

Eeschema 〇介

The KiCad Team

# Table of Contents

Eeschema 简介	2
描述	2
技术概述	2
通用 Eeschema 命令	3
鼠标命令	3
坐标	4
格点	7
放置	7
示光坐标	7
菜单	7
上方工具	8
右侧工具	10
左侧工具	11
输出和快速	12
主菜单	15
文件菜单	15
首选项菜单	15
帮助菜单	22
通用部件工具	23
表格管理	23
搜索工具	23
网表工具	24
批注工具	25
电气工具	26
物料清单工具	29
字段工具	31
用于封装分配的输入工具	32
管理符号	34
符号表	34
原理图构建和	37
简介	37
一般考虑	37
开始	37
符号放置和	37
源端	40
充电	46
保存的符号	48
分原理	50
简介	50
在层次结构中导航	50
本地、分部和全局	51
层次构建摘要	51

工作表符号 .....	51
□接 - 分□引脚 .....	52
□接 - 分□□□ .....	53
复□□次□构 .....	55
平面□次□构 .....	55
符号批注工具 .....	58
□介 .....	58
一些例子 .....	59
使用□气□□□□□行□计□□ .....	62
□介 .....	62
如何使用 ERC .....	62
ERC 的示例 .....	63
□示□断 .....	63
□源引脚和□源□志 .....	65
配置 .....	65
ERC □告文件 .....	66
□建网□列表 .....	67
概述 .....	67
网表格式 .....	67
网表示例 .....	68
关于网表的□明 .....	70
其他格式 .....	72
□□和打印 .....	74
□介 .....	74
常□的打印命令 .....	74
在 Postscript 中□制 .....	74
以 PDF 格式□制 .....	75
在 SVG 中□□ .....	76
在 DXF 中□□ .....	76
在 HPGL 中□□ .....	76
在□上打印 .....	77
符号□□□器 .....	79
关于符号□的一般信息 .....	79
符号□概述 .....	79
符号□□□器概述 .....	79
□□□与□□ .....	82
□建□符号 .....	84
□形元素 .....	87
每个符号多个□位和替代体型式 .....	89
引脚□建和□□ .....	91
符号字段 .....	94
□源符号 .....	96
LibEdit - 符号 .....	98
概述 .....	98

定位符号点 .....	98
符号名 .....	99
符号字段 .....	99
符号文档 .....	100
符号 .....	103
符号器 .....	104
介绍 .....	104
主屏幕 .....	104
符号器部工具 .....	105
建立自定义网表和 BOM 文件 .....	106
中网表文件格式 .....	106
新的网表格式 .....	108
XSLT 方法 .....	108
命令行格式：python 脚本的示例 .....	116
中网表结构 .....	117
有关 xsltproc 的更多信息 .....	122
仿真器 .....	126
分配模型 .....	126
Spice 指令 .....	131
仿真 .....	131

## 参考手册

### 版权

本文档由其贡献者拥有版权 ©2010-2018，如下所列。您可以根据 GNU 通用公共许可 (http://www.gnu.org/licenses/gpl.html)，版本 3 或更高版本，或知识共享署名许可 (http://creativecommons.org/licenses/by/3.0/)，版本 3.0 或更高版本的条款分派和/或修改它。

本指南中的所有商标均属于其合法所有者。

### 贡献者

Jean-Pierre Charras, Fabrizio Tappero.

### 翻译人

taotieren <[admin@taotieren.com](mailto:admin@taotieren.com)>, 2019, 2020, 2021.

Telegram 简体中文交流群: [https://t.me/KiCad\\_zh\\_CN](https://t.me/KiCad_zh_CN)

### 反馈

请将任何报告、建议或新版本引导到此：

- 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 翻译: <https://gitlab.com/kicad/code/kicad-i18n/issues>

### 出版日期和软件版本

2015 年 5 月 30 日出版。

# Eeschema 简介

## 描述

Eeschema 是一个原理图设计软件，作为 KiCad 的一部分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 Eeschema 文件都可以从一个操作系统100%兼容到另一个操作系统。

Eeschema 是一个集成的应用程序，其中原理图控制，布局，元件管理和 PCB 设计的所有功能都在 Eeschema 本身内进行。

Eeschema 打算与 PcbNew 合作，后者是 KiCad 的印刷电路板设计软件。它可以输出网表文件，其中列出了其他元件包的所有电气连接。

Eeschema 包含一个符号编辑器，可以创建和管理符号。它集成了替代原理图捕捉元件所需的以下附加但必不可少的功能：

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

## 技术概述

Eeschema 受可用内存的限制。因此，元件、元件引脚，连接或板的数量没有限制。在多个表格的情况下，表示是分段的。

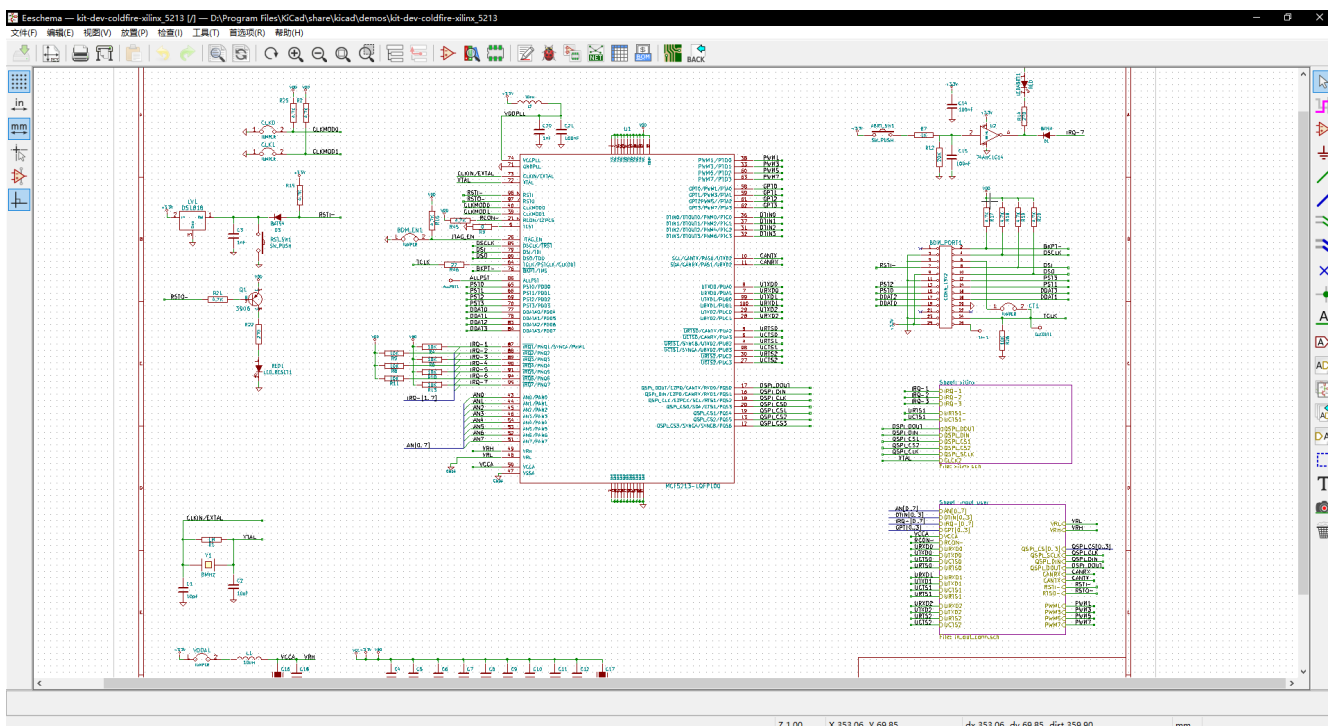
Eeschema 可以通过以下几种方式使用多表格表：

- 单一的层次结构（每个原理图只使用一次）。
- 复用的层次结构（一些原理图在多个图中不止一次使用）。
- 扁平层次结构（原理图未在主图中明确连接）。

# 通用 Eeschema 命令

命令可以通过以下方式执行：

- 菜单（屏幕顶部）。
- 屏幕顶部的快捷命令（常用命令）。
- 点击屏幕右侧的快捷命令（特定命令或“工具”）。
- 屏幕左侧的快捷命令（指示器）。
- 按下鼠标（重要的补充命令）。特别是右击打开光栅下元素的上下文菜单（放置、网格和元素等）。
- 功能键 F1, F2, F3, F4, Insert 和 Space 具体来 Esc 取消正在执行的命令。Insert 允许复制最后创建的元素。
- 按通常执行工具命令并在当前光标位置开始工具操作。有关列表，参见“帮助 → 列出”菜单或按“Ctrl+F1”。



## 鼠标命令

### 基本命令

左键

- 在状态栏中指示光栅下的符号或文本的特征。
- 双击元素如果元素可编辑符号或文本。

右键

- 打开弹出菜单

## 阻止操作

您可以在所有 Eeschema 菜中移拖复制和除所区域。

通使用鼠左在目周制一个框来区域。

在期按住 “Shift”， “Ctrl” 或 “Shift + Ctrl” 分行复制， 拖和除：

鼠左	移
Shift + 鼠左	复制
Ctrl + 鼠左	拖
Ctrl + Shift + 鼠左	除

拖或复制您可以：

- 再次以放置元素。
- 右或按 Esc 取消。

如果已启移命令， 可以使用右出菜另一个命令。



- “Ctrl+F1” 示当前列表。
- 可以在“原理器” 框的“控件” 卡中重新定义菜“首” → “常”）

是默认列表：

帮助（此窗口）	Ctrl+F1
放大	F1
小	F2



□放重□	F3
□放中心	F4
适合屏幕	Home
□放到□□	@
重置本地坐□	Space
□□□目	E
□除□目	Del
旋□□目	R
拖□□目	G
撤消	Ctrl+Z
重做	Ctrl+Y
鼠□左□□□	Return
鼠□左□双□	End
保存原理□	Ctrl+S
加□原理□	Ctrl+O
□找□目	Ctrl+F
□找下一个□目	F5
□找下一个DRC□□	Shift+F5
□找和替□	Ctrl+Alt+F
重复最后一□	Ins
移□□ → 拖□□	Tab
复制□	Ctrl+C
粘□□	Ctrl+V
切□	Ctrl+X
移□原理□□目	M
重复的符号或□□	C
添加符号	A
添加□源	P

像 X	X
像 Y	Y
定向普通符号	N
符号	V
符号参考	U
符号封装	F
使用符号器	Ctrl+E
开始画	W
开始	B
端路	K
添加	L
添加分	H
添加全局	Ctrl+L
添加接点	J
添加无接志	Q
添加表	S
添加入口	Z
添加入口	/
添加形折	I
添加形文字	T
从原理更新到PCB	F8
自置域	O
留下表	Alt+BkSp
除点	BkSp
突出示接	Ctrl+X

可以使用器重新定义所有菜首 → 常 → 《首-控件，控件》）。

可以使用菜（首 → 入和出 → 入/出）入/出置。

## 格点

在 Eeschema 中，光标始终在网格上移动。网格可以自定义：

- 可以使用“出菜”或使用“首/”菜更改大小。
- 可以在“原理器”对话框的“色”卡中更改色（菜首 → 常）
- 可以使用左工具按钮切换可性。

默认网格尺寸为 50mil (0.050") 或 1.27mm。

是在符号器中计符号将符号和放置在原理中以及放置引脚的首网格。

人可以使用 25mil 到 10mil 的小网格。用于计符号体或放置文本和注不建用于放置引脚和

## 放

要更改放：

- 右以打开出菜然后所需的放。
- 或使用功能：
  - F1：放大
  - F2 小
  - F4 或只需鼠中不移鼠光指位置居中
- 窗口放：
  - 鼠放大/小
  - Shift + 鼠向上/向下平移
  - Ctrl + 鼠向左/向右平移

## 示光坐

示位英寸或毫米。但是，Eeschema 是使用 0.001 英寸 (mil/thou) 作其内部元。

窗口右下角示以下信息：

- 放系数
- 光的位位置
- 光的相位置

按空格可以将相坐重置零。于量两点之的距离或象很有用。

	Z 5.50	X 348.00 Y 60.96	dx 348.00 dy 60.96 dist 353.30	mm	
--	--------	------------------	--------------------------------	----	--

## 菜

部菜允打开和保存原理程序配置和看文档。

文件(F)	编辑(E)	视图(V)	放置(P)	检查(I)	工具(T)	首选项(R)	帮助(H)
-------	-------	-------	-------	-------	-------	--------	-------



	□用CvPcb□符号分配封装。
	□出网表（Pcbnew, SPICE和其他格式）。
	□□符号字段。
	生成物料清单（BOM）。
	□用 Pcbnew □行 PCB 布局。
	返回□入封装分配（使用 CvPcb 或 Pcbnew □□□到“封装”字段中。

## 右工具

此工具包含以下工具：

- 放置符号，交叉点，文本等。
- 建分子表和接符号。

	取消激活的命令或工具。
	通过不同颜色、粗细和网宽来突出显示网络。如果 KiCad 以工程模式运行，那么用于所连网络的线将在 Pcbnew 中也会高亮。
	显示符号器件框以放置新符号。
	显示源符号器件框以放置源符号。
	画一根线。
	画一根总线。
	控制入口点。某些元素是形化的，不会建立连接，因此它们不用于将元件接在一起。
	控制到网络的入口点。
	放置“无连接”标志。某些标志放在符号引脚上意味着没有连接。它是为了通知用户器件特定引脚缺少连接是故意的，并不警告。
	放置一个交叉点。连接两根交叉线或一根线和一个引脚，当它可能是模糊的（即，如果端或引脚不是直接的连接到另一个端）。
	放置一个本地-本地连接，位于同一层中的物品。对于两个不同工作表之间的连接，您必须使用全局或分表。
	放置一个全局连接，即使具有相同名称的所有全局元件也已连接，位于不同的层上。
	放置分表-分表用于建立子表与包含它的父表之间的连接。
	放置分表。您必须指定此子表的文件名。
	从子表输入分引脚。命令只能在行-行分表。它将建立于分表的分引脚，放置在目标子表中的层。
	在子表中放置分引脚。命令只能在行-行分表。它会建立任意分引脚，即使它也是如此，目标子表中不存在。
	划一条线，只是形化的，不能连接任何东西。
	放置文本批注。
	放置位图像。
	删除所有元素。

## 左工具栏

此工具栏管理显示









	切换网格可见性。
	将单位切换为英寸。
	将单位切换为毫米。
	光标形状（全屏/小）。
	切换“隐藏”引脚的可见性。
	切换 自由角度/90 度 对齐和放置。

## 出菜和快速

右击可打开所选元素的上下文菜单，包含：

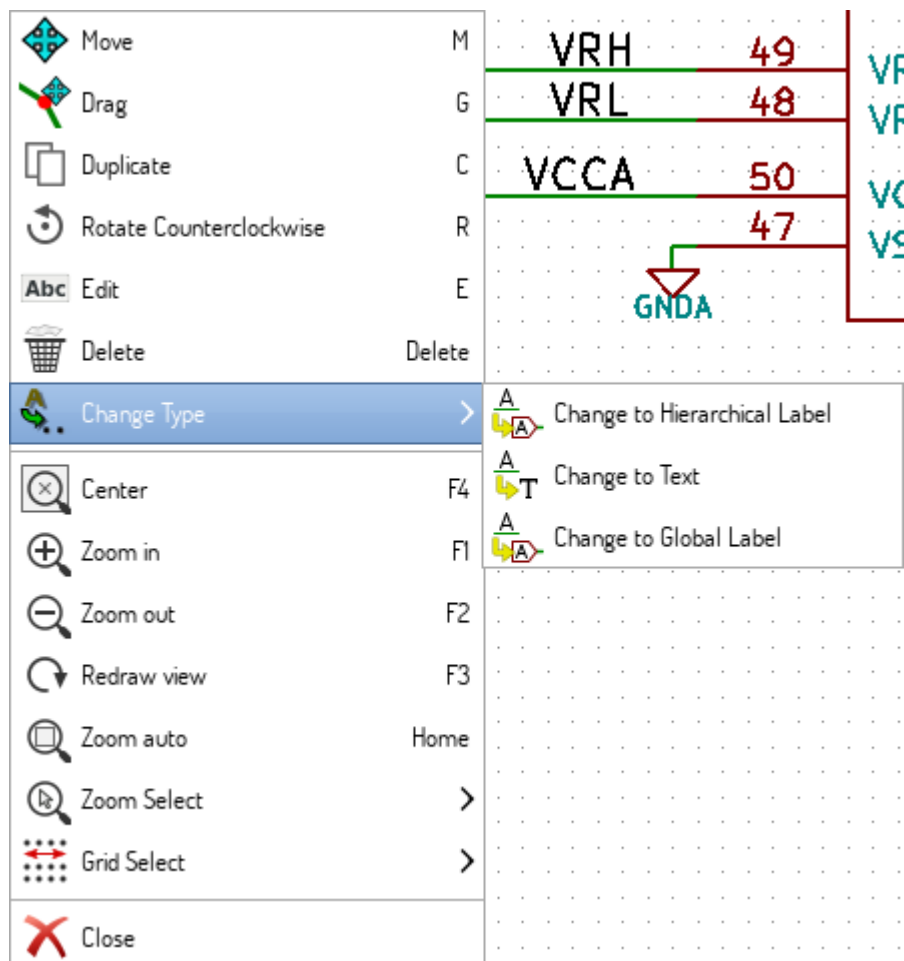
- 缩放系数。
- 网格调整。
- 通常所选元素的参数。

没有固定元素的弹出窗口。

	Center	F4
	Zoom in	F1
	Zoom out	F2
	Redraw view	F3
	Zoom auto	Home
	Zoom Select	>
	Grid Select	>
	Close	

□□□□□





□□符号。



# 主菜

## 文件菜

文件(F) 编辑(E) 视图(V) 放置(P) 检查(I) 工具(T) 首选项(R) 帮助(H)

新建	关当前原理并启一个新原理在独立模式下。
打开	加原理目在独立模式下。
打开最近	从最近打开的文件列表中打开原理目在独立模式下。
附加示意	将另一个工作表的内容插入当前工作表。
入非 Kicad 原理文件	入以其他文件格式保存的原理目。
保存	保存当前工作表及其所有子工作表。
保存当前工作表	保存当前工作表，而不保存目中的其他工作表。
将当前工作表另存...	以新名称保存当前工作表。
面置	配置面尺寸和
打印	打印原理目（另“打印和打印，打印和打印”一章）。
	出 PDF，PostScript，HPGL 或 SVG 格式（参“和打印，和打印”一章）。
关	止用程序。

## 首菜



管理符号表	添加/删除符号
配置路径	置默认搜索路径。
常	首位，网格大小，字段名称等）。
置言	界面言。
	可性置。
入和出	将首到/从文件

管理符号表

符号库

库的范围

全局库工程专用库

文件: C:\Users\taoti\AppData\Roaming\kicad\sym-lib-table

	活动的	别名	库路径	插件类型
1	<input checked="" type="checkbox"/>	4xxx	\${KICAD_SYMBOL_DIR}/4xxx.lib	Legacy
2	<input checked="" type="checkbox"/>	4xxx_IEEE	\${KICAD_SYMBOL_DIR}/4xxx_IEEE.lib	Legacy
3	<input checked="" type="checkbox"/>	74xGxx	\${KICAD_SYMBOL_DIR}/74xGxx.lib	Legacy
4	<input checked="" type="checkbox"/>	74xx	\${KICAD_SYMBOL_DIR}/74xx.lib	Legacy
5	<input checked="" type="checkbox"/>	74xx_IEEE	\${KICAD_SYMBOL_DIR}/74xx_IEEE.lib	Legacy
6	<input checked="" type="checkbox"/>	Amplifier_Audio	\${KICAD_SYMBOL_DIR}/Amplifier_Audio.lib	Legacy
7	<input checked="" type="checkbox"/>	Amplifier_Buffer	\${KICAD_SYMBOL_DIR}/Amplifier_Buffer.lib	Legacy
8	<input checked="" type="checkbox"/>	Amplifier_Current	\${KICAD_SYMBOL_DIR}/Amplifier_Current.lib	Legacy
9	<input checked="" type="checkbox"/>	Amplifier_Difference	\${KICAD_SYMBOL_DIR}/Amplifier_Difference.lib	Legacy
10	<input checked="" type="checkbox"/>	Amplifier_Operational	\${KICAD_SYMBOL_DIR}/Amplifier_Operational.lib	Legacy
11	<input checked="" type="checkbox"/>	Amplifier_Instrumentation	\${KICAD_SYMBOL_DIR}/Amplifier_Instrumentation.lib	Legacy
12	<input checked="" type="checkbox"/>	Amplifier_Video	\${KICAD_SYMBOL_DIR}/Amplifier_Video.lib	Legacy
13	<input checked="" type="checkbox"/>	Analog	\${KICAD_SYMBOL_DIR}/Analog.lib	Legacy
14	<input checked="" type="checkbox"/>	Analog_ADC	\${KICAD_SYMBOL_DIR}/Analog_ADC.lib	Legacy
15	<input checked="" type="checkbox"/>	Analog_DAC	\${KICAD_SYMBOL_DIR}/Analog_DAC.lib	Legacy
16	<input checked="" type="checkbox"/>	Analog_Switch	\${KICAD_SYMBOL_DIR}/Analog_Switch.lib	Legacy
17	<input checked="" type="checkbox"/>	Audio	\${KICAD_SYMBOL_DIR}/Audio.lib	Legacy
18	<input checked="" type="checkbox"/>	Battery_Management	\${KICAD_SYMBOL_DIR}/Battery_Management.lib	Legacy
19	<input checked="" type="checkbox"/>	Comparator	\${KICAD_SYMBOL_DIR}/Comparator.lib	Legacy
20	<input checked="" type="checkbox"/>	Connector	\${KICAD_SYMBOL_DIR}/Connector.lib	Legacy
21	<input checked="" type="checkbox"/>	Connector_Generic	\${KICAD_SYMBOL_DIR}/Connector_Generic.lib	Legacy

浏览库...

