

CvPcb

Table of Contents

CvPcb 介	3
CvPcb 特性	3
手 或自	3
后 Cvpcb	3
CvPcb 命令	4
主界面	4
主界面工具	4
主界面 快捷方式	5
CvPcb 配置	6
封装 管理	7
重要提示:	7
封装 表	7
使用封装 列表添加向	12
看当前封装	16
封装命令	16
看当前 3D 模型	19
使用 CvPcb 向元件分配封装	21
手 分配封装	21
封装列表	21
自	25
Equivalence 文件	25
Equivalence 文件格式	25
自 元件分配封装	26

参考手册

版

本文档版 所有 © 2010-2018 , 其 献者如下所列。您可以根据 GNU 通用公共 可 (<https://www.gnu.org/licenses/gpl.html>) , 版 本 3 或更高版本 , 或知 共享署名 可 (<https://creativecommons.org/licenses/by/3.0/>) , 版本 3.0 或更高版本的条款分 和/或修改它。

本文档中的所有商 都属于其合法所有者。

献者

Jean-Pierre Charras, Fabrizio Tappero, Wayne Stambaugh.

翻 人

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

taotieren <admin@taotieren.com>, 2019, 2020, 2021.

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

反

将所有的 Bug, 建 以及新版本重定向于此:

- 于 KiCad 文档: <https://gitlab.com/kicad/services/kicad-doc/issues>
- 于 KiCad 件: <https://gitlab.com/kicad/code/kicad/issues>
- 于 KiCad 件国 化: <https://gitlab.com/kicad/code/kicad-i18n/issues>

行日期及 件版本

布于 2015-05-22。

CvPcb 介

CvPcb 能 原理 中的元器件与 行PCB布局 的封装分配 二者的 系将被添加入由原理 建程序 Eeschema 建的网 列表文件中。

当在元器件的封装字段初始化后,由 Eeschema 生成的网 列表文件才会包含元器件 PCB 封装与原理 端口的系。

种情况下, 封装和原理 之 的 是用 在 原理 通 置元件的封装字段 建的。此外封装也可能被 定义于原理 符号 中,在用 从 中加 元器件 ,其封装会被自 置。

CvPcb 提供了在 建原理 的 程中 元器件分配 PCB 封装的 便方法。它 有封装列表 ,封装 以及 3D 模型 功能。些功能旨在提高分配封装 的准确率。

用 可以手 元器件分配 的封装。通 建 .equ 文件,也可以 封装的自 分配。 .equ 文件包含了元器件和其 封装的相 信息。

我 认 使用 种交互式的封装分配方法,比起直接在 制原理 的 候 行封装分配,更加 ,并且 有更高的正确率。

使用 CvPcb,你可以看到所有可能可用的封装列表。此外,你 能在窗口中看 不同封装的真 几何外形,可以帮助你 原理 中的元器件 正确的封装。

CvPcb 只能通 Eeschema 后 ,其入口位于 Eeschema 的 部工具 无 Eeschema 是通 Kicad 的 目管理器后 ,是作 独立 件 独后 ,都可以通 其 部工具 按 CvPcb。

从通 Kicad 目管理器后 的 Eeschema CvPcb 通常来 是更好的 ,是因 :

- CvPcb 需要 取 目配置文件,以确定哪些封装 需要被加
- 当 目文件和打开的原理 文件 于同一目 ,Cvpcb 可以初始化元器件的封装 置字段。

从通 Kicad 目管理器后 的 Eeschema CvPcb,可以确保上述要求自 得到 足。

WARNING

尽管用 确 能 从 独后 的 Eeschema 中 Cvpcb,但是 注意,独打开的原理 文件由于可能缺失相 的 目文件,而 致缺失相 的文件,会使 Cvpcb 的工作出 如果在 独打开的原理 文件所在的目 中,不包括其他 fp-lib-table 文件,那么工程封装 列表也是不可用的。

CvPcb 特性

手 或自

Cvpcb 同 支持交互式的手 封装分配和通 .equ 文件 行的自 封装分配。

后 Cvpcb

Cvpcb 能从原理 制程序 Eeschema 中后

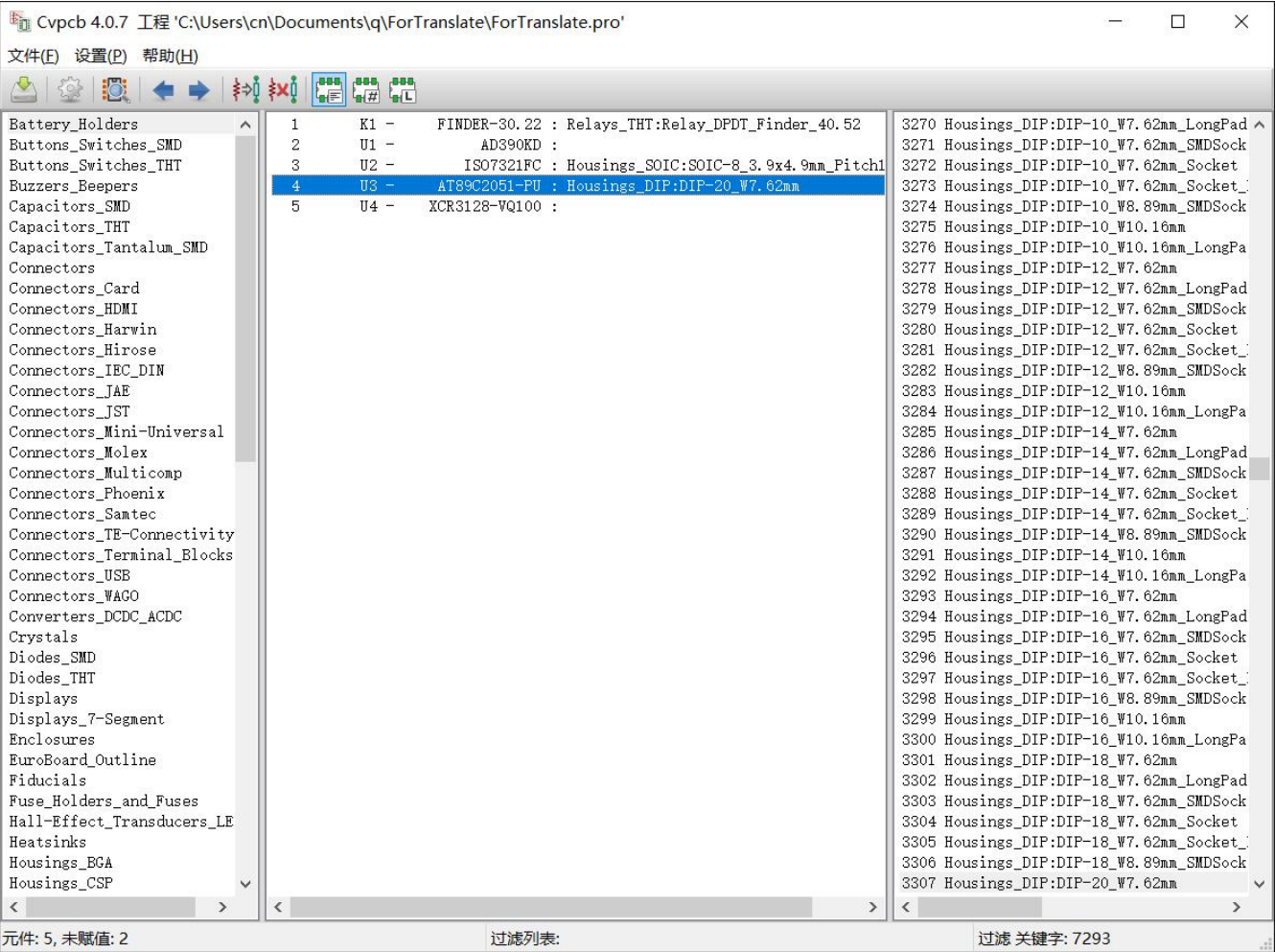


Eeschmea 会自动的将一些信息 (例如当前原理 中元器件列表和可用的封装), CvPcb。在 用 CvPcb 之前,用 唯一需要做的就是 原理 中的各个元件 号。

CvPcb 命令

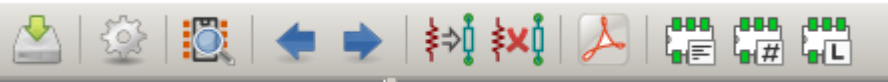
主界面

下面的 片展示了 CvPcb 的主界面。



在主界面中, 左窗格包括了所有与 目 的可用的封装 文件名列表; 中 窗格包括了从网 文件中 入的所有元器件列表; 右窗格包括了所有从与 目 的封装 中加 的可用的封装列表。 如果没有加 网 文件, 那么元器件列表将是空的; 似的, 如果没有找到可用的封装 , 那么位于右 的封装列表也将是空的。

主界面工具



部的工具 提供了下列命令的快速 接口 :

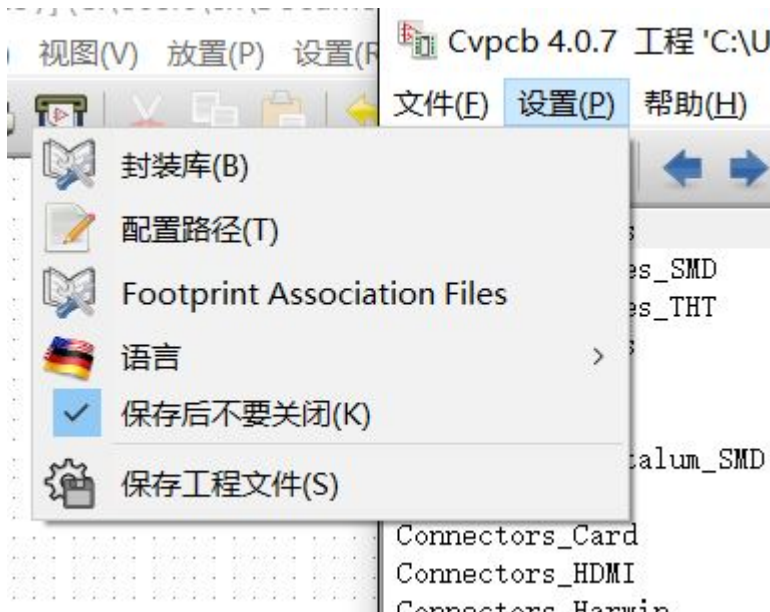
	将当前封装 移到 Eeschema 是封装字段的内容）。
	用 CvPcb 配置菜
	示在封装中 的元件的封装 窗口。
	在没有的情况下自 列表中的上一个元件 封装
	自 列表中的下一个元件而不占用封装
	使用等价文件自 将封装与 元件相
	除所有封装分配。
	使用默认 打开 定的封装文档 pdf 文件 pdf 看器。
	启用或禁用 以限制封装列表 所 元件的封装 器。
	启用或禁用 以限制使用的封装列表 所 元件的引脚数。
	启用或禁用 以使用。限制封装列表 定的

