	`*.g\*, *.drl, etc.`
Bitmap2Component	将位图转换为元件符号或封装	`*.lib, *.kicad_mod, *.kicad_wks`
PCB Calculator	便携式计算器，可以用于方便的计算线宽/电气间隙/色环代码等等	无
Pl Editor	图框编辑器	`*.kicad_wks`

Note

上述文件扩展名列表不完整，仅包含 KiCad 支持的文件的子集。但是它能帮助你了解 KiCad 使用的基本的文件。

有充分的理由认为, KiCad 已足够成熟, 并可以用于开发和维护复杂的电路板。

KiCad 对电路板的大小不做任何限制, 它可以轻松地处理多达 32 个铜层、多达 14 个技术层和多达 4 个辅助层的电路板。KiCad 可以创建制造印刷电路板所需的所有文件、用于照片绘图仪的 Gerber 文件、钻孔文件、元件位置文件等等。

作为开源 (GPL 许可) 软件, KiCad 是面向有意创建开源电子硬件的项目的工程师的理想工具。

在互联网上, KiCad 的主页是:

http://www.kicad.org/

1.1 下载和安装 KiCad

KiCad 运行在 GNU/Linux，Apple macOS 和 Windows 上。您可以在以下位置找到最新的说明和下载链接：

http://www.kicad.org/download/

Important

根据 KiCad 稳定版本发布政策，KiCad 定期发布。新功能不断添加到开发分支中。如果您想利用这些新功能并通过测试帮助，请下载适用于您平台的最新每晚构建包。每夜构建可能会引入诸如文件损坏，生成坏 Gerbers 等错误，但 KiCad 开发团队的目标是在新功能开发期间尽可能保持开发分支的可用性。

1.1.1 在 GNU/Linux 下

KiCad 的稳定版本，包括如 KiCad 和 kicad-doc，可以在大多数发行版的软件包管理器中找到。如果您的发行版没有提供最新的稳定版本，请按照针对不稳定版本发布的说明进行操作，并安装最新的版本。

在 Ubuntu 下，安装不稳定的夜间构建 KiCad 的最简单方法是通过 *PPA* 和 *Aptitude*。在终端中键入以下内容：

```
sudo add-apt-repository ppa:js-reynaud/ppa-kicad
sudo aptitude update && sudo aptitude safe-upgrade
sudo aptitude install kicad kicad-doc-en
```

在 Debian 下，安装 KiCad 的回滚构建的最简单方法是：

```
# b'' 设 b''b'' 置 b'' Debian b'' 回 b''b'' 滚 b''
echo -e "
# b'' 伸 b''b'' 展 b''-b'' 回 b''b'' 滚 b''
deb http://ftp.us.debian.org/debian/ stretch-backports main contrib non-free
deb-src http://ftp.us.debian.org/debian/ stretch-backports main contrib non-free
" | sudo tee -a /etc/apt/sources.list > /dev/null

# b'' 运 b''b'' 行 b''b'' 更 b''b'' 新 b''b'' 并 b''b'' 安 b''b'' 装 b'' KiCad
sudo apt-get update
sudo apt-get install -t stretch-backports kicad
```

在 Fedora 下安装不稳定的夜间构建的最简单方法是通过 *copr*。要通过 copr 安装 KiCad，请将以下内容输入到 copr 中：

```
sudo dnf copr enable @kicad/kicad
sudo dnf install kicad
```

或者，您可以下载并安装 KiCad 的预编译版本，或直接下载源代码，编译和安装 KiCad。

1.1.2 在 Apple macOS 下

Stable builds of KiCad for macOS can be found at: https://downloads.kicad.org/kicad/macos/explore/stable

Unstable nightly development builds can be found at: https://downloads.kicad.org/kicad/macos/explore/nightlies

1.1.3 在 Windows 下

Stable builds of KiCad for Windows can be found at: https://downloads.kicad.org/kicad/windows/explore/stable

For Windows you can find nightly development builds at: https://downloads.kicad.org/kicad/windows/explore/-nightlies

1.2 支持

如果您有想法，评论或问题，或者您只是需要帮助：

- Visit the Forum
- Join users on IRC or Discord
- Watch Tutorials

Chapter 2

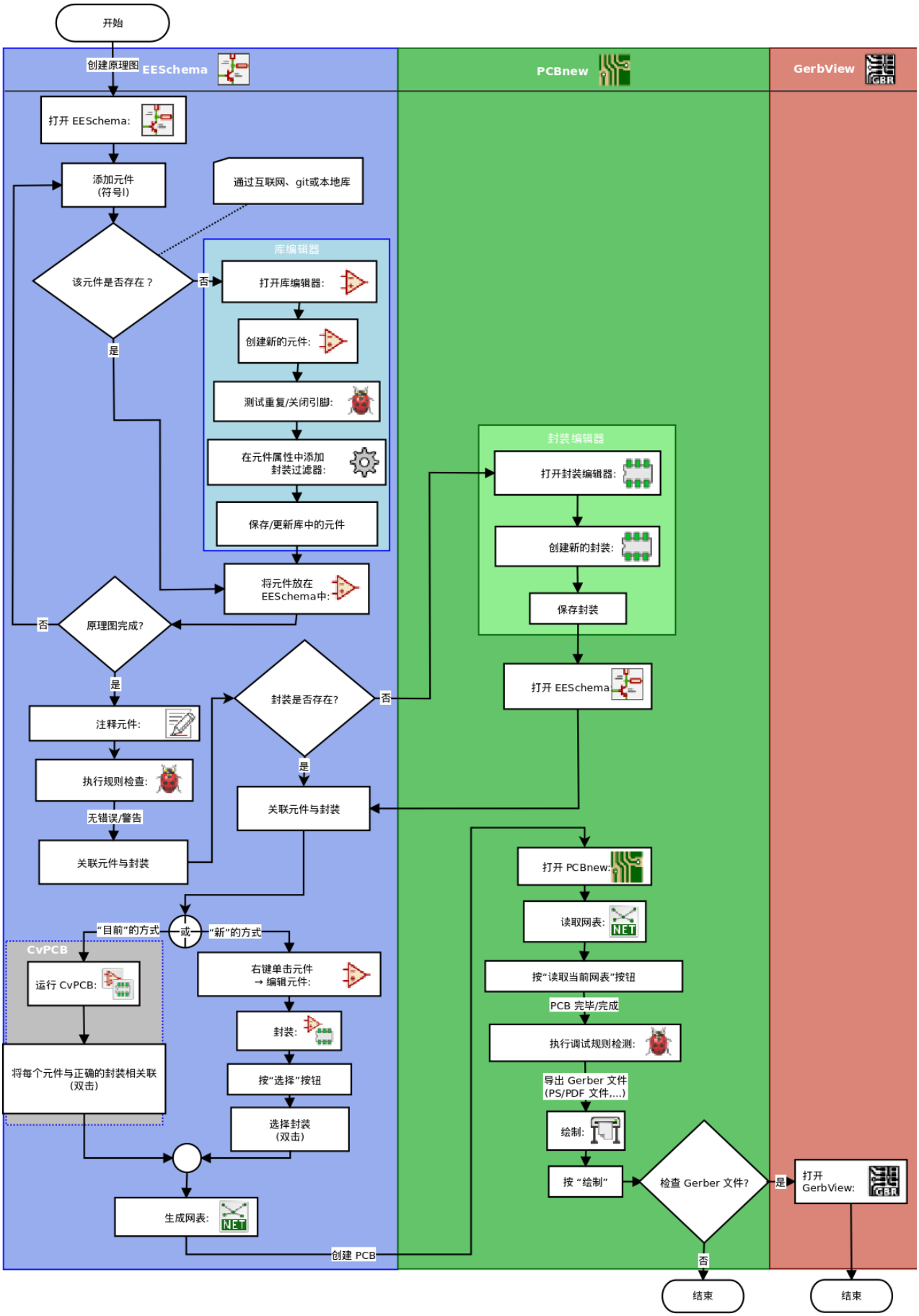
KiCad 工作流程

尽管与其他 PCB 设计软件具有相似性，但 KiCad 的特点是独特的工作流程，其中原理图元件和封装是分开的。仅在创建原理图后才会为元件分配封装。

2.1 概述

KiCad 工作流由两个主要任务组成: 绘制原理图和布置电路板。原理图元件库和 PCB 封装库对于这两个任务都是必需的。KiCad 库中包括许多元件和封装, 并且还具有创建新元件的工具。

在下图中, 您将看到一个表示 KiCad 工作流的流程图。流程图说明了您需要采取哪些步骤, 以及以何种顺序采取这些步骤。在适用的情况下, 为方便起见, 添加了图标。


开始
创建原理图
EESchema
PCBnew
GerbView
打开 EESchema:
添加元件
(符号I)
通过互联网、git或本地库
该元件是否存在？
否
是
库编辑器
打开库编辑器:
创建新的元件:
测试重复/关闭引脚:
在元件属性中添加
封装过滤器:
保存/更新库中的元件
将元件放在
EESchema中:
原理图完成?
否
是
注释元件:
执行规则检查:
无错误/警告
关联元件与封装
封装是否存在?
否
是
关联元件与封装
封装编辑器
打开封装编辑器:
创建新的封装:
保存封装
打开 EESchema
"目前"的方式
或
"新"的方式
CvPCB
运行 CvPCB:
将每个元件与正确的封装相关联
(双击)
右键单击元件
→ 编辑元件:
封装:
按"选择"按钮
选择封装
(双击)
生成网表:
创建 PCB
打开 PCBnew:
读取网表:
按"读取当前网表"按钮
PCB 完毕/完成
执行调试规则检测:
导出 Gerber 文件
(PS/PDF 文件,...)
绘制:
按 "绘制"
检查 Gerber 文件?
是
否
打开
GerbView:
结束
结束

有关创建元件的详细信息, 请阅读《make-schematic-symbols-in-kicad(制作-原理图-符号-在-kicad),Making schematic symbols（制作原理图符号）》。有关如何创建新封装的信息, 请参阅《make-component-footprints(制作-元件-封装),Making component footprints(制作元件封装)》。

Quicklib 是一个工具，允许您使用基于 Web 的界面快速创建 KiCad 库元件。有关 Quicklib 的更多信息，请参阅《make-schematic-components-with-quicklib(制作-原理图-元件-使用-quicklib)，Making Schematic Components With Quicklib(使用 Quicklib 制作原理图元件)》。

2.2 前向和后向注释

完成电子原理图绘制后，下一步就是将其传输到 PCB。通常，可能需要添加其他元件，将部件更改为不同的大小，进行网络重命名等。这可以通过两种方式完成：前向注释或后向注释。

正向注释是将原理图信息发送到相应 PCB 布局的过程。这是一个基本功能，因为您必须至少执行一次才能将原理图初始导入 PCB。之后，正向注释允许向 PCB 发送增量原理图更改。有关前向注释的详细信息，请参阅《forward-annotation-in-kicad（向前-注释-在-kicad），Forward Annotation（向前注释）》部分。

向后注释是将 PCB 布局更改发送回其相应原理图的过程。向后注释的两个常见原因是门交换和引脚交换。在这些情况下，存在功能相同的门或引脚，但可能仅在布局期间存在选择精确栅极或引脚的强有力的情况。一旦在 PCB 中做出选择，则该更改将被推回原理图。

Chapter 3

使用 KiCad

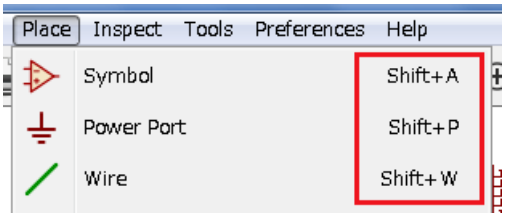
3.1 快捷键

KiCad 有两种相关但不同的快捷键: 快捷键 (accelerator keys) 和热键 (hotkeys)。两者都用于通过使用键盘而不是鼠标来执行命令以提高使用 KiCad 的生产效率。

3.1.1 快捷键

快捷键的效果与单击菜单或工具栏图标的效果相同: 将输入该命令, 但在单击鼠标左键之前不会发生任何情况。如果要进入命令模式, 但不希望立即执行任何操作, 请使用快捷键。

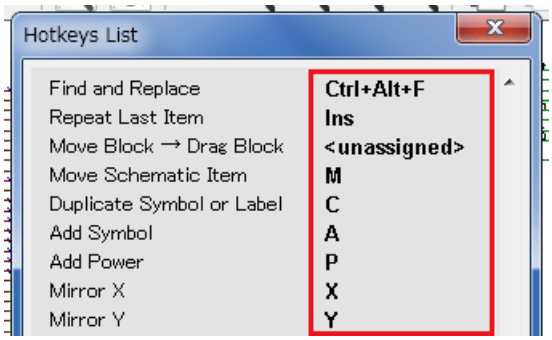
快捷键显示在所有菜单窗格的右侧:



3.1.2 热键

热键等于加速器键加上鼠标左键单击。使用热键会立即在当前光标位置启动该命令。使用热键可以在不中断工作流的情况下快速更改命令。

要查看任何 KiCad 工具中的热键，请转到 **帮助** → **列出热键**或按 *Ctrl+F1* :