添加库

移除库

上移

下移

替换路径:

	环境变量	路径段
1	KICAD_SYMBOL_DIR	D:\Program Files\KiCad\share\kicad\library
2	KIPRJMOD	D:\Program Files\KiCad\share\kicad\demos\interf_u

确定

取消

Eeschema 使用两个表来存可用符号列表，有些符号因范而异：

- 全局

每个目都可以使用全局表中列出的 它保存在您的主目中的 **sym-lib-table** 中（确切路径取决于操作系;表上方的路径）。

- 目用

目中列出的可用于当前打开的目。它保存在目目中的 **sym-lib-table** 文件中（表格上方的路径）。

您可以通过表下方的 全局 或 目用 卡来看任一列表。

### 添加一个新

通... 按并文件或 “附加” 并入文件的路径来添加 定的将添加到当前打开的表（全局/目用）中。

### 除

通一个或多个并 除 按来除

### 属性

表中的每一行都存了几个描述 的字段：

活	启用/禁用 减少加的集很有用。
昵称	昵称是用于将符号分配元件的短唯一符。符号 由'<Library Nickname> : <Symbol Name>'字符串表示。
路径	路径指向位置。
插件型	确定文件格式。
	存特定如果插件使用）。
明	要描述内容。

### 常

示

原理图编辑选项

显示

编辑

控制

颜色

字段名称模板

网格尺寸(G):

50.0

mils

总线宽度(B):

12

mils

线宽 (L):

6

mils

部件ID符号(P):

A

图标比例:

50

275 %

100

☒ 自动

☒ 显示网格(S)

☒ 限制总线和连线垂直或水平绘制(R)

☐ 显示隐藏引脚(H)

☒ 显示页面范围(T)

☐ 符号选择器中的封装预览(实验)

确定

取消

网格尺寸	网格大小  建议使用普通网格（0.050英寸或1,27毫米）。小网格用于元件构建。
厚度	用于制图的笔大小。
线条粗细	用于制图没有象的象的笔大小 指定的笔大小。
元件 ID 表示法	用于表示符号元的后式（U1A, U1.A, U1-1等）
比例	调整工具大小。
显示网格	网格可性置。
将和限制 H 和 V 方向	如果和用垂直或水平制。否可以在任何方向放置和
显示藏的引脚：	通常示不可或（藏）引脚源引脚。
显示面限制	如果中，在屏幕上示面界。
符号器中的封装	示封装框和 放置新符号的封装器。  注意：可能会致或延使用自

原理图编辑选项

×

显示

编辑

控制

颜色

字段名称模板

测量单位(M):

mm

▼

重复项目的水平间距(H):

0

▲

▼

mils

重复项目的垂直间距(V):

100

▲

▼

mils

重复标签的增量(I):

1

▲

▼

默认文本尺寸(A):

50

▲

▼

mils

自动保存间隔时间(A):

10

▲

▼

分钟

☒ 自动放置符号字段(U)

☒ 允许字段自动对齐放置(L)

☐ 自动放置字段对齐到50mil网格(W)

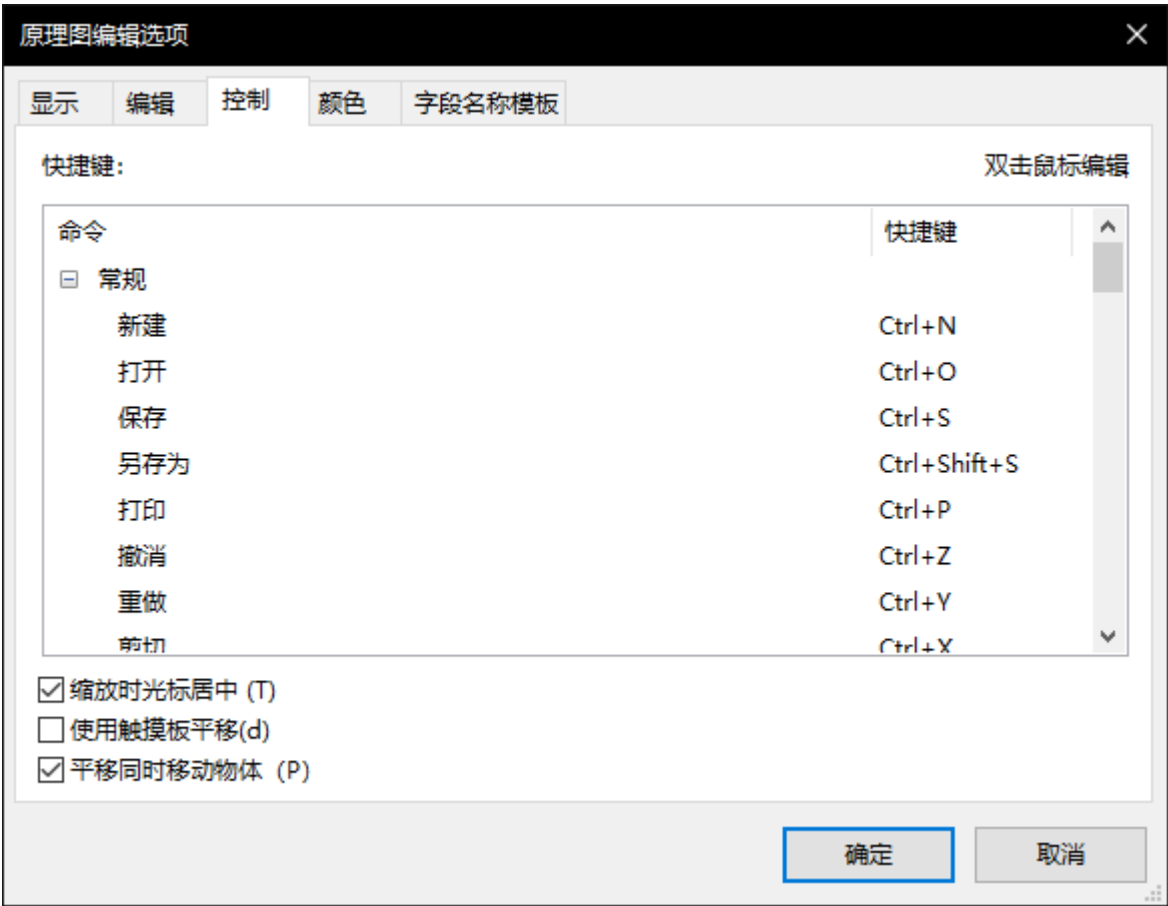
确定

取消

□量□位	□□□示和光□坐□□位 （英寸或毫米）。
重复□目的水平□距	元素复制期□ X □□上的增量（默认□□□□ （在放置符号， □□或□□等物品后， Insert □复制）
重复□目的垂直□距	在 Y □期□增量 元素复制（默认□□□0.100英寸或2,54毫米）。
重复□□的增量	在复制文本□束期□□□□的增量 在一个数字中，例如□□成□□通常□1或-1）。
默认文本大小	□建新文本□或□□□使用的文本大小。
自□保存□□□隔	保存□份之□的□□□分□□□
自□放置符号字段	如果□中， □□符号字段（例如， □和 □可能会移□新放置的符号以避免与其□生冲突 其他□目。
允□字段自□放置更改□□	□展“自□ 放置符号字段”的□□□放置□启用符号字段的文本□□□整 一个新的部分。
始□将自□播放的字段与 50mil 网格□□	自□□展 放置符号字段的□□□如果□中， □使用 50mil 自□放置字段 网格， 否□它□是自由放置的。

控制

重新定义□□并□置用□界面行□□



通过双击操作或新的或右操作以显示菜单

	操作定义新的与双相同）。
撤消更改	原操作的最近更改。
恢复默认	将操作置其默认
撤消所有更改	原操作的所有最近更改。
全部恢复默认	将所有操作置其默认

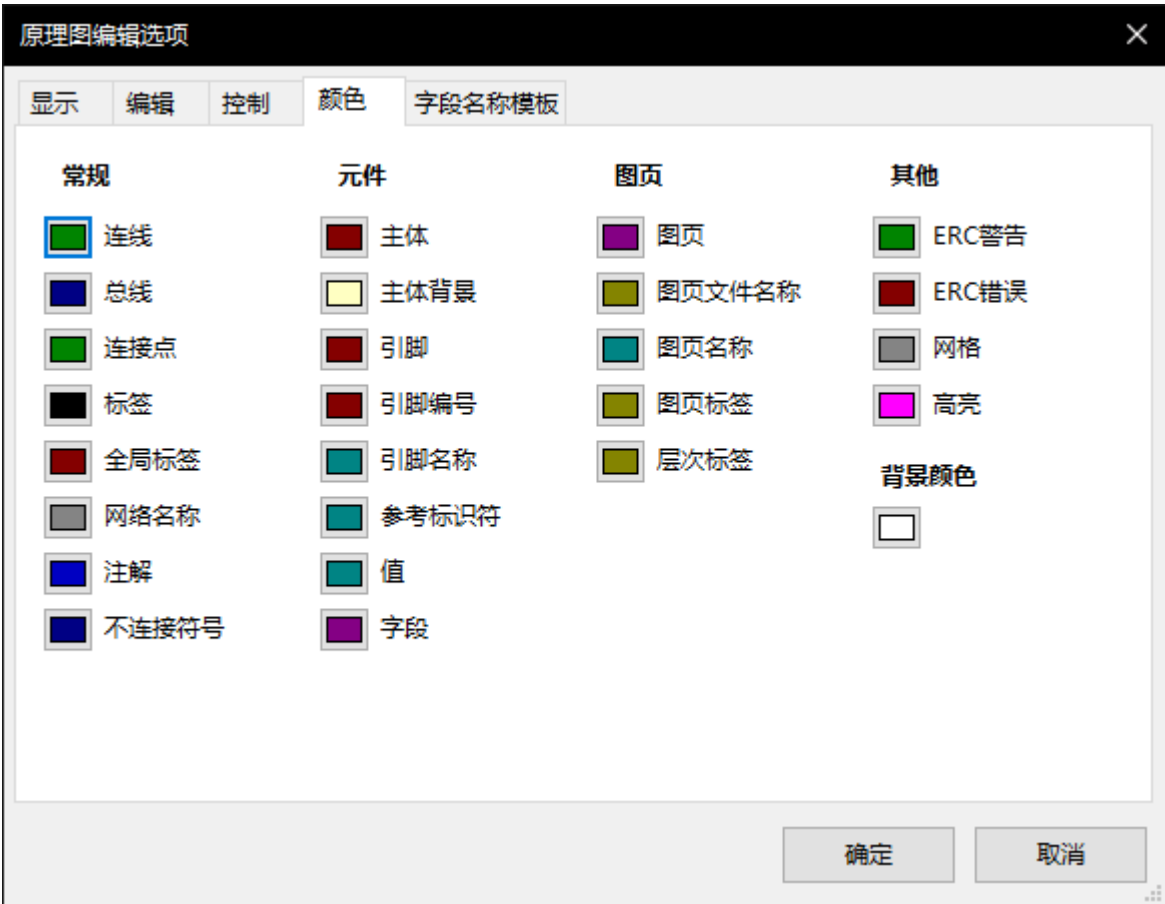
说明:

中心和形光标焦	如果中，指向的位置会形 放大/小到屏幕中心。
使用触摸板行平移	启用或使用或。) 平移 触摸板手和放一个需要按住 Ctrl 否 放大/小和 Ctrl/Shift 是平移修改器。
移对象平移	如果中，自平移窗口 如果光在或移程中离开窗口。

## 色

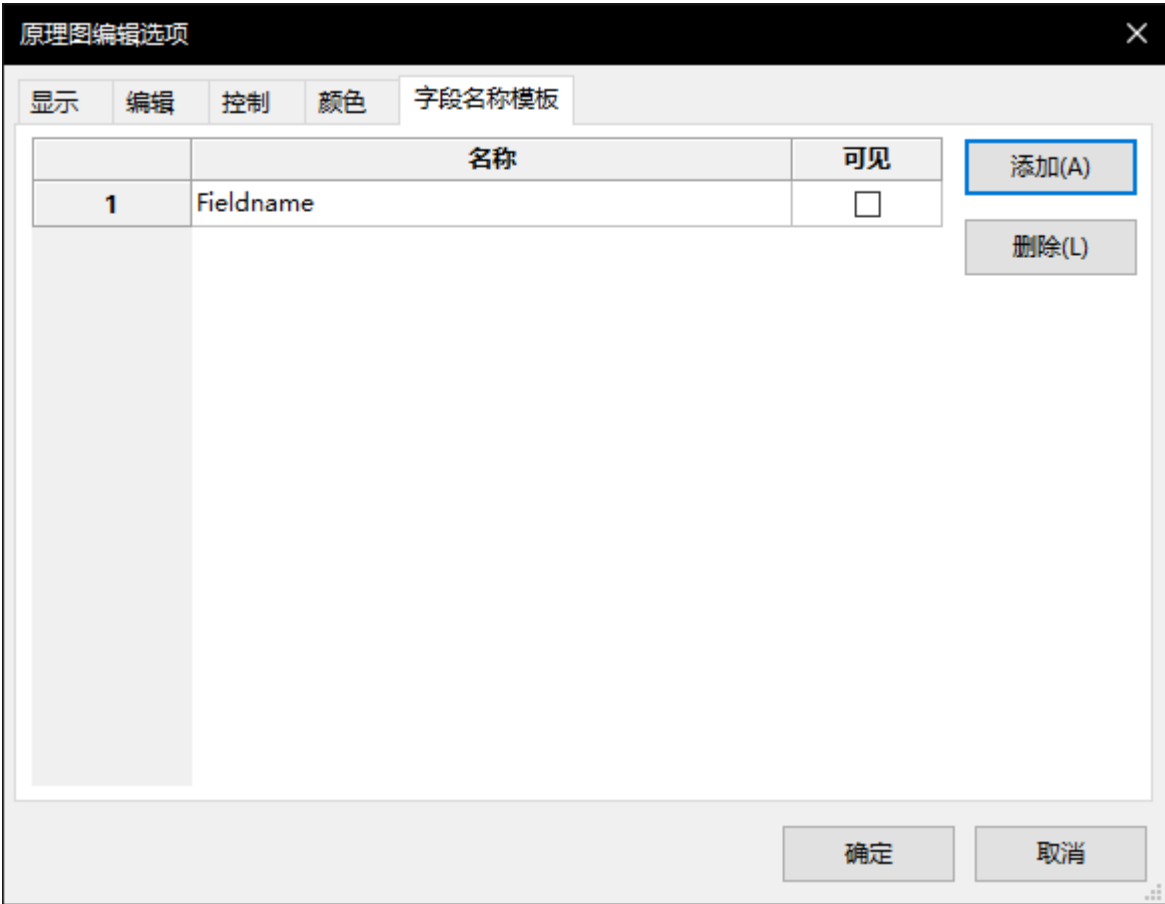
各种形元素的配色方案。 任何色本以特定元素的新色。





## 默认字段

定义将在新放置的符号中显示的其他自定义字段和相关的



## 帮助菜

在帮助（本文档），取有关 KiCad 的广泛教程。

在提交告使用“复制版本信息”来您的构建和系

# 通用部工具

## 表格管理

“置” ( ) 允您定义尺寸和的内容。

页面设置

图纸

尺寸:  
A3 297x420mm

方向:  
横向

自定义尺寸:  
高度: 279.40 宽度: 431.80

布局预览

标题栏字段设置

共 1 页 第 1 页

更改日期  
Sun 22 Mar 2015 <<< 2019/ 2/18

☐ 导出到其他图页

版次  
2B

☐ 导出到其他图页

标题  
UNIVERSAL INTERFACE

☐ 导出到其他图页

公司  
KICAD

☐ 导出到其他图页

注释 1  
Comment 1

☐ 导出到其他图页

注释 2  
Comment 2

☐ 导出到其他图页

注释 3  
Comment 3

☐ 导出到其他图页

注释 4  
Comment 4

☐ 导出到其他图页

页面布局描述文件

浏览

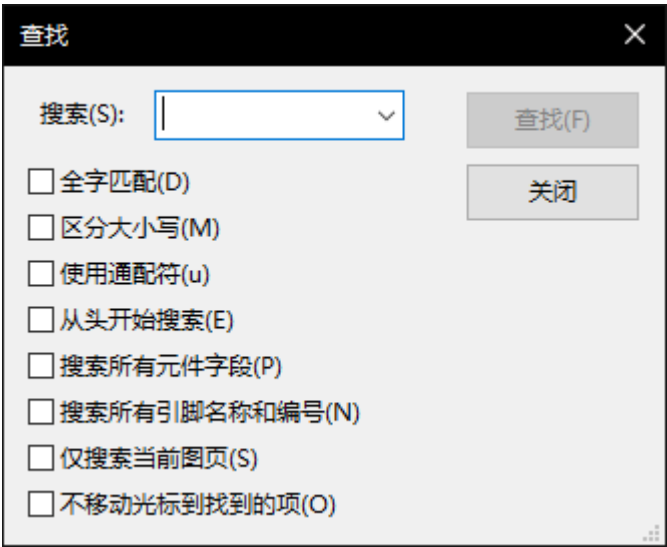
确定

取消

工作表号会自更新。您可以通过按“布日期”按左箭按将日期置今天，但不会自更改。


## 搜索工具

“找” ( ) 可用于搜索工具。



您可以在当前工作表或整个工程结构中搜索引用，或文本字符串。找到后，光标将定位在相关子表中的找到元素上。

## 网表工具

网表工具  打开网表生成工具。

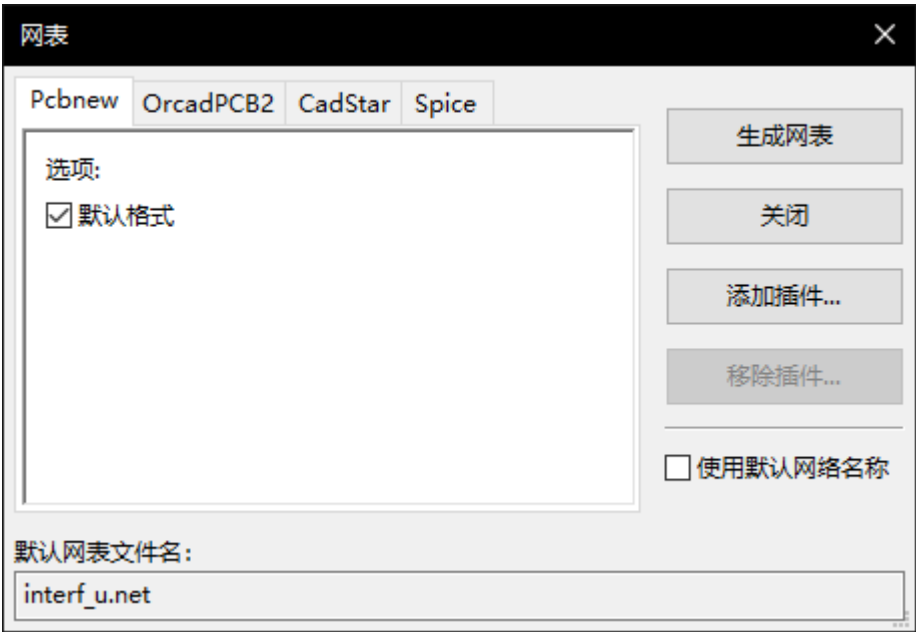
该工具建立一个文件，描述整个工程结构中的所有连接。

在多表工程结构中，任何本地元件在其所属的工作表内可唯一。例如：表3的元件 LABEL1 与表5的元件 LABEL1 不同（如果没有故意引入连接以连接它们，是因为工作表名称路径在内部与本地元件相关）。

- NOTE

即使 Eeschema 中的元件没有文本长度限制，考虑到生成网表的其他程序可能存在此限制。
- NOTE

避免元件中的空格，因为它将在生成的文件中显示为独特的元件。它不是 Eeschema 的限制，而是多网表格式的限制，通常假设元件没有空格。



格式

默认格式	其中以 Pcbnew 作为默认格式。
------	--------------------

可以生成其他格式:

- Orcad PCB2
- CadStar
- Spice (simulators)

可以添加外部插件来展网表格式列表（上中添加了 PadsPcb 插件）。

有关在《create-a-netlist, Create a Netlist》一章中建网表的更多信息。

## 批注工具

后批注工具。此工具分配元件的引用。

于多部件元件（例如包含4个的 7400 TTL分配了多部件后因此，指定 U3 的 7400 TTL将分 U3A, U3B, U3C 和 U3D）。

您可以无条件地批注所有元件或批注新元件，即之前未批注的元件。

批注原理图

范围：

☒ 使用整个原理图

☐ 仅使用当前页面

选项：

☒ 保持现有的批注

☐ 重置现有的批注

☐ 重置，但保持多单元器件的顺序

☐ 保持对话框打开

☐ 不要求确认

顺序：

☒ X方向排序元件 (X)

☐ Y方向排序元件 (Y)

编号：

☒ 使用该数字之后的编号：

0

☐ 参考编号X100

☐ 参考编号X1000

批注

清除批注

关闭

批注信息：

显示：☒ 所有 ☒ 错误 ☒ 警告 ☒ 相关信息 ☒ 活动

保存报告文件

范

使用整个原理图	所有工作表都重新批注（默认）。
仅使用当前页面	仅重新批注当前工作表（此选项在特殊情况下使用，例如 仅估当前表中的电阻数量。
保留已有批注	条件批注，只有新的 元件将被重新批注（默认）。
重置已有批注	所有的无条件批注 元件将被重新批注（此选项将在那里使用 是重复的参考）。
重置，但不要交回任何已批注的多元件部件	保持 当重新批注所有多个元件（例如U2A，U2B）在一起。