主界面 快捷方式

下面的表格列出了 CvPcb 主窗口中的 快捷方式：

右箭 / 卡	激活当前激活窗格右 的下一个窗格。如果最后一个窗格当前已激活， 行到第一个窗格。
左箭	激活当前激活的左 的下一个窗格 窗格。如果第一个窗格当前已激活， 行到最后一个窗格。
向上箭	当前所 列表的上一个 目。
向下箭	当前所 列表的下一个 目。
Page Up	当前所 目的整 名
Page Down	当前所 目的整 内容 名
Home	当前所 列表的第一
End	当前所 列表的最后一

CvPcb 配置



CvPcb 可以在保存封装 系后自 或手

点 菜 中的“置”—“封装 ”将会打开封装 配置 框。

根据 CvPcb 版本得不同， 存在两种封装 管理方式:

- 方式是使用 .mod 文件 行管理, 用 可以看到封装 文件列表。
- 新的管理方式使用“Pretty” 格式。 种管理方式将会使用一个文件 列表, 每个文件 (文件 名 *.pretty) 就是一个新的管理方式允 用 使用来自 gEDA/gPCB 中的 以及 Eagle 件中 xml 格式的

封装 管理

重要提示:

本 内容只与2013 年12 月之后 行的KiCad 相

封装 表

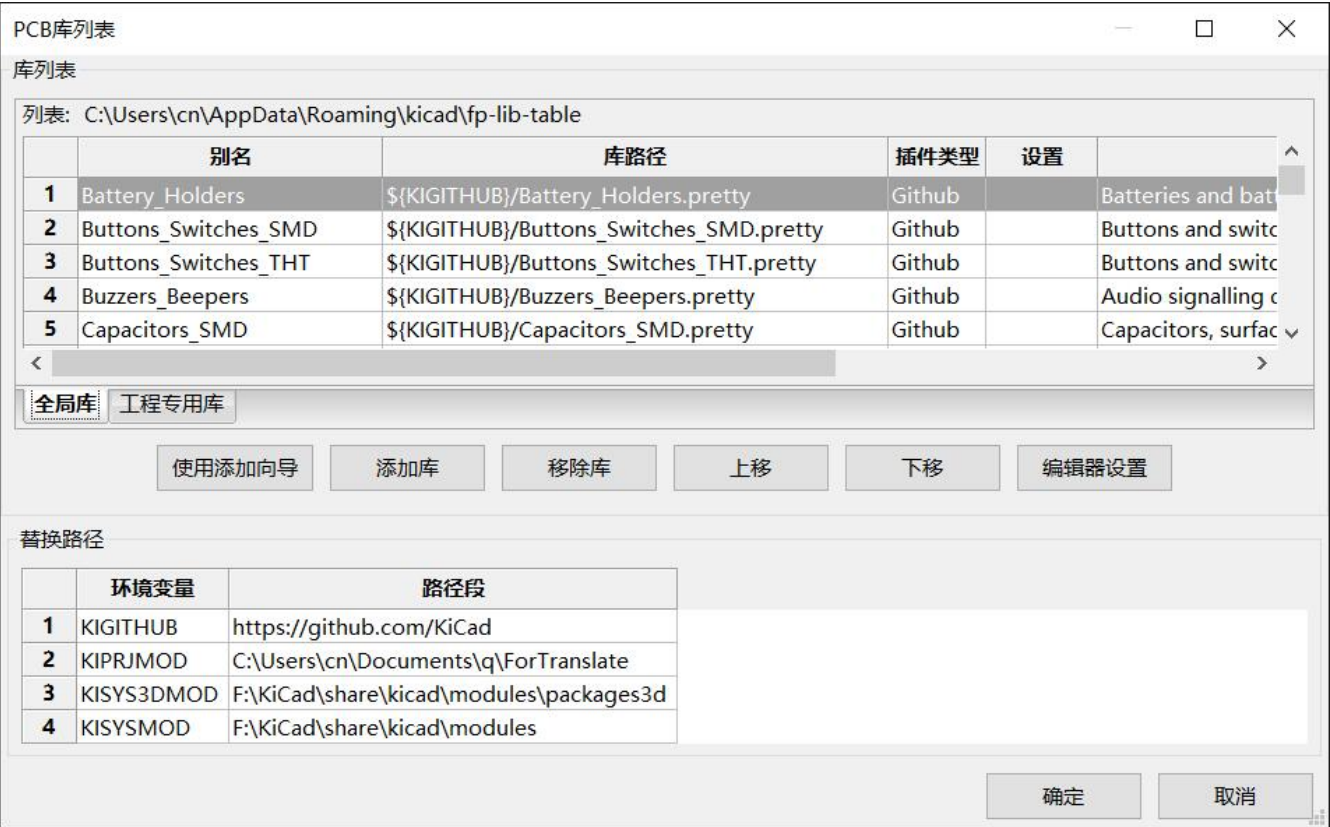
从 2013 年 12 月以后， Pcbnew 和 CvPcb 使用了新的基于 **封装 列表**的 管理工具，新的管理工具允 用 通 以下方 式 **直接使用封装**

- KiCad 封装 文件 (.mod 文件)
- KiCad 新式 .pretty 封装 (在用 的本地磁 中, 有 .pretty 展名并包含了 .kicad_mod 文件的文件)
- KiCad 新式 .pretty 封装 (托管在官方或第三方 的 Github 中)
- GEDA 封装 (包含了 .fp 文件的文件)
- Eagle 封装

NOTE

- 用 能改写位于本地磁 上的 Kicad .pretty 封装 (以及 些文件 中包括的 .kicad_mod 文件)。
- 其余的格式都是只 的。

下面的 片展示了封装 列表 框， 框可以通 菜 中的“置”-“封装 ”打开。



封装 列表的作用是 Kicad 支持的封装 分配一个 名。可利用 名而不是之前基于封装 路径 找 序的方法, 行封装 的 找.

功能能 使 Cvp pcb 控制不同封装 的加 从而使得 那些位于不同封装 但是 有相同名称的封装可以被正确 此外, 功能 能使 Kicad 来自其他 PCB 器, 例如 Eagle 和 GEDA 的封装

全局封装 列表

全局封装 列表包括了那些在任何 目中都可 的封装 表格的配置存 在用 主目 下的 fp-lib-table 文件中。用 主 目的具体位置由用 使用的操作系 决定。

目封装 列表

目封装 列表包括了 在当前打开 目内能 的封装 目封装 列表 能在 目的网 列表文件被加 才能 如果当前 没有打开任何 目, 或是在打开的 目目 中没有封装 列表文件, 那么系 将 建一个新的可供 的表格文件。

初始 置

首次运行 Pcbnew 或 Cvp pcb 如果在用 主目 下无法找到全局封装 列表文件 **fp-lib-table**, 那么 Pcbnew 或 CvPcb 将 将存 在 KiCad 模板文件 中的默认封装 列表文件 制到用 主目 中。

如果默认的 fp-lib-table 文件无法被找到, 那么将会在用 主目 下 建一个新的封装 列表文件。 种情况下, 用 可以从 制 fp-lib-table 文件, 或是手 行封装 配置。

默认的封装 列表将作 kicad 的一部分而被安装, 其中包括了 多 准的封装。

然, 用 首先需要根据 需求, 修改 列表(添加/移除 目)。

(加 多的封装 会耗 多)

添加列表 目

如果要使用一个封装 它必 先被添加到全局封装 列表或工程封装 列表中。 当用 当前 有一个网 列表文件 工程封装 列表才是可用的。

封装 列表中的 目 名不可重

“ 名” 字段可以由用 自行决定，不必和封装 的路径/封装 文件名等相 名中不可以出 冒号:。列表中的每一 需要有一个可用的路径。根据封装 型的不同，路径的具体表 形式可能不同。“路径” 字段中的内容可以是 路径，相路径，或者是 境 量(下文会 一步讨)

了正确 取封装 列表中每一 的“插件 型” 字段必 被正确 目前 KiCad 支持的 型包括 KiCad legacy, KiCad Pretty, Eagle, 和 GEDA 封装

列表中的“描述” 字段，用于 添加 外的 注信息。列表中的“ 置” 字段在当前版本尚未使用，修改 字段没有任何效果。

- 注意，用 无法在同一个封装列表中 两 分配相同的 名。但是可以在全局封装列表和工程封装列表中使用相同的 名。
- 如果重名 生，工程封装列表中的名称将 先被使用。定义在工程封装列表中的 目，将会被写入当前网 列表所在 目 下的 fp-lib-table 文件中。

境 量替

境 量替 是封装 列表的 大功能之一。它将允 用 使用 境 量定义自定义的封装路径。使用 境 量替 ,需要在封装 列表的“路径” 字段中，遵守以下 法: `+${ENV_VAR_NAME}`

运行 KiCad 默认定义 两个 境 量:

- **KIPRJMOD** 境 量。 量指向当前 目的目 不可被改
- **KISYSMOD** 境 量。 量指向默认随 KiCad 安装的默认封装 目

用 可以通 在菜 中的“ 置” -“配置目 ” 中重 KISYSMOD 的 因此用 可以使用自定义的封装 替 Kicad 的默认封装

如果当前 目的网 列表文件已被加 , CvPcb 会将 KIPRJMOD 的 置 网 列表文件的目 (即 目目)。

在加 一个 路板文件 Pcbnew 也会 置 境 量。

境 量允 用 在不知道 目 目的情况下，将封装 存 于 目目 下。

使用 GitHub 插件

GitHub 插件提供了只 那些包含 Kicad pretty 封装 文件的 GitHub 的接口。 插件也提供了 COW(“ 制 写入”) 功能。 功能是可 它将允 用 从 GitHub 取的封装 并且将它 保存在本地。因而，“GitHub” 插件用于 只 托管于 <https://github.com/> 的 程 pretty 封装 如果要向封装 列表中添加 GitHub 其“路径” 字段需要被 置 一个合法的 GitHub 地址。

例如:

https://github.com/liftoff-sr/pretty_footprints

或

<https://github.com/KiCad>

典型的 GitHub URL 以下形式:

https://github.com/user_name/repo_name

“插件 型”字段必 被 置 “GitHub”。如果要使用 COW 功能,必 向 置字段内添加 允 写 个目 的 . 的 用于 置 GitHub 上封装 的修改副本的存 路径.存 在 目 中的封装将和 GitHub 上的只 部分共同构成封装 .如果没有声明 ,那么 GitHub 封装 就完全是只 的.如果声明了 ,那么 “混合”封装 的修改,将存 到本地的 .pretty 文件中.注意 “混合”封装 中位于 GitHub 中的那一部分始 是只 的, 意味着你无法直接 除或修改特定 GitHub 中的内容. 一步的讨 认 混合 仍然属于“GitHub” 型,只是它包括了本地的 /写部分及 程的只 部分.