您可以从 **首选项 → 热键选项**菜单编辑热键的分配，并导入或导出它们。

Note

在本文档中，热键用括号表示，如下所示：[a]。**如果看到** [a]，**只需在键盘上键入** *a* **键即可**。

3.1.3 例子

考虑在原理图中添加走线的简单示例。

要使用快捷键, 请按 *Shift + W* 调用 *添加*导线命令 (请注意光标将会更改)。接下来, 左键单击所需的导线开始位置, 开始绘制导线。

使用热键，只需按 [w]，线将立即从当前光标位置开始。

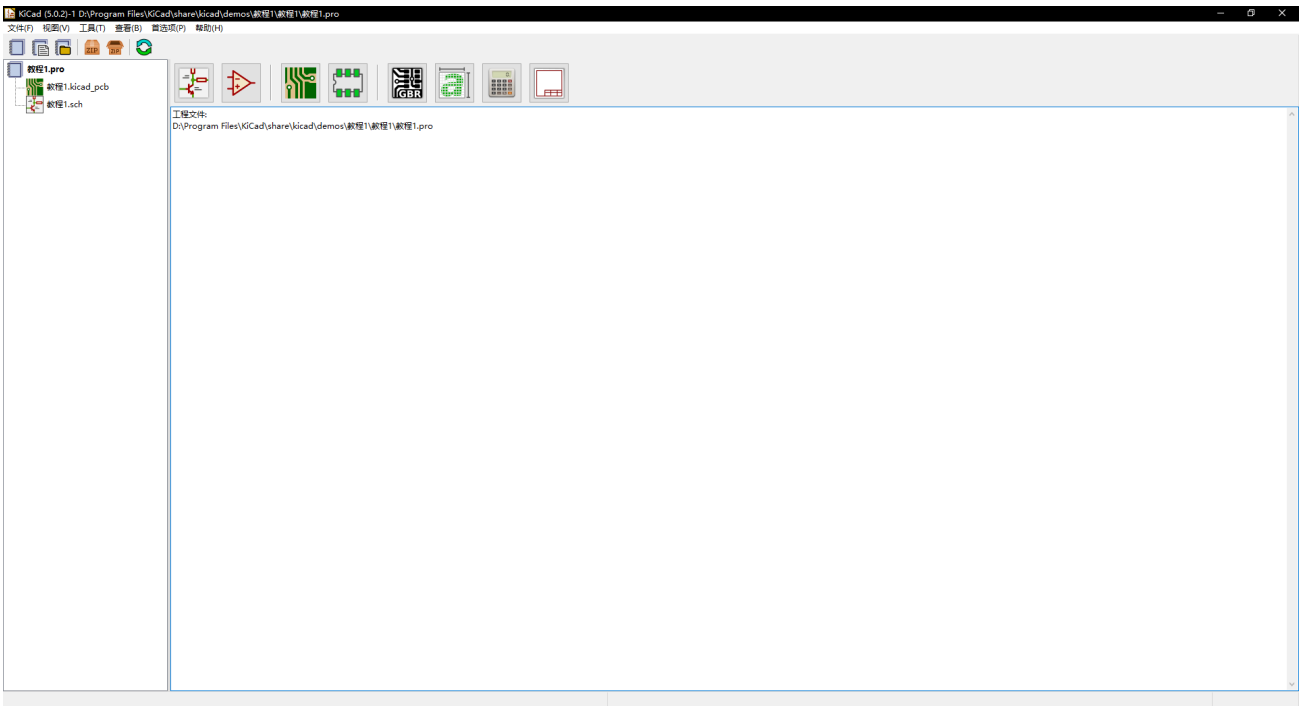
Chapter 4

绘制电路原理图

在本节中, 我们将学习如何使用 KiCad 绘制电路原理图。

4.1 使用 Eeschema

1. 在 Windows 下运行 kicad.exe。在 Linux 下，在终端中键入 *kicad* 。您现在位于 KiCad 项目管理器的主窗口中。从这里您可以访问八个独立的软件工具：*Eeschema*，原理图库编辑器，*Pcbnew*，*PCB* 封装编辑器，*Pcbnew*，*GerbView*，*Bitmap2Component*，*PCB* 计算器和 图框编辑器。请参阅工作流程图，了解如何使用主要工具。



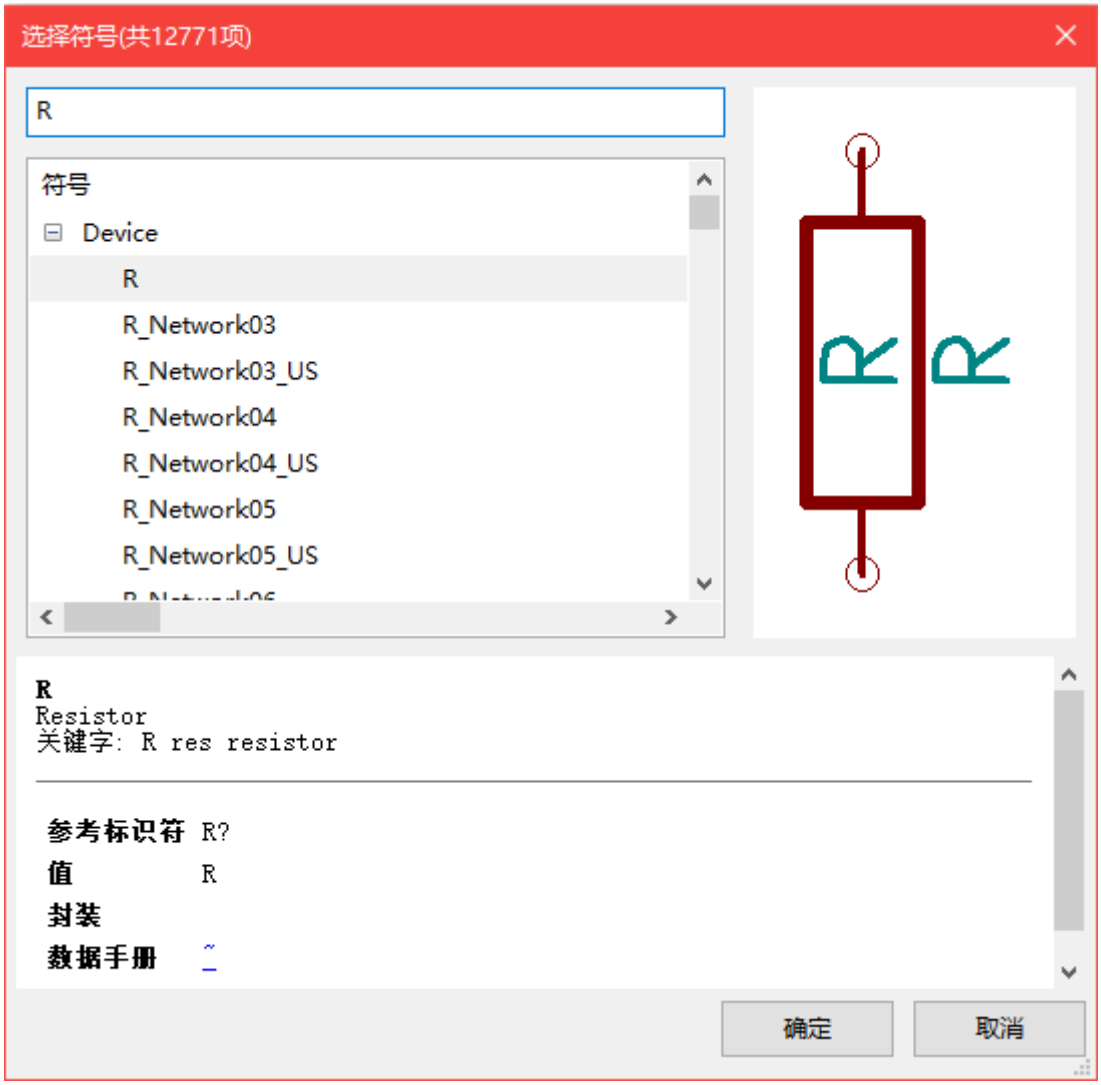
2. 创建一个新项目：**文件** → **新** → **项目**。将项目文件命名为 *教程 1*。项目文件将自动采用扩展名 *.pro*。对话框的确切外观取决于使用的平台，但应该有一个用于创建新目录的复选框。除非您已有专用目录，否则请保持检查状态。您的所有项目文件都将保存在那里。

3. 从创建原理图开始。开始原理图编辑器 *eeschema*，  。它是左边的第一个按钮。

4. 单击顶部工具栏上的 页面设置图标  。设置适当的 纸张尺寸（*A4*，'8.5x11' 等）并输入标题为 *教程 1*。如有必要，您将在此处看到可以输入更多信息。单击确定。此信息将填充右下角的原理图表。使用鼠标滚轮放大。保存整个原理图：**文件** → **保存**

5. 我们现在将放置第一个元件。单击右侧工具栏中的 放置符号图标  。您也可以按 添加符号热键 [a]。

6. 单击原理图工作表的中间部分。屏幕上将出现 选择符号窗口。我们要放一个电阻器。搜索/过滤 **R**esistor 的 *R*。您可能会注意到电阻器上方的 设备标题。此 设备标题是元件所在库的名称，这是一个非常通用且有用的库。



7. 双击它。这将关闭 选择符号窗口。通过单击元件在原理图中的的位置将其放置在原理图中。

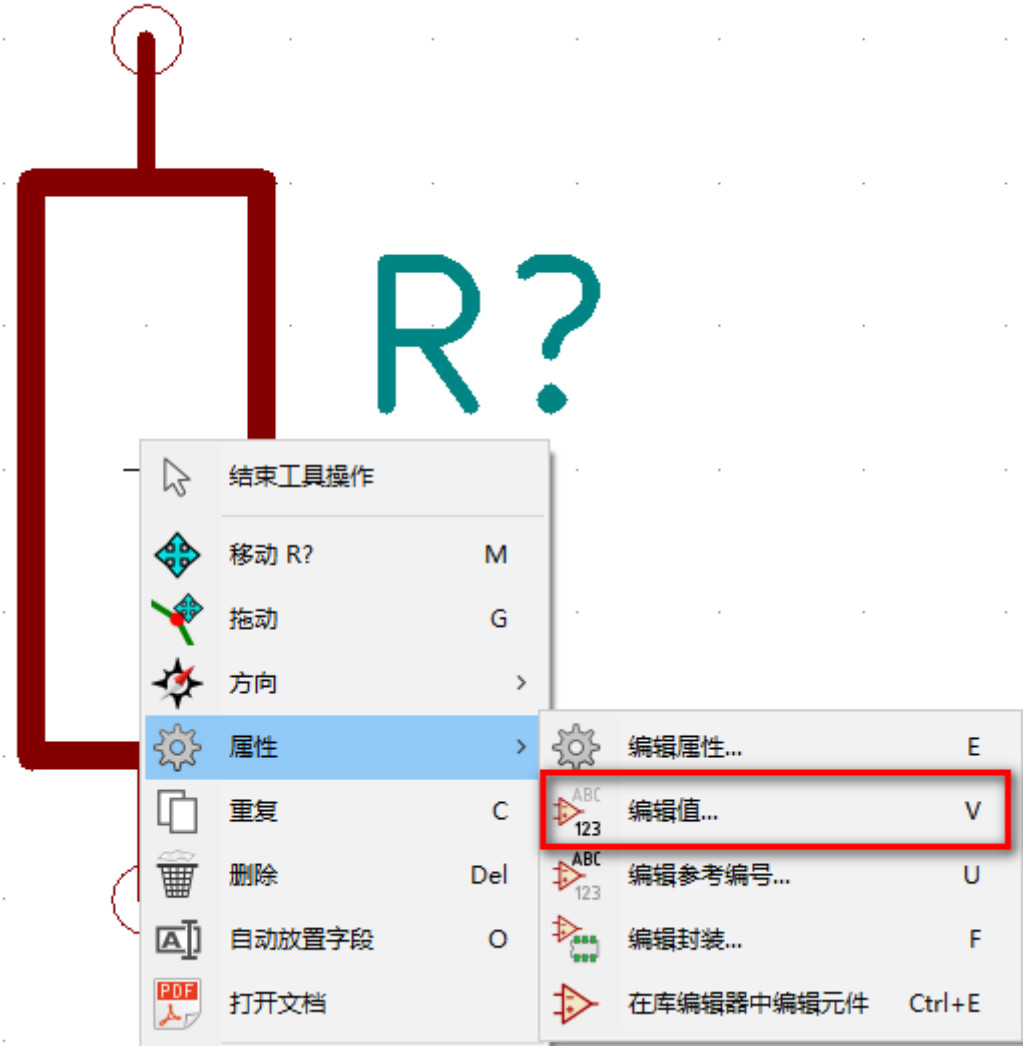
8. 单击放大镜图标可放大元件。也可以使用使用鼠标滚轮进行放大和缩小。按下滚轮 (中央) 鼠标按钮将可以水平和垂直平移。

9. 尝试将鼠标悬停在元件 *R* 上，然后按 [r]。元件应旋转。您不需要实际单击元件来旋转它。

Note

有时，如果您的鼠标也在其他地方，则会出现一个菜单。您将在 KiCad 中经常看到澄清选择菜单；它允许处理彼此重叠的对象。在这种情况下，如果出现菜单，请告诉 KiCad 您要对 _ 符号…R…_ 执行操作。

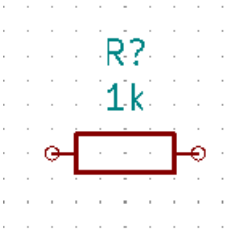
10. 右键单击元件的中间，然后选择 **属性** → **编辑值**。您可以通过将鼠标悬停在元件上并按 [v] 来实现相同的结果。或者，按下 [e] 将打开到更通用的 属性窗口。请注意，下面的右键单击菜单显示所有可用操作的热键。



11. 将出现 编辑值字段窗口。用 *1k* 替换当前值 *R*。单击确定。

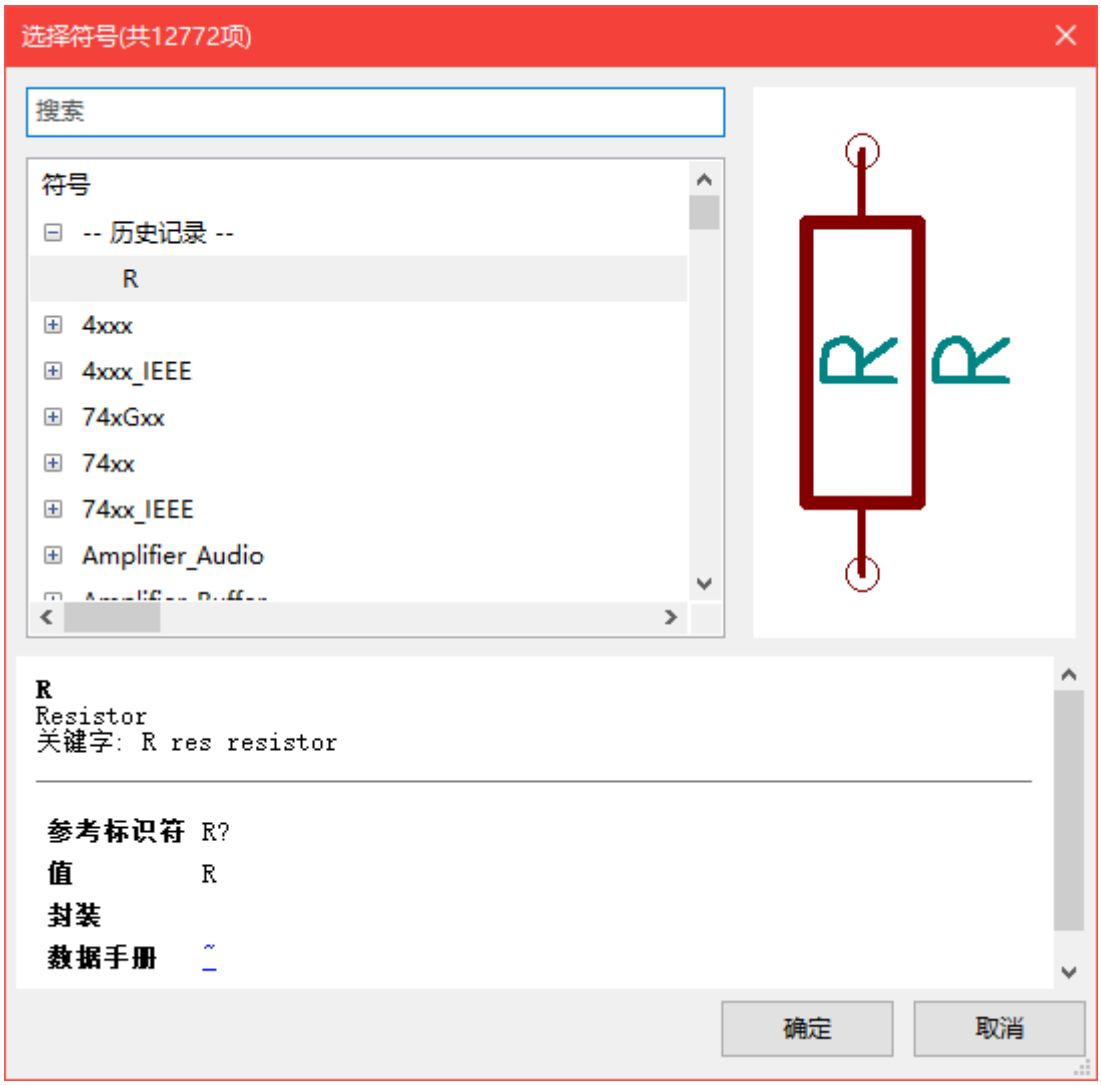
Note

不要更改参考字段（R?），稍后会自动完成。电阻器上方的值现在应为 *1k* 。



12. 要放置另一个电阻，只需单击要显示电阻的位置。符号选择窗口将再次出现。

13. 您之前选择的电阻现在位于历史列表中，显示为 *R* 。单击 *确定* 并放置元件。



14. 如果您输入错误并想要删除元件，请右键单击该元件并单击 *删除*。这将从原理图中删除元件。或者，您可以将鼠标悬停在要删除的元件上，然后按 [Delete]。