**批注顺序**

元件编号的顺序（水平或垂直）。

**批注**

指定的参考格式。

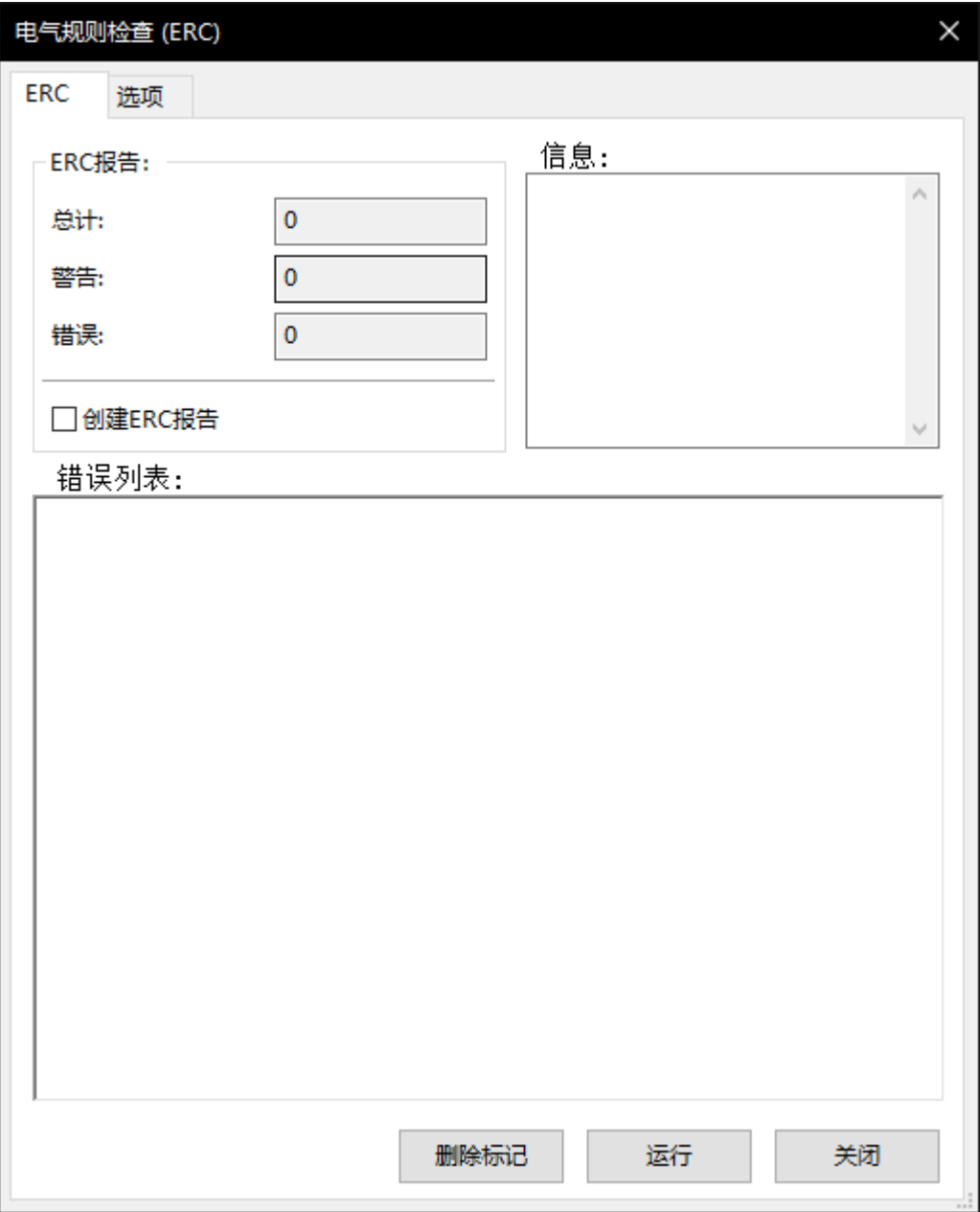
**电气规则检查工具**

运行  后（子程序ERC）工具。

工具运行计划可能被遗忘的接口和不一致。

运行 ERC 后，Eeschema 会放置图标以突出显示错误左边的后示说明。可以生成报告文件。

主要 ERC 对话框



显示在 Electrical Rules Checker 对话框中：

- 和警告的数。
- 计数。
- 警告计数。

创建 ERC 文件	中此可生成 ERC 告文件。
-----------	----------------

命令：

除	除所有ERC/警告
运行	后气
关	关框。

- 消息将跳到原理中的相应

## ERC 框



此卡允您定义引脚之的接; 您可以每种情况3个

- 无
- 警告
- 

可以通修改元格的每个方格。




网

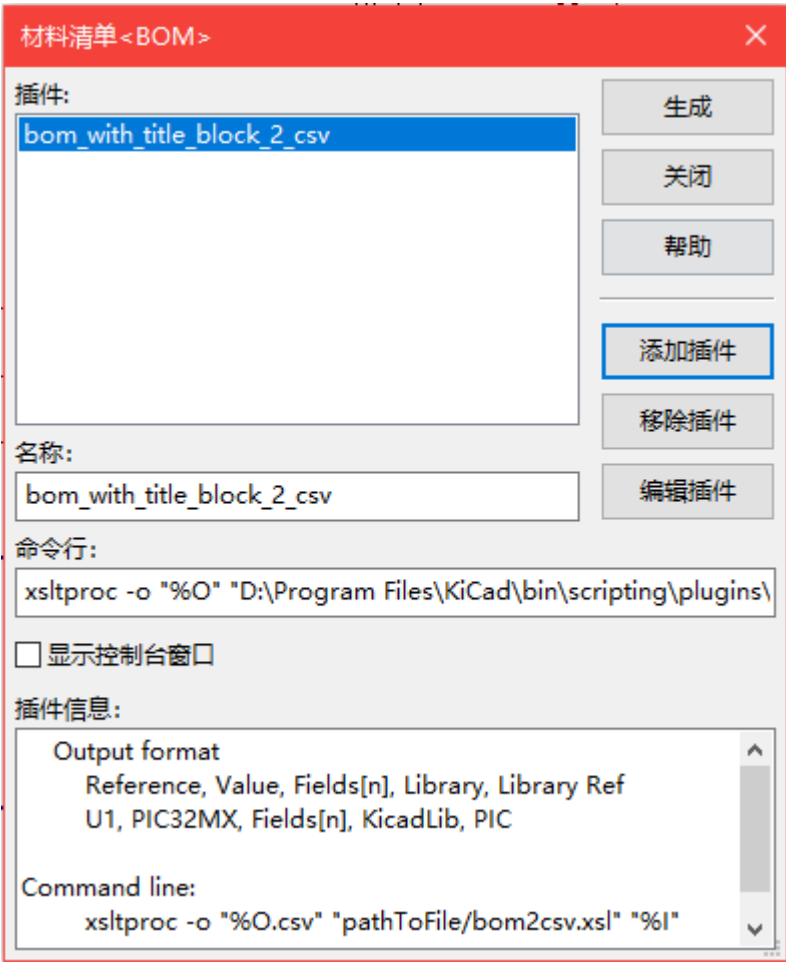
网似网	告网只有字母大小写（例如 lable/Lable/LaBeL）。网名称区分大小写，因此些网被网独的网
网独特的全局网	告网出网一次的全局网 特网。通常需要至少有两个网接。

命令：

初始化网默认网	恢复原始网置。
---------	---------

物料清网工具

网  后网物料清网(BOM) 生成器。此工具生成一个列出元件和/或分网接（全局网网的文件。



Eeschema 的 BOM 生成器使用外部插件，可以是 XSLT 或 Python 脚本。KiCad 程序文件目网中安装了一些示例。

用于 BOM 的网有用的元件属性包括：

- 网 - 使用的每个部件的唯一名称。
- 封装 - 手网入或反网注（网下文）。
- 字段1 - 制造商的名称。
- 字段2 - 制造商的元件号。

字段3 - 分商的元件号。

例如：

符号属性

单元:

A

方向 (度):

0

+90

+180

-90

方向:

默认

X轴镜像

Y轴镜像

转换形状

库符号:

kit-dev-coldfire-xilinx\_5213\_schlib:CONN\_13X2

验证

修改

符号 ID:

461BAEE7

编辑Spice模型

重置字段属性

更新字段值

字段:

名称	值
参考标识符	BDM_PORT1
值	CONN_13X2
封装	Connector_PinHeader_2.54m...
数据手册	

水平位置:

左对齐

居中

右对齐

垂直位置:

顶部对齐

居中

底部对齐

可见性:

显示

旋转

字体风格:

标准

斜体

加粗

加粗斜体

字段名称:

数据手册

字段值:

显示Datasheet数据手册

字体大小:

1.524

mm

位置 X:

0.000

mm

位置 Y:

0.000

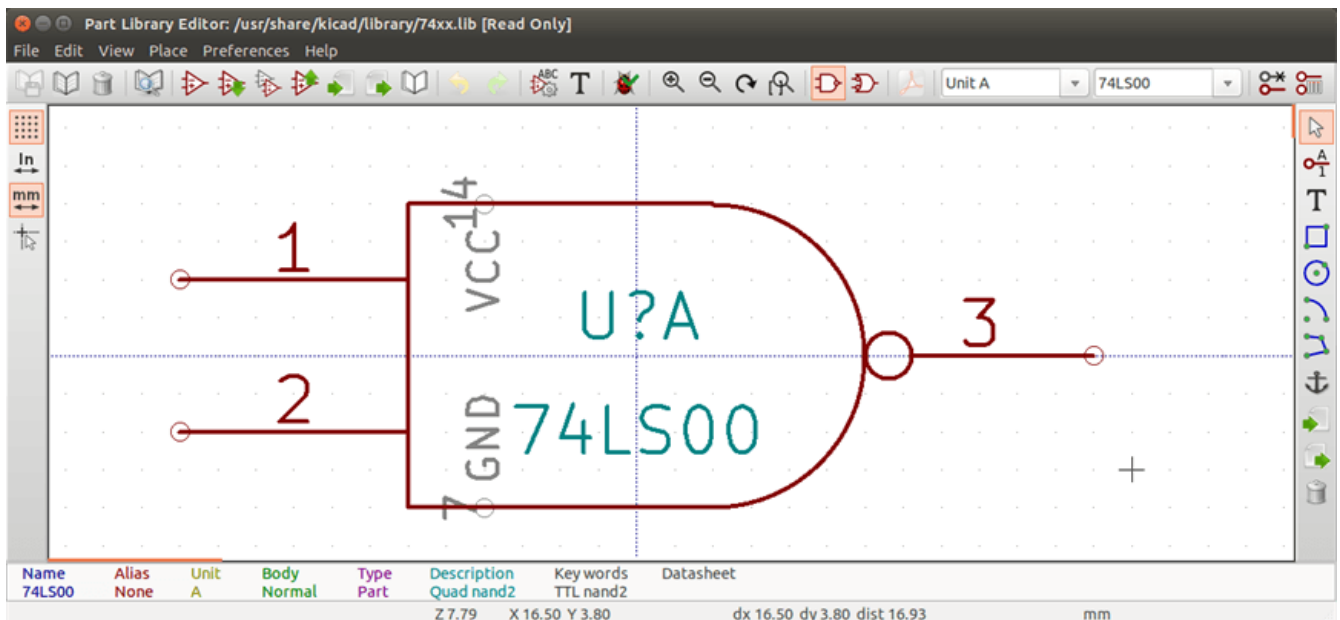
mm

确定


取消

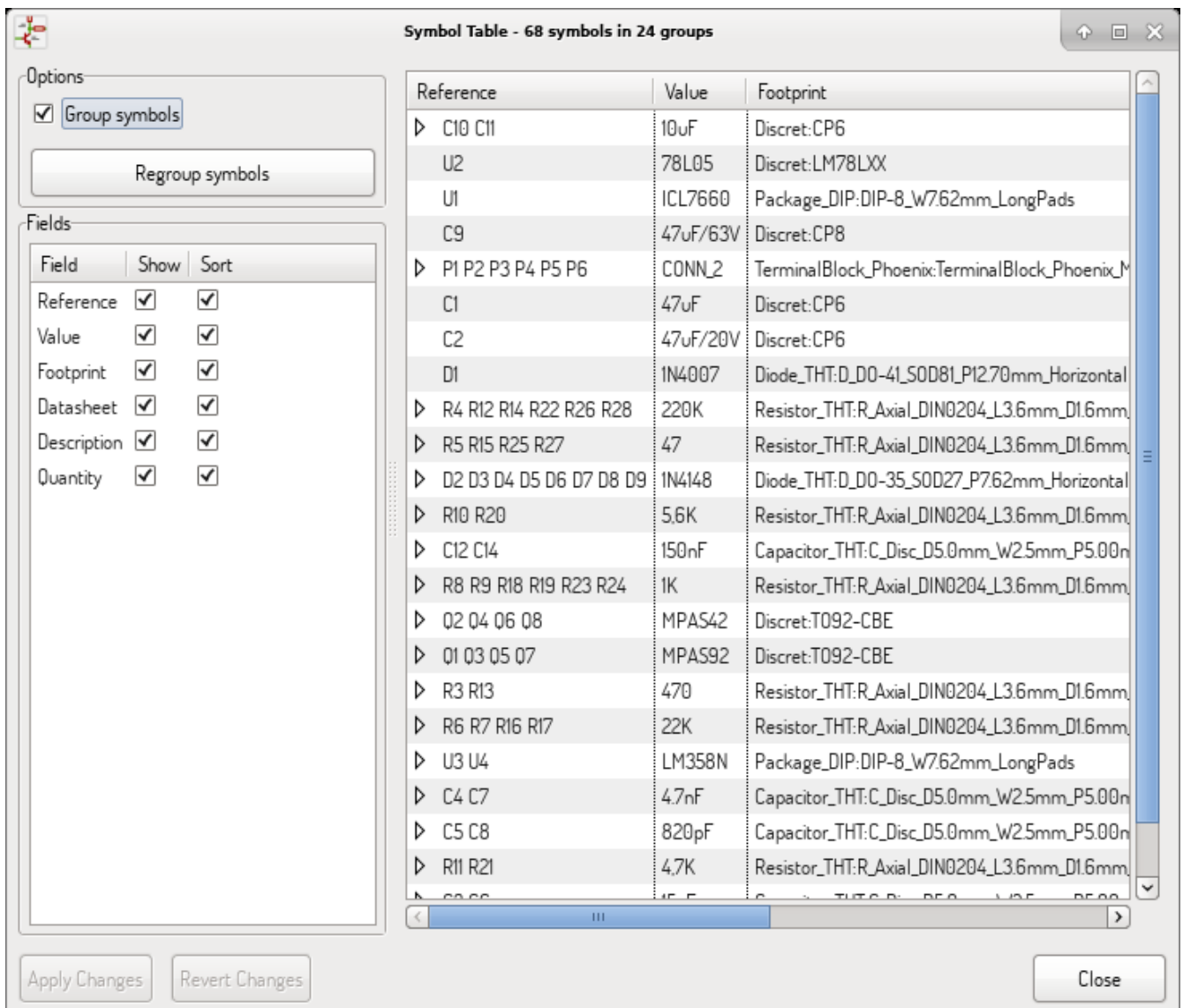
在 MS Windows 上，BOM 生成器窗口有一个特殊图标（由蓝色箭头指示），用于控制外部插件窗口的可见性。+ 默认情况下，BOM 生成器命令行控制台窗口隐藏，点击重定向到 *Plugin info* 字段。点击此图标可显示正在运行的命令的窗口。如果插件提供了图形用户界面，这可能是必要的。

30



## □□ 字段工具

□□  打开□子表格以□看和修改所有符号的字段□□



修改字段□后，您需要通□□□ “□用” 按□接受更改，或通□□□ 恢复 按□撤消更改。

化学字段填充的技巧

子表格中有几种特殊的复制/粘贴方法。在输入在少数元件中重复的字段时它可能很有用。

些方法如下所示。

复制 (Ctrl+C)		黏贴 (Ctrl+V)																																																
<table><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><tr><td></td><td></td><td></td></tr></table>	abc															<table><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><tr><td></td><td></td><td></td></tr></table>	abc															<table><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>abc</td><td>abc</td><td></td></tr><tr><td>abc</td><td>abc</td><td></td></tr><tr><td></td><td></td><td></td></tr></table>	abc									abc	abc		abc	abc				
abc																																																		
abc																																																		
abc																																																		
abc	abc																																																	
abc	abc																																																	
<table><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><tr><td></td><td></td><td></td></tr></table>	11	12	13													<table><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><tr><td></td><td></td><td></td></tr></table>	11	12	13													<table><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>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><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><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><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><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><tr><td></td><td></td><td></td></tr></table>	11	12		21	22											<table><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><tr><td></td><td></td><td></td></tr></table>	11	12		21	22											<table><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><tr><td></td><td></td><td></td></tr></table>	11	12		21	22													
11	12																																																	
21	22																																																	
11	12																																																	
21	22																																																	
11	12																																																	
21	22																																																	
<table><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><tr><td></td><td></td><td></td></tr></table>	11	12	13	21	22	23										<table><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><tr><td></td><td></td><td></td></tr></table>	11	12	13	21	22	23										<table><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><tr><td></td><td></td><td></td></tr></table>	11	12	13	21	22	23												
11	12	13																																																
21	22	23																																																
11	12	13																																																
21	22	23																																																
11	12	13																																																
21	22	23																																																

NOTE 些技巧也可以在具有网格控制元素的其他框中使用。

用于封装分配的输入工具

□□□

□□  后□反□注工具。

此工具允许将 PcbNew 中□建的封装更改□入 Eeschema 中的封装字段。

# 管理符号

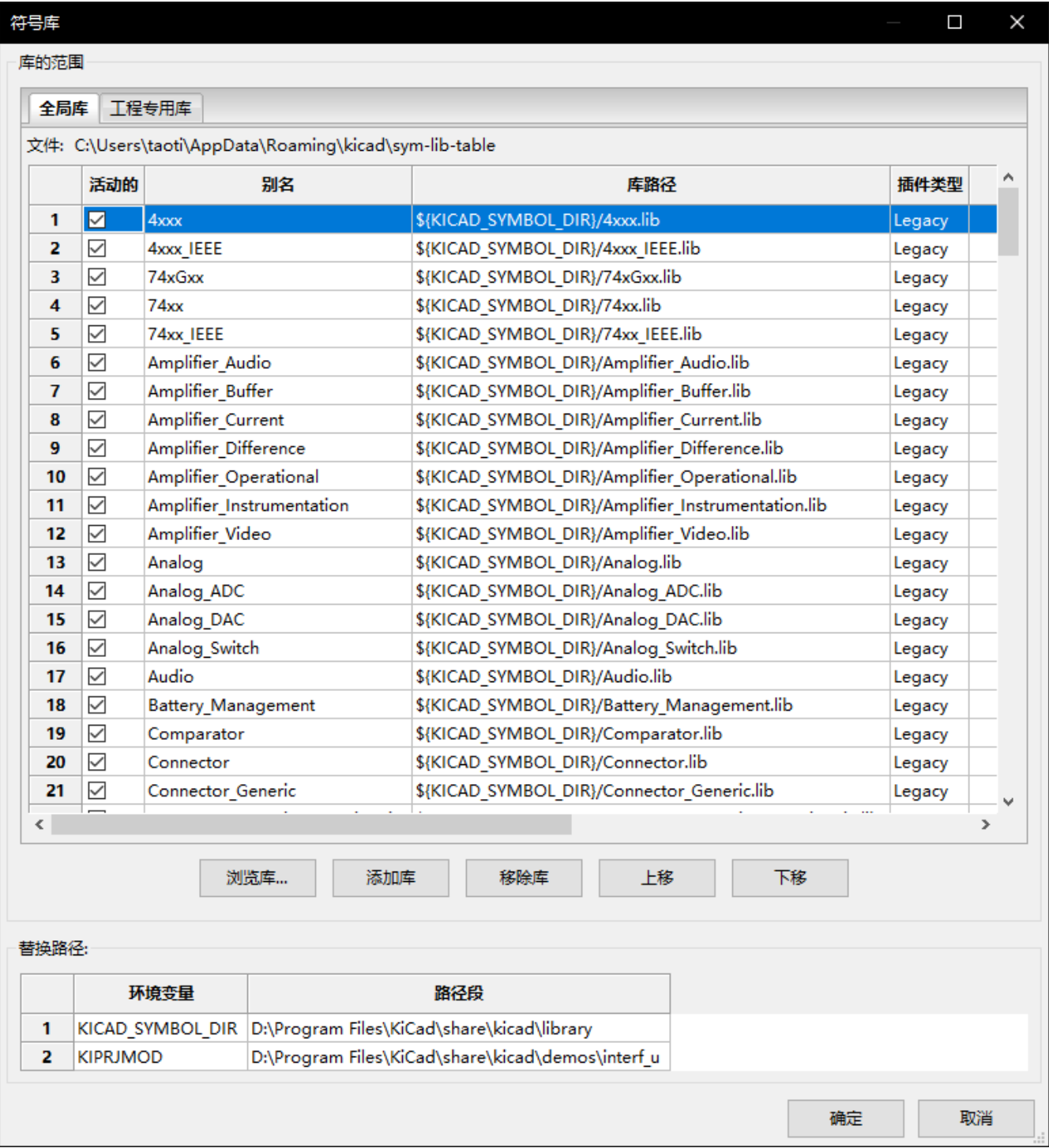
符号包含建原理使用的符号集合。原理中的每个符号由一个全名唯一标识。全名由昵称和符号名称构成。一个例子是“音AD1853”。

## 符号表

符号表包含 KiCad 知道的所有文件的列表。符号表由全局符号表文件和项目特定符号表文件构成。

添加符号Eeschema 使用昵称（在我的示例中“音”）来寻找符号表中的位置。

下图显示了符号表对话框，可以通过用 首 菜单中的 管理符号表 条目来打开对话框。



## 全局符号表

全局符号表包含始用的列表，与当前加的目文件无关。表保存在用主文件的文件符号列表文件中。此文件的位置取决于所使用的操作系统

## 目特定符号表

目特定符号表包含用于当前加的目文件的列表。目特定符号表只能在与目文件一起加行 如果未加目文件或当前目路径中没有符号表文件，会建一个空表，可以其行然后将其与目文件一起保存。

## 初始配置

第一次运行 Eeschema 并且在用的主文件中找不到全局符号表文件 **sym-lib-table** Eeschema 将复制存 在系的 KiCad 模板文件中的默认符号表文件 sym-lib-table 到用主文件中的文件 sym-lib-table。如果找不到默认模板 sym-lib-table 文件，会出一个框，提示入 sym-lib-table 文件的位置。如果未找到 sym-lib-table 或解除框，将在用的主文件中建空符号表。如果生种情况，用手复制 sym-lib-table 或手配置表。

### NOTE

默认符号表包括作 KiCad 的一部分安装的所有符号 根据用途和系的速度，可能是也可能不是所希望的。加符号所需的与符号表中的数量成正比。如果符号加从全局表中除很少和/或从未使用的并根据需要将它添加到目表中。

## 添加表

了使用符号，必首先将其添加到全局表或目特定表中。特定于目的表适用于打开目文件的情况。

每个条目必有一个独特的昵称。

不必以任何方式与文件名或路径相关。冒号 “:” 和 “/” 字符不能在昵称中的任何位置使用。每个条目必具有有效的路径和/或文件名，具体取决于的类型。路径可以定义相或境量替参下面的部分）。

必适当的插件类型才能正确取 KiCad 目前支持旧版符号文件插件。

有一个描述字段用于添加条目的描述。此不使用字段，因此在加添加将不起作用。

- 注意，您不能在同一个表中包含重复的昵称。但是，您可以在全局和目特定的符号表中包含重复的昵称。
- 当出重复的昵称目特定的表条目将先于全局表条目。
- 在目特定表中定义条目包含些条目的 sym-lib-table 文件将写入当前打开的目文件的文件中。

## 境量替代

符号表的最大功能之一是境量替 允定义符号存 在境量中的自定义路径。使用路径中的法 `{ENV_VAR_NAME}` 支持境量替

默认情况下，在运行 KiCad 定义 两个境量：

- KIPRJMOD** 境量，始指向当前打开的目目 **KIPRJMOD** 无法修改。
- KICAD\_SYMBOL\_DIR** 境量。指向使用 KiCad 安装的默认符号的路径。