下面的表格展示了没有 允 写 个目 置的封装 列表 :

昵称	路径	插件 型		描述。
github	https://github.com/liftoff-sr/pretty_footprints	Github		Liftoff's GH footprints

下面的表格展示了开启 COW 功能的封装 列表 注意 境 量 \${HOME} 做 例用. github.pretty 目 位于 \${HOME}/pretty/. 如果用 使用了 允 写 个目 提前手 建上述目 并以 .pretty 尾。

昵称	路径	插件 型		描述。
github	https://github.com/liftoff-sr/pretty_footprints	Github	allow_pretty_writing_to_this_dir=\${HOME}/pretty/github.pretty	Liftoff's GH footprints

于 置了 允 写 个目 的列表 将首先加 位于本地的封装 一旦用 使用封装 器 封装 行修改并保存在了 COW 本地文件 中, 那么 GitHub 中,和用 修改并保存 的的封装同名的任何封装的更新将不会被看

始 每一个 GitHub 封装 使用不同的本地 .pretty 目 不要 通 多次引用同一目 的方式来 合两个不同的 GitHub 封装

也不要不同的封装 列表 中使用相同的 COW (*.pretty) 目 将 来不少

置字段中, 允 写 个目 的 置 将会通 使用 \${\} 表示的 境 量来 建路径, 就和在 路径字段中使用 境 量一

那 COW 的目的究竟是什么? 其 它是 了更好的促 封装 的分享。

如果用 周期性的将通 COW 修改的 pretty 元件 反 到 GitHub 的 者 ,那么用可以帮助更新 GitHub 中的封装 用 可以 将 COW 文件 中的 *.kicad_mod 文件 送 的 者.当用 得知他 的修改被同步到了 GitHub 他 就可以 除本地的 COW 文件, 然后使用 GitHub 插件提供的只 功能.用 尽量向位于 <https://github.com> 的主 多多提交, 让自己的 COW 文件尽可能的小。

使用模式

封装 既可以在全局定义, 也可以在当前工程的范 内定义. 全局定义的封装 将被存 在用 主目 下的 fp-lib-table 文件中, 它 将可以在所有的 目中使用。

全局封装 始终可以 即使当前没有加 目。

工程封装 能在当前打开的网 列表文件中使用。

工程封装 列表存 在 目文件 内的 fp-lib-table 文件中。你可以自由 在哪个列表中定义封装

两种方法各有 劣。你可以在全局封装 列表中定义所有你可能会用到的封装， 可以 随用随取。 做的缺点是你必 在非常多的封装中 找你所需要的封装。你也可以根据 目的需求定义你的封装

做的好 在于你 需要定义你在某个 目中需要用到的封装， 将大大减小搜索的 度。

做的缺点是，每新建一个 目，你都得重新手 定义每一个 目中需要用到的封装。你也可以同 在全局范 和工程范 内定义你的封装

一种使用模式是，将所有常用的封装 定义在全局封装 列表中，将一些只在特定 目中使用的封装 定义在工程封装 列表中。 而言之，用 完全可以自主决定封装 的管理方式。

使用封装 列表添加向

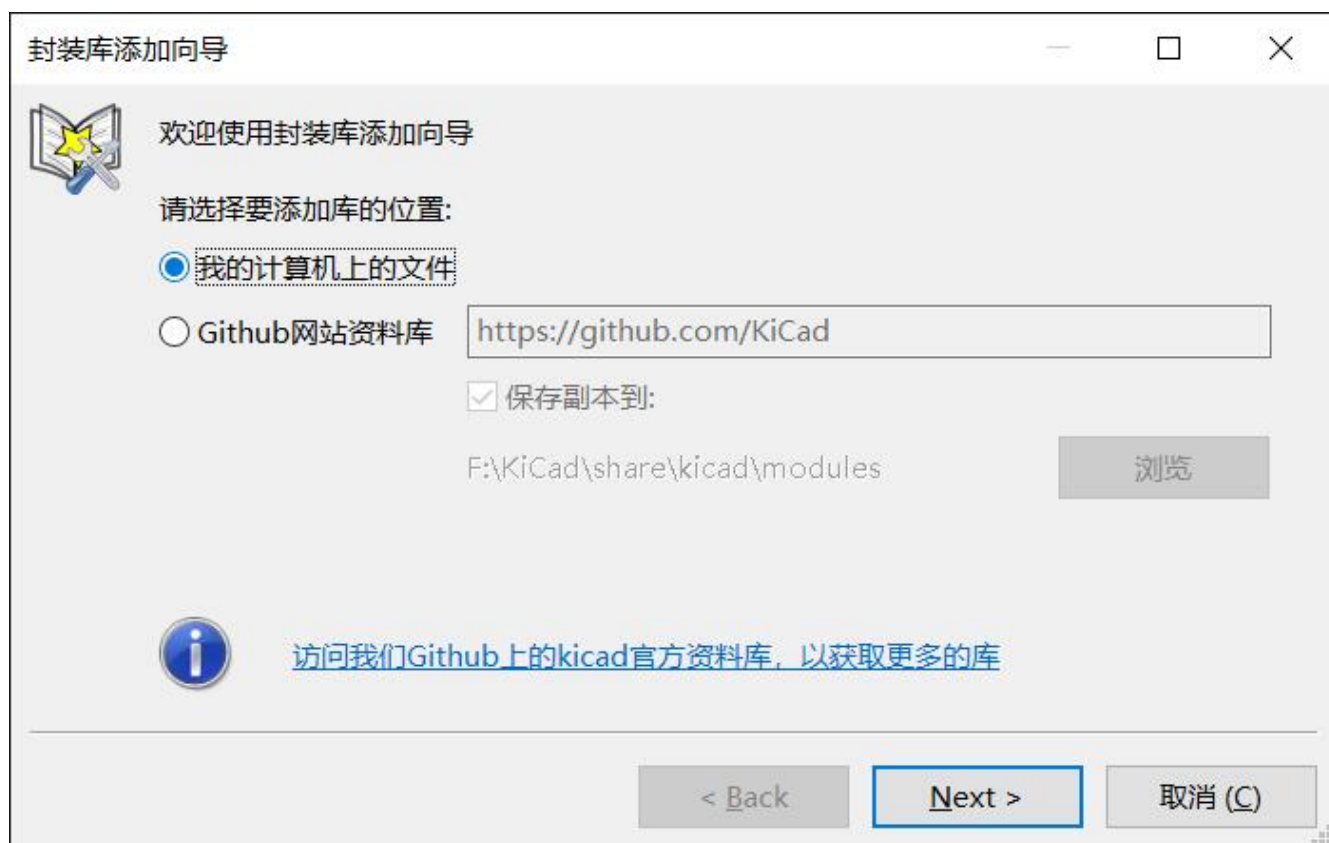
添加向 用于向封装 列表中添加封装 可以在 封装 列表 框中打开添加向

待添加的封装 可以是任何 Kicad 支持的 型。

它可以是本地的封装 或是 GitHub 上的封装

当 位于 GitHub 中 它 可以被添加 程 也可以 将它 下 到本地作 本地 添加。

此 本地封装 默认被 中。



封装库添加向导

欢迎使用封装库添加向导


请选择要添加库的位置:

☒ 我的计算机上的文件

☐ Github网站资料库

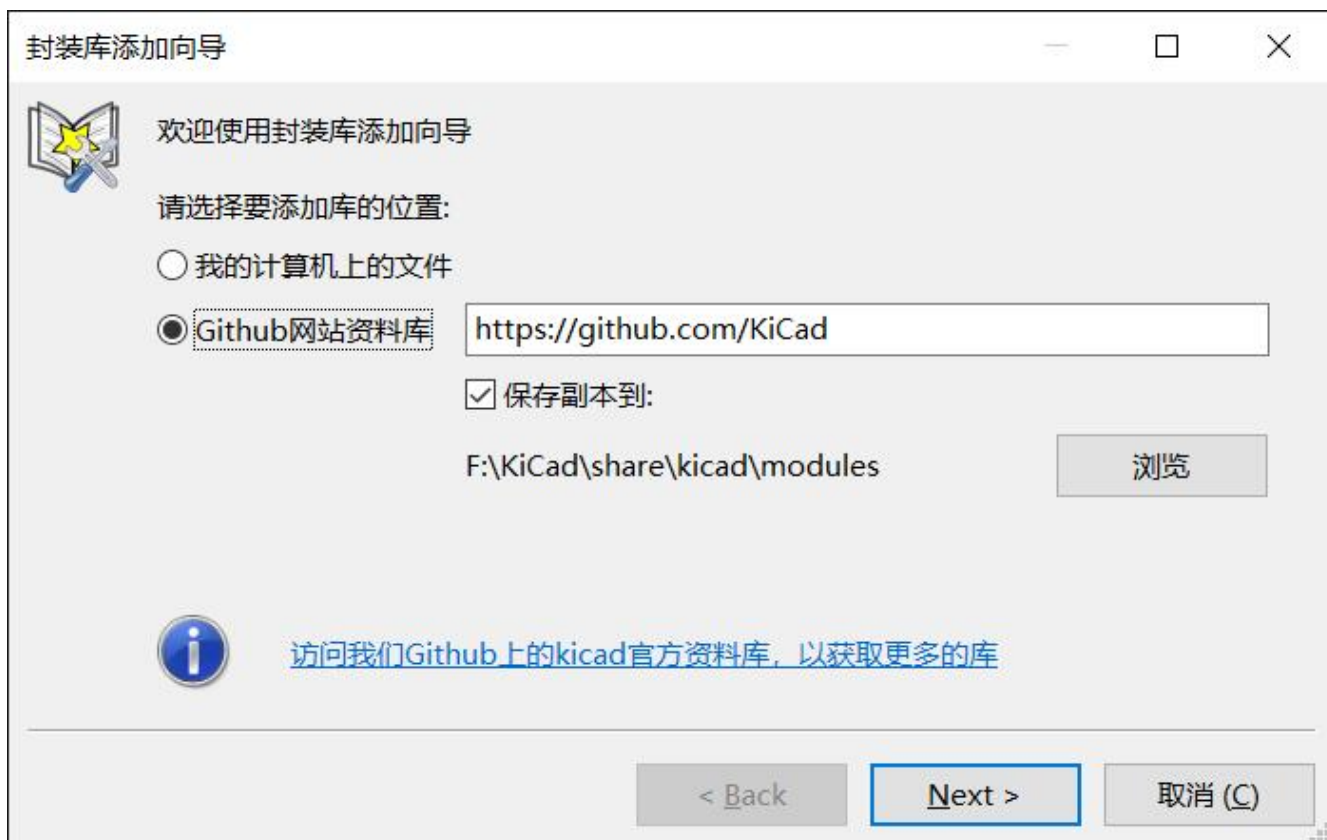
☒ 保存副本到:

F:\KiCad\share\kicad\modules

 [访问我们Github上的kicad官方资料库，以获取更多的库](#)

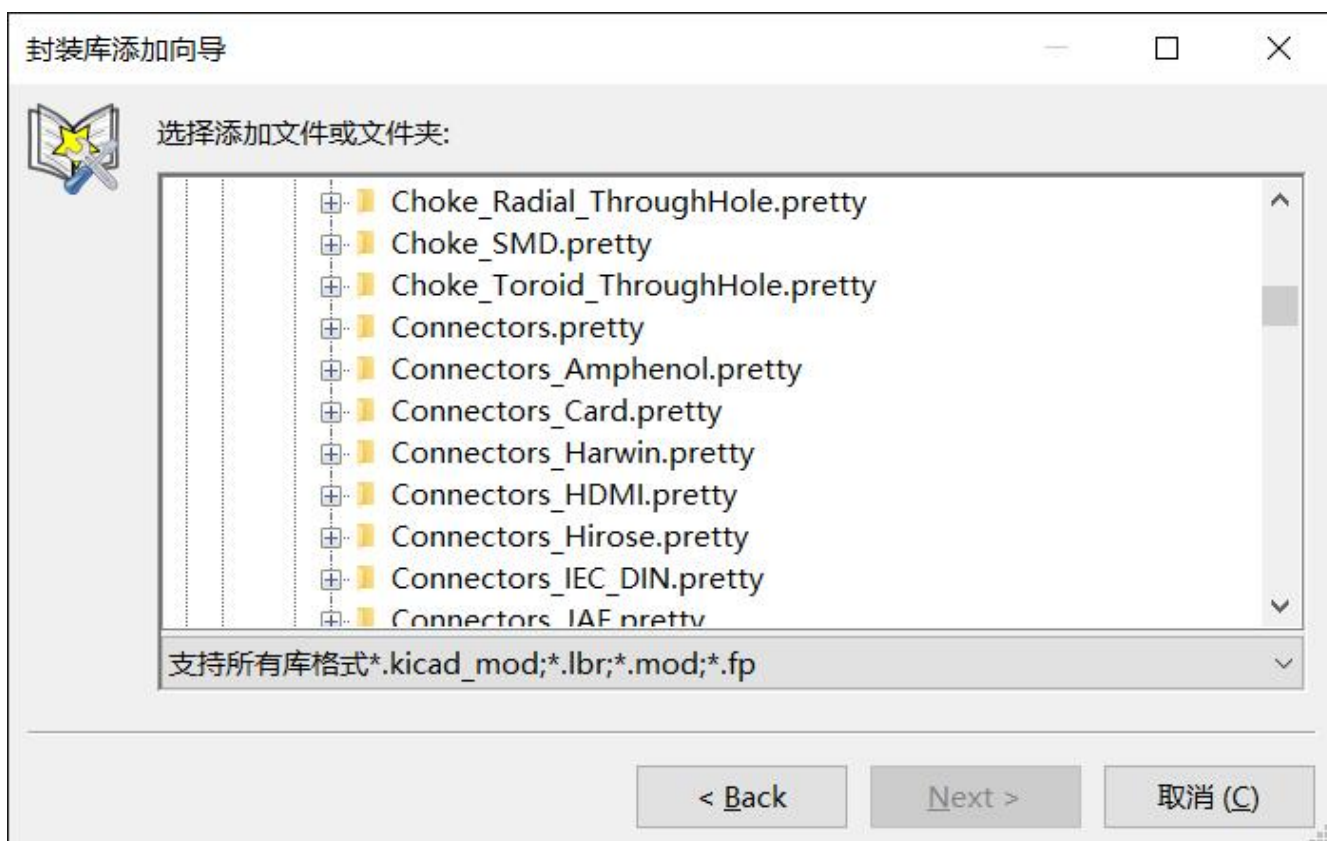
< Back Next > 取消 (C)

此 程封装 被 中。

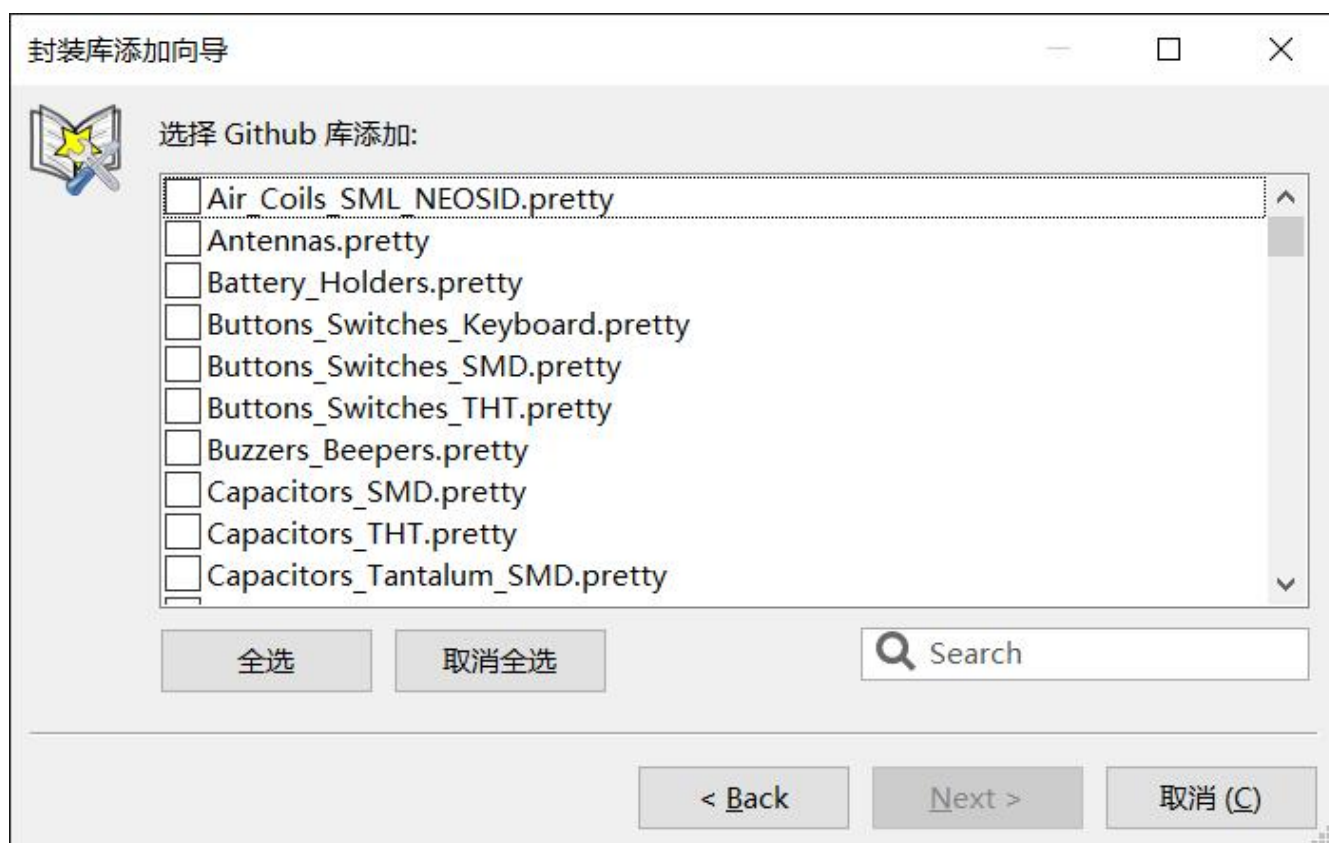


根据 的不同，将会 示 些 面中的一个,用于让用 要添加的封装 列表:

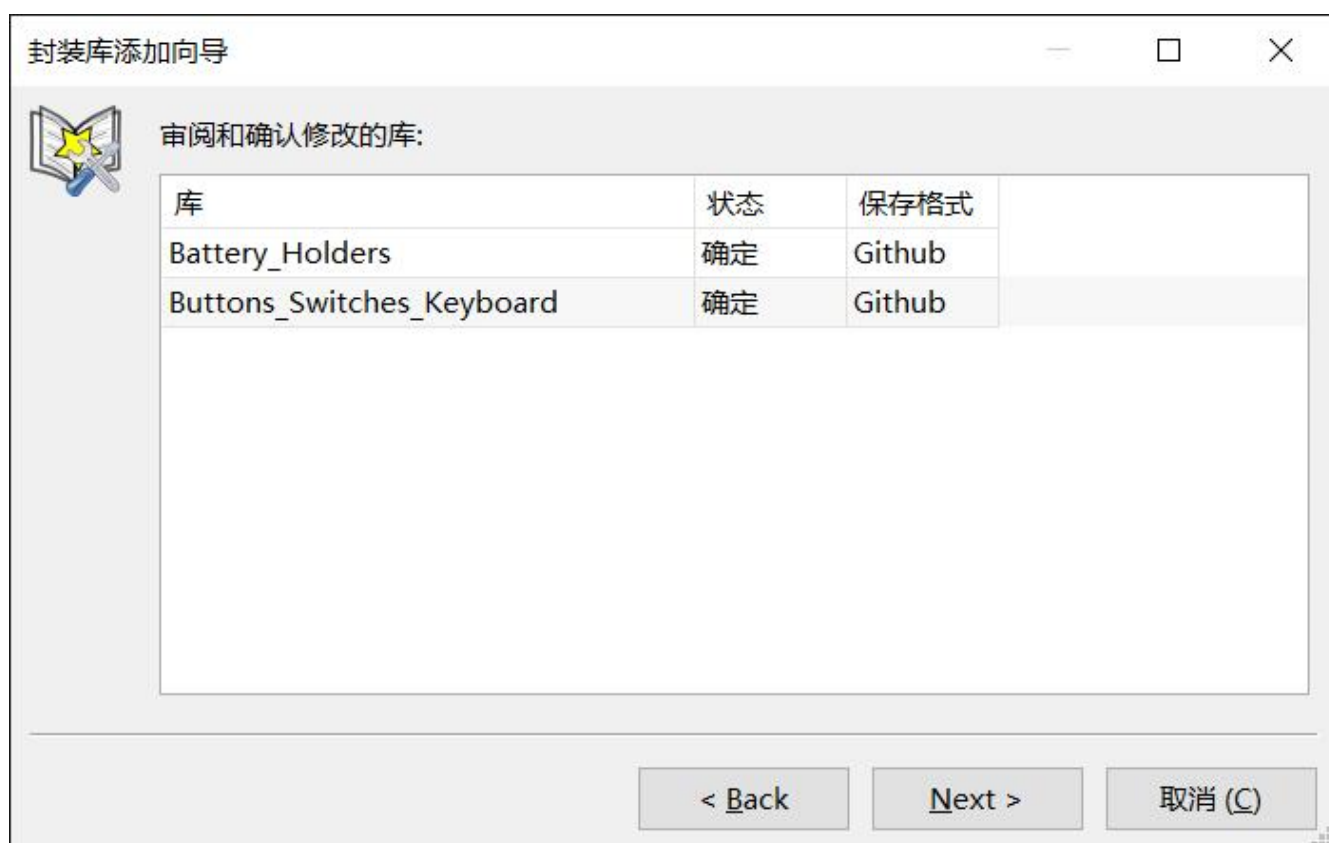
此 本地 被 中.



此，程 被 中。



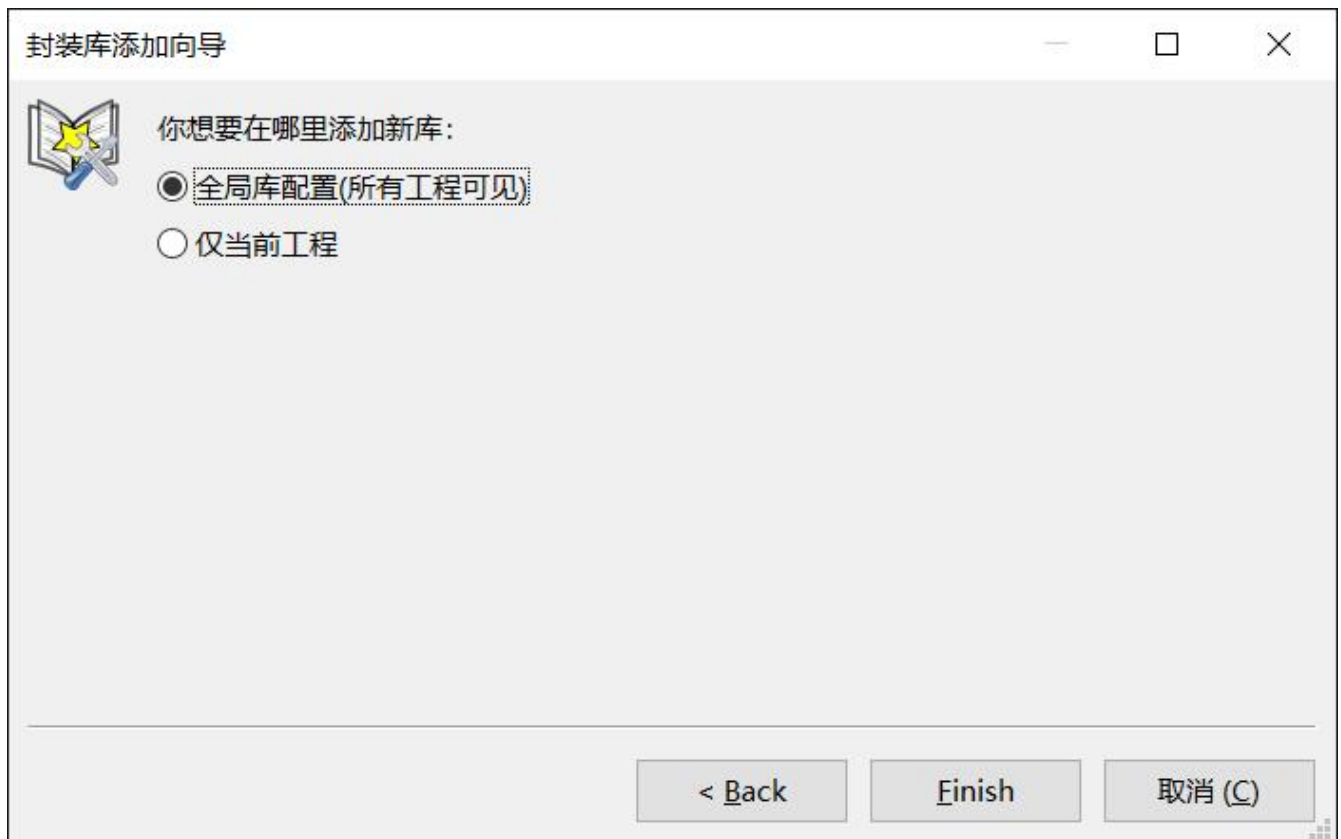
当一系列封装 被 中后,下一个 面将 :



如果一些 中的 不正确(不支持,不是封装 ,...),它 将被 “不可用”。

最后一个 面是 需要 出的封装 列表 :

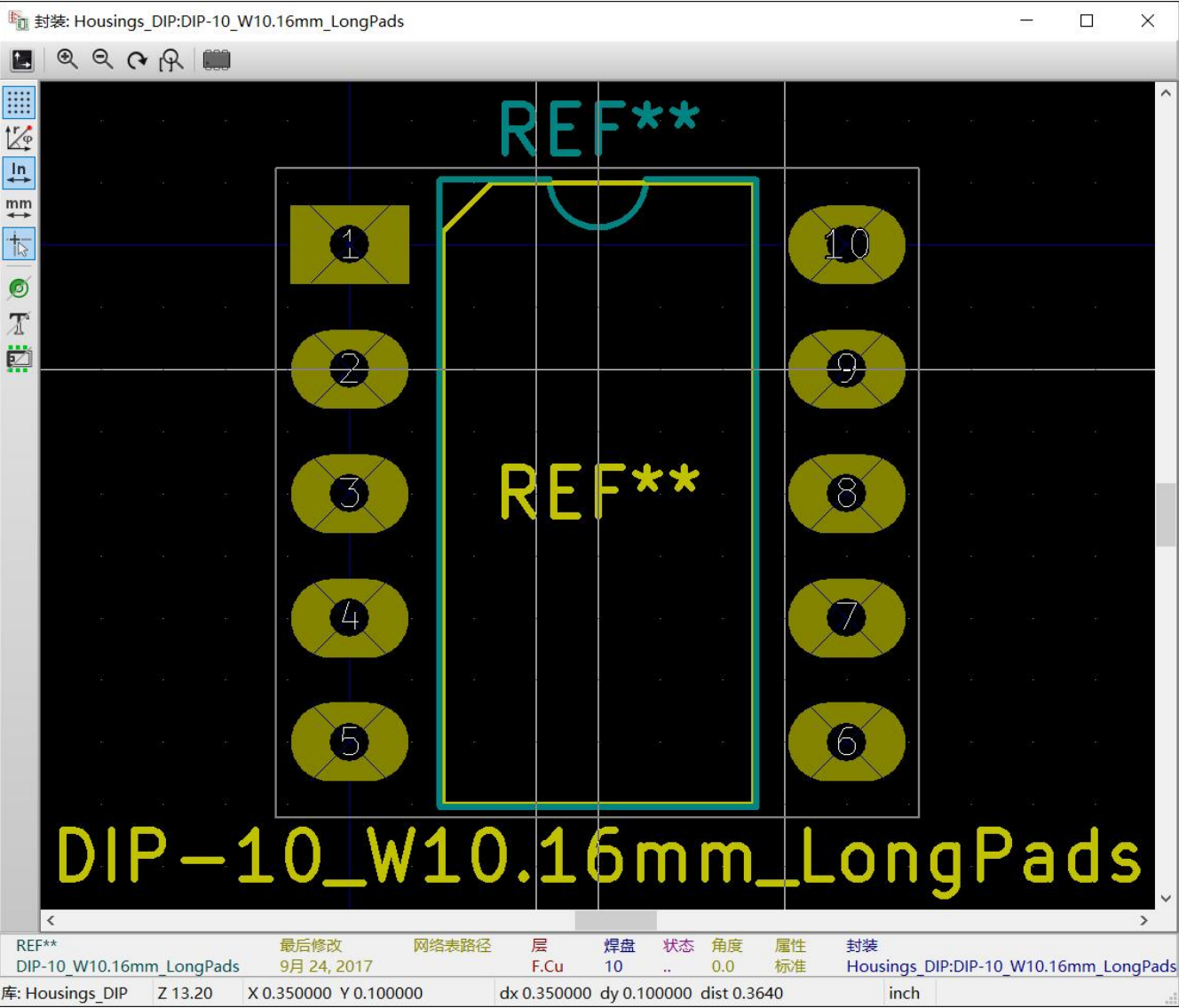
- 全局 配置



看当前封装

封装命令

封装命令会将当前 中的封装 示在 封装 窗口中.如果封装已 被分配了一个 3D 模型,那么三 模型也会在 窗口中示。下面的 片展示了封装 看器窗口。



状 信息

状 位于 CvPcb 主窗口的底部, 它 用 提供了 多有用的信息。下面的表格列出了状 中不同窗格的定义。

左	元件 计: 数量, 未分配封装的数量
中	中元件的 列表
右	器状 以及可用封装的数量

快捷

F1	放大
F2	小
F3	刷新 示
F4	将光 移 到窗体中心
Home	是封装大小自适 于当前窗体
空格	置相 于当前光 的相 坐
右箭	将光 向右移 一个网格 位
左箭	将光 向左移 一个网格 位
上箭	将光 向上移 一个网格 位
下箭	将光 向下移 一个网格 位