15. 您还可以通过将鼠标悬停在原理图页上并按 [C] 来复制已经存在于原理图页上的元件。单击要放置新复制元件的位置。

16. 右键单击第二个电阻。选择 *拖动*。重新定位元件并左键单击以放下。将鼠标悬停在元件上并按 [g] 可以实现相同的功能。[r] 将旋转元件，而 [x] 和 [y] 将围绕其 x 轴或 y 轴翻转。

Note

右键单击 → 移动或 [m] 也是一个有价值的选项移动任何东西，但最好只将它用于元件标签和元件尚未连接。我们稍后会看到为什么会这样。

17. 将鼠标悬停在第二个电阻上，然后按 [V] 键编辑该电阻。将 *R* 替换为 *100* 。可以使用 Ctrl+Z 撤消任何编辑操作。

18. 更改网格大小。您可能已经注意到，在原理图表上，所有元件都被捕捉到大间距网格上。您可以通过 **右键单击 → 网格**轻松更改网格的大小。通常，建议使用 *50.0mils* 的网格作为原理图表 _ 。

19. 我们将从库中添加一个可能未在默认项目中配置的元件。在菜单中，选择 **首选项 → 管理符号库**。在符号库窗口中，您可以看到两个选项卡：全局库和项目专有库。每个都有一个符号表库文件。要使库（.lib 文件）可用，它必须位于其中一个符号表库文件中。如果文件系统中有库文件但尚未提供，则可以将其添加到其中一个符号表库文件中。为了练习，我们现在将添加一个已经可用的库。

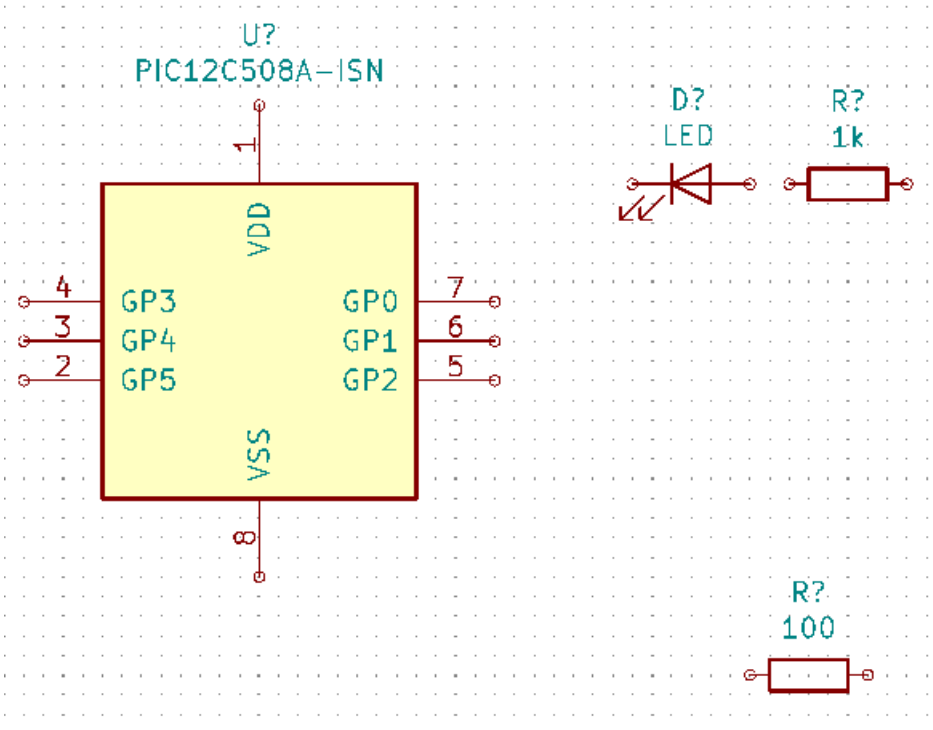
20. 选择项目专用表。单击表下方的文件浏览器按钮。您需要找到计算机上安装官方 KiCad 库的位置。查找包含一百个 *.dcm* 和 *.lib* 文件的 库目录。试试 *C:\Program Files (x86)\KiCad\share* (Windows) 和 */usr/share/kicad/library/* (Linux)。找到目录后，选择并添加 *MCU_Microchip_PIC12.lib* 库并关闭窗口。它将添加到列表的末尾。现在单击其昵称并将其更改为 *microchip_pic12mcu* 。单击确定关闭符号库窗口。

21. 重复添加元件步骤，但这次选择 *microchip_pic12mcu* 库而不是 设备库并选择 *PIC12C508A-ISN* 元件。

22. 将鼠标悬停在微控制器元件上。请注意，[x] 和 [y] 再次翻转元件。保持符号围绕 Y 轴镜像，使引脚 G0 和 G1 指向右侧。

23. 重复添加元件步骤，这次选择 设备库并从中选择 *LED* 元件。

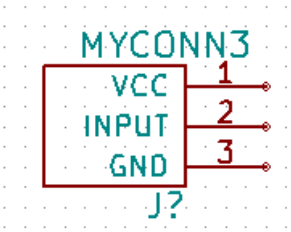
24. 组织原理图纸上的所有元件，如下所示。



25. 我们现在需要为我们的 3 针连接器创建原理图元件 “MYCONN3”。您可以跳转到标题为《make-schematic-symbols-in-kicad（制作-原理图-符号-在-kicad），Make Schematic Symbols in KiCad（在 KiCad 中制作原理图符号)》的部分，了解如何从头开始制作该元件，然后返回本节继续使用该板。

26. 您现在可以放置新制作的元件。按 [a] 并选择 *myLib* 库中的 *MYCONN3* 元件。

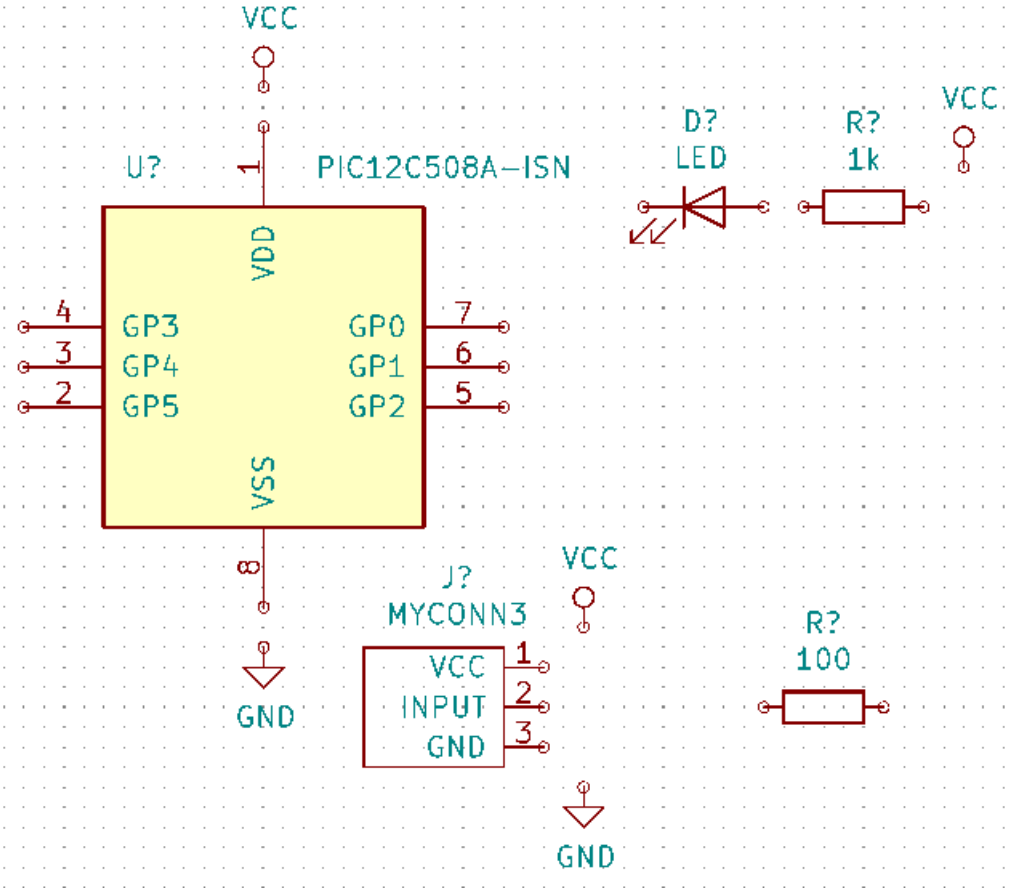
27. 元件标识符 *J?* 将出现在 *MYCONN3* 标签下。如果要更改其位置，请右键单击 *J?* 然后单击 *移动字段*（相当于 [m]）。在执行此操作之前/之后放大可能会有所帮助。重新定位 J? 在如下所示的元件下。标签可以随意移动多次。



28. 是时候放置电源和接地符号了。单击右侧工具栏上的 *放置电源端口*按钮  。或者，按 [p]。在元件选择窗口中，向下滚动并从 *电源*库中选择 *VCC* 。单击确定。

29. 单击 1k 电阻的引脚上方以放置 VCC 部件。单击微控制器 *VDD* 上方的区域。在 *元件选择历史记录*部分中，选择 *VCC* 并将其放在 VDD 引脚旁边。再次重复添加过程，并将 VCC 部分放在 *MYCONN3* 的 VCC 引脚上方。如果需要，可以移动引用和值。

30. 重复添加引脚步骤，但这次选择 GND 部分。将 GND 部分放在 *MYCONN3* 的 GND 引脚下。在微控制器的 VSS 引脚左侧放置另一个 GND 符号。您的原理图现在应该如下所示：



31. 接下来，我们将连接所有元件。单击右侧工具栏上的 放置电线图标 。

Note

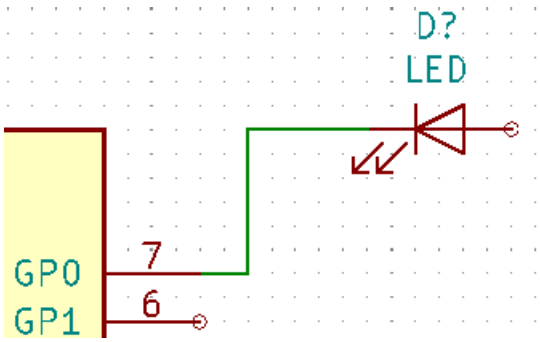
小心不要选择 放置总线，它位于此按钮的正下方，但线条较粗。《bus-connections-in-kicad（总线-连接-在-kicad），Bus Connections in KiCad（KiCad 中的总线连接）》 **部分将解释如何使用总线部分。**

---

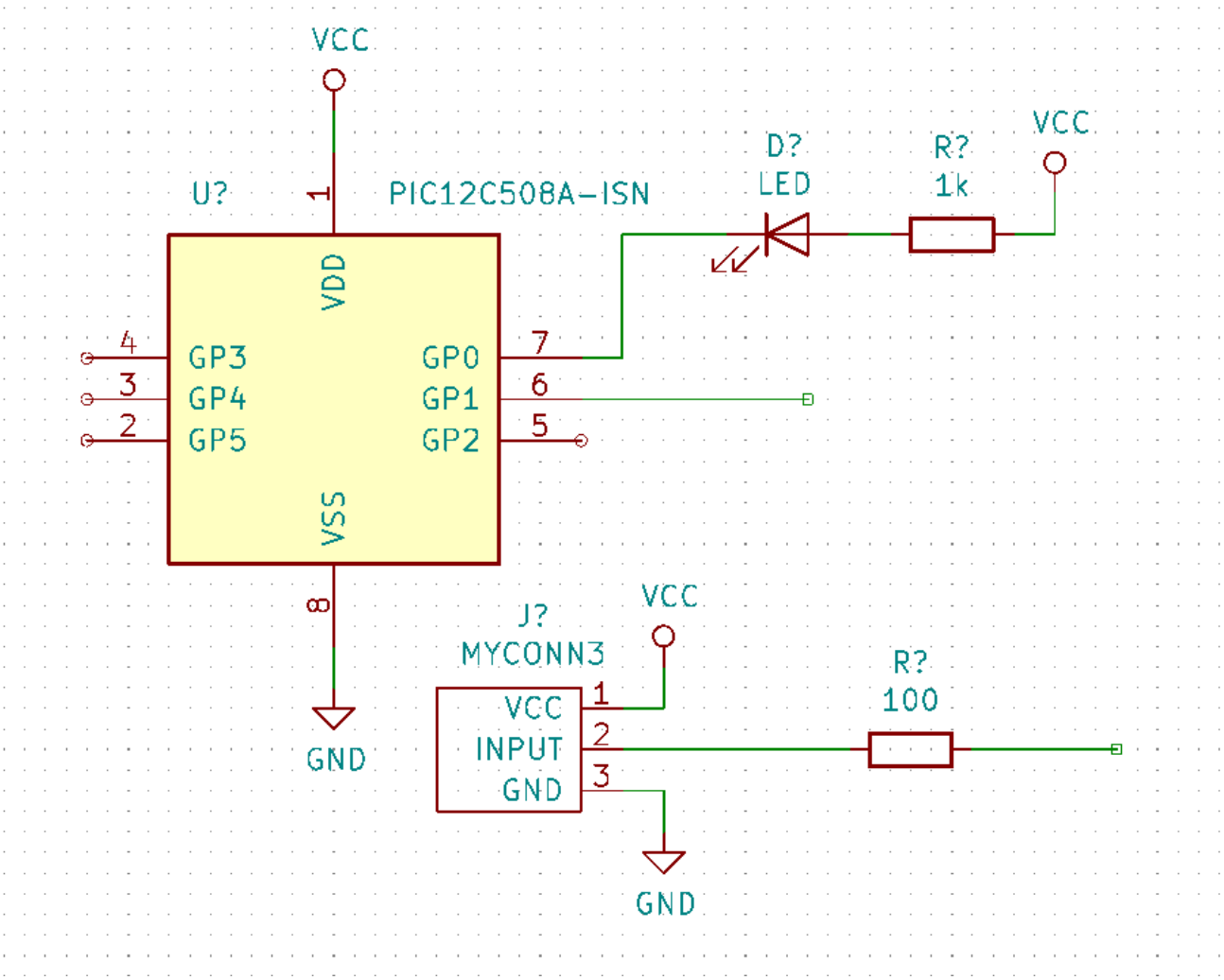
32. 单击微控制器引脚 7 末端的小圆圈，然后单击 LED 引脚 1 上的小圆圈。在绘制导线时单击一次以创建拐角。您可以在放置连接时放大。