您可以重写 KICAD\_SYMBOL\_DIR, 方法是在 “首/配置路径” 中自己定义它，路径允您替自己的，以取代默认的 KiCad 符号

**KIPRJMOD** 允您在没有目路径的情况下存 必定义路径（并不 是已知） 目特定符号表中的

使用模式

符号可以全局定义，也可以定义到当前加的目。用全局表中定义的符号始可用，并存用在主文件的sym-lib-table 文件中。目特定符号表当前打开的目文件有效。

每种方法都有点和缺点。在全局表中定义所有意味着它将在需要始可用。做的缺点是加会增加。

在目特定的基上定义所有符号意味着您只有目所需的会减少符号加 缺点是您必始住添加每个目所需的每个符号

一种使用模式是全局定义常用而只需要目特定表中的目。如何定义没有限制。

留目重新映射

加在符号表之前建的原理Eeschema 将原理中的符号接重新映射到相的表符号。一程的成功取决于几个因素：

- 原理中使用的原始仍然可用，并且在符号添加到原理保持不
- 在到所有恢复行行所有恢复行以建恢复或使有恢复保持最新状
- 目符号存的完整性尚未坏。

WARNING	重新映射将份在重新映射期在目文件中的 rescue-backup 文件中更改的所有文件。在重新映射之前，必份目以防万一出
WARNING	即使已禁用恢复操作以行恢复操作以确保正确的符号可用于重新映射。勿取消此操作，否重映射将无法正确重新映射原理符号。任何坏的符号接都必手修复。
NOTE	如果已除原始并且未行恢复，可以将存用作恢复作最后的手段。将存复制到新文件名，并在符号表之前使用Eeschema版本将新文件添加到列表的部。



# 原理图构建和

## 简介

原理图可以用图表示，但是，如果足够大，它需要多张图

由几张图表示的示意图是分层的，并且其所有图每个图由其自己的文件表示）构成Eeschema 项目。分层的原理图的操作将在《hierarchical-schematics, Hierarchical Schematics》章节中描述。

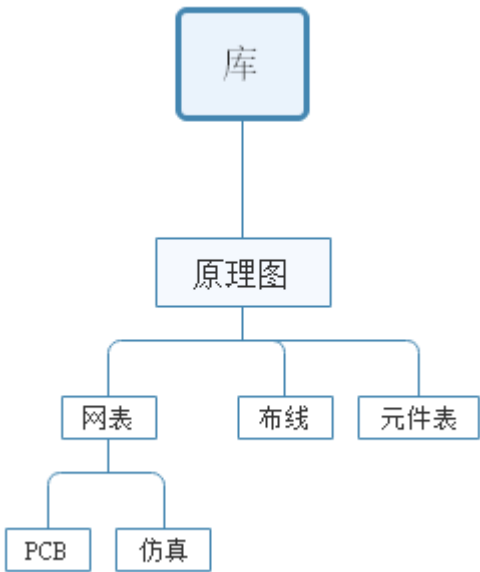
## 一般考虑

使用 Eeschema 设计的原理图不但是电子元件的图形表示。它通常是开发的入口点，允许

- 通过一整套ERC电气规则检查以发现和漏。
- 自动生成物料清单 (BOM) 。
- 用于仿真元件（如 SPICE）的（构建 - 定制 - 网表和文件 - 文件，生成网表）。
- 构建 - 定制 - 网表和网表文件，生成网表），用于得到 PCB 布局。

原理图主要由符号，互连节点，源和源端组成。为了清晰起见您可以放置纯粹的图形元素，如条目，注释和折

## 开始



符号从符号库添加到原理图中。在制作原理图之后，生成一个网表，稍后用于将接口和封装集入 PcbNew。

## 符号放置和

### 找到并放置一个符号

要将符号添加到原理图中，可以使用 。使用输入框可以输入要添加的符号的名称。



“选择符号”对话框将根据您在搜索字段中输入的内容按名称，关键字和明确符号。只需输入高电平器件即可使用它。

- **通配符**：分使用字符“?”和“\*”表示“任意字符”和“任意数量的字符”。
- **关键字** 如果部分的描述或关键字包含格式“Key : 123”，您可以通过输入相等于匹配“Key> 123”（大于），“Key<123”（小于）等。数字可能包括以下不区分大小写的后缀之一：

p	n	u	m	k	meg	g	t
10 <sup>-12</sup>	10 <sup>-9</sup>	10 <sup>-6</sup>	10 <sup>-3</sup>	10 <sup>3</sup>	10 <sup>6</sup>	10 <sup>9</sup>	10 <sup>12</sup>

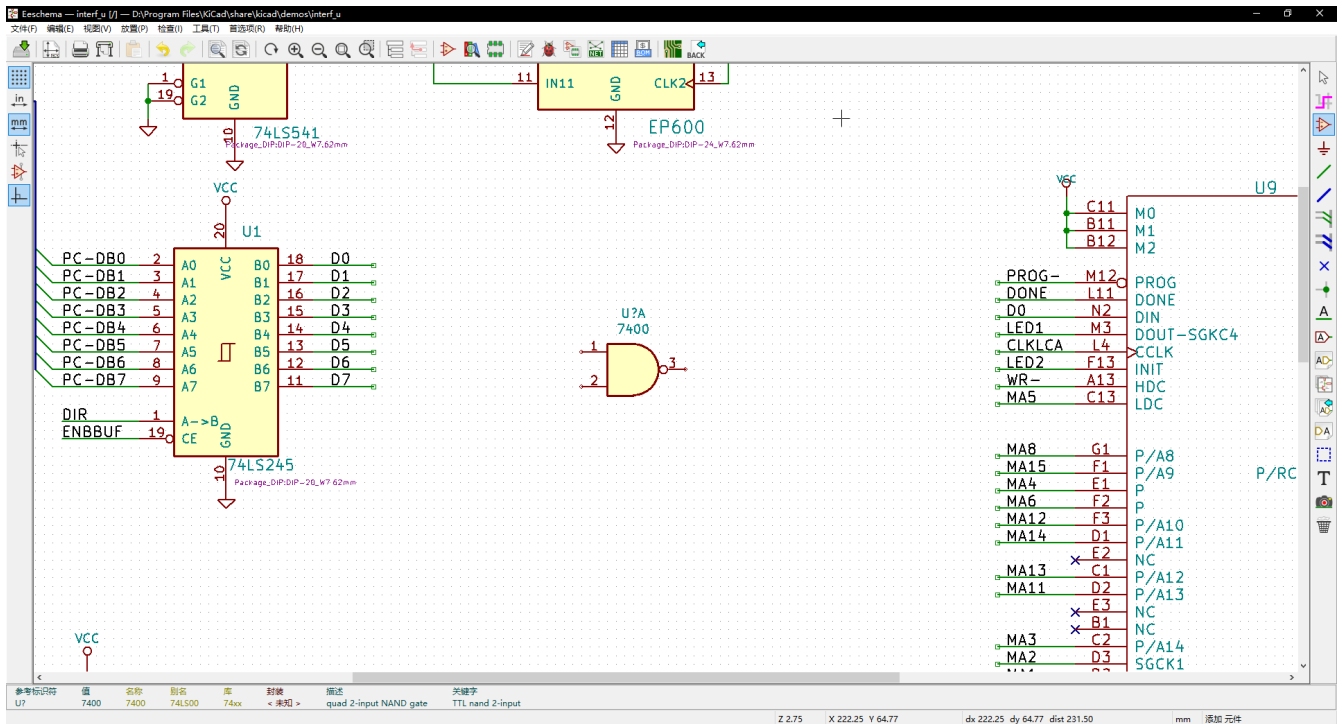
  

ki	mi	gi	ti
2 <sup>10</sup>	2 <sup>20</sup>	2 <sup>30</sup>	2 <sup>40</sup>


- **正则表达式**：如果你熟悉正则表达式，这些也可以用。使用的正则表达式像是 [wxWidgets 高正则表达式](#) 类似于 Perl 常表达式。

在将符号放置在原理图之前，您可以使用或右击上下文菜单其行旋像和其字段。可以在放置后以相同的方式完成。

□是放置期□的符号：



## □源端口

□源端口符号是符号（符号在 □源□中分□□□因此可以使用符号□□器放置它□□ 但是，由于□源放置□繁，因此可以使用  工具。□个工具很相似，只是搜索直接在 □源□中完成。

## 符号□□和修改（已放置的元素）

□□符号有两种方法：

- 符号本身的修改：多□元符号上的位置，方向，□位□□□
- 修改符号的其中一个字段：引用，□□覆盖区等。

□□放置符号□□您可能需要修改其□□特□是□阻器，□容器等），但是立即□其分配参考□号或□□□元是没有用的（除了元件之外）□定□位，您必□手□分配）。□可以通□批注功能自□完成。

## 符号修改

要修改符号的某些功能，□将光□放在符号上，然后□行以下任一操作：

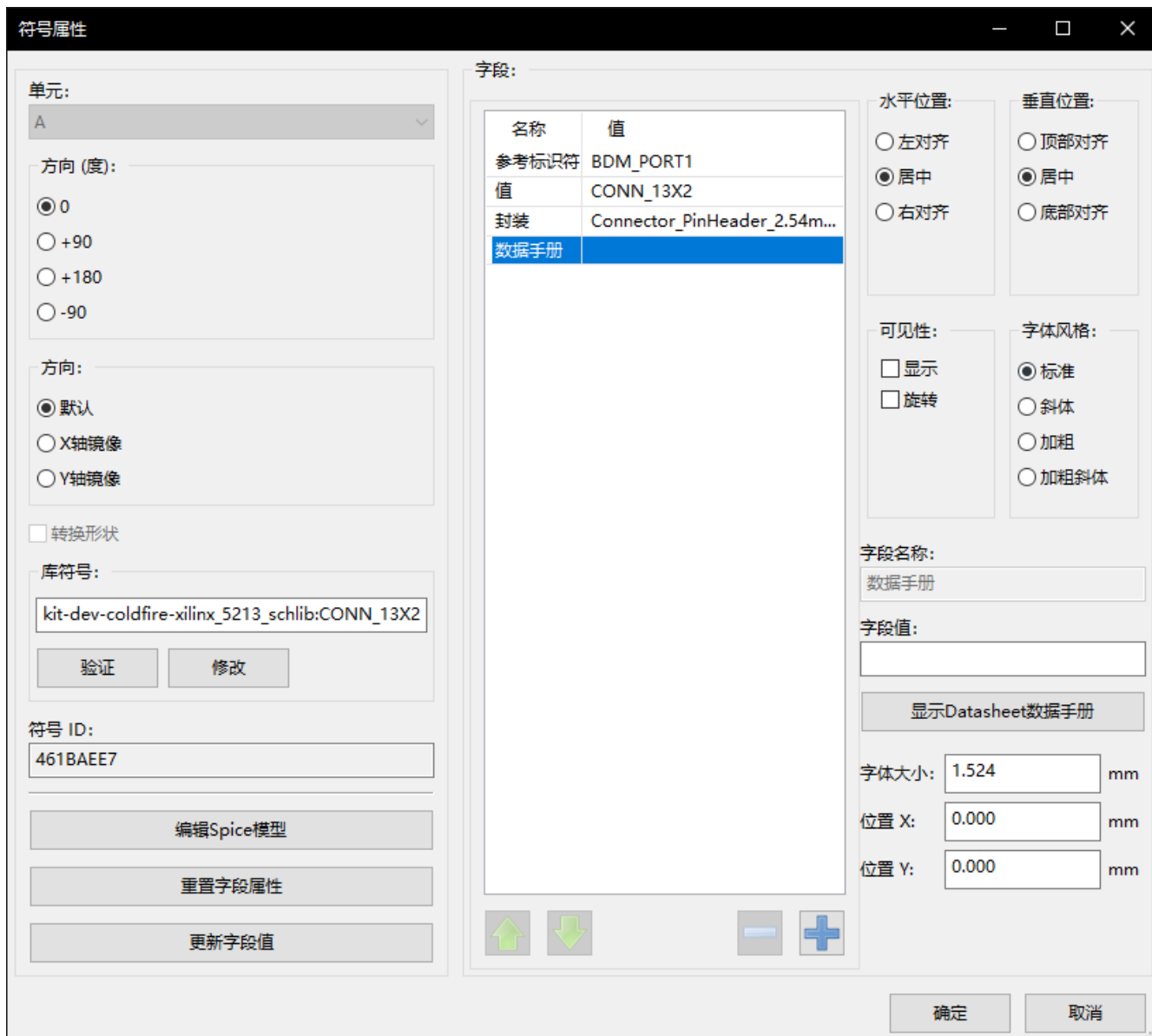
- 双□符号以打开完整的□□□□框。
- 右□□□以打开上下文菜□并使用以下命令之一：移□□方向，□□□□除等。

## 文本字段修改

您可以修改字段的参考，□□位置，方向，文本大小和可□性：

- 双□文本字段□行修改。
- 右□□□以打开上下文菜□并使用以下命令之一：移□□旋□□□□□□□除等。

要□得更多□□□□或者要□建字段，□双□□符号以打开 符号属性□□框。



每个字段都可以是可见的或隐藏的，并且可以水平或垂直对齐。始终正常对齐的符号（无旋转或镜像）指示对齐的位置，并且相对于符号的原点。

重置默认将符号置回原始方向，并重置每个字段的字体大小和位置。但是，文本字段不会被修改，因为它们可能会破坏原理图。

## 源端口

### 简介

所有某些元素也可以与垂直右侧工具上的工具一起放置。

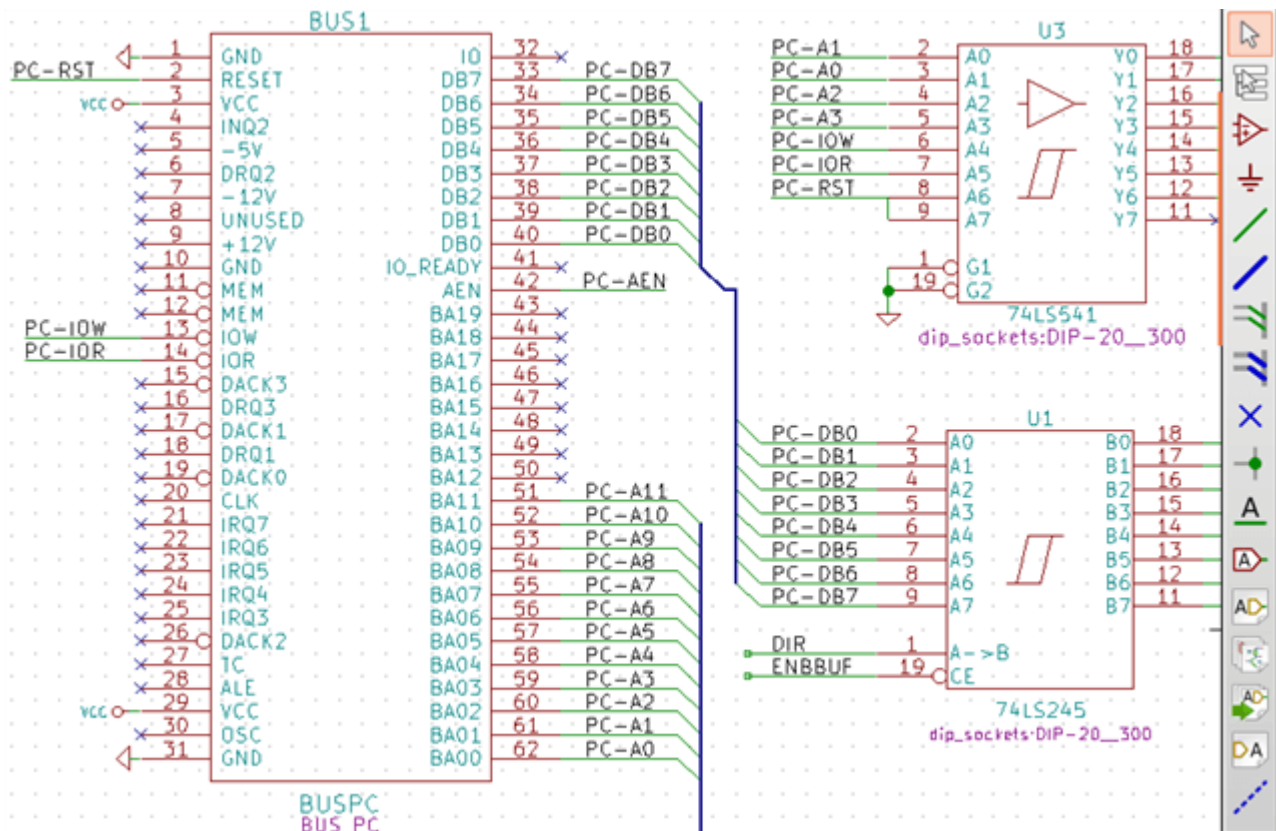
某些元素是：

- 符号之的大多数连接。
- 以圆形方式加入
- 折用于演示。
- 接点：用于在交叉或之间建立连接。
- 入口：表示和之间的连接。



## 网络（网络）

在下面的原理图中，许多引脚连接到网络



## 网络号

网络号是一种在原理图中对相关信号进行分类的方法，以简化复杂的电路。可以使用网络号工具将网络号制成网络号并使用与信号相同的网络号命名。KiCad 6.0 及更高版本中有两种类型的网络号矢量和网络号

一个 **向量网络号** 是以公共前缀并以数字结尾的信号集合。向量网络号命名 '**<PREFIX> [M..N]**'，其中 '**PREFIX**' 是任何有效的信号名称，'**M**' 是第一个后数字，'**N**' 是最后一个后数字。例如，网络号 '**DATA [0..7]**' 包含信号 '**DATA0**'，'**DATA1**'，依此类推，直到 '**DATA7**'。指定 '**M**' 和 '**N**' 的顺序无关紧要，但两者都必须是非空的。

一个 **网络号** 是一个或多个信号和/或向量网络号的集合。网络号可用于将相关信号捆绑在一起，即使它们具有不同的名称。网络号使用特殊命名法：

**<OPTIONAL\_NAME>{SIGNAL1 SIGNAL2 SIGNAL3}**

网络号的成分列在由空格字符分隔的花括号（**{}**）内。网络号的可识别名称位于左大括号之前。如果网络号未命名，在 PCB 上生成的网络号将只是成分内的信号名称。如果网络号具有名称，生成的网络号将具有名称作为前缀其中句点（**.**）将前缀与信号名称分开。

例如，网络号 '**{SCL SDA}**' 有两个信号成分在网表中这些信号将是 '**SCL**' 和 '**SDA**'。网络号 '**USB1 {DP DM}**' 将生成名称 '**USB1.DP**' 和 '**USB1.DM**' 的网络号用于在几个相似路上重复使用大型网络的电路，使用这种技术可以节省空间

网络号可以包含向量网络号 例如，网络号 '**MEMORY {A [7..0] D [7..0] OE WE}**' 包含向量网络号和普通信号，并将生成名称如 '**MEMORY.A7**' 和 '**MEMORY.OE**' 之网络号在 PCB 上的。

网络号可以与信号相同的方式制作和连接，包括使用连接点在交叉点之间建立连接。与信号一样，网络号不能有多个名称 - 如果两个冲突的网络号接到同一网络号会生成 ERC 错误

## 成之的接

成之的接接在的相同成之的接必通接。无法将引脚直接接到; Eeschema 将忽略种型的接。

在上面的示例中，接是通放置在接到引脚的上的行的。到的入口（45度段）是形化的，并不是形成接所必需的。

上，使用重复命令（*Insert* 如果元件引脚按增序排列，可以通过以下方式快速建立接（上在存器，微处理器等元件上的常情况）：

- 放置第一个例如 PCA0)
- 尽可能多地使用重复命令来放置成 Eeschema 将自建垂直的下一个 PCA1, PCA2 .....），理上是在其他引脚的位置上。
- 在第一个下画 然后使用重复命令将其他放在下。
- 如果需要，以相同的方式放置条目（放置第一个条目，然后使用重复命令）。

### NOTE

在“首/”菜中，您可以置重复参数：

- 垂直步
- 水平步
- 增量（因此可以增 2,3 或减）。

## 正在展开

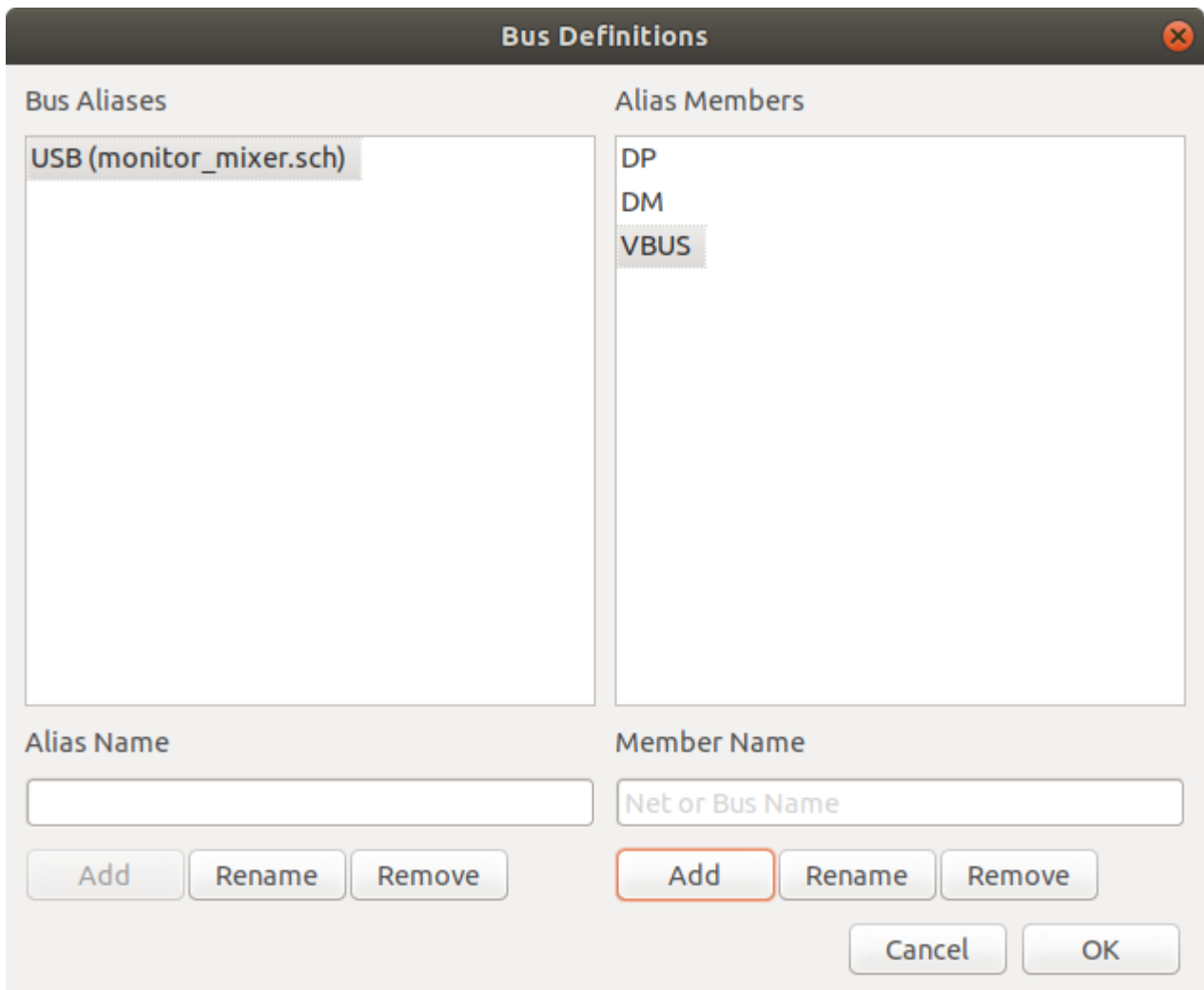
展开工具允您快速分离来自的信号。要展开信号，右象（等）并“展开”。或者，当光位于象上时使用“展开”默认：“D”）。菜允您要展开的成