鼠 操作

	在当前游 位置放大或 小
Ctrl +	水平方向平移
Shift +	垂直方向平移
鼠 右	打开右 菜

右 菜

通 点 鼠 右 打开右 菜



放	当前 放的倍数
网格	网格大小

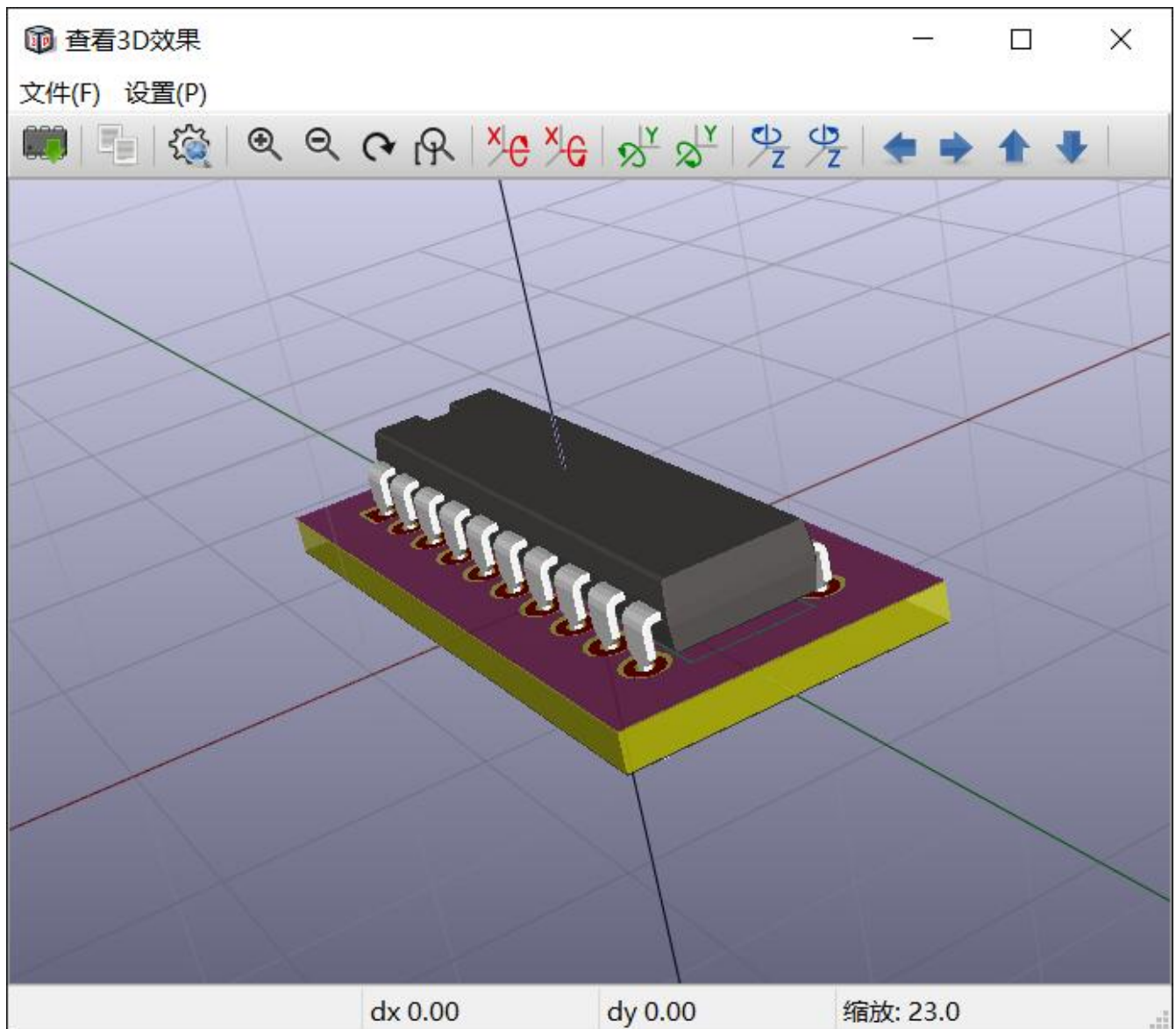
水平菜

	示 示 框
	放大
	小
	重
	在 示区域适合
	打开 3D 模型 看器

垂直工具

	示或 藏网格
	以极坐 或矩形表示法 示坐
	以英寸 示坐
	示以毫米 位的坐
	切 指 式
	在草 或正常模式下在 板之 切
	在草 或普通模式下 制文本之 切
	在草 或正常模式下 制 之 切

看当前 3D 模型



鼠 操作

鼠	在光 放大或 小
Ctrl+ 鼠	水平方向平移
Shift + 鼠	垂直方向平移

水平菜

	重新加 3D 模型
	将 3D 像 制到剪 板
	置 3D 看器
	放大
	小
	重
	在 示区域适合
	沿 X 向后旋
	沿 X 向前旋
	沿 Y 向后旋
	沿 Y 向前旋
	沿 Z 向后旋
	沿 Z 向前旋
	左
	右
	上
	下
	打开和 正交投影模式


使用 CvPcb 向元件分配封装

手 分配封装

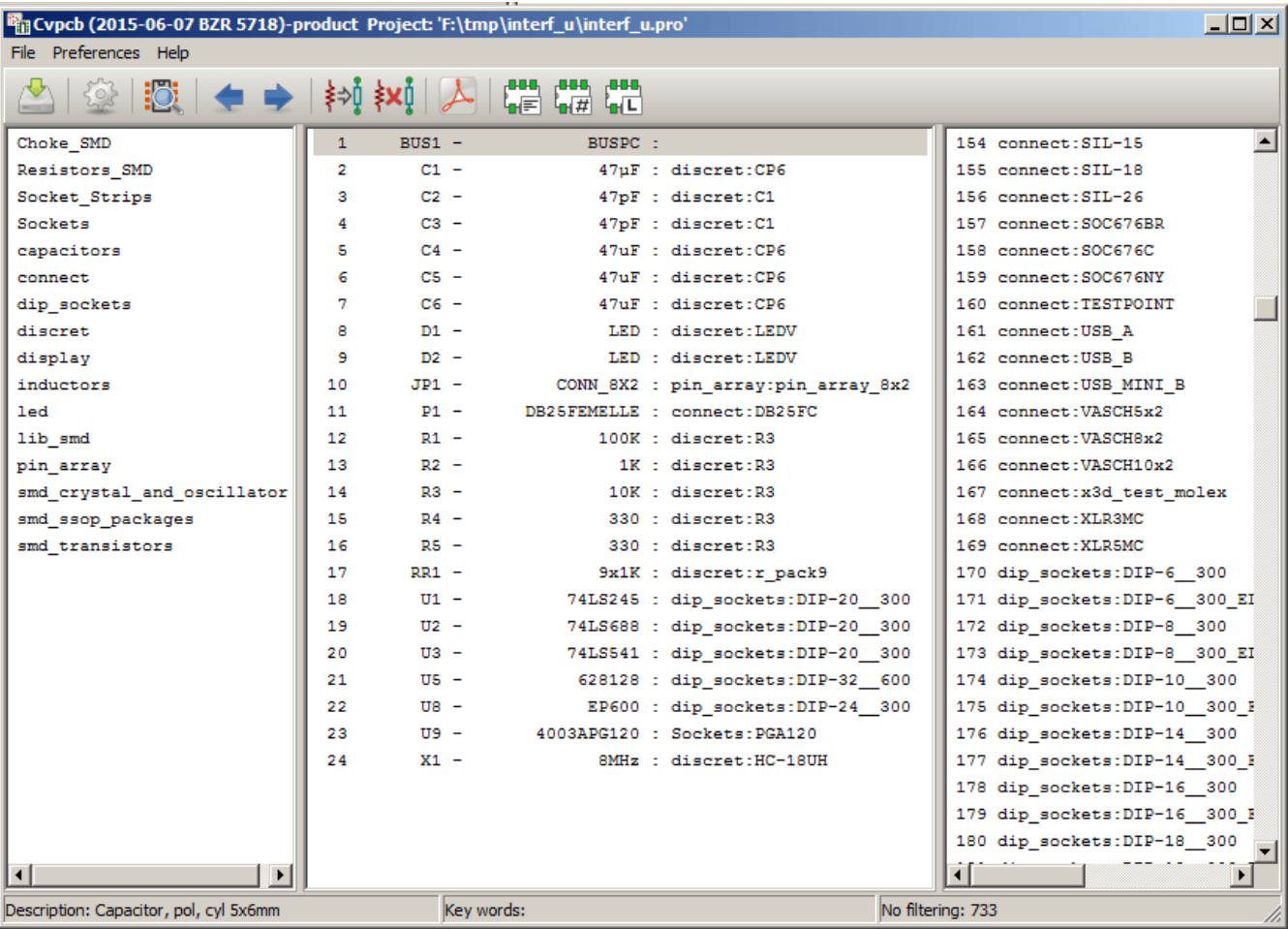
如果要手 分配封装,首先在需要位于窗口中部的元件窗格中 一个元器件.然后在右 的封装窗格中,鼠 左 双 想要分配的封装.然后 封装就会被分配 的元器件.此外,分配完成后,下一个未分配封装的元件将会被自 中.改 元件的封装 ,操作 似.