Note

如果要重新定位连线元件，重要的是使用 [g]（**抓取**）**而不是** [m]（**移动**）**。使用抓取将保持电线连接。如果您忘记了如何移动元件，请查看步骤** 24。



33. 重复此过程并连接所有其他元件，如下所示。要双击终止电线。当连接 VCC 和 GND 符号时，导线应接触 VCC 符号的底部和 GND 符号的中间顶部。请参见下面的截图。



34. 我们现在将考虑使用标签建立连接的另一种方法。通过单击右侧工具栏上的 放置网络标签图标 来选择网络标签工具。你也可以用 [l]。

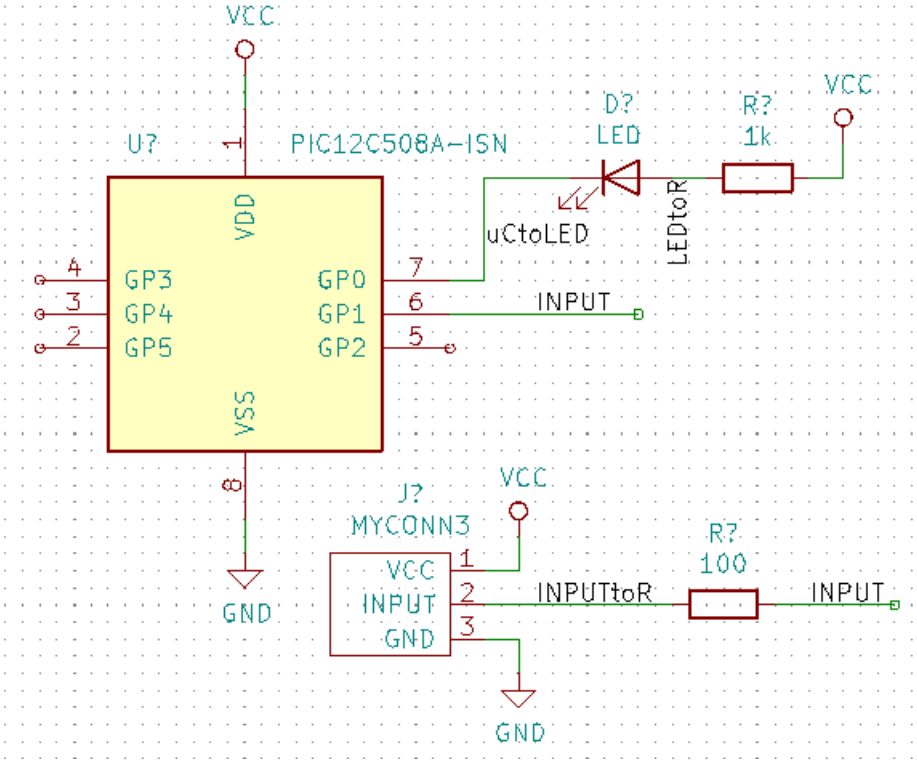
35. 单击连接到微控制器引脚 6 的导线中间。将此标签命名为 “INPUT（输入）”。标签仍然是一个独立的项目，您可以移动，旋转和删除。标签的小锚矩形必须正好在导线或引脚上才能使标签生效。

36. 按照相同的步骤在 100 欧姆电阻的右侧放置另一个标签。也可以将其命名为 *INPUT* 。这两个标签具有相同的名称，在 PIC 的引脚 6 和 100 欧姆电阻之间产生不可见的连接。当在复杂设计中连接导线时，这是一种有用的技术，其中绘制线条会使整个原理图变得更加混乱。要放置标签，您不一定需要电线，只需将标签贴在引脚上即可。

37. 标签也可用于简单地标记电线以用于提供信息。在 PIC 的引脚 7 上放置一个标签。输入名称 *uCtoLED* 。将电阻和 LED 之间的导线命名为 *LEDtoR* 。将 *MYCONN3* 和电阻之间的导线命名为 *INPUTtoR* 。

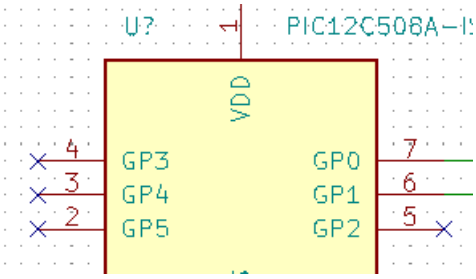
38. 您不必标记 VCC 和 GND 线，因为标签是从它们所连接的电源对象中隐含的。

39. 下面是最终的结果。



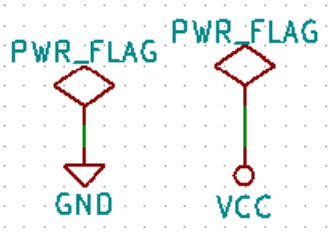
40. 我们现在处理未连接的电线。当 KiCad 检查时，任何未连接的引脚或电线都会产生警告。为了避免这些警告，您可以指示程序未经连接的电线是故意的，也可以手动将每个未连接的电线或引脚标记为未连接。

41. 单击右侧工具栏上的 *放置无连接标志图标*  。单击引脚 2,3,4 和 5，X 似乎表示缺少有线连接是故意的。



42. 某些元件具有不可见的电源引脚。您可以通过单击左侧工具栏上的 *显示隐藏的引脚图标*  使其可见。如果遵守 VCC 和 GND 命名，隐藏的电源引脚会自动连接。一般来说，你应该尽量不要制作隐藏的电源引脚。

43. 现在需要添加一个 *Power Flag* 来向 KiCad 表明电源是从某个地方进来的。按 [a] 并搜索 *电源库中的 PWR_FLAG* 。放置其中两个。将它们连接到 GND 引脚和 VCC，如下所示。



Note

这将避免经典的原理图检查警告：引脚连接到其他一些引脚但没有引脚来驱动它。

44. 有时候在这里和那里写注释是件好事。要在原理图上添加注释，请使用右侧工具栏上的 *放置文本图标* 。

45. 现在，所有元件都需要具有唯一标识符。事实上，我们的许多元件仍被命名为 *R?* 还是 *J?* 。通过单击顶部工具栏上的 *注释原理图符号图标* ，可以自动完成标识符分配。

46. 在注释原理图窗口中，选择 *使用整个原理图并单击 注释按钮*。点击 *关闭*。注意所有 *?* 已被数字取代。每个标识符现在都是唯一的在我们的例子中，它们被命名为 *R1*，*R2*，*U1*，*D1* 和 *J1* 。

47. 我们现在将检查原理图的错误。单击顶部工具栏上的 *执行电气规则检查图标* 。单击 *运行按钮*。生成一个报告，通知您任何错误或警告，例如断开的电线。你应该有 0 个错误和 0 个警告。如果出现错误或警告，原理图中将出现一个小绿色箭头，指示错误或警告所在的位置。选中 *创建 ERC 文件报告并再次按 运行按钮*以接收有关错误的更多信息。

Note

如果您收到 未找到默认编辑器，则必须选择它的警告，请尝试将路径设置为 *c:\windows\notepad.exe* (windows) **或** */usr/bin/gedit* (Linux)。

48. 原理图现已完成。我们现在可以创建一个网表文件，我们将添加每个元件的封装。单击顶部工具栏上的 *生成网表图标* 。单击 *生成网表按钮*并保存在默认文件名下。

Note

在以前版本的 KiCad 中，网表是必要的。在最近的版本中，您可以忽略它，而是使用 * 工具 * → 从原理图更新 PCB。如果这样做，您必须首先为符号指定封装。

49. 生成网表文件后，单击顶部工具栏上的 *运行 Cvpcb* 图标 。如果弹出丢失的文件错误窗口，请忽略它并单击 *确定*。

Note

还有很多方法可以为符号添加封装。

- **右键单击符号 → 属性 → 编辑封装双击符号，或右键单击符号 → 属性 → 编辑属性 → 封装**
- **工具 → 编辑符号字段在 Eeschema 的首选项中，检查符号选择器中的显示封装预览并在选择要放置的新符号时选择封装**

50. *Cvpcb* 允许您将原理图中的所有元件与 KiCad 库中的封装链接起来。中心的窗格显示原理图中使用的所有元件。在这里选择 *D1*。在右侧窗格中，您可以看到所有可用的占用空间，此处向下滚动到 *LED_THT: LED-D5.0mm* 并双击它。

51. 右侧窗格可能只显示可用封装的一个子集。点击图标  ,  和  以启用或禁用这些筛选器。

52. 对于 *U1*，选择 *Package_DIP*: *DIP-8_W7.62mm* 封装。对于 *J1*，选择 *Connector*: *Banana_Jack_3Pin* 封装。对于 *R1* 和 *R2*，选择 *Resistor_THT*: *R_Axial_DIN0207_L6.3mm_D2.5mm_P2.54mm_Vertical* 封装。

53. 如果您有兴趣知道您选择的封装是什么样的，您可以单击 查看所选封装图标  以预览当前封装。

54. 完成以后，可以通过点击 **文件** → **保存原理图**或使用 应用、保存原理图和继续按钮保存原理图。

55. 您可以关闭 *Cvpcb* 并返回 *Eeschema* 原理图编辑器。如果您没有将其保存在 Cvpcb 中，请单击 **文件** → **保存**立即保存。再次创建网表。您的网表文件现已更新，包含所有封装。请注意，如果您缺少任何设备的占地面积，则需要制作自己的封装。这将在本文档的后续部分中解释。

    Note

    现在每个符号都有封装。您可以使用 工具 → 从原理图更新 PCB，**而不是网表和接下来的两个步骤。如果您这样做，**Pcbnew **将使用 原理图更新** *PCB* **对话框中的打开。单击 更新** *PCB* **。然后，您可以按照本教程的** Pcbnew **部分中的说明进行操作。**

56. 切换到 KiCad 项目管理器。您可以在文件列表中看到网表文件。

57. 网表文件描述了所有元件及其各自的引脚连接。网表文件实际上是一个文本文件，您可以轻松地检查，编辑或编写脚本。

    Note

    库文件（ **.lib*）**也是文本文件，它们也很容易编辑或编写脚本。**

58. 要创建物料清单（BOM），请转到 *Eeschema* 原理图编辑器，然后单击顶部工具栏上的 生成物料清单图标  。默认情况下，没有处于活动的插件。您可以通过单击 **添加插件**按钮添加一个。选择要使用的 *.xsl 文件，在这种情况下，我们选择 *bom2csv.xsl*。

Note
Linux:
如果缺少 xsltproc，您可以下载并安装它：

```
sudo apt-get install xsltproc
```

对于像 Ubuntu 这样的 Debian 派生发行版，或者

```
sudo yum install xsltproc
```

对于 RedHat 派生的发行版。如果您不使用这两种发行版，请使用您的发行版软件包管理器命令来安装 xsltproc 软件包。
xsl 文件位于：*/usr/lib/kicad/plugins/*。
Apple OS X:
如果缺少 xsltproc，您可以从应该包含它的 Apple 站点安装 Apple Xcode 工具，或者下载并安装它：

```
brew install libxslt
```

xsl 文件位于：*/Library/Application Support/kicad/plugins/*。
Windows:
xsltproc.exe 和包含的 xsl 文件将分别位于 _《KiCad 安装目录》\bin_ 和 《*KiCad* 安装目录》*\bin\scripting\plugins*。
所有平台：
您可以通过以下方式获取最新的 bom2csv.xsl：
https://gitlab.com/kicad/code/kicad/raw/master/eeschema/plugins/xsl_scripts/bom2csv.xsl

KiCad 自动生成命令，例如：

```
xsltproc -o "%O" "/home/<user>/kicad/eeschema/plugins/bom2csv.xsl" "%I"
```

您可能想要添加扩展名，因此请将此命令行更改为：

```
xsltproc -o "%O.csv" "/home/<user>/kicad/eeschema/plugins/bom2csv.xsl" "%I"
```

按 *帮助*按钮获取更多信息。

59. 现在按 *生成*。该文件（与项目同名）位于项目文件夹中。使用 LibreOffice Calc 或 Excel 打开 * **.csv** 文件。将出现导入窗口，按 OK。