成后，下一次将把成放在所需位置。工具自生成入口和通向位置。放置后，您可以放置其他段（例如，接到件引脚）并以任何正常方式完成

## 名

名是一种快捷方式，可让您更有效地使用大型 它允您定义并其指定一个短的名称，然后可以在原理中使用名称而不是完整的名。

要建名，在“工具”菜中打开“定义”框。



别名可以被命名任何有效的信号名称。使用列表框，您可以向别名添加信号或矢量。作为一种快捷方式，您可以输入或粘贴由空格分隔的信号和/或列表，并将它们全部添加到别名定义中。在这个例子中，我定义了一个名为“USB”的别名，其成员为“DP”，“DM”和“VBUS”。

定义别名后，可以通过将别名放在大括号内来用于创建信号，例如“{USB}”。这与“{DP DM VBUS}”具有相同的效果。您还可以添加前缀名称，例如“USB1 {USB}”会生成如上所述的“USB1.DP”等网络。对于重复使用别名可以使原理图上的网络更短。记住，别名只是一个快捷方式，别名的名称不包含在网表中。

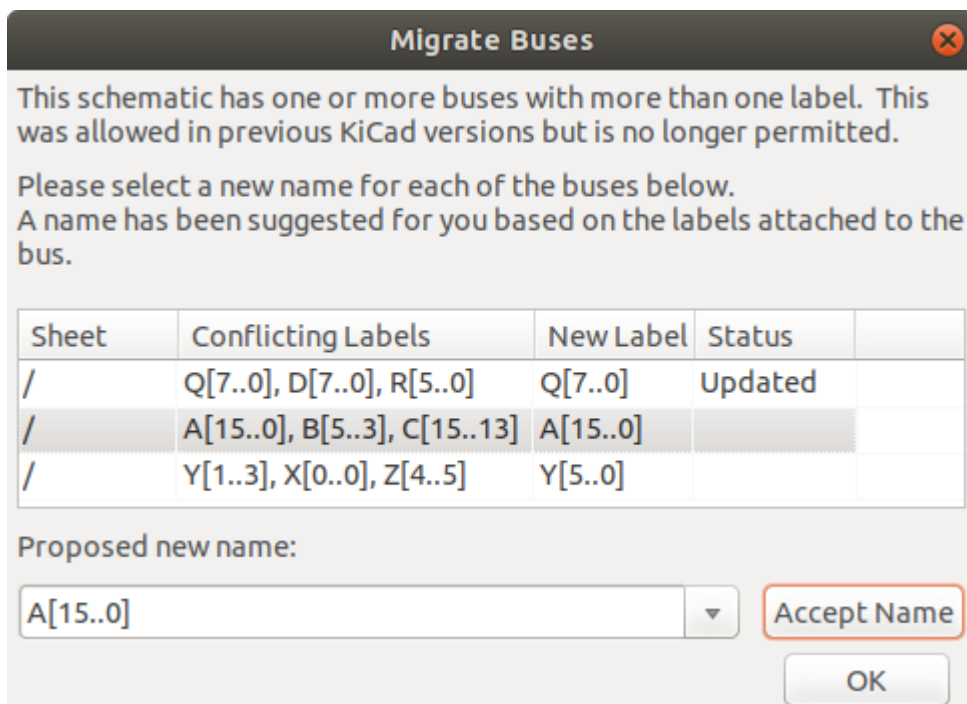
别名保存在原理图文件中。在特定的原理图工作表中创建的任何别名都可用于同一文档结构中的任何其他原理图工作表。

## 有多个别名的情况

KiCad 5.0 及更早版本允许将具有不同别名的信号接在一起，并且在网表列表期间将加入一些信号的成本。此行已在 KiCad 6.0 中删除，因为它与 EDA 不兼容，并且可能导致令人困惑的网表，因此不容易确定信号将接收的名称。

如果您在旧版本的 KiCad 中打开使用此功能的文档，您将看到“迁移”对话框，该对话框将提示您更新原理图以便在任何给定的信号路上只存在一个信号。





由于具有多个冲突的每行，您必须保留的行。下拉名称框允许您在设计中存在的行之一行或者您可以通过手动将其输入新名称字段来输入其他名称。

## 源端口连接

当符号的源引脚可连接它必须连接，就像任何其他信号一样。

和触器等符号可能有不可连接的源引脚。必须小心一些原因。

- 由于它不可连接你无法连接。
- 你不知道他的名字。

此外，将它可连接并将它像其他引脚一样连接将是一个坏主意，因为原理图将不可连接并且不符合通常的惯例。

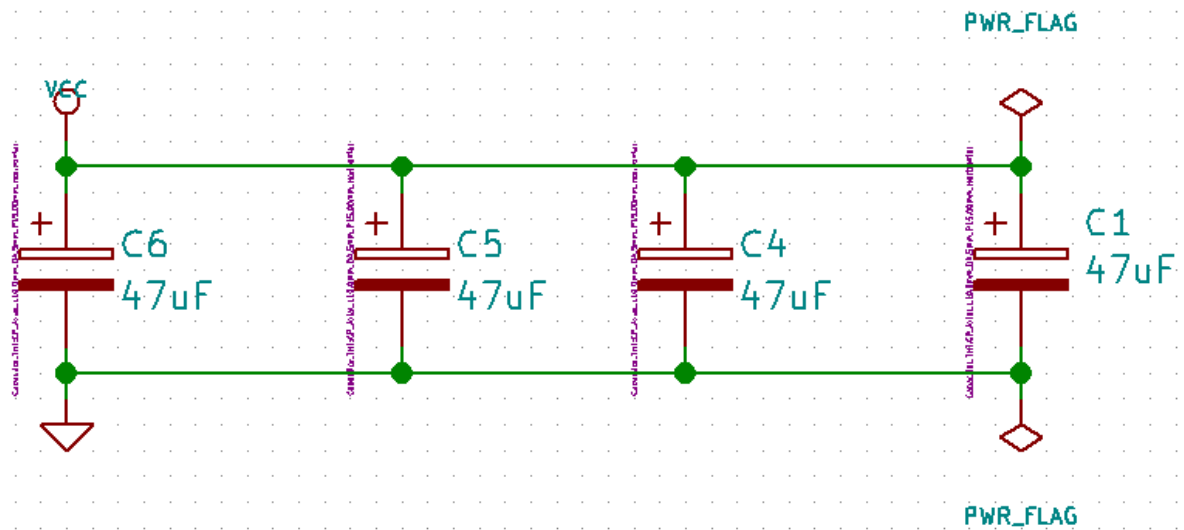
### NOTE

如果要抑制显示一些不可连接的源引脚，必须在主菜单的 首选项/显示框中 中 显示不可连接的源引脚 或者 左边的工具上的。

Eeschema 自动将同名的源引脚连接到名称的源网。可能需要连接不同名称的源网（例如，TTL 元件中的 GND 和 MOS 元件中的 VSS）；因此使用源端口。

建议不要使用任何行源连接。它只有一个\_本地\_连接范围不会连接不可连接的源引脚。

下面显示了源端口连接的示例。




在示例中，地（GND）接到源端口 VSS，源端口 VCC 接到 VDD。

可以看到两个 PWR\_FLAG 符号。它表明两个源端口 VCC 和 GND 确接到源。如果没有两个标志，ERC 工具将断出：警告：源端口未通。

所有些符号都可以在“源”符号中找到。



## “无接”标志

些符号于避免意外的 ERC 警告非常有用。气确保没有接意外地保持未接状

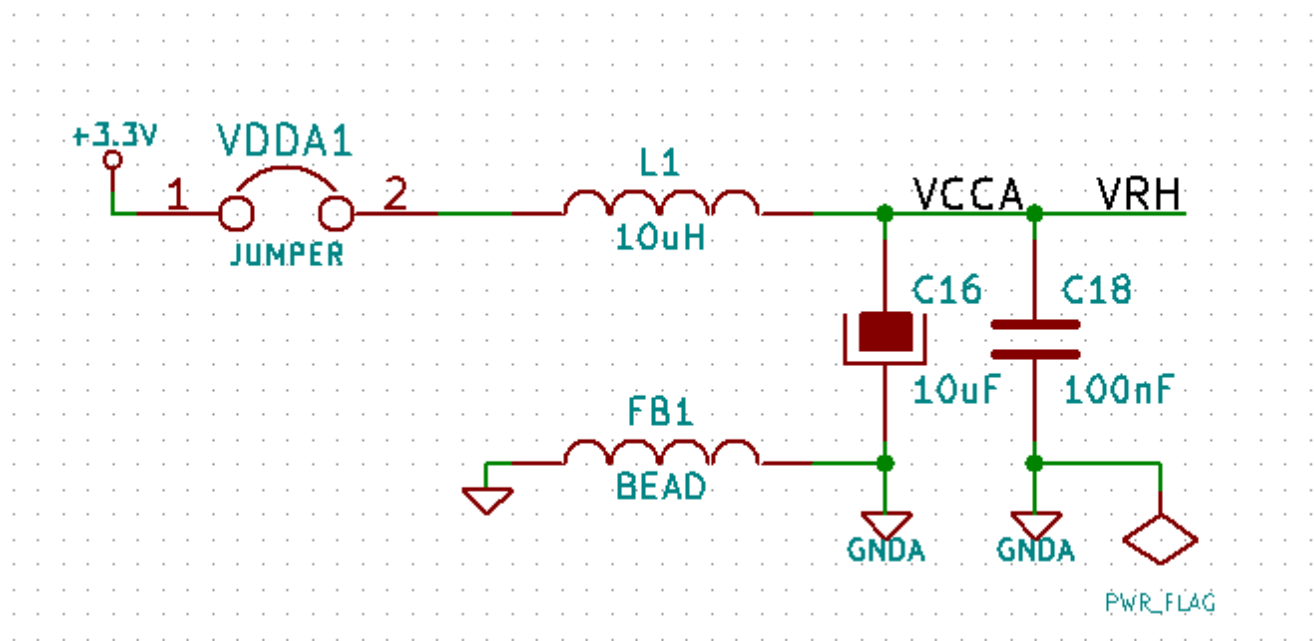
如果引脚必保持未接状必在些引脚上放置“无接”标志（工具 ）。些符号生成的网表没有任何影响。

## 充

### 文本注

放置批注(例如文本字段和框架)可能很有用(有助于理解原理)。文本字段(工具 )和多段(工具 )用于此用途，而和是接元素。

在里，您可以找到有文本注的框架示例。



## 表格

使用工具。

页面设置

图纸

尺寸:

A3 297x420mm

方向:

横向

自定义尺寸:

高度:

279.40

宽度:

431.80

布局预览



标题栏字段设置

共 1 页 第 1 页

更改日期

Sun 22 Mar 2015

<<<

2019/ 2/18

导出到其他图页

版次

2B

导出到其他图页

标题

UNIVERSAL INTERFACE

导出到其他图页

公司

KICAD

导出到其他图页

注释 1

Comment 1

导出到其他图页

注释 2

Comment 2

导出到其他图页

注释 3

Comment 3

导出到其他图页

注释 4

Comment 4

导出到其他图页

页面布局描述文件

浏览

确定

取消

47

Comment 4		
Comment 3		
Comment 2		
Comment 1		
<b>KICAD</b>		
Sheet: /		
File: interf_u.sch		
<b>Title: UNIVERSAL INTERFACE</b>		
Size: A3	Date: Sun 22 Mar 2015	<b>Rev: 2B</b>
KiCad E.D.A. kicad (5.0.2)-1		Id: 1/1

工作表号（工作表X/Y）会自更新。

## 救存的符号

默认情况下，Eeschema 根据置的路径和从目加符号序。在加非常旧的目可能会致如果已更改或已被除，或者自使用后已不存在。在目中，目中的目将自替新版本。新版本可能没有正确或者可能方向不同，致出破的原理

保存目将包含具有当前符号内容的存以及原理 允在没有完整的情况下分目。如果加其存和系中存在符号的目，Eeschema 将描以找冲突。找到的任何冲突都将在以下框中列出：

This project uses symbols that no longer match the ones in the system libraries. Using this tool, you can rescue these cached symbols into a new library.

Choose "Rescue" for any parts you would like to save from this project's cache, or press "Cancel" to allow the symbols to be updated to the new versions.

All rescued components will be renamed with a new suffix of "-RESCUE-kicad\_test" to avoid naming conflicts.

**Symbols with cache/library conflicts:**

scue symbol	Symbol name
<input checked="" type="checkbox"/>	DIODE

**Instances of this symbol:**

Reference	Value
D1	DIODE
D2	DIODE
D3	DIODE


**Cached Part:**



**Library Part:**



Never Show Again

 Cancel

 OK

您可以在此示例中看到项目最初使用的是阴极朝上的二极管，但在项目中包含阴极朝下的二极管。这种改变会打破原理图。在此按 OK 将使符号缓存保存到特殊的 恢复 中，并重命名所有符号以避免命名冲突。

如果按 取消 不会进行任何恢复，因此 Eeschema 默认会添加所有新元件。如果此保存原理图将覆盖缓存并且旧符号将无法恢复。如果已保存原理图仍可以通过“工具”菜单中的“恢复缓存元件”再次调用恢复对话框，再次运行恢复功能。

如果您不想看到此对话框，可以按 从不再显示。默认设置是不进行任何操作并允许添加新元件。可以在 首选项中更改此设置。

# 分原理

## 介

对于大于几百的项目，分表示通常是一个很好的解决方案。如果要管理此项目，需要：

- 使用大表会致打印和理
- 使用多个工作表，将引入入次构。

然后，完整的原理包含一个主要的原理表，称根表，以及构成次构的子表。此外，巧妙地将计分独立的表格通常会提高其可性。

从根表中，您必能找到所有子表。借助可通部工具的集成次构航器，Eeschema 可以轻松分原理管理。

有两种型的次构可以同存在：第一种次构被起并且具有普遍用途。第二个包括在中建符号，些符号在原理中看起来像符号，但上于描述其内部构的示意

第二种型用于开集成路，因在种情况下，您必在制的原理中使用函数

Eeschema 目前不会理第二种情况。

次构可以是：

- 定：定的工作表只使用一次
- 复：定的工作表被多次使用 (倍数例)
- 平面：是一个的次构, 但不会制工作表之的接。

Eeschema 可以理所有些次构。

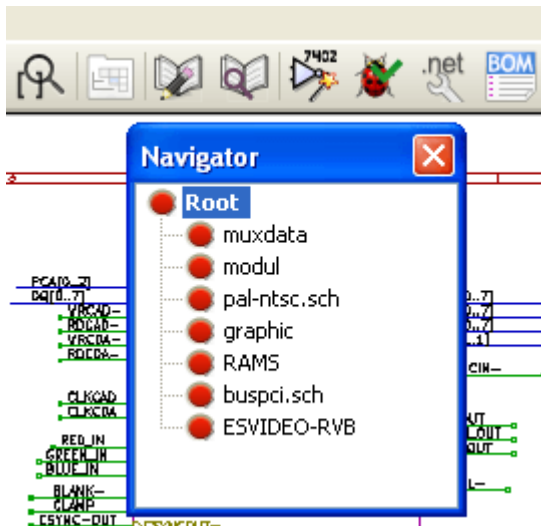
建分原理很容易，整个次构从根原理开始理，就像您只有一个原理一

要理解的两个重要步是：

- 如何建子表。
- 如何在子表之建立气接。

## 在次构中航

通使用可通部工具上的按的航工具来子表之的航。




其名称即可每个工作表。要快速右工作表名称，然后“入工作表”或双工作表的范

要将当前工作表退出到父工作表，右原理中没有象的任何位置，然后在上下文菜单中“离开工作表”或按“Alt + Backspace”。

## 本地、分和全局

### 属性

本地工具  在工作表内接信号。分工具  在工作表内接信号，并接到父工作表中的分引脚。

全局工具  接所有次构中的信号。源引脚（型源入和源出）不可就像全局一因它在所有次构中被接在它之

**NOTE** 在次构（或复中，可以使用分和/或全局

## 次构建摘要


您必：

- 在根工作表中放置一个名工作表符号的次构符号。
- 使用航器入新原理子工作表）并制它，就像任何其他原理一
- 通将全局HLabels）放在新的原理子表）中，并在根表中使用相同名称的称 SheetLabels制两个原理之的接。些 SheetLabel 将接到根表的工作表符号，接到原理的其他元素，如准符号引脚。

## 工作表符号

制一个由两个角点定义的矩形，表示子表格。

此矩形的大小必允您放置以后特定次构引脚，于子表中的全局HLabels）。

些似于通常的符号引脚。工具 .

以放置矩形的左上角。再次以放置右下角，具有足大的矩形。

然后，系将提示您此子表入文件名和表名称（以便使用次构航器相的原理

您必至少提供一个文件名。如果没有工作表名称，文件名将用作工作表名称（通常的方法）。

## 接 - 分引脚

您将在此建建的符号的接点（次构引脚）。

些接点似于普通符号引脚，但是只需一个接点就可以接一个完整的

有两种方法可以做到一点：

- 在制子表之前放置不同的引脚（手放置）。
- 在制子表和全局半自放置）后放置不同的引脚。

第二种解决方案是非常的。

手放置：

- 工具 。
- 要放置引脚的次构符号。

有关建名“接”的分引脚的示例，参下文：

您可以在建期或之后定义引脚的名称，大小和方向，方法是右引脚并在出菜中引脚。

在工作表内部，必使用与“分引脚”相同的名称“分”。注意正确匹配些名称必手完成，就是下面第二种方法首的原因。

自放置：




工具 。

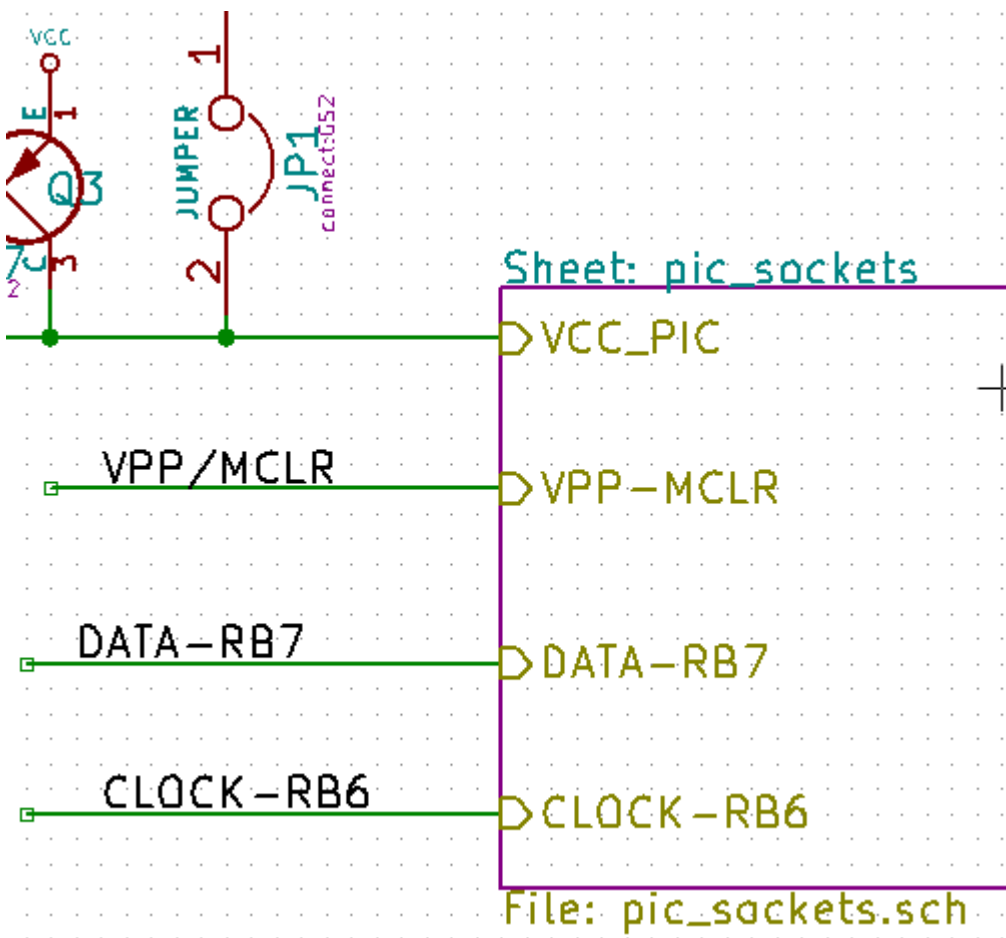
- 要从中输入与相原理中放置的全局引脚的次结构符号。如果存在新的全局输出引脚，即不于已放置的引脚。
- 要放置此引脚的位置。

因此可以快速且无地放置所有必需的引脚。他的方面符合相的全局

## 接 - 分

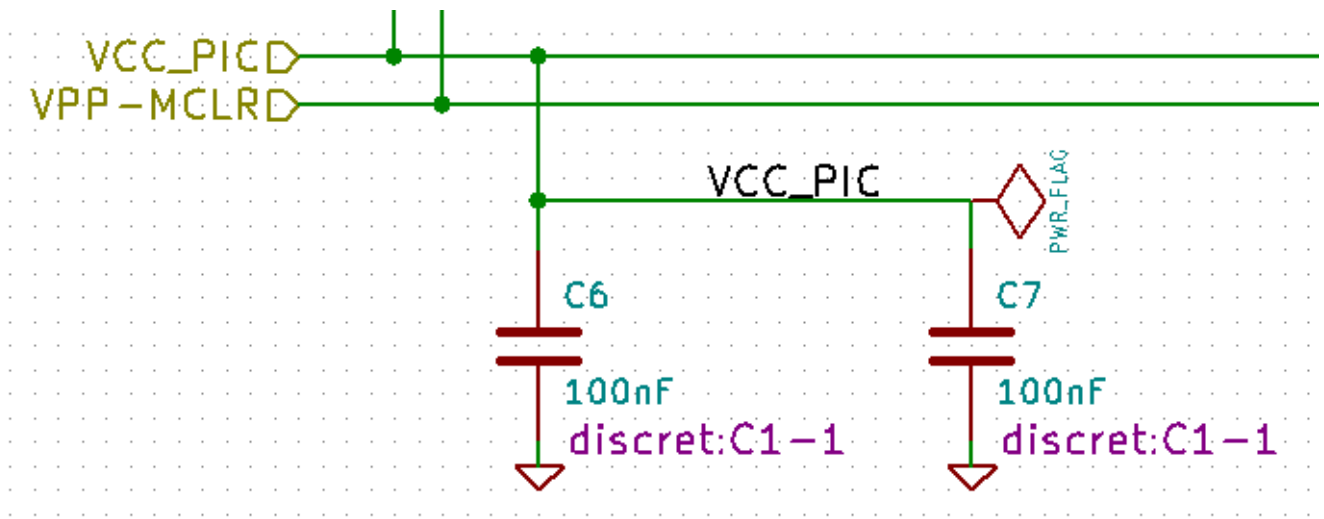
建的工作表符号的每个引脚必于子工作表中名分 分与似，但它提供子表和根表之的接。两个互 (pin 和 HLabel) 的形表示是似的。使用工具像建分 .