封装列表

如果高亮 中某元器件/封装 ,有一个或多个 已 打开,那么,右 的封装窗格将自 示 后的封装列表.

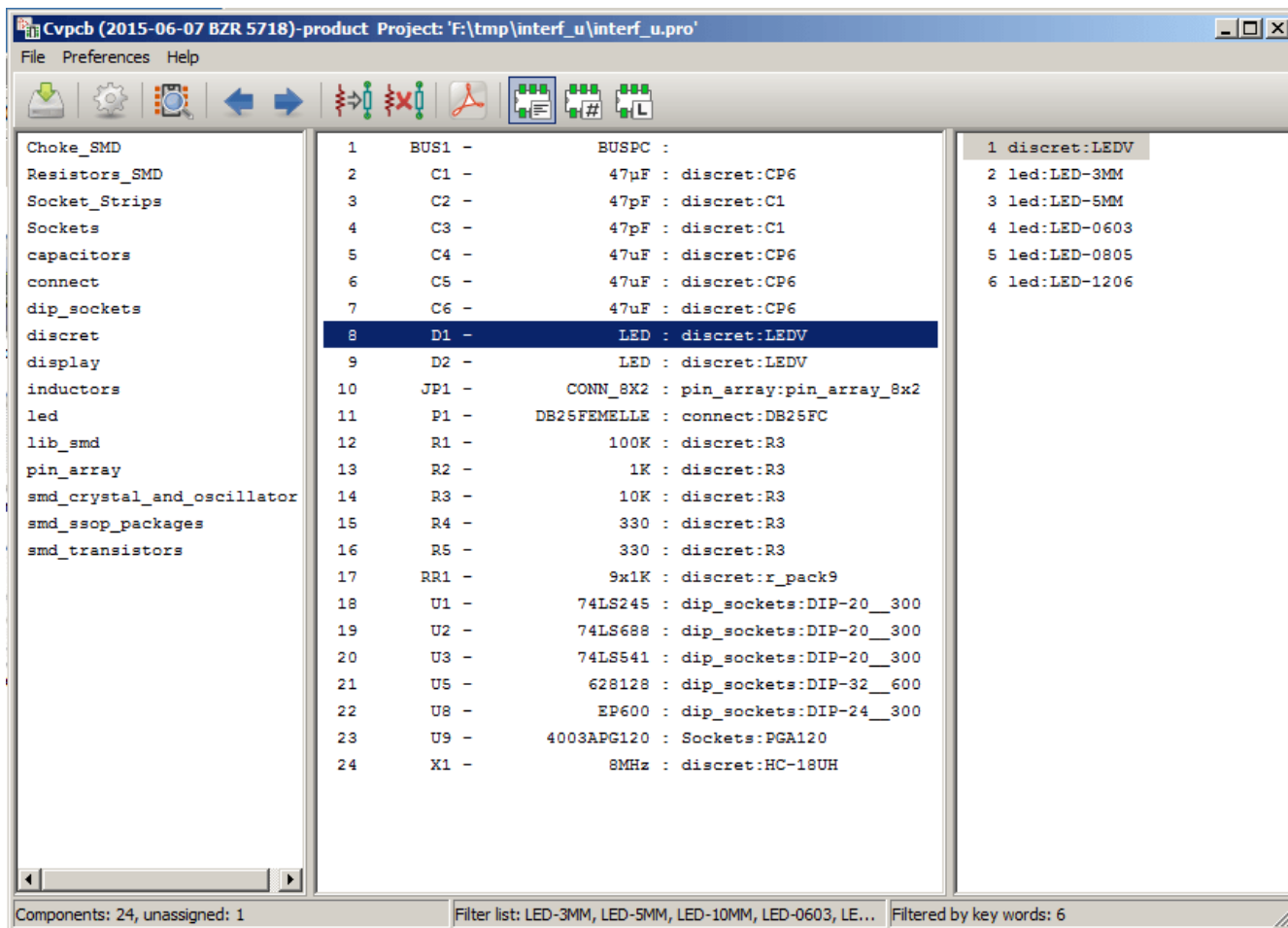
 点 些按 将打开或 器.当所有的 器都 于 状 ,将会在右 示所有可用的封装.

器:

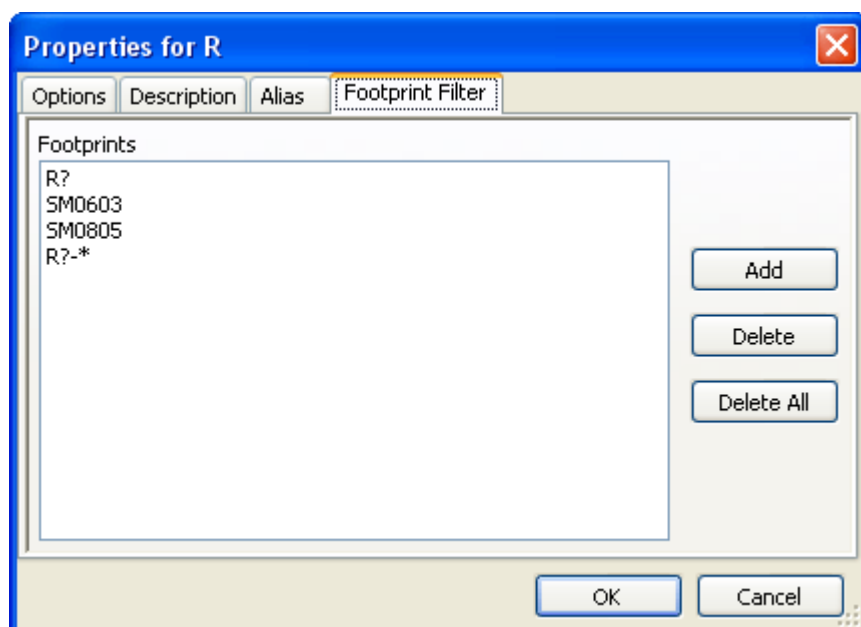


根据 用于 定元器件的 器 封装列表 行 . 定元器件中启用的 器 示于主窗口底部的状 中部.