您现在可以转到 PCB 布局部分，这将在下一节中介绍。但是，在继续之前，让我们快速了解如何使用总线连接元件引脚。

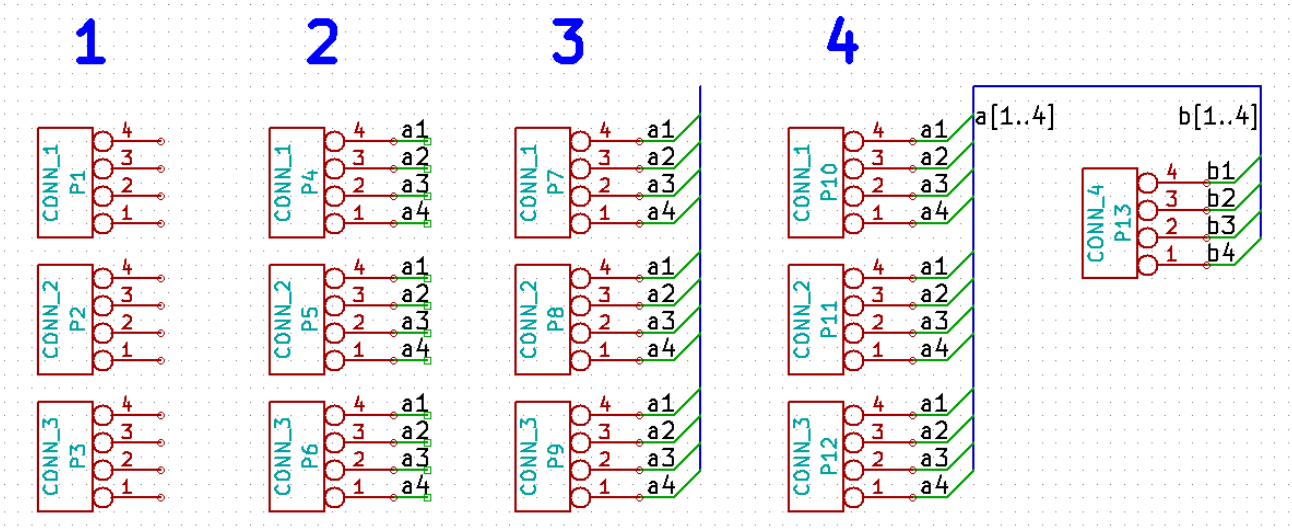
4.2 KiCad 的总线连接

有时您可能需要将元件 A 的多个顺序引脚与元件 B 的其他顺序引脚连接。在这种情况下，您有两个选项：我们已经看到的标记方法或使用总线连接。让我们看看如何做到这一点。

1. 让我们假设您有三个 4 针连接器，您想要将引脚连接在一起。使用标签选项（按 [l]）标记 P4 部件的引脚 4。将此标签命名为 *a1* 。现在按 [Insert] 将相同的项目自动添加到引脚 4（引脚 3）下方的引脚上。注意标签是如何自动重命名为 *a2* 。
2. 再按 [Insert] 两次。此键对应于 重复最后一项操作，它是一个无限有用的命令，可以让您的生活更轻松。
3. 在另外两个连接器 CONN_2 和 CONN_3 上重复相同的标签操作，您就完成了。如果继续制作 PCB，您将看到三个连接器相互连接。图 2 显示了我们描述的结果。出于美观目的，还可以使用图标图像添加一系列 将电线放入总线入口  和使用图标图像的总线线路：  ，如图 3 所示。但是，请注意，对 PCB 没有影响。
4. 应该指出的是，连接到图 2 中的引脚的短导线不是严格必需的。实际上，标签可以直接应用于引脚。
5. 让我们更进一步，假设你有一个名为 CONN_4 的第四个连接器，无论出于什么原因，它的标签恰好有点不同（b1，b2，b3，b4）。现在我们想要以引脚到引脚的方式将 _Bus a_ 与 *Bus b* 连接起来。我们希望不使用引脚标记（这也是可能的），而是使用总线上的标签，每个总线一个标签。
6. 使用之前说明的标记方法连接并标记 CONN_4。将引脚命名为 b1，b2，b3 和 b4。使用图标  将引脚连接到一系列 电线到总线入口，并使用图标  连接到总线。见图 4。
7. 在 CONN_4 的总线上放一个标签（按 [l]）并命名为 *b[1..4]* 。
8. 在前一个总线上放一个标签（按 [l]）并将其命名为 *a[1..4]* 。
9. 我们现在可以做的是使用带有按钮图像的总线连接总线 *a[1..4]* 和总线 *b[1..4]* ：  。
10. 通过将两条总线连接在一起，引脚 a1 将自动连接到引脚 b1，a2 将连接到 b2，依此类推。图 4 显示了最终结果的样子。

> **Note**
>
> **通过 [Insert] 可访问的 重复上一项选项可以成功用于重复期间项目插入。例如，连接到图 2，图 3 和图 4 中所有引脚的短导线都已放置此选项。**

11. 通过 [Insert] 访问的 重复上一项选项也被广泛用于使用图标  放置许多系列的 电线到总线入口。



Chapter 5

布局印刷电路板

现在是时候使用您生成的网表文件来布局 PCB 了。这是通过 *Pcbnew* 工具完成的。

Note

如果您使用来自 *Eeschema* 的 原理图更新到 *PCB*，则不需要网表和步骤 5. 您现在可以像步骤 6 和 7 一样将脚印放入板中，然后按步骤 2~4 输入表单信息和设计规则。

5.1 使用 Pcbnew

1. 从 KiCad 项目管理器，单击 *Pcb 布局编辑器*图标  。您还可以使用 *Eeschema* 中的相应工具栏按钮。*Pcbnew* 窗口将打开。如果您收到一条消息，指出 **.kicad_pcb* 文件不存在并询问您是否要创建它，只需单击是。

2. 首先输入一些原理图信息。单击顶部工具栏上的 *页面设置*图标  。将相应的 *纸张尺寸*（A4，8.5x11 等）和标题设置为 *教程 1* 。

3. 最好将 **间距**和 **最小布线宽度**设置为 PCB 制造商要求的宽度。通常，您可以将间隙设置为 *0.25mm*，将最小轨道宽度设置为 *0.25mm* 。单击 **设置** → **设计规则**菜单。如果它尚未显示，请单击 *网络类编辑器*选项卡。将窗口顶部的 *间距*字段更改为 *0.25mm*，将 *布线宽度*字段更改为 *0.25mm*，如下所示。这里的测量单位是 *mm*。

网络类:

	间距	布线宽度	过孔外径	过孔内径	微孔外径	微孔内孔	差分对宽度	差分对间距
Default	0.254	0.4	1.4	0.6	0.5	0.127	0.2	0.25
Power	0.254	0.5	1.6	0.6	0.5	0.127	0.2	0.25

4. 单击 全局设计规则选项卡，将 最小布线宽度设置为 *0.25mm* 。单击 确定按钮以提交更改并关闭 设计规则编辑器窗口。

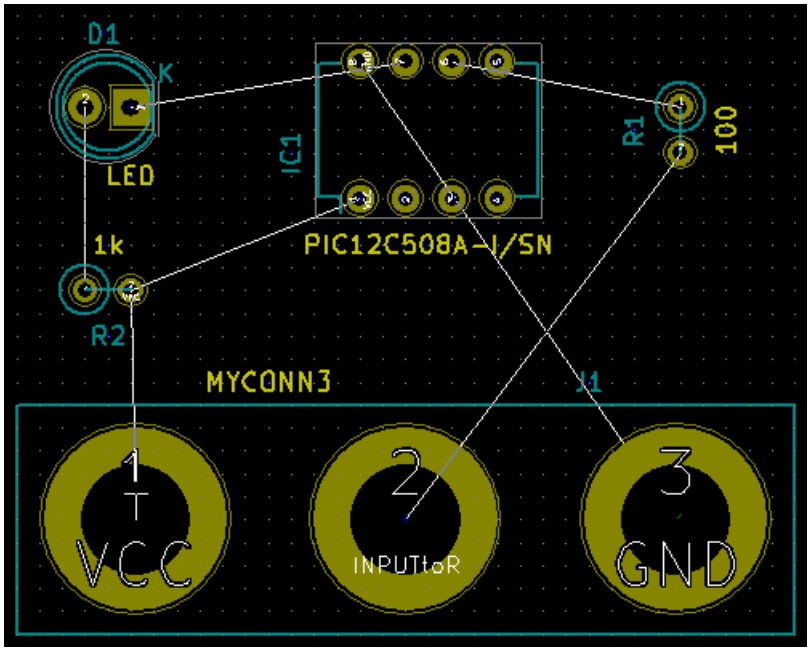
5. 现在我们将导入网表文件，如果你创建了一个。单击顶部工具栏上的 读取网表图标 。如果网表文件 *教程 1.net* 是从 Eeschema 创建的，则应在 网表文件字段中选择。点击 读取当前网表。然后单击 关闭按钮。

6. 现在应该可以看到所有元件。选择它们并按照鼠标光标。

7. 将元件移动到板的中间。如有必要，您可以在移动元件时放大和缩小。单击鼠标左键。

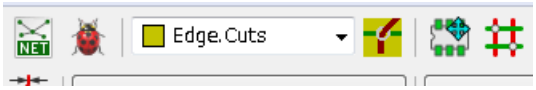
8. 所有元件都通过称为 飞线的一组细线连接。确保按下 显示/隐藏板飞线按钮 。通过这种方式，您可以看到链接所有元件的最快速度。

9. 您可以通过将每个元件悬停在其上并按 [m] 来移动它们。单击要放置它们的位置。或者，您可以通过单击选择元件然后拖动它。按 [r] 旋转元件。移动所有元件，直到最小化电线交叉的数量。



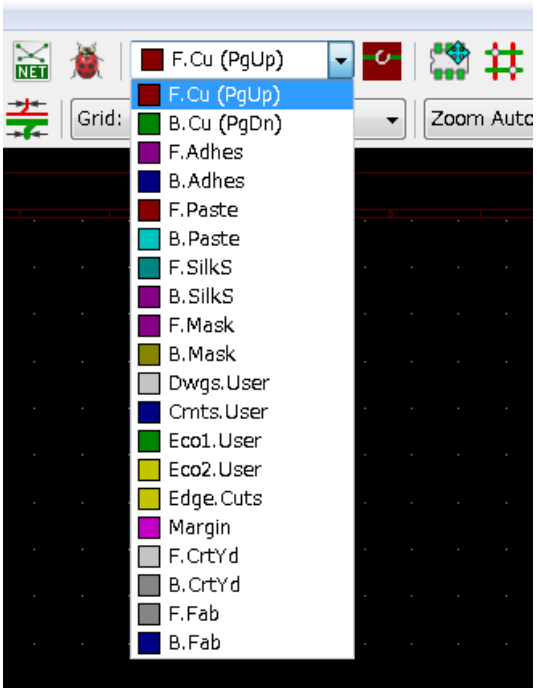
10. 注意 100 欧姆电阻的一个引脚如何连接到 PIC 元件的引脚 6。这是用于连接引脚的标记方法的结果。标签通常比实际的电线更受欢迎，因为它们使原理图更加杂乱。

11. 现在我们将定义 PCB 的边缘。从顶部工具栏的下拉菜单中选择 边切（边缘切割）图层。单击右侧工具栏上的 添加图形线图标 。沿着电路板边缘，在每个角落点击，并记住在绿色边缘和 PCB 边缘之间留一个小间隙。



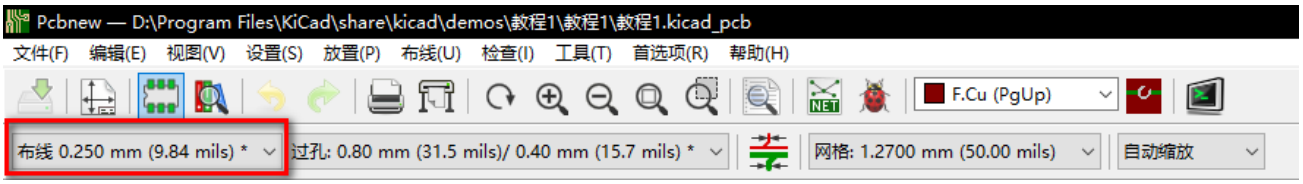
12. 接下来，连接除 GND 之外的所有电线。事实上，我们将使用放置在电路板底部铜线（称为 *B.Cu* ）的接地层一次连接所有 GND 连接。

13. 现在我们必须选择我们想要处理的铜层。在顶部工具栏的下拉菜单中选择 *F.Cu*（*PgUp*）。这是前顶部铜层。



14. 例如，如果您决定改为使用 4 层 PCB，请转到 **设置** → **层设置**并将 *铜层*更改为 4. 在 *层*表中，您可以命名图层并确定它们的含义用于。请注意，可以通过 *预设图层分组*菜单选择非常有用的预设。

15. 单击右侧工具栏上的 *布线*图标 。单击 *J1* 的第 1 针并运行轨道以填充 *R2* 。双击以设置轨道结束的点。该轨道的宽度将默认为 0.250mm。您可以从顶部工具栏的下拉菜单中更改布线宽度。请注意，默认情况下，您只有一个可用的布线宽度。



16. 如果您想添加更多的轨道宽度，请转到：**设置** → **设计规则** → **全局设计规则**选项卡，在此窗口的右下角添加您希望可用的任何其他宽度。然后，您可以在布置电路板时从下拉菜单中选择布线的宽度。请参阅下面的示例（英寸）。

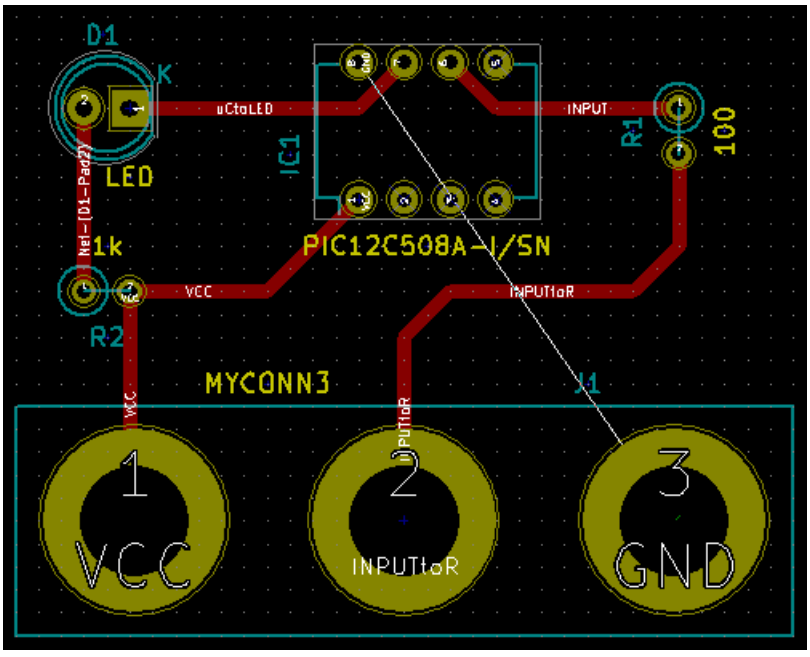
自定义布线宽度:

	宽度
布线 1	0.381
布线 2	0.762
布线 3	
布线 4	
布线 5	
布线 6	
布线 7	
布线 8	

17. 或者，您可以添加一个 网络类，在其中指定一组选项。转到 **设置** → **设计规则** → **网络类编辑器**并添加一个名为 电源的新类。将布线宽度从 8mil（表示为 0.0080）更改为 24mil（表示为 0.0240）。接下来，将除地面之外的所有内容添加到 电源类（左侧选择 默认，右侧选择 电源并使用箭头）。

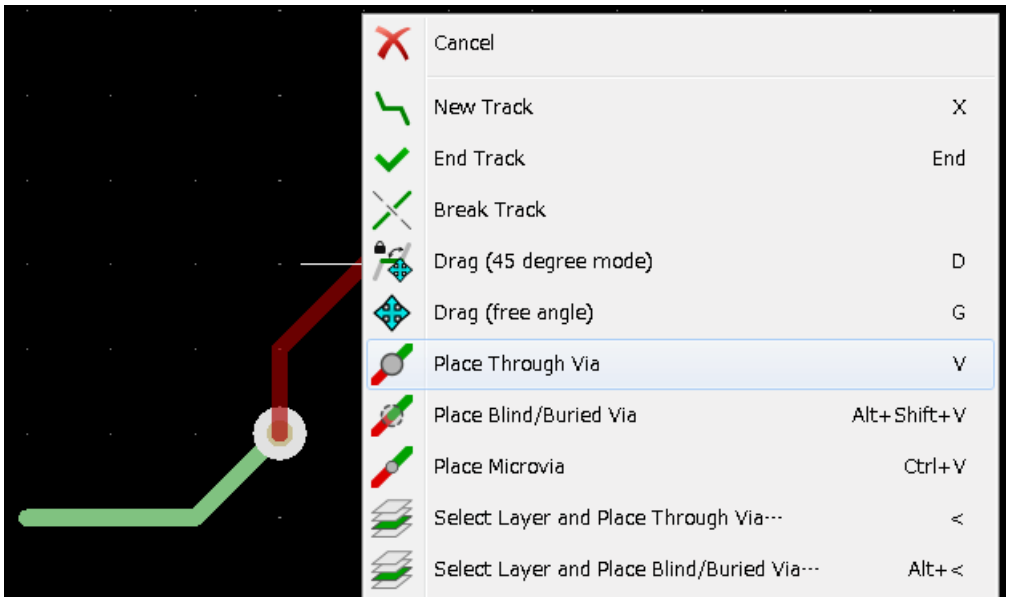
18. 如果要更改网格大小，**右键单击** → **网格**。在放下元件并将它们与轨道连接在一起之前或之后，请务必选择合适的网格尺寸。

19. 重复此过程，直到连接除 J1 的引脚 3 之外的所有电线。您的电路板应如下所示。



20. 现在让我们在 PCB 的另一个侧上运行布线。在顶部工具栏的下拉菜单中选择 *B.Cu* 。点击 布线图标 。在 J1 的引脚 3 和 U1 的引脚 8 之间画一条线。这实际上没有必要，因为我们可以用地平面做到这一点。注意轨道的颜色是如何变化的。

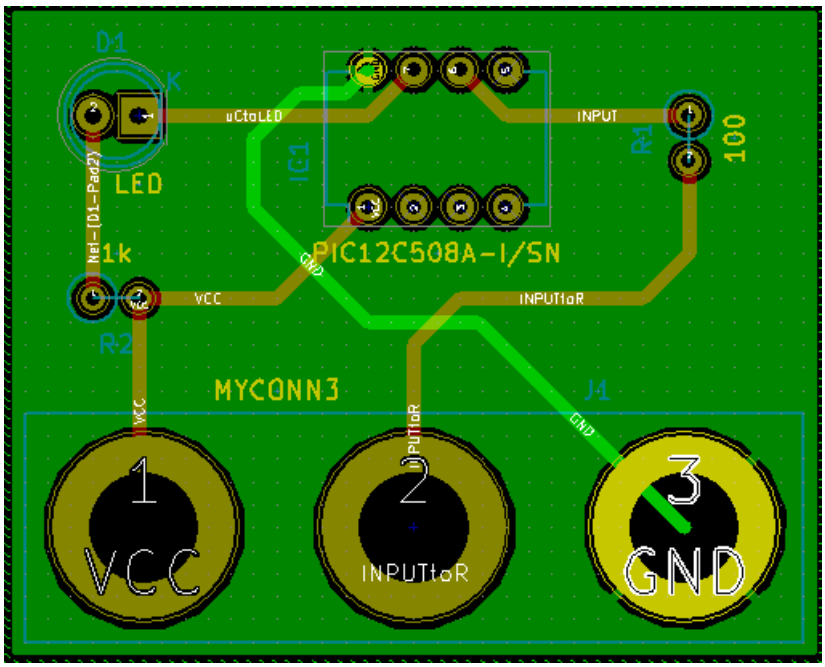
21. **通过更改图层**从引脚 A 到引脚 B。有可能通过放置布线来运行时更改铜平面过孔通过当您在上部铜平面上运行布线时，右键单击并选择 放置过孔或只需按 [v]。这将带您到最底层，您可以完成您的布线。



22. 当您想要检查特定连接时，可以单击右侧工具栏上的 高亮网络图标 。点击 J1 的引脚 3。布线本身和连接到它的所有焊盘都应高亮显示。

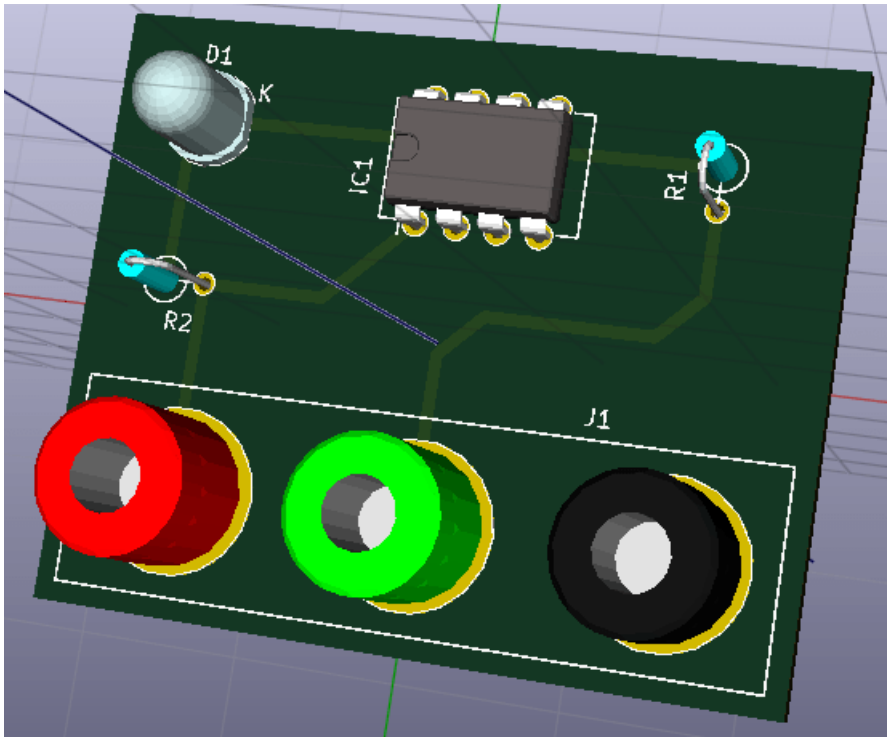
23. 现在我们将制作一个连接到所有 GND 引脚的接地层。单击右侧工具栏上的 添加填充区域图标 。我们将在板周围布线一个矩形，因此单击您想要其中一个角的位置。在出现的对话框中，将 默认焊盘连接设置为 防散热（花焊盘），将 轮廓角度设置为 仅限 *H*，*V* 和 *45* 度，然后单击 确定。

24. 通过单击旋转中的每个角来跟踪板的轮廓。通过第二次单击第一个角完成矩形。右键单击刚刚跟踪的区域。单击 区域 → 填充或重新填充所有区域。板应填充为绿色，看起来像这样：



25. 单击顶部工具栏上的 执行设计规则检查图标 运行设计规则检查器。点击 开始 *DRC* 。应该没有错误。点击 列表未连接。应该没有未连接的项目。单击 确定关闭 *DRC* 控制对话框。

26. 单击 **文件** → **保存**保存文件。要以 3D 方式观察您的电路板，请单击 **视图** → **3D 查看器**。



27. 您可以拖动鼠标来旋转 PCB。

28. 你的板是完整的。要将其发送给制造商，您需要生成所有 Gerber 文件。

5.2 生成 Gerber 文件

完成 PCB 后，您可以为每一层生成 Gerber 文件，并将它们发送给您最喜欢的 PCB 制造商，他们将为您制作电路板。

1. 从 KiCad，打开 *Pcbnew* 板编辑器。
2. 点击 **文件** → **绘制** 。选择 *Gerber* 作为 绘制格式并选择放置所有 Gerber 文件的文件夹。单击 绘制按钮继续。
3. 要生成钻孔文件，请从 *Pcbnew* 再次转到 **文件** → **绘制**选项。默认设置应该没问题。
4. 这些是您制作典型的 2 层 PCB 时需要选择的层：

层	**KiCad 层名**	**默认 Gerber 扩展**	**“使用 Protel 文件扩展名” 已启用**
Bottom Layer（底层）	B.Cu（底层）	.GBR	.GBL
Top Layer（顶层）	F.Cu （顶层）	.GBR	.GTL
Top Overlay （顶层丝印层）	F.SilkS （顶层丝印层）	.GBR	.GTO
Bottom Solder Resist（底层阻焊层）	B.Mask （底层阻焊层）	.GBR	.GBS

层	KiCad 层名	默认 Gerber 扩展	“使用 Protel 文件扩展名” 已启用
Top Solder Resist（顶层阻焊层）	F.Mask（顶层阻焊层）	.GBR	.GTS
Edges（边缘（板框）层）	Edge.Cuts （边缘（板框）层）	.GBR	.GM1

5.3 使用 Gerbview

1. 要查看所有 Gerber 文件，请转到 KiCad 项目管理器并单击 *GerbView* 图标。在下拉菜单或图层管理器中，选择 *图形图层 1* 。单击 **文件 → 打开 Gerber 文件**或单击图标  。选择并打开所有生成的 Gerber 文件。注意它们如何一个显示在另一个之上。
2. 使用 **文件 → 打开 Excellon 钻孔文件**打开钻孔文件。
3. 使用右侧的图层管理器选择/取消选择要显示的图层。在发送生产之前仔细检查每一层。
4. 该视图与 Pcbnew 类似。在视图内右键单击并单击 网格以更改网格。

5.4 使用 FreeRouter 自动布线

手动布线板既快速又有趣，但对于具有大量元件的电路板，您可能希望使用自动布线器。请记住，您应首先手动路由关键迹线，然后设置自动布线器以执行无聊位。它的工作只会解释未布线的痕迹。我们将在这里使用的自动布线器是 FreeRouting。

Note

FreeRouting **是一个开源的** Java **应用程序。目前**，FreeRouting **存在于几个或多或少相同的副本中，您可以通过互联网搜索** *freerouting* **找到它们。它可以在源代码形式或预编译的** Java **包中找到**。

1. 从 *Pcbnew* 单击 **文件 → 导出 → Specctra DSN** 并在本地保存文件。启动 FreeRouter 并单击 打开您自己的设计按钮，浏览 *dsn* 文件并加载它。
2. FreeRouter 具有 KiCad 目前不具备的一些功能，包括手动路由和自动路由。FreeRouter 主要有两个步骤：首先，对电路板进行布线，然后对其进行优化。完全优化可能需要很长时间，但您可以随时停止它。
3. 您可以通过单击顶部栏上的 自动布线器按钮来启动自动路由。底栏为您提供有关正在进行的路由过程的信息。如果 通过计数超过 30，则您的电路板可能无法使用此路由器自动执行。更多地展开您的元件或更好地旋转它们并再试一次。零件的旋转和位置的目标是降低速率的交叉走线的数量。
4. 左键单击鼠标可以停止自动路由并自动启动优化过程。另一次左键单击将停止优化过程。除非你真的需要停下来，否则最好让 FreeRouter 完成它的工作。
5. 单击 **文件 → 导出 Specctra 会话文件**菜单并使用 *.ses* 扩展名保存板文件。您实际上不需要保存 FreeRouter 规则文件。

6. 回到 *Pcbnew*。您可以通过单击 **文件** → **导入** → **Spectra 会话**并选择 *.ses* 文件来导入刚刚布线的电路板。

如果有任何您不喜欢的布线，可以使用 [Delete] 和路由工具删除它并重新路由它，这是 布线图标  在右侧工具栏上添加。

Chapter 6

在 KiCad 中转发注释

完成电子原理图，封装分配，电路板布局和生成 Gerber 文件后，您就可以将所有内容发送给 PCB 制造商，以便您的电路板成为现实。

通常，这种线性工作流程并非如此单向。例如，当您必须修改/扩展您或其他人已经完成此工作流程的板时，您可能需要移动元件，替换其他元件，更改占用空间等等。在此修改过程中，您不想做的是从头开始重新布线整个电路板。相反，这是你如何做到这一点：

1. 假设您想要用 CON2 替换假设的连接器 CON1。
2. 您已经拥有完整的原理图和完全布线的 PCB。
3. 从 KiCad 开始 *Eeschema*，通过删除 CON1 并添加 CON2 进行修改。使用图标图像保存原理图项目  并点击顶部工具栏上的 网表生成图标  。
4. 点击 网表然后点击 保存。保存为默认文件名。你必须重写旧的。
5. 现在为 CON2 分配封装。单击顶部工具栏上的 运行 *Cvpcb* 图标  。将封装分配给新设备 CON2。其余元件仍然具有分配给它们的先前封装。关闭 *Cvpcb* 。
6. 返回原理图编辑器，单击 文件 → 保存整个原理图项目保存项目。关闭原理图编辑器。
7. 从 KiCad 项目管理器，单击 *Pcbnew* 图标。*Pcbnew* 窗口将打开。
8. 旧的，已经布线的电路板应该自动打开。让我们导入新的网表文件。单击顶部工具栏上的 读取网表图标  。
9. 单击 浏览网表文件按钮，在文件选择对话框中选择网表文件，然后单击 读取当前网表。然后单击 关闭按钮。
10. 此时，您应该能够看到已布置所有先前元件的布局。在左上角，您应该看到所有未布线的元件，在我们的例子中是 CON2。用鼠标选择 CON2。将元件移动到板的中间。
11. 放置 CON2 并布线它。完成后，保存并继续照常生成 Gerber 文件。

这里描述的过程可以根据需要轻松地重复多次。除了上面描述的前向注释方法之外，还有另一种方法称为后向注释。此方法允许您从 Pcbnew 修改已布线的 PCB，并在原理图和网表文件中更新这些修改。然而，后向注释方法没有那么有用，因此这里不描述它。