参下面的根表示例：



注意引脚 VCC\_PIC 接到连接器 JP1。

以下是子表中的相接：



您再次找到两个相等的分表提供两个分表之的接口。

**NOTE** 根据前面描述的语法（Bus [N..m]），您可以使用分表和次构引脚接两条

## 分表和全局和形源引脚

以下是有关提供接口的各种方法的一些注而不是有接口。

### 表的

具有局部能力，即限于放置它的示意图是因

- 每都有一个号。
- 此工作表号与相关

因此，如果在 n°3 中放置 “TOTO” 上真正的 “TOTO\_3”。如果您在工作表 n°1（根表）中放置了 “TOTO” 上放置一个名 “TOTO\_1” 的与 “TOTO\_3” 不同。即使只有一也是如此。

### 分

于而言，于分也是如此。

因此，在同一中，分 “TOTO” 被认接到本地 “TOTO”，但没有接到另一中称 “TOTO” 的分或分被认接到放置在父表中的分符号中的相等表引脚符号。

### 形源引脚

可以看出，如果形源引脚具有相同的名称，它将接在一起。因此，所有声称 形源引脚 并命名 VCC 的源引脚都将所有符号 VCC 的源引脚接在它所放置的工作表内。

意味着如果将 VCC 放在子表中，它将不会接到 VCC 引脚，因上是 VCC\_n，其中 n 是表号。

如果您希望此 VCC 真正接到整个原理的 VCC 必通 VCC 源符号将其明确接到不可的源引脚。

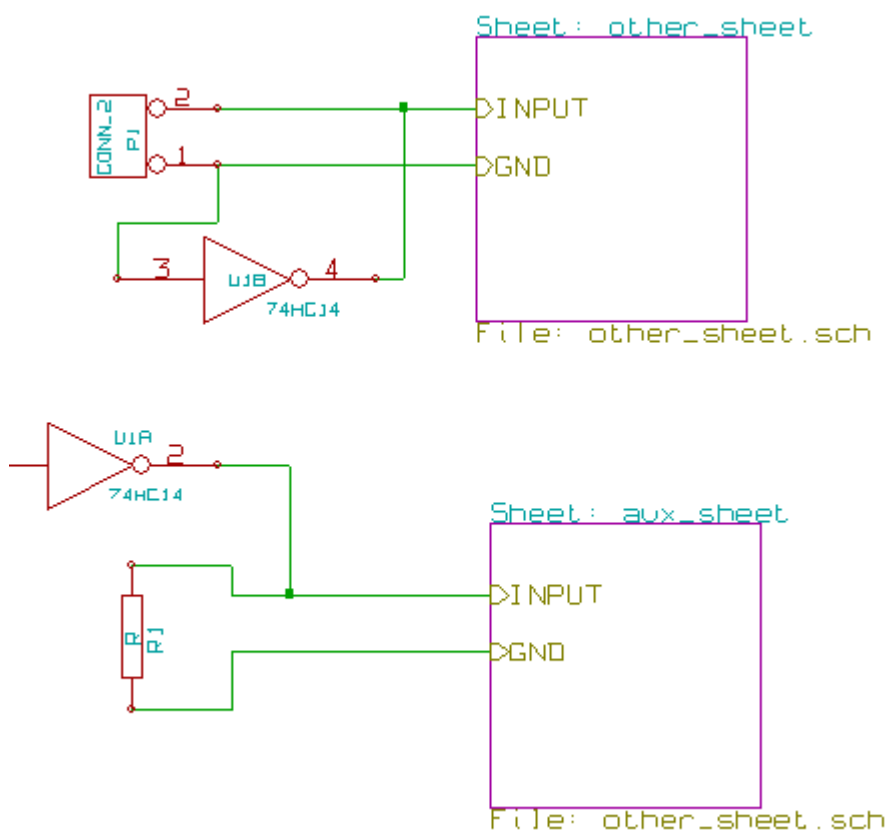
## 全局

具有相同名称的全局跨整个次构接。

（像vcc的力.....是全局

## 复次构

是一个例子。相同的原理使用两次（两个例）。两个工作表共享相同的原理因两个工作表的文件名相同（*other\_sheet.sch*）。工作表名称必须是唯一的。



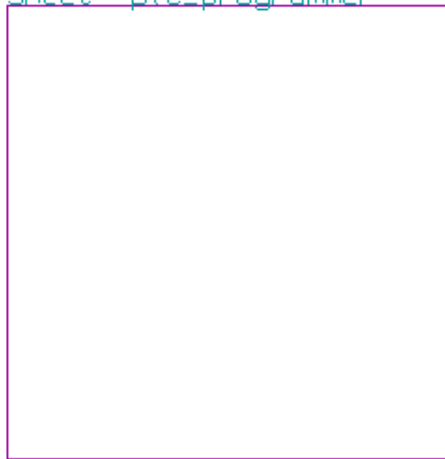
## 平面次构

如果遵守以下规则可以使用多个工作表构建项目，而无需在些工作表（平面次构）之间构建连接：

- 构建包含其他工作表的根工作表，些工作表充当其他工作表之的接口。
- 不需要明确的接口。
- 在所有工作表中使用全局而不是分

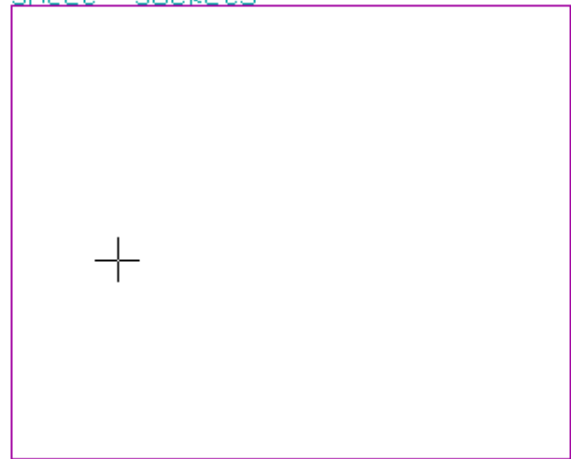
以下是根表的示例。

Sheet: pic\_programmer



File: pic\_programmer.sch

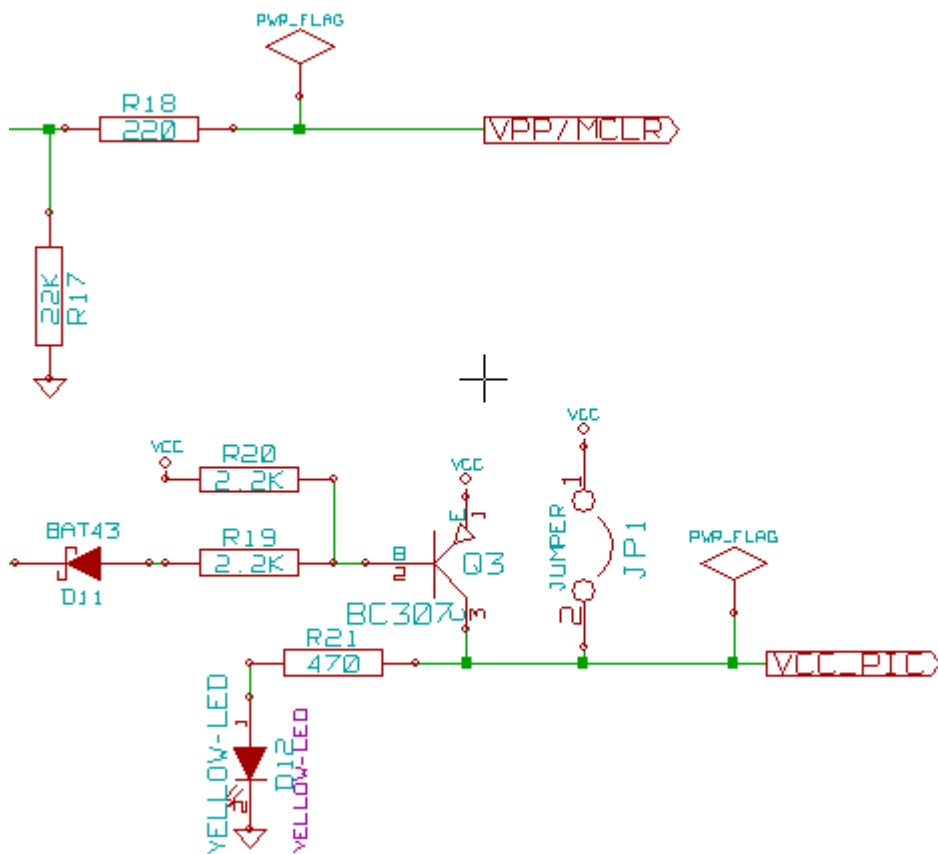
Sheet: sockets



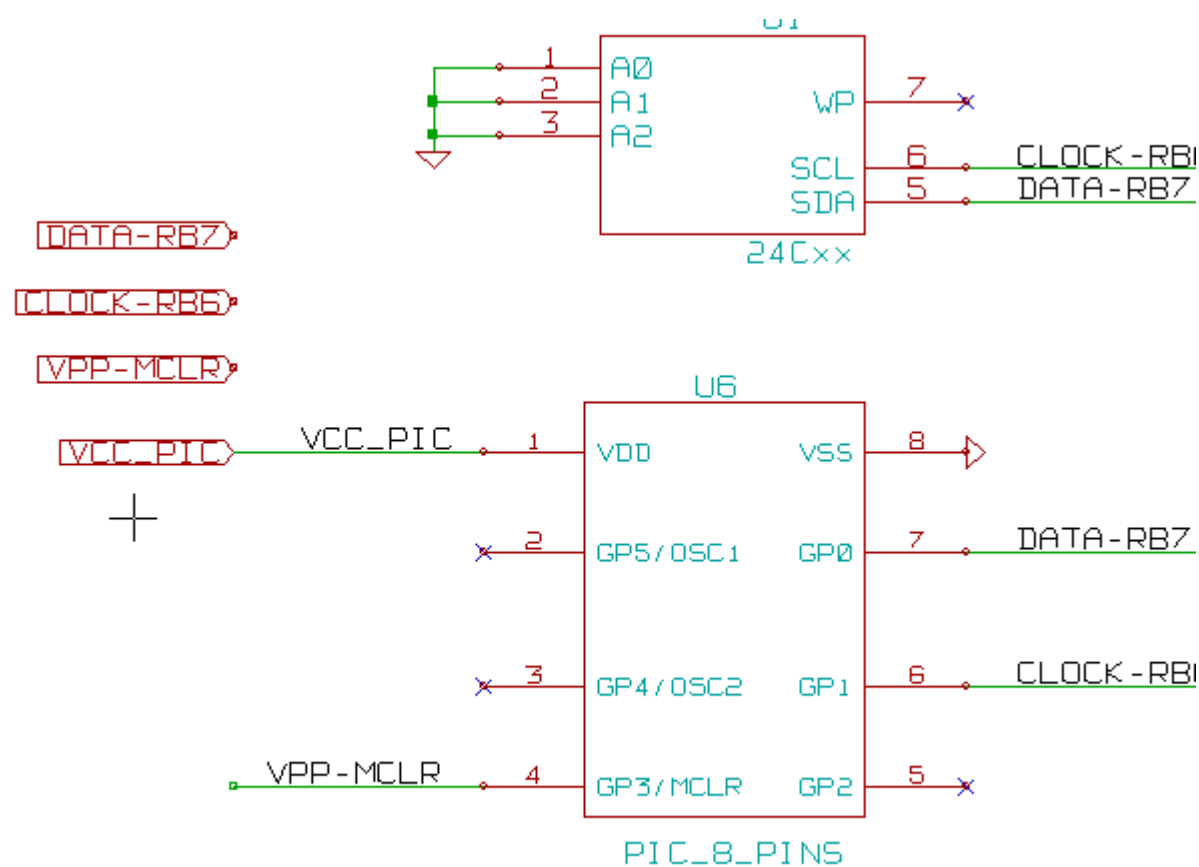
File: pic\_sockets.sch

□是两□由全局□□□接。

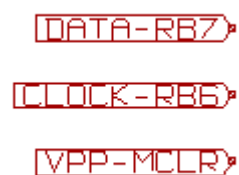
□是 pic\_programmer.sch。



□是 pic\_sockets.sch。



看全局



# 符号批注工具

## 简介

批注工具允许您自原理图中的符号指定一个指示符。具有多个位的符号的批注将分配一个唯一的后, 以最大限度地减少些符号的数量。批注批注工具可通过像: image:images/icons/annotate.png [icons\_annotate\_png]。在里, 您可以找到它的主窗口。

批注原理图

范围:

☒ 使用整个原理图

☐ 仅使用当前页面

顺序:

☒ X方向排序元件 (X)

☐ Y方向排序元件 (Y)



选项:

☒ 保持现有的批注

☐ 重置现有的批注

☐ 重置, 但保持多单元器件的顺序

编号:

☒ 使用该数字之后的编号:

☐ 参考编号X100

☐ 参考编号X1000

☐ 保持对话框打开

☐ 不要求确认

批注

清除批注

关闭

批注信息:

显示: ☒ 所有 ☒ 错误 ☒ 警告 ☒ 相关信息 ☒ 活动

保存报告文件

可用的批注方案:

- 批注所有符号 (重置有批注)
- 批注所有符号, 但不要交任何以前批注的多元元件。
- 批注当前未批注的符号。未批注的符号将具有以 "?" 字符尾的指示符。
- 批注整个次构 (使用整个原理)。
- 批注当前工作表 (使用当前)。

“重置, 但不交任何批注的多元件” 保留具有多元的符号之的所有有关例如, U22 和 U2A 可能分重新批注 U1B 和 U1B, 但它永不会重新批注到 U1B 和 U2A, 也不会重新批注 U2B 和 U2A。如果要确保保持引脚分, 很有用。

批注序提供了用于在次构的每个工作表内置参考号的方法。

除特定情况外, 如果您不想修改以前的批注, 自批注将用于整个目 (所有工作表) 和新元件。

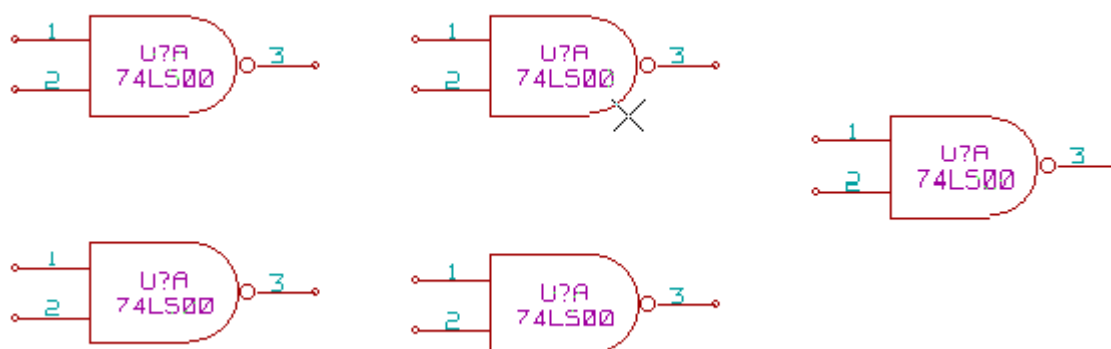
“批注” 出了用于计算参考的方法:

- 在原理中使用第一个空号: 元件从 1 开始批注 (于每个引用前 如果存在先前的批注, 使用未使用的数字。
- 从号 \*100 开始并使用第一个空号: 从 1 的 101 开始批注, 从 2 的 201 开始, 等等。如果在工作表内有超 99 个具有相同参考前 (U, R) 的目 在 1 中, 批注工具使用数字 200 和更多, 并且工作表 2 的批注将从下一个空号开始。
- 从表格号 \*1000 开始并使用第一个空号。于 1, 批注从 1001 开始, 从 2 的 2001 开始。

## 一些例子

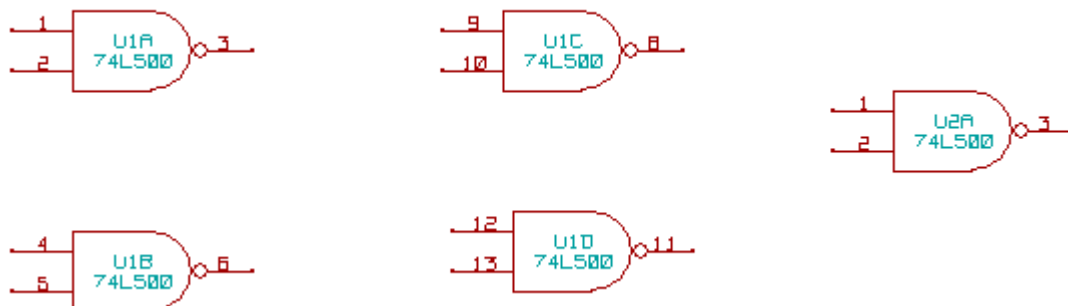
### 批注序

此示例示放置了 5 个元素, 但未批注。

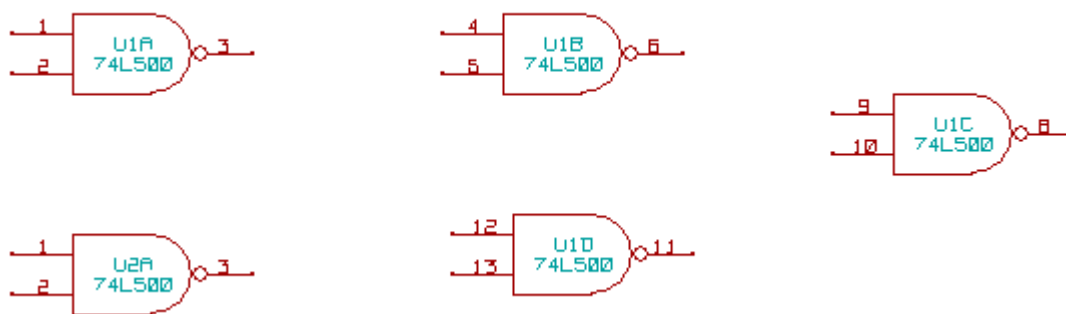


行批注工具后, 得以下果。

按 X 位置排序。



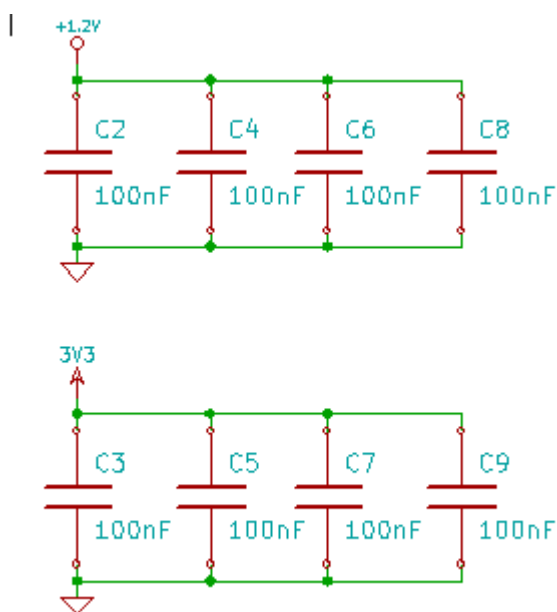
按 Y 位置排序。



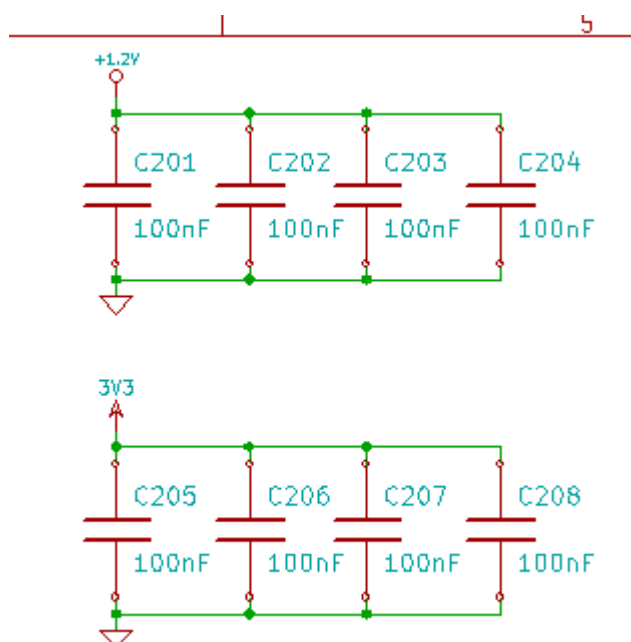
您可以看到四个 74LS00 分布在 U1 包中，第五个 74LS00 已分配给下一个 U2。

## 批注

是表 2 中的批注，其中使用原理中的第一个空号。

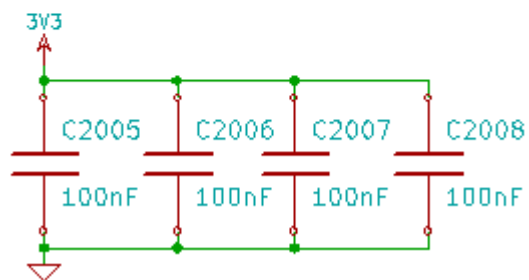
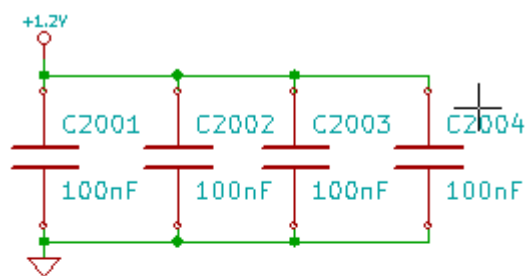