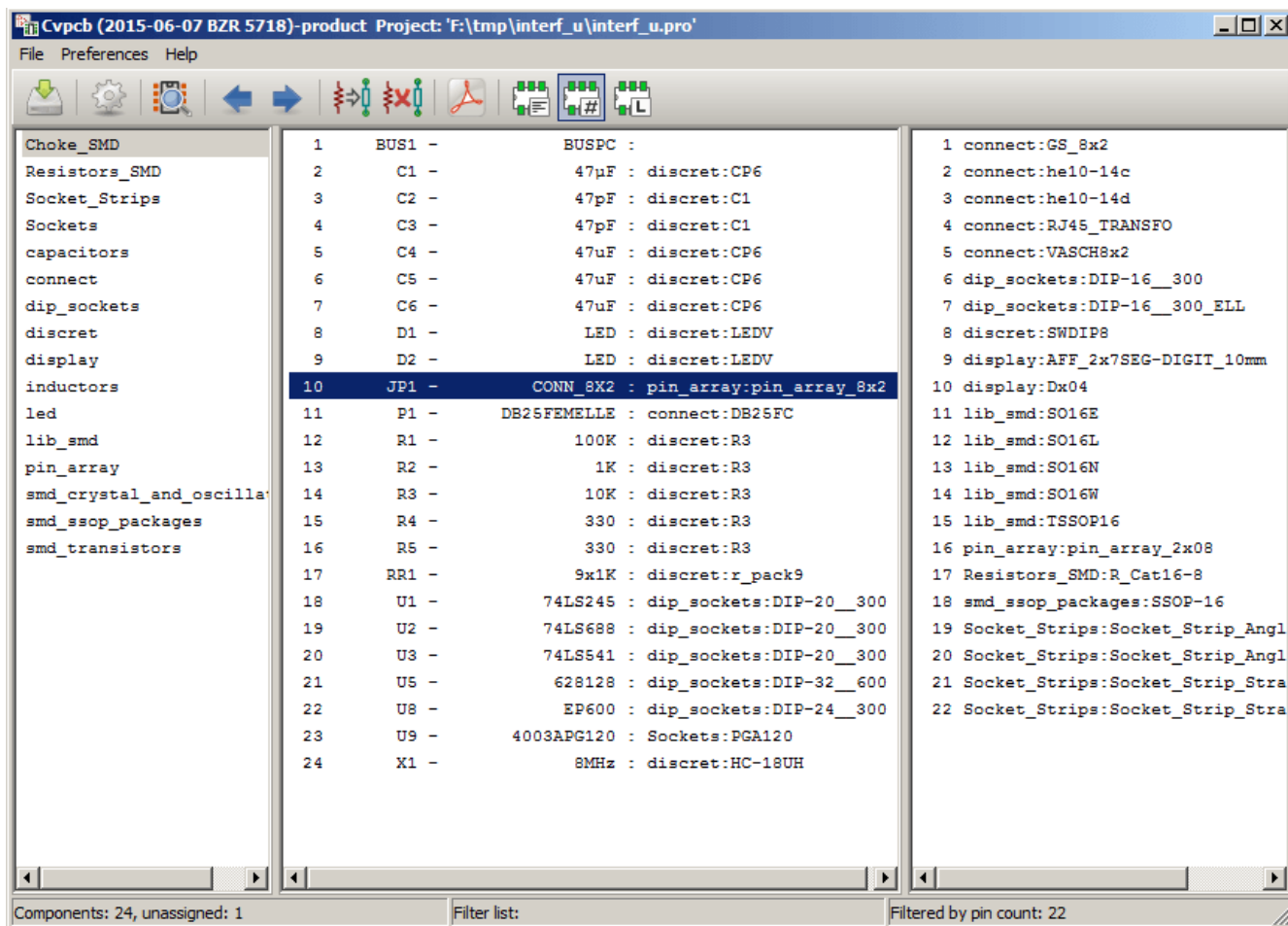
根据 用于 定元器件的 器 封装列表 行



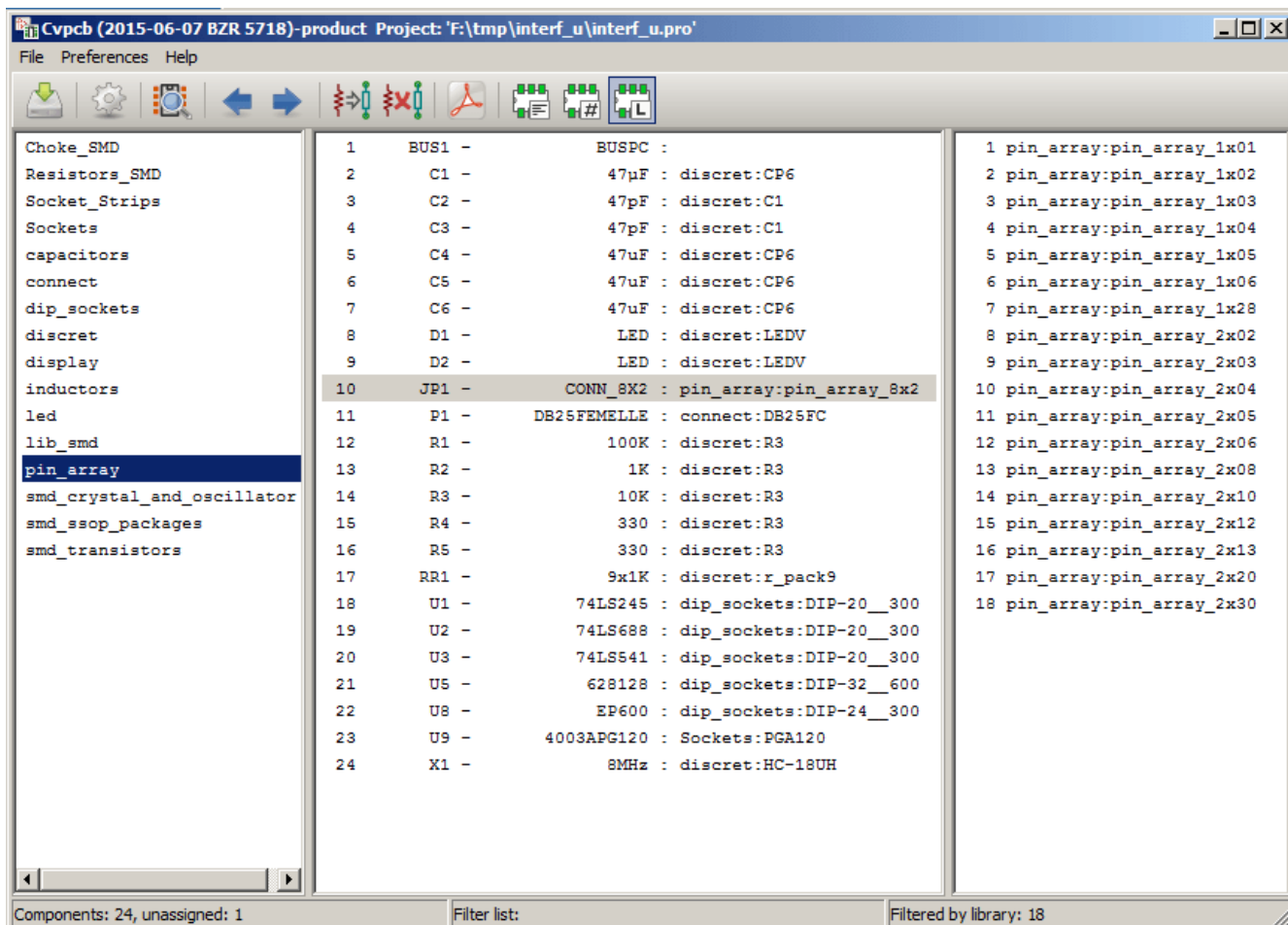
在 Eeschema 的元件 器中,用 可以在元件属性 框的“封装” 卡中 置元件的可用封装列表元件属性 框如下所示。



根据 中的元件的引脚数目 行 :

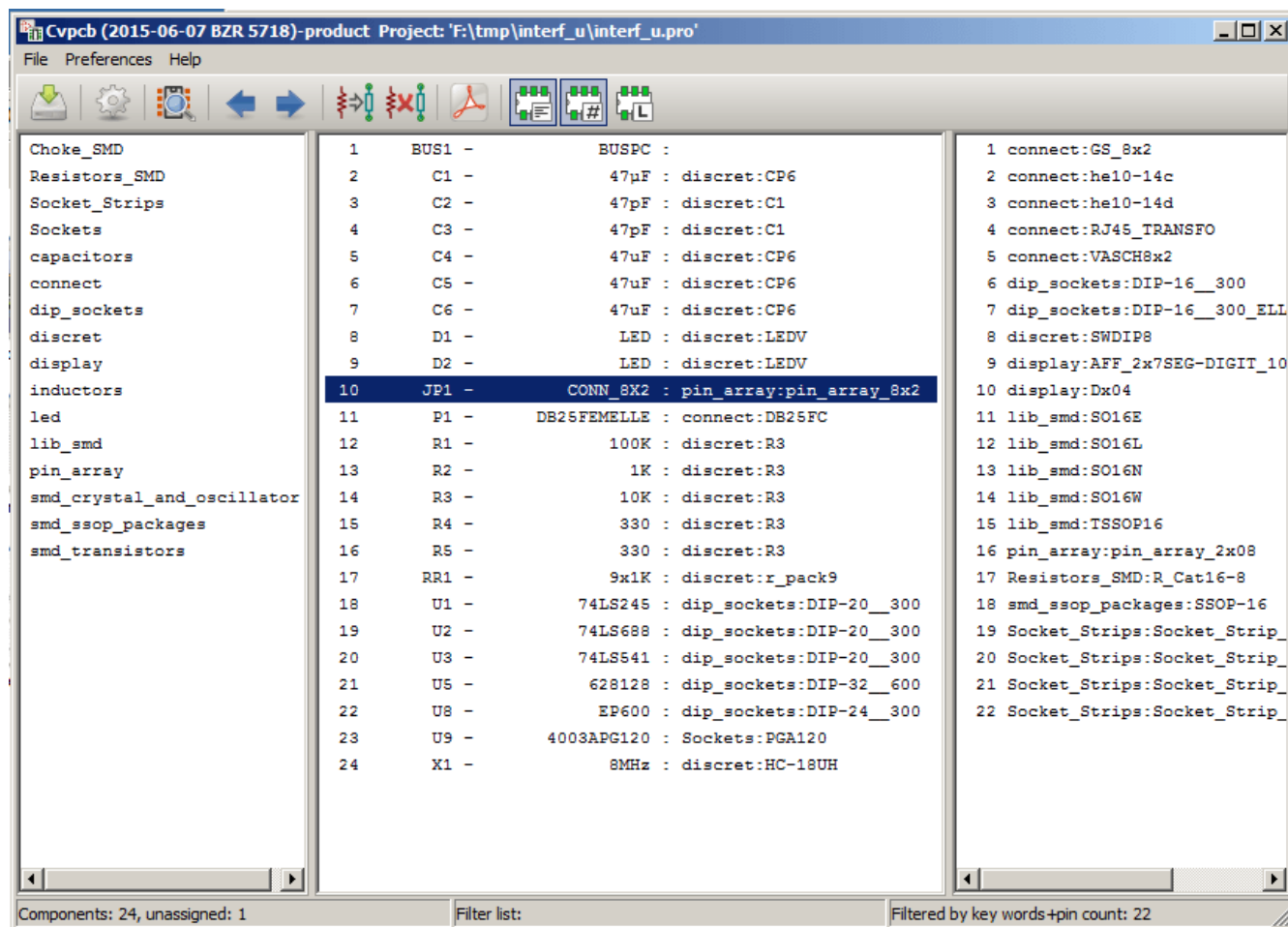


根据 的封装 行



不同的 器可以叠加作用, 的需求. 可以帮助减少右 窗格中的封装数目, 方便 .

根据 定元器件的引脚和元件 器 行 :



自

Equivalence 文件

Equivalence 文件可以帮助用 自 元件分配封装.

它会根据元件的名称属性(*value field*)列出与之 的封装. Equivalence 文件的文件 展名 **.equ**。

Equivalence 文件格式

equ 文件中每一行 一个元件. 每行的格式如下:

“元器件 ” “封装名”

每个名称都 被 引号括起, 不同的封装名 由一个或者多个空格隔开。

例子:

如果 U3 是 14011, 它的封装是 14DIP300,那么其 行 写 :

“14011” “14DIP300”

开 的行 注 行.

Equivalence 文件示例:

```
#Integrierte Schaltkreise (SMD):
```

```
'74LV14' 'S014E'  
'74HCT541M' 'S020L'  
'EL7242C' 'S08E'  
'DS1302N' 'S08E'  
'XRC3064' 'VQFP44'  
'LM324N' 'S014E'  
'LT3430' 'SSOP17'  
'LM358' 'S08E'  
'LTC1878' 'MSOP8'  
'24LC512I/SM' 'S08E'  
'LM2903M' 'S08E'  
'LT1129_S08' 'S08E'  
'LT1129CS8-3.3' 'S08E'  
'LT1129CS8' 'S08E'  
'LM358M' 'S08E'  
'TL7702BID' 'S08E'  
'TL7702BCD' 'S08E'  
'U2270B' 'S016E'
```

```
#Xilinx
```

```
'XC3S400PQ208' 'PQFP208'  
'XCR3128-VQ100' 'VQFP100'  
'XCF08P' 'BGA48'
```

```
#upro
```

```
'MCF5213-LQFP100' 'VQFP100'
```

```
#Spannungsregler
```

```
'LP2985LV' 'SOT23-5'
```

自 元件分配封装

点 位于 部工具 的自 分配按 以解析 Equivalence 文件.

所有在 .equ 文件中能找到相 的元件,将会被自 分配封装。