Chapter 7

在 KiCad 中制作原理图符号

有时，您想要放置在原理图中的符号不在 KiCad 库中。这很正常，没有理由担心。在本节中，我们将了解如何使用 KiCad 快速创建新的原理图符号。不过，请记住，您始终可以在互联网上找到 KiCad 元件。

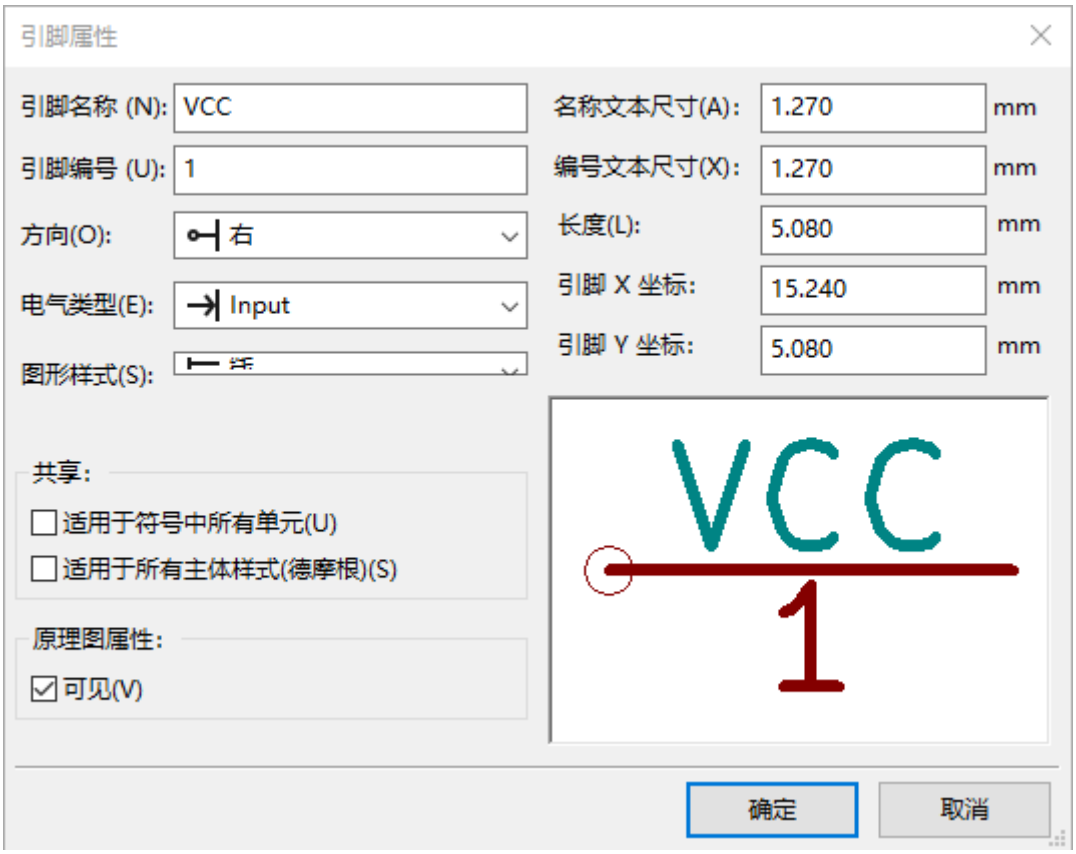
在 KiCad 中，符号是一段以 *DEF* 开头并以 *ENDDEF* 结尾的文本。一个或多个符号通常放在库文件中，扩展名为 *.lib*。如果要将符号添加到库文件，只需使用文本编辑器的剪切和粘贴命令即可。

7.1 使用元件库编辑器

1. 我们可以使用 元件库编辑器（*Eeschema* 的一部分）来创建新元件。在我们的项目文件夹 *教程 1* 中，让我们创建一个名为 *库*的文件夹。在我们创建新元件后，我们将在内部放置新的库文件 *myLib.lib* 。

2. 现在我们可以开始创建新元件了。从 KiCad 开始 *Eeschema*，单击 *库编辑器*图标  ，然后单击 *新元件*图标  。将出现 *元件属性*窗口。将新组件命名为 *MYCONN3* ，将 *默认参考指示符*设置为 *J* ，将 *每个包的单位数*设置为 *1* 。单击确定。如果出现警告，则单击 *是*。此时，元件仅由其标签组成。让我们添加一些引脚。单击右侧工具栏上的 *添加引脚*图标  。要放置引脚，请在 *MYCONN3* 标签正下方的零件编辑器工作表中间单击鼠标左键。

3. 在出现的引脚属性窗口中，将引脚名称设置为 *VCC* ，将引脚编号设置为 *1* ，将 *电气类型*设置为 *电源输入*，然后单击 *确定*。

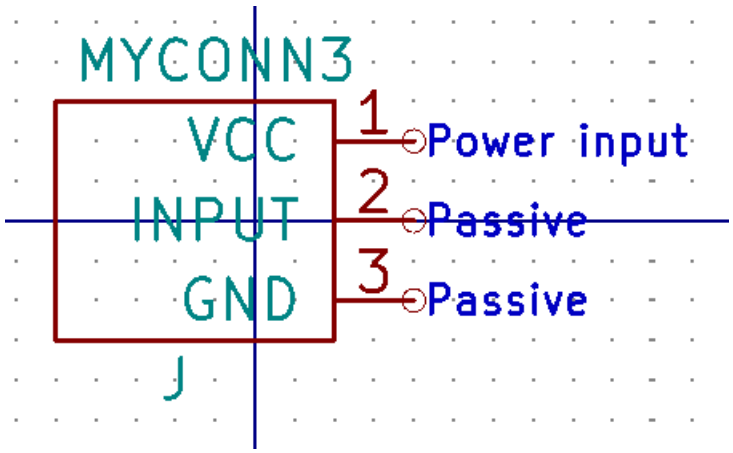


4. 在 *MYCONN3* 标签下方单击您想要的位置放置引脚。

5. 重复执行引脚步骤，这次 *引脚名*应为 *输入*，*引脚号*应为 *2*，*电气类型*应为 *被动*。

6. 重复放置引脚步骤，这次 *引脚名称*应为 *GND*，*引脚编号*应为 *3*，*电气类型*应为 *被动*。将销钉一个放在另一个的顶部。元件标签 *MYCONN3* 应位于页面的中心（蓝线交叉的位置）。

7. 接下来，绘制元件的轮廓。单击 *添加矩形图标*  。我们想在引脚旁边绘制一个矩形，如下所示。为此，请单击矩形左上角的位置（不要按住鼠标按钮）。再次单击矩形右下角的位置。



8. 如果要用黄色填充矩形，请在 **首选项** → **选择颜色方案**中将填充颜色设置为 *黄色 4*，然后使用 [e] 在编辑屏幕中选择矩形，选择 *填充背景*。

9. 将元件保存在库 *myLib.lib* 中。单击 新建库图标  ，导航到 教程 *1/*库*/* 文件夹并保存名为 *myLib.lib* 的新库文件。

10. 转到 **首选项** → **元件库**并在 用户定义的搜索路径中添加 教程 *1/*库*/*，在 元件库文件中添加 *mylib.lib* 。

11. 单击 选择工作库图标  。在 选择库窗口中，单击 *myLib* ，然后单击 确定。注意窗口的标题如何表示当前正在使用的库，现在应该是 *myLib* 。

12. 单击顶部工具栏中的 更新当前库中的当前元件图标  。单击顶部工具栏中的 在磁盘上保存当前加载的库图标  保存所有更改。在出现的任何确认消息中单击 是。现在，新的逻辑示意图元件已在窗口标题栏中指示的库中完成并可用。

13. 您现在可以关闭 元件库编辑器窗口。您将返回到原理图编辑器窗口。您的新元件现在可以从库 *myLib* 中使用。

14. 您可以通过将库 *.lib* 文件添加到库路径来使其可用。从 *Eeschema* 开始，转到 **首选项** → **库**并在 用户定义的搜索路径中添加路径，在 元件库文件中添加 *file.lib* 。

7.2 导出，导入和修改库元件

而不是从头开始创建库元件，有时从已经制作并修改它更容易。在本节中，我们将了解如何将元件从 KiCad 标准库设备导出到您自己的库 *myOwnLib.lib* ，然后进行修改。

1. 从 KiCad 开始 *Eeschema*，点击 库编辑器图标  ，点击 选择工作库图标  并选择库的 设备。单击 从当前库中加载元件进行编辑图标 image:images/icons/import_cmp_from_lib.png[从当前库中加载元件进行编辑图标] 并导入 *RELAY_2RT* 。

2. 单击“导出元件”图标  ，导航到 库*/* 文件夹并保存名为 *myOwnLib.lib* 的新库文件。

3. 您可以将此元件和整个库 *myOwnLib.lib* 添加到库路径中，从而使您可以使用它。从 *Eeschema* ，转到 **首选项** → **元件库**并在 用户定义的搜索路径中添加 *library/* ，在 元件库文件中添加 *myOwnLib.lib* 。关闭窗口。

4. 单击 选择工作库图标  。在选择库窗口中，单击 *myOwnLib* 并单击 OK。注意窗口的标题如何表示当前正在使用的库，它应该是 *myOwnLib* 。

5. 单击 从当前库中加载元件进行编辑图标  并导入 *RELAY_2RT* 。

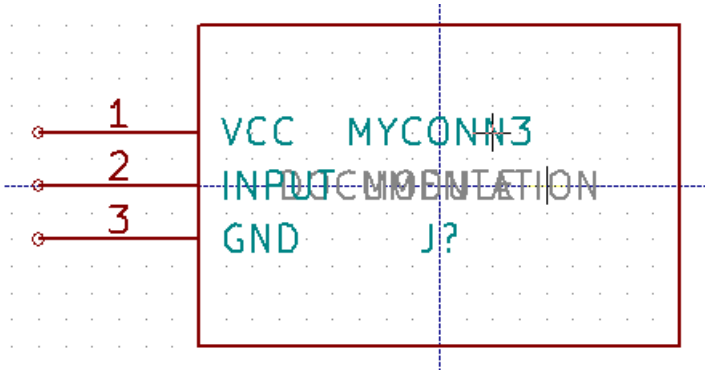
6. 您现在可以根据需要修改元件。将鼠标悬停在 *RELAY_2RT* 标签上，按 [e] 并将其重命名为 *MY_RELAY_2RT* 。

7. 单击顶部工具栏中的 更新当前库中的当前元件图标  。单击顶部工具栏中的 在磁盘上保存当前加载的库图标  保存所有更改。

7.3 使用 quicklib 制作原理图元件

本节介绍使用互联网工具 *quicklib* 为 MYCONN3 创建原理图元件的另一种方法(参见上面的《myconn3, MYCONN3》)。

1. 前往 *quicklib* 网页：http://kicad.rohrbacher.net/quicklib.php
2. 使用以下信息填写页面：元件名称：MYCONN3 参考前缀：J 引脚布局样式：SIL 引脚数，N:3
3. 单击 分配引脚图标。使用以下信息填写页面：引脚 1：VCC; 引脚 2：输入引脚;3：GND。类型：所有 3 个引脚都被动。
4. 单击 预览图标，如果您满意，请单击 构建库元件。下载文件并将其重命名为 *教程 1/库/myQuickLib.lib* 。你完成了!
5. 使用 KiCad 查看它。从 KiCad 项目管理器，启动 *Eeschema*，单击 库编辑器图标  ，单击 导入元件图标  ，导航到 *教程 1/库/* 并选择 *myQuickLib.lib* 。



6. 您可以通过将此元件和整个库 *myQuickLib.lib* 添加到 KiCad 库路径来使其可用。从 *Eeschema* ，转到 **首选项** → **元件库**并在 用户定义的搜索路径中添加 *library* ，在 元件库文件中添加 *myQuickLib.lib* 。

正如您可能猜到的，当您想要创建具有大引脚数的元件时，这种创建库元件的方法非常有效。

7.4 制作高引脚数的原理图元件

在 *quicklib* 中标题为 制作原理图元件的部分中，我们了解了如何使用 *quicklib* 基于 Web 的工具制作原理图元件。但是，您偶尔会发现需要创建一个具有大量引脚（几百个引脚）的原理图元件。在 KiCad 中，这不是一项非常复杂的任务。

1. 假设您要为具有 50 个引脚的器件创建原理图元件。通常的做法是使用多个低引脚数的图纸来绘制它，例如两个图纸，每个图纸有 25 个引脚。该元件表示允许简单的引脚连接。
2. 创建元件的最佳方法是使用 *quicklib* 分别生成两个 25 引脚元件，使用 Python 脚本重新编号它们的引脚，最后通过使用复制和粘贴将它们合并为一个单独的 DEF 和 ENDDEF 元件。
3. 您将在下面找到一个简单的 Python 脚本示例，它可以与 *in.txt* 文件和 *out.txt* 文件一起使用以重新编号该行：`X PIN1 1 -750 600 300 R 50 50 1 1 I` into `X PIN26 26 -750 600 300 R 50 50 1 1 I` 这对文件 *in.txt* 中的所有行都已完成。

简单的脚本

```
#!/usr/bin/env python
''' b'' 操 b''b'' 作 b'' KiCad b'' 元 b''b'' 件 b''b'' 引 b''b'' 脚 b''b'' 编 b''b'' 号 b''b'' 的
    b''b'' 简 b''b'' 单 b''b'' 脚 b''b'' 本 b'' '''
''' simple script to manipulate KiCad component pins numbering'''
import sys, re
try:
    fin=open(sys.argv[1],'r')
    fout=open(sys.argv[2],'w')
except:
    print "oh, wrong use of this app, try:", sys.argv[0], "in.txt out.txt"
    sys.exit()
for ln in fin.readlines():
    obj=re.search("(X PIN)(\d*)(\s)(\d*)(\s.*)",ln)
if obj:
    num = int(obj.group(2))+25
    ln=obj.group(1) + str(num) + obj.group(3) + str(num) + obj.group(5) +'\n'
    fout.write(ln)
fin.close(); fout.close()
#
# b'' 有 b''b'' 关 b''b'' 正 b''b'' 则 b''b'' 表 b''b'' 达 b''b'' 式 b''b'' 语 b''b'' 法 b''b'' 和
    b'' KiCad b'' 元 b''b'' 件 b''b'' 生 b''b'' 成 b''b'' 的 b''b'' 详 b''b'' 细 b''b'' 信 b''b'' 息
    b''b'': b''
# for more info about regular expression syntax and KiCad component generation:
# http://gskinner.com/RegExr/
# http://kicad.rohrbacher.net/quicklib.php
```