开始到工作表号 \*100 并使用第一个空号出以下果。





从开始到工作表A1 \*1000并使用第一个空单元格输出以下结果。

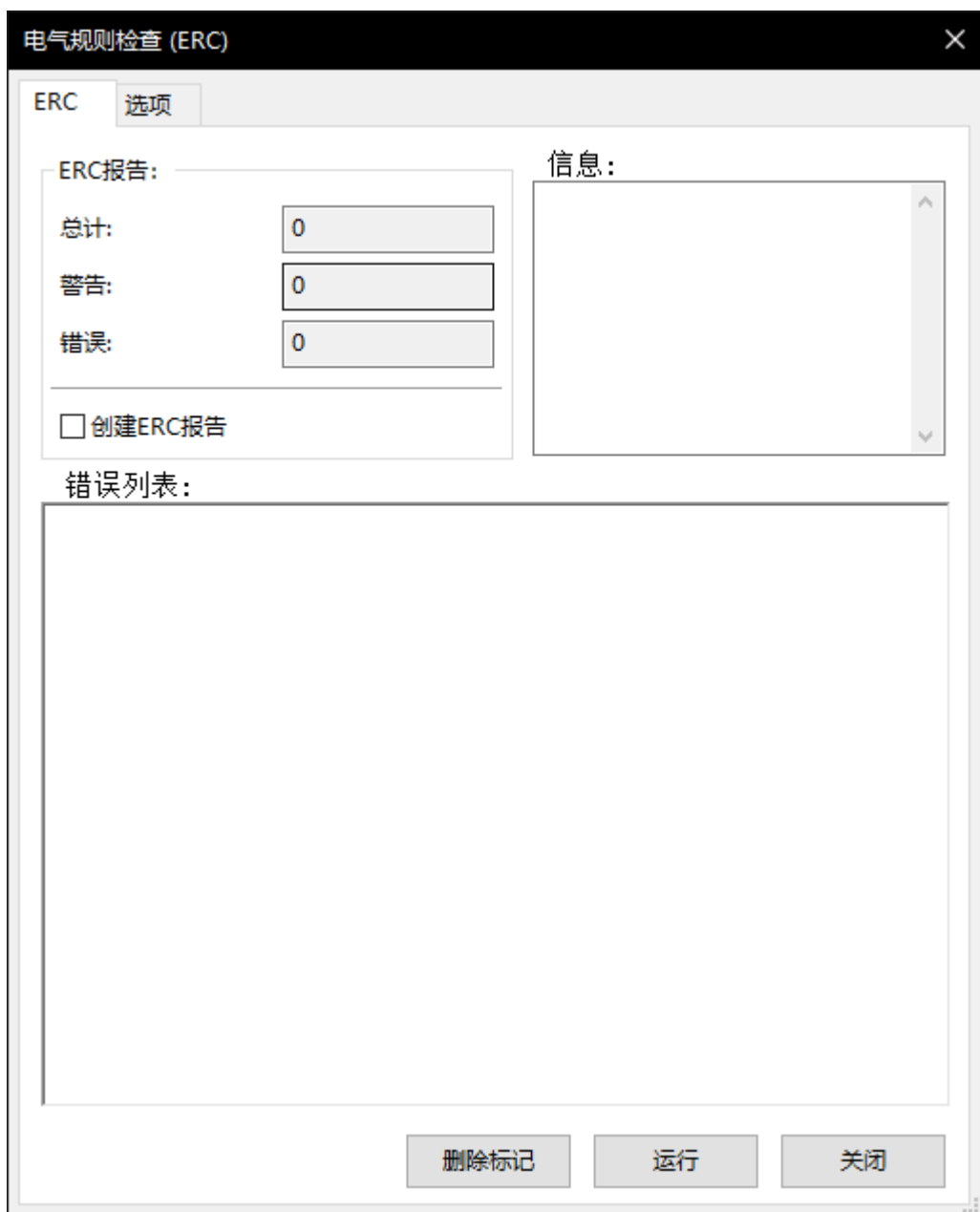


# 使用电气规则检查

## 简介

电气规则检查(ERC)工具会自原理图ERC工作表中的任何例如未连接的引脚，未连接的分号，短路等。当然，自不是可靠的，并且可以所有设计的件不是100%完成。的非常有用，因为它允许多疏忽和小

上，必须所有到的然后在正常行之前行正。ERC的量与在符号建设期声明引脚属性所采取的慎直接相关。ERC出告或警告。



## 如何使用 ERC

像可以后 ERC : images/icons/erc.png[ERC icon]。

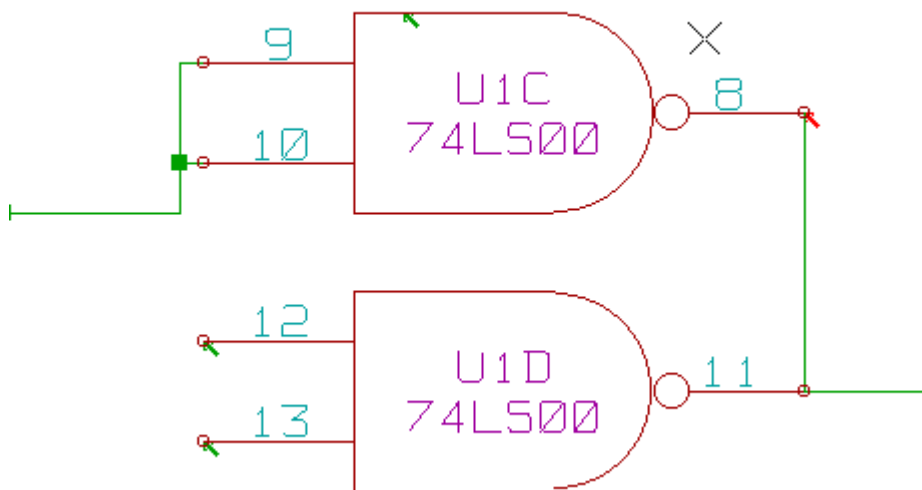
在原理图元素上置警告, 以引 ERC (引脚或)。

## NOTE

- 在此对话框中，消息您可以跳到原理图中的相应位置。
- 在原理图中，右击以显示断消息。

您可以从框中删除。

## ERC 的示例

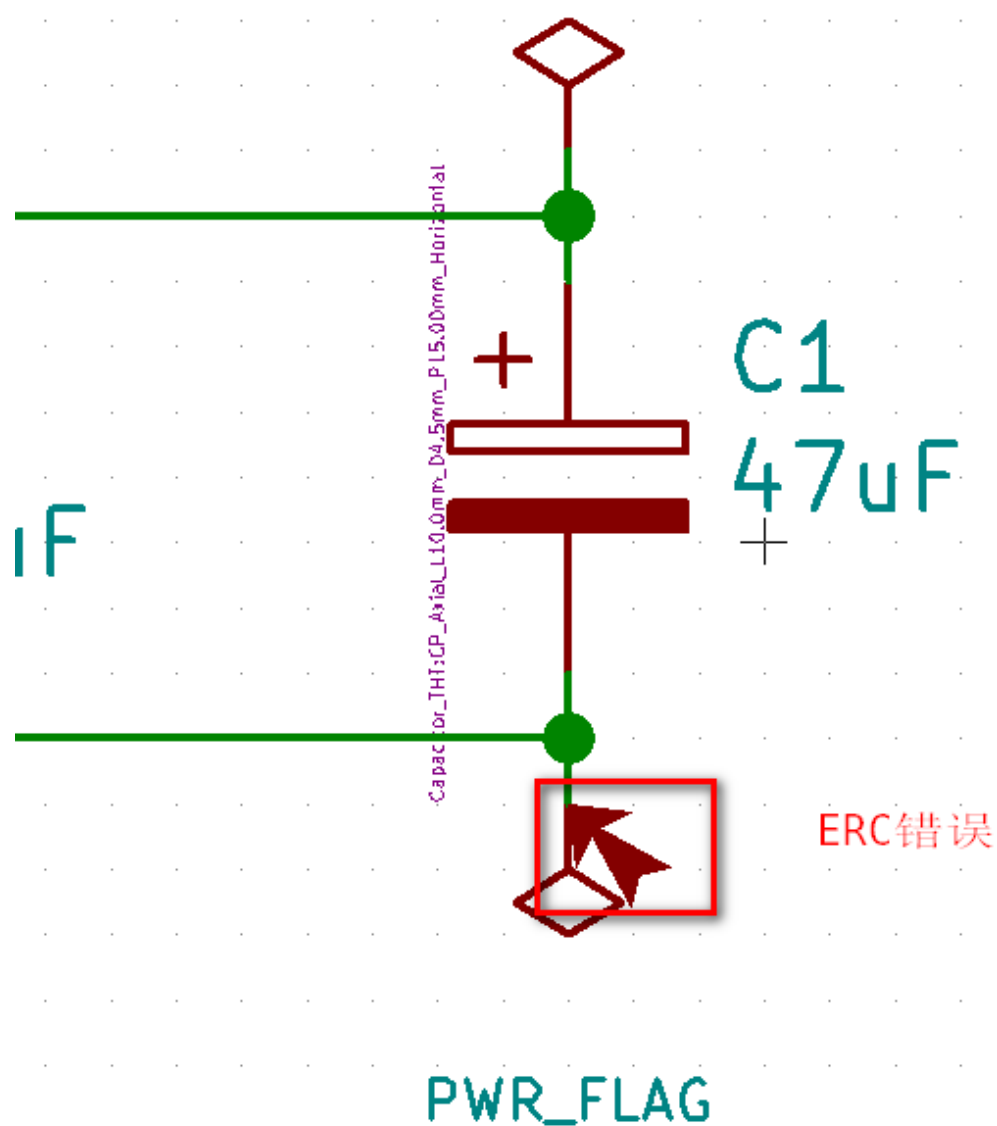


在这里，您可以看到四个：

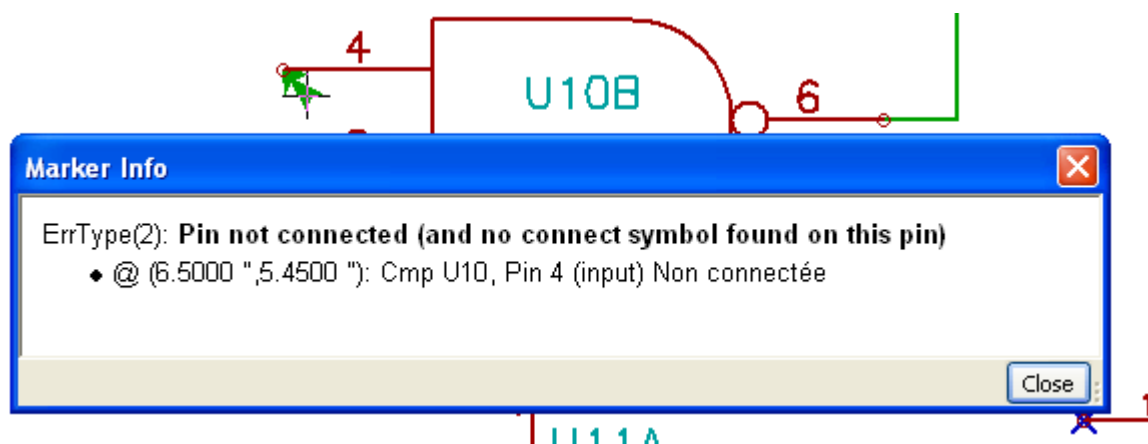
- 两个输出端接在一起（红色箭头）
- 两个输入未接（红色箭头）
- 隐藏源端口输出缺少源标志（部分红色箭头）

## 示例断

通过右击输出菜允许您 ERC 断窗口。



当信息您可以得的描述。

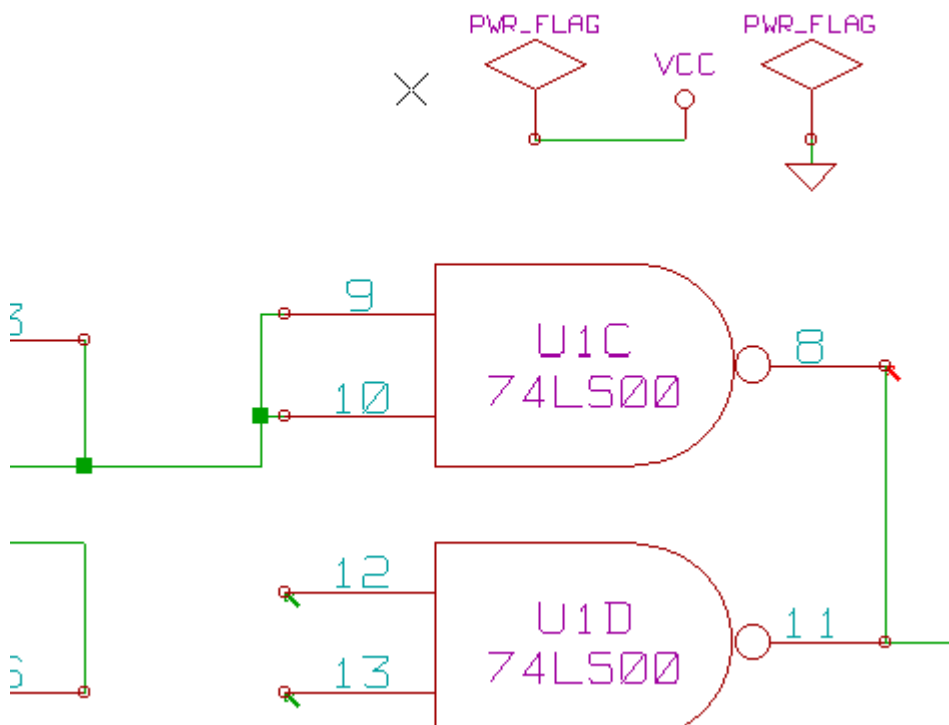


## 源引脚和源标志

通常在源引脚上出错误或警告，即使一切看起来都很正常。上面的例子。之所以会产生这种情况，是因为在大多数设计中，源是由不是源的连接器提供的（如器件输出，它被声明为源输出）。

因此，ERC 不会检测到任何源输出引脚来控制并声明它不是由源输出的。

要避免此警告，您必须在此源端口上放置“PWR\_FLAG”。看一下下面的例子：

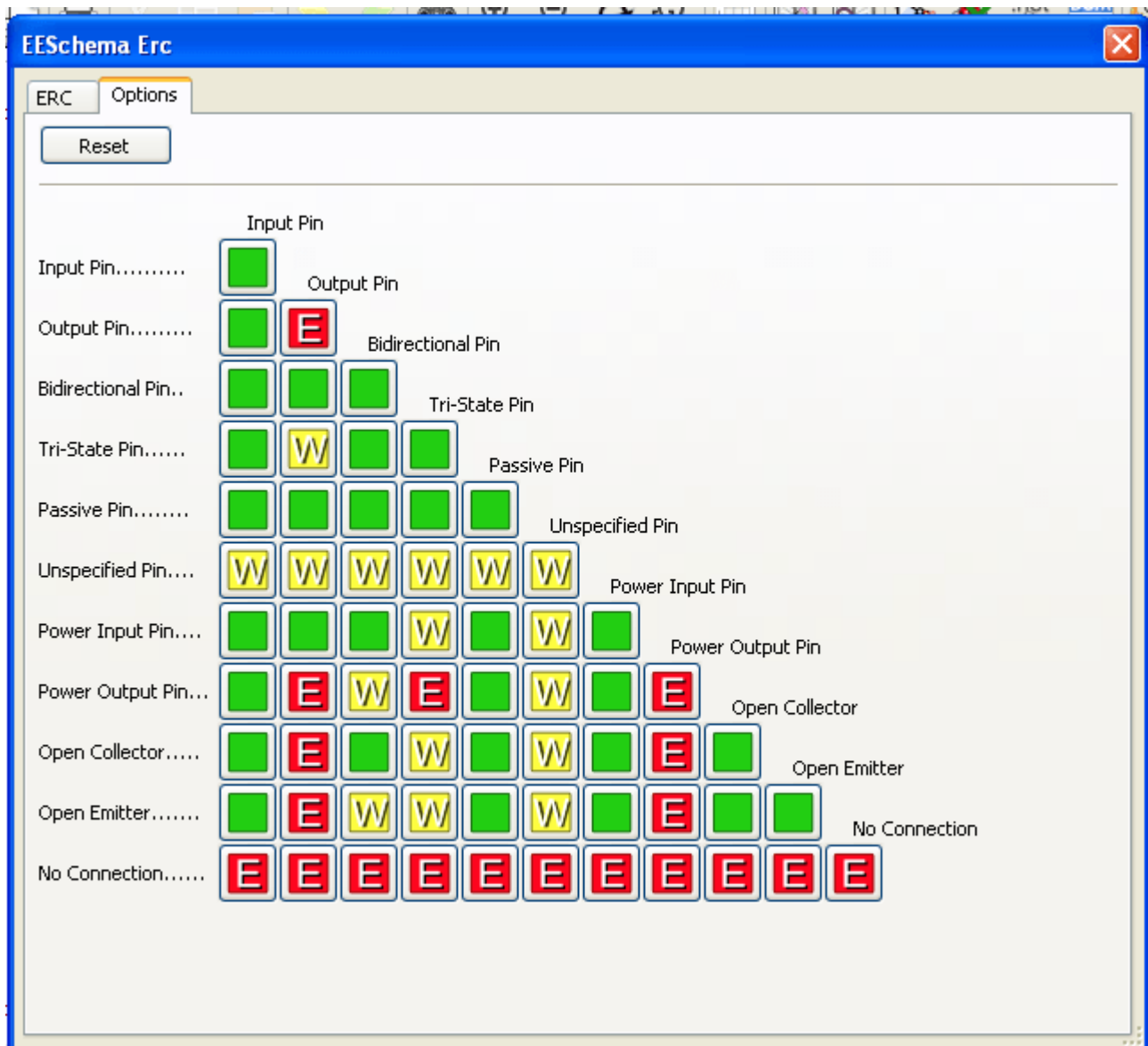


然后警告将消失。

大多数情况下，PWR\_FLAG 必须连接到 GND，因为器件的输出声明为源但接地引脚永远不会断（正常属性是源输入），因此，在没有源标志的情况下，接地不会与源相连。

## 配置

面板允许您配置接口以定义警告的气条件。



□□所需的□元格方□可以更改□□□使其循□□□□正常，警告，□□□

## ERC □告文件

通□□中写入 ERC □告□□□可以生成并保存 ERC □告文件。ERC □告文件的文件□展名□.erc。以下是 ERC □告文件的示例。

```
ERC control (4/1/1997-14:16:4)
```

```
***** Sheet 1 (INTERFACE UNIVERSAL)
```

```
ERC: Warning Pin input Unconnected @ 8.450, 2.350
```

```
ERC: Warning passive Pin Unconnected @ 8.450, 1.950
```

```
ERC: Warning: BiDir Pin connected to power Pin (Net 6) @ 10.100, 3.300
```

```
ERC: Warning: Power Pin connected to BiDir Pin (Net 6) @ 4.950, 1.400
```

```
>> Errors ERC: 4
```

# 建网列表

## 概述

网表是描述符号之间的连接的文件。有些连接称网表 在网表文件中，您可以找到：

- 符号列表
- 符号之间的连接 (网) 列表。


存在多种不同的网表格式。有符号列表和网表列表是两个单独的文件。网表是使用原理图捕获件的基因为网表是与其他子 CAD 文件的连接，例如：

- PCB 布局件。
- 原理图和信号模拟器。
- CPLD（和其他可编程 IC 器件）。

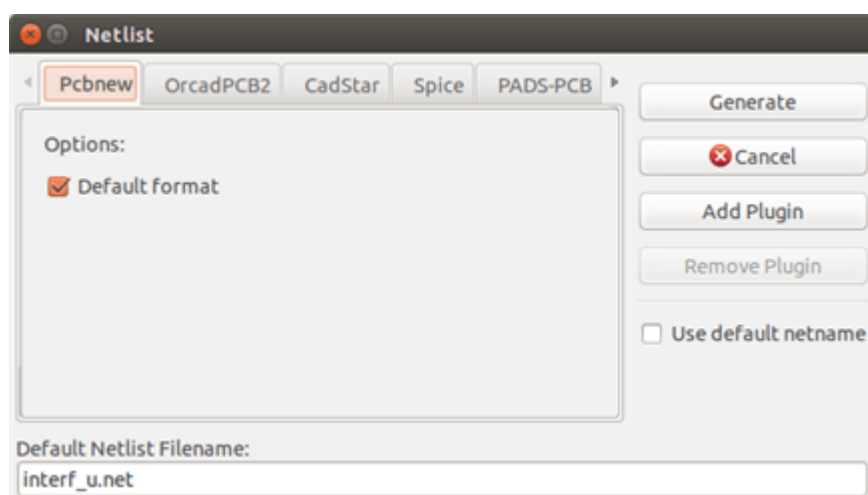
Eeschema 支持几种网表列表格式。

- PCBNEW 格式（印刷电路）。
- ORCAD PCB2 格式（印刷电路）。
- CADSTAR 格式（印刷电路）。
- Spice 格式，适用于各种模拟器（其他模拟器也使用 Spice 格式）。