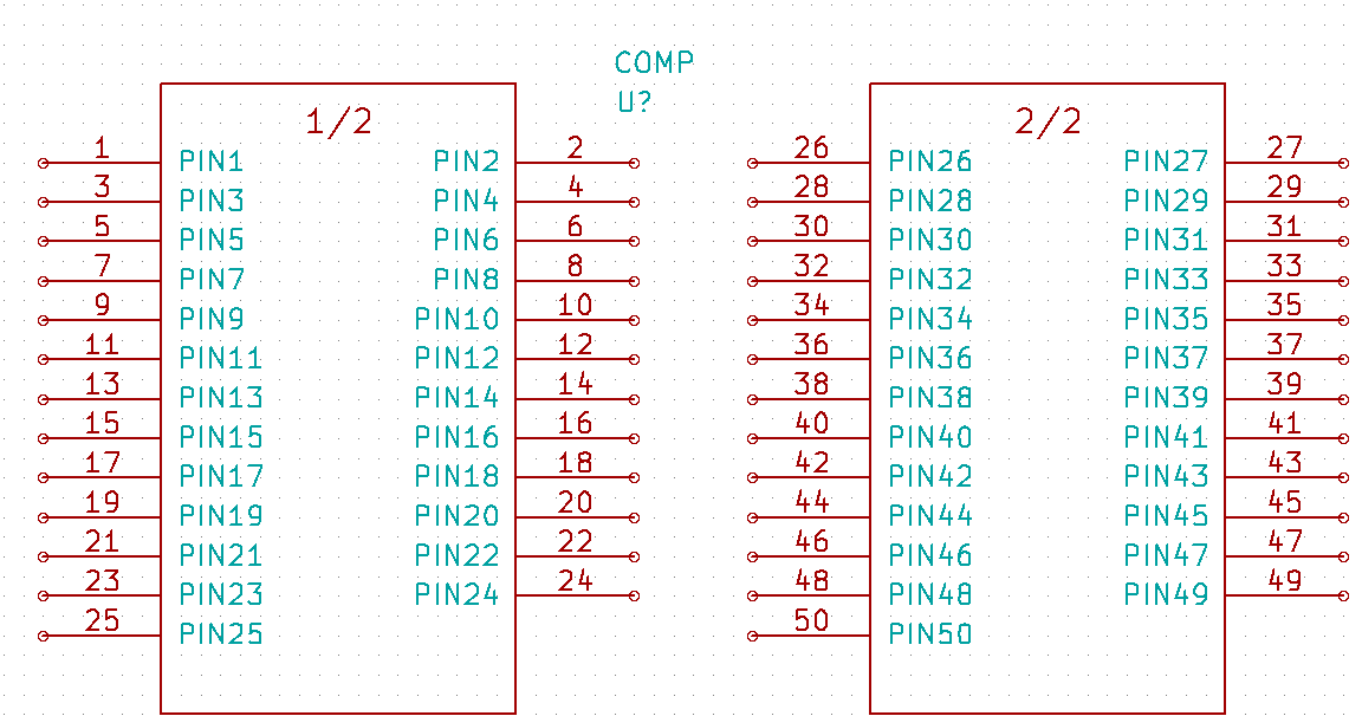
1. 在将两个元件合并为一个元件时，有必要使用 Eeschema 的库编辑器移动第一个元件，以便第二个元件不会在它上面移动。您将在下面找到最终的.lib 文件及其在 *Eeschema* 中的表示形式。

***.lib 文件的内容**

```
EESchema-LIBRARY Version 2.3
#encoding utf-8
# COMP
DEF COMP U 0 40 Y Y 1 F N
F0 "U" -1800 -100 50 H V C CNN
F1 "COMP" -1800 100 50 H V C CNN
DRAW
S -2250 -800 -1350 800 0 0 0 N
S -450 -800 450 800 0 0 0 N
X PIN1 1 -2550 600 300 R 50 50 1 1 I

...

X PIN49 49 750 -500 300 L 50 50 1 1 I
ENDDRAW
ENDDEF
#End Library
```



1. 这里介绍的 Python 脚本是一个非常强大的工具，用于操作引脚号和引脚标签。然而，请注意，它的所有功能都来自于神秘而且非常有用的正则表达式语法：*http://gskinner.com/RegExr/*.

Chapter 8

制作元件封装

与其他 EDA 软件工具不同，其中一种类型的库包含原理图符号和封装变化，KiCad *.lib* 文件包含原理图符号，*.kicad_mod* 文件包含封装。*Cvpcb* 用于将封装映射到符号。

对于 *.lib* 文件，*.kicad_mod* 库文件是可以包含从一个部分到多个部分的任何文本文件。

有一个广泛的封装库与 KiCad，但有时您可能会发现您需要的封装不在 KiCad 库中。以下是在 KiCad 中创建新 PCB 封装的步骤：

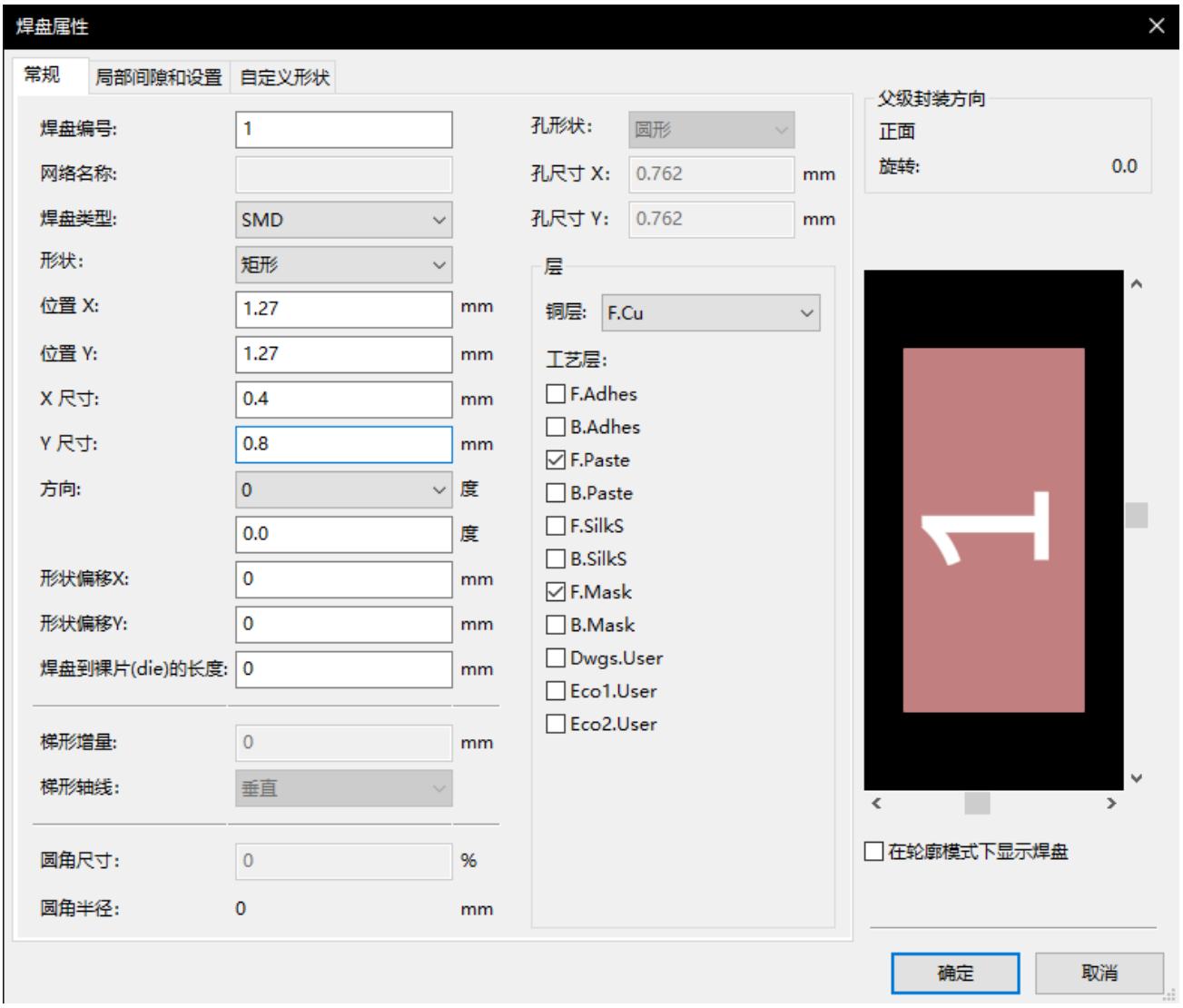
8.1 使用封装编辑器

1. 从 KiCad 项目管理器开始 *Pcbnew* 工具。单击顶部工具栏上的 *打开封装编辑器图标*  。这将打开 *封装编辑器*。

2. 我们将在新的封装库 *myfootprint* 中保存新的封装 *MYCONN3* 。在 *教程 1/* 项目文件夹中创建一个新文件夹 *myfootprint.pretty* 。单击 **首选项** → **封装库管理器**并按 *附加库*按钮。在表格中，输入 *myfootprint* 作为昵称，输入 *${KIPRJMOD}/myfootprint.pretty* 作为库路径，并输入 *KiCad* 作为插件类型。按 *确定*关闭 *PCB 库表*窗口。单击顶部工具栏上的 *选择活动库*图标  。选择 *myfootprint* 库。

3. 单击顶部工具栏上的 *新封装*图标  。输入 *MYCONN3* 作为 *封装名称*。在屏幕中间将出现 *MYCONN3* 标签。在标签下，您可以看到 “REF*” 标签。*右键单击* MYCONN3 *并将其移到* “*REF**” 上方。右键单击 “REF___*”，选择 *编辑文本*并将其重命名为 *SMD* 。将 *显示值*设置为 *不可见*。

4. 在右侧工具栏中选择 *添加焊盘*图标  。单击工作表以放置焊盘。右键单击新焊盘，然后单击 *编辑焊盘*。你也可以用 [e]。



5. 将 *焊盘编号* 设置为 *1* ，*焊盘形状* 设置为 *矩形*，*焊盘类型* 设置为 *SMD* ，*形状大小 X* 设置为 *0.4* ，*形状大小 Y* 设置为 *0.8* 。单击确定。再次点击 *添加焊盘* 并再放两个焊盘。

6. 如果要更改网格大小，**右键单击 → 网格选择**。在放下元件之前，请务必选择合适的网格尺寸。

7. 将 *MYCONN3* 标签和 *SMD* 标签移开，使其看起来像上图所示。

8. 放置焊盘时，通常需要测量相对距离。将光标放在您想要相对坐标点 （*0,0*）的位置，然后按空格键。移动光标时，您将看到光标在页面底部位置的相对指示。可以随时按空格键设置新原点。

9. 现在添加一个封装边框。单击右侧工具栏中的 *添加图形线或多边形按钮*  。绘制元件周围连接器的边框。

10. 单击顶部工具栏上的 *在活动库中保存封装图标*  ，使用默认名称 MYCONN3。

Chapter 9

关于 KiCad 项目文件的可移植性的注意事项

您需要将哪些文件发送给某人才能完全加载和使用您的 KiCad 项目?

当你有一个 KiCad 项目与某人分享时，重要的是原理图文件 *.sch*，板文件 *.kicad_pcb* ，项目文件 *.pro* 和网表文件 *.net*，与两个原理图一起发送库文件 *.lib* 和封装文件 *.kicad_mod* 。只有这样，人们才能完全自由地修改原理图和电路板。

使用 KiCad 原理图，人们需要包含符号的 *.lib* 文件。需要在 *Eeschema* 首选项中加载这些库文件。另一方面，使用板（*.kicad_pcb* 文件），封装可以存储在 *.kicad_pcb* 文件中。您可以向某人发送 *.kicad_pcb* 文件，而不是其他任何内容，他们仍然可以查看和编辑该板。但是，当他们想要从网表加载元件时，脚本库（*.kicad_mod* 文件）需要存在并加载到 *Pcbnew* 首选项中，就像原理图一样。此外，有必要在 *Pcbnew* 的首选项中加载 *.kicad_mod* 文件，以便在 *Cvpcb* 中显示这些封装。

如果有人向您发送了一个带有封装的 *.kicad_pcb* 文件，您可以在另一个板上使用，您可以打开封装编辑器，从当前板上加载封装，并将其保存或导出到另一个封装库中。您也可以通过 **Pcbnew** → **文件** → **压缩** → **封装 - 创建封装压缩**，这将创建一个新的一次导出 *.kicad_pcb* 文件中的所有封装。*.kicad_mod* 文件包含所有板的封装。

最重要的是，如果 PCB 是你想要分发的唯一东西，那么电路板文件 *.kicad_pcb* 就足够了。但是，如果您想让人们完全能够使用和修改您的原理图，其元件和 PCB，强烈建议您压缩并发送以下项目目录:

```
b'' 教 b''b'' 程 b''1/
|-- b'' 教 b''b'' 程 b''1.pro
|-- b'' 教 b''b'' 程 b''1.sch
|-- b'' 教 b''b'' 程 b''1.kicad_pcb
|-- b'' 教 b''b'' 程 b''1.net
|-- library/
|   |-- myLib.lib
|   |-- myOwnLib.lib
|   \-- myQuickLib.lib
|
|-- myfootprint.pretty/
|   \-- MYCONN3.kicad_mod
|
\-- gerber/
    |-- ...
```

```
    \-- ...
```

Chapter 10

有关 KiCad 文档的更多信息

这是 KiCad 中大多数功能的快速指南。有关更详细的说明，请参阅可通过每个 KiCad 模块访问的帮助文件。点击 **帮助** → **手册**。

KiCad 为其所有四个软件元件提供了一套非常好的多语言手册。

所有 KiCad 手册的英文版都随 KiCad 一起发布。

除了手册之外，KiCad 还随本教程一起发布，该教程已被翻译成其他语言。本教程的所有不同版本均免费分发所有最新版本的 KiCad。本教程以及手册应与您的 KiCad 版本一起打包在您的指定平台上。

例如，在 Linux 上，典型位置位于以下目录中，具体取决于您的确切分布：

```
/usr/share/doc/kicad/help/zh/
/usr/local/share/doc/kicad/help/zh
```

在 Windows 上它位于：

```
<installation directory>/share/doc/kicad/help/zh
```

在 OS X 上：

```
/Library/Application Support/kicad/help/zh
```

10.1 网上的 KiCad 文档

最新版本的 KiCad 文档可以在 http://docs.kicad.org 上找到多种语言。由于官方的 CI/CD 环境过时官网提供的中文文档版本存在排版错乱的问题。KiCad 中文文档排版优化版本下载地址：https://gitee.com/KiCAD-CN/KiCad-doc_build/releases