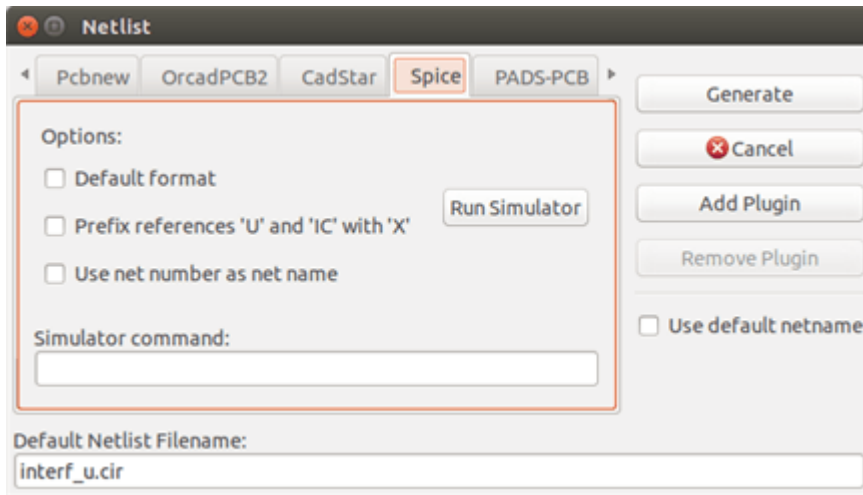
## 网表格式

工具  以打开网表构建框。

的 Pcbnew



的 Spice

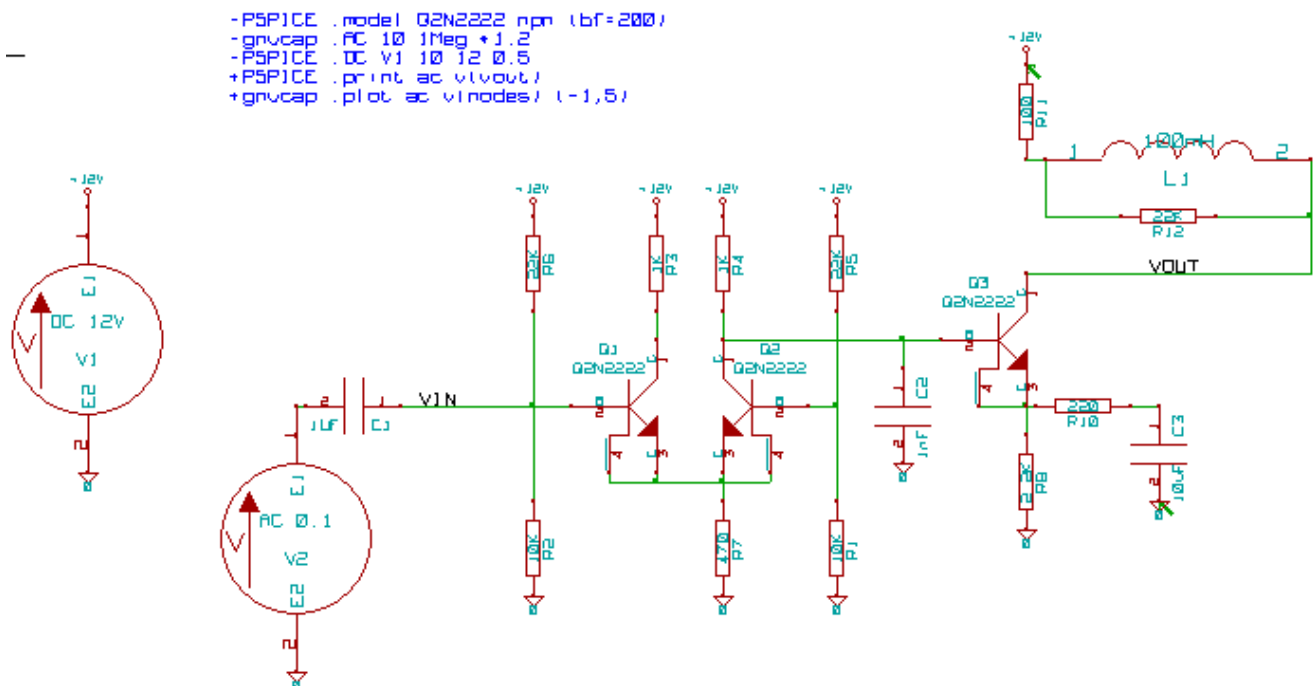


使用不同的卡，您可以所需的格式。在 Spice 格式中，您可以使用网名称生成网表，使得 SPICE 文件更易于或者使用旧版 Spice 使用的网号。通“网表”按钮将要求您提供网表文件名。

**NOTE** 于大型原理，网表生成最多可能需要几分钟

## 网表示例

您可以在下面看到使用 PSPICE 的原理计：



PCBNEW 网表文件示例：



```

# Eeschema Netlist Version 1.0 genereee le 21/1/1997-16:51:15
(
(32E35B76 $noname C2 1NF {Lib=C}
(1 0)
(2 VOUT_1)
)
(32CFC454 $noname V2 AC_0.1 {Lib=VSOURCE}
(1 N-000003)
(2 0)
)
(32CFC413 $noname C1 1UF {Lib=C}
(1 INPUT_1)
(2 N-000003)
)
(32CFC337 $noname V1 DC_12V {Lib=VSOURCE}
(1 +12V)
(2 0)
)
(32CFC293 $noname R2 10K {Lib=R}
(1 INPUT_1)
(2 0)
)
(32CFC288 $noname R6 22K {Lib=R}
(1 +12V)
(2 INPUT_1)
)
(32CFC27F $noname R5 22K {Lib=R}
(1 +12V)
(2 N-000008)
)
(32CFC277 $noname R1 10K {Lib=R}
(1 N-000008)
(2 0)
)
(32CFC25A $noname R7 470 {Lib=R}
(1 EMET_1)
(2 0)
)
(32CFC254 $noname R4 1K {Lib=R}
(1 +12V)
(2 VOUT_1)
)
(32CFC24C $noname R3 1K {Lib=R}
(1 +12V)
(2 N-000006)
)
(32CFC230 $noname Q2 Q2N2222 {Lib=NPN}
(1 VOUT_1)
(2 N-000008)
(3 EMET_1)
)
(32CFC227 $noname Q1 Q2N2222 {Lib=NPN}
(1 N-000006)
(2 INPUT_1)
(3 EMET_1)
)
)
# End

```

在 PSPICE 格式中，网表如下：

```
* Eeschema 网表 1.1 版(Spice 格式)  建日期：18/6/2008-08:38:03

.model Q2N2222 npn (bf=200)
.AC 10 1Meg \*1.2
.DC V1 10 12 0.5

R12  /VOUT N-000003 22K
R11  +12V N-000003 100
L1   N-000003 /VOUT 100mH
R10  N-000005 N-000004 220
C3   N-000005 0 10uF
C2   N-000009 0 1nF
R8   N-000004 0 2.2K
Q3   /VOUT N-000009 N-000004 N-000004 Q2N2222
V2   N-000008 0 AC 0.1
C1   /VIN N-000008 1uF
V1   +12V 0 DC 12V
R2   /VIN 0 10K
R6   +12V /VIN 22K
R5   +12V N-000012 22K
R1   N-000012 0 10K
R7   N-000007 0 470
R4   +12V N-000009 1K
R3   +12V N-000010 1K
Q2   N-000009 N-000012 N-000007 N-000007 Q2N2222
Q1   N-000010 /VIN N-000007 N-000007 Q2N2222

.print ac v(vout)
.plot ac v(nodes) (-1,5)

.end
```

## 关于网表的说明

### 网表名称注意事项

许多使用网表的工具不接受元件名称、引脚、网表或其他信息中的空格。避免在网表中使用空格, 或元件或其引脚的名称和引脚字段, 以确保最大程度的兼容性。

同网表字母和数字以外的特殊字符也可能导致问题。注意，此限制与 Eeschema 无关，而是与网表格式无关，后者可以忽略不可识别的字符使用网表文件的元件。

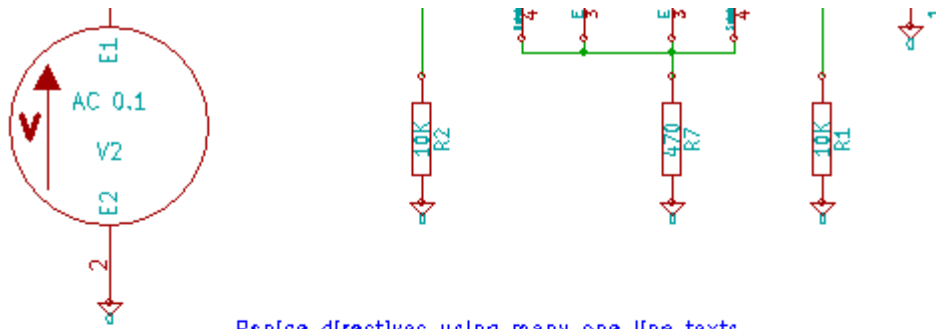
### PSPICE 网表

对于 Pspice 模拟器，您必须在网表本身（.PROBE，.AC 等）中包含一些命令行。

从以关键字 **-pspice** 或 **-gnuap** 开始的示意图中包含的任何文本行都将在网表的开头插入（不相关关键字）。

以关键字 **+pspice** 或 **+gnuap** 开头的示意图中包含的任何文本行都将在网表的末尾插入（不相关关键字）。

下面是一个使用多行文本和一个多行文本的示例:



Pspice directives using many one line texts

```
-PSPICE .model Q2N2222 npn (bf=200)
-gnucap .AC dec 10 1Meg *1.2
-PSPICE .DC V1 10 12 0.5
+PSPICE .print ac v(vout)
+gnucap .plot ac v(nodes) (-1.5)
```

Pspice directives using one multiline text:

```
+PSPICE .model NPN NPN
.model PNP PNP
.lib C:\Program Files\LTC\LTspiceIV\lib\cmp\standard.bjt
.backanno
```

例如，如果您输入以下文本（不要使用空行）

```
-PSPICE .PROBE
```

一行 .PROBE 将被插入网表中。

在前面的例子中，在网表的开头插入了三行，用种技插入了两行。

如果您使用的是多行文本，只需要 +pspice 或 +gnucap 关键字一次：

```
+PSPICE .model NPN NPN
.model PNP PNP
.lib C:\Program Files\LTC\LTspiceIV\lib\cmp\standard.bjt
.backanno
```

建四行:

```
.model NPN NPN
.model PNP PNP
.lib C:\Program Files\LTC\LTspiceIV\lib\cmp\standard.bjt
.backanno
```

另注意，Pspice 的 GND 网必须命名 0（零）。

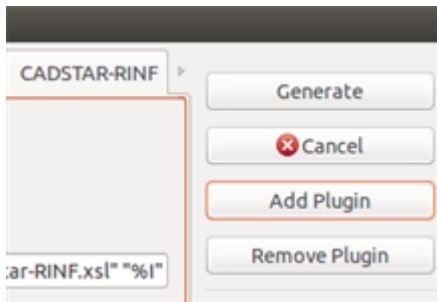
## 其他格式

对于其他网表格式，您可以以插件的形式添加网表器。一些器由 Eeschema 自后第14章出了器的一些解和示例。

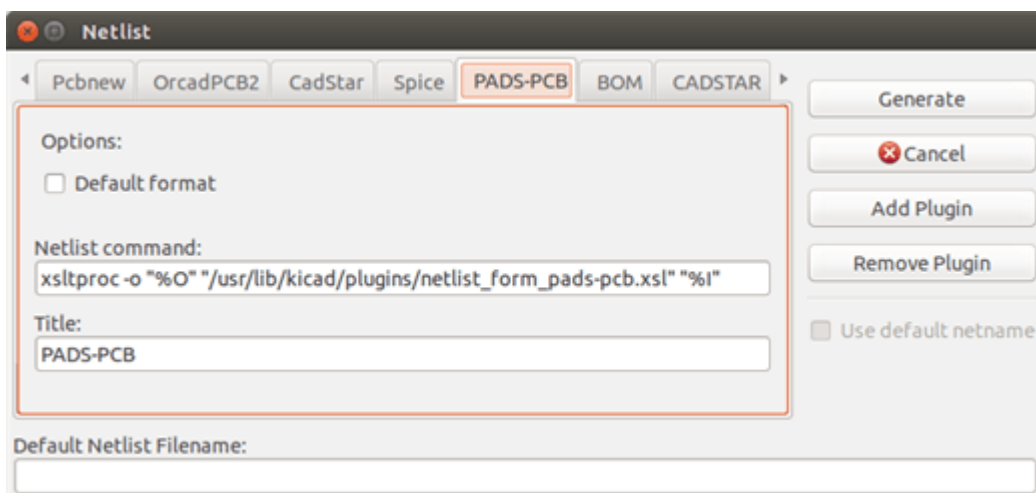
器是一个文本文件（xsl 格式），但可以使用其他言，如 Python。使用 xsl 格式工具（xsltproc.exe 或 xsltproc 取 Eeschema 建的中文件和器文件以建出文件。在这种情况下，器文件（工作表式）非常小并且非常容易写。

## 在框窗口中

您可以通过 添加插件 按钮添加新的网表插件。



是插件 PadsPcb 置窗口：



置将需要：

- 例如，网表格式的名称）。
- 要后的插件。

生成网表

1. Eeschema 建一个中文件 \*.tmp，例如 test.tmp。
2. Eeschema 运行插件，取 test.tmp 并建 test.net。

## 命令行格式

下面是一个示例，使用 xsltproc.exe 作 .xsl 文件的工具，使用文件 netlist\_form\_pads-pcb.xml 作表式：

```
f:/kicad/bin/xsltproc.exe -o %O.net f:/kicad/bin/plugins/netlist_form_pads-pcb.xml %I
```

附：

f:/kicad/bin/xsltproc.exe	A tool to read and convert xsl file
-o %O.net	Output file: %O will define the output file.
f:/kicad/bin/plugins/netlist_form_pads-pcb.xsl	File name converter (a sheet style, xsl format).
%I	Will be replaced by the intermediate file created by Eeschema (*.tmp).

由于名 test.sch 的原理的命令行是：

```
f:/kicad/bin/xsltproc.exe -o test.net f:/kicad/bin/plugins/netlist_form_pads-pcb.xsl test.tmp.
```

器和工作表式 (插件)

是一个非常的件，因它的目的只是将入文本文件（中文本文件）另一个文本文件。此外，从中文本文件中，您可以建 BOM 清

使用 xsltproc 作器工具生成工作表式。

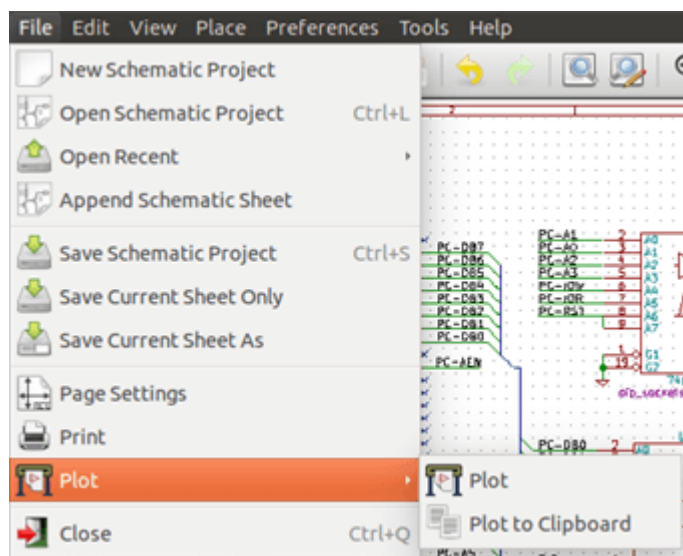
中文表文件格式

有关 xslproc 的更多明，中文文件格式的明以及器的工作表式的一些示例，参第14章。

# □□和打印

## □介

您可以通□文件菜□□□打印和□□命令。



支持的□出格式是 Postscript, PDF, SVG, DXF 和 HPGL。您也可以直接打印到您的打印机。

## 常□的打印命令

### □制当前□面

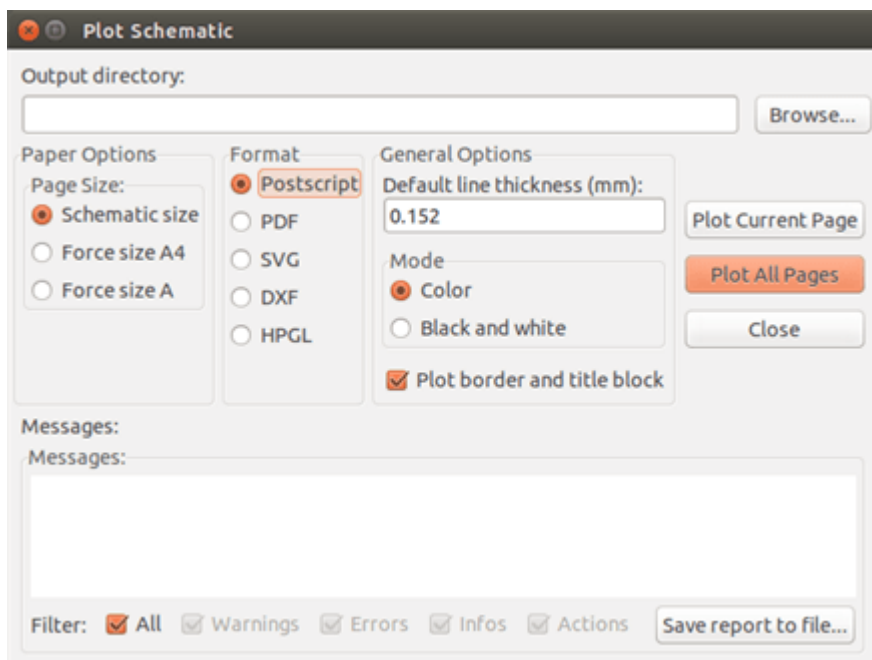
□打印当前工作表的一个文件。

### □制所有□面

允□您□制整个□次□构（□每个工作表生成一个打印文件）。

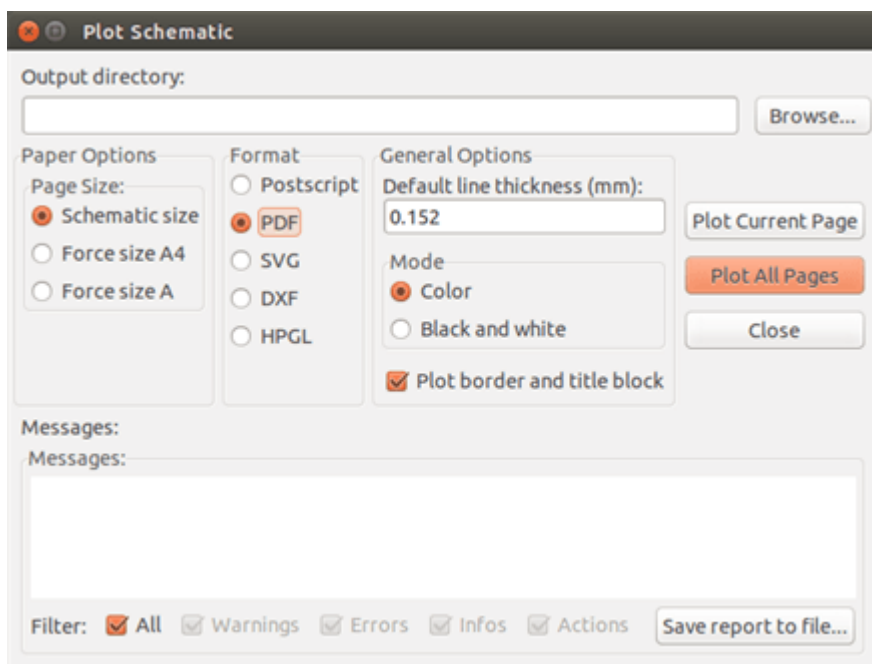
## 在 Postscript 中□制

此命令允□您□建 PostScript 文件。



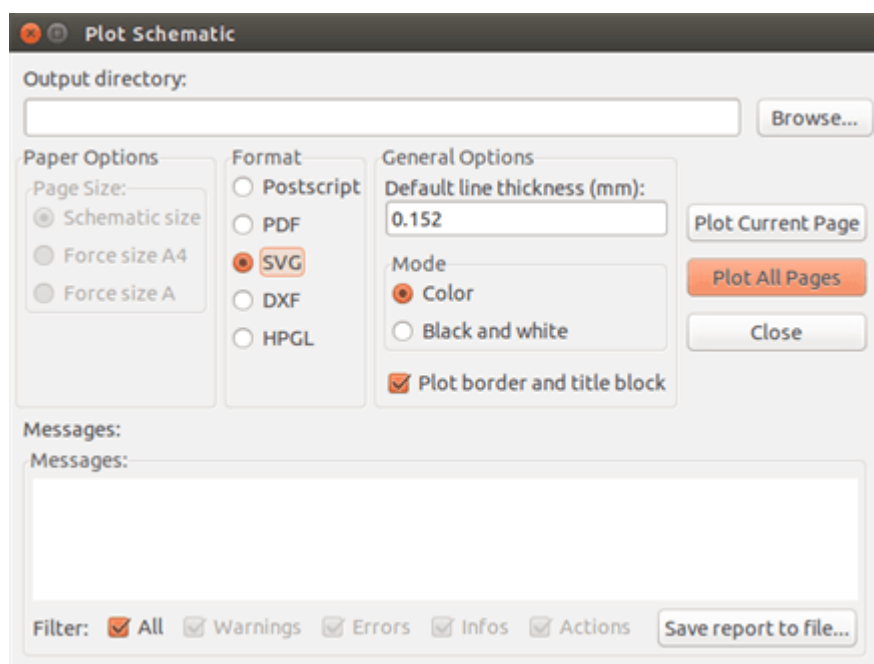
文件名是“展名”.ps 的工作表名称。您可以禁用“制框和” 如果要建用于封装的 postscript 文件（格式 .eps 通常用于在文字理件中插入表，非常有用。消息窗口示建的文件名。

## 以 PDF 格式制



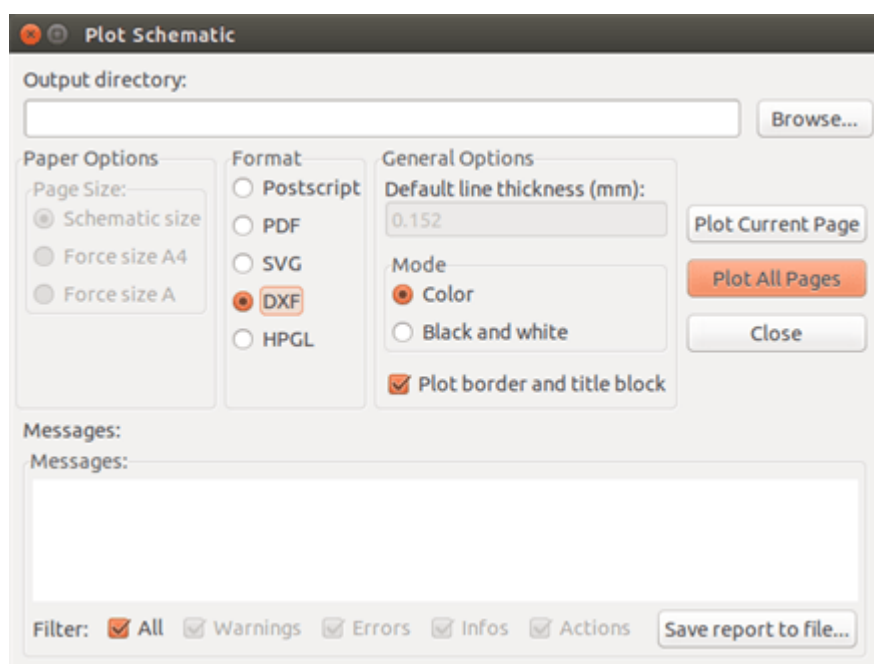
允您使用 PDF 格式建打印文件。文件名是“展名”.pdf 的工作表名称。

## 在 SVG 中



允许您使用矢量格式 SVG 创建打印文件。文件名是展名.svg 的工作表名称。

## 在 DXF 中



允许您使用 DXF 格式创建打印文件。文件名是展名.dxf 的工作表名称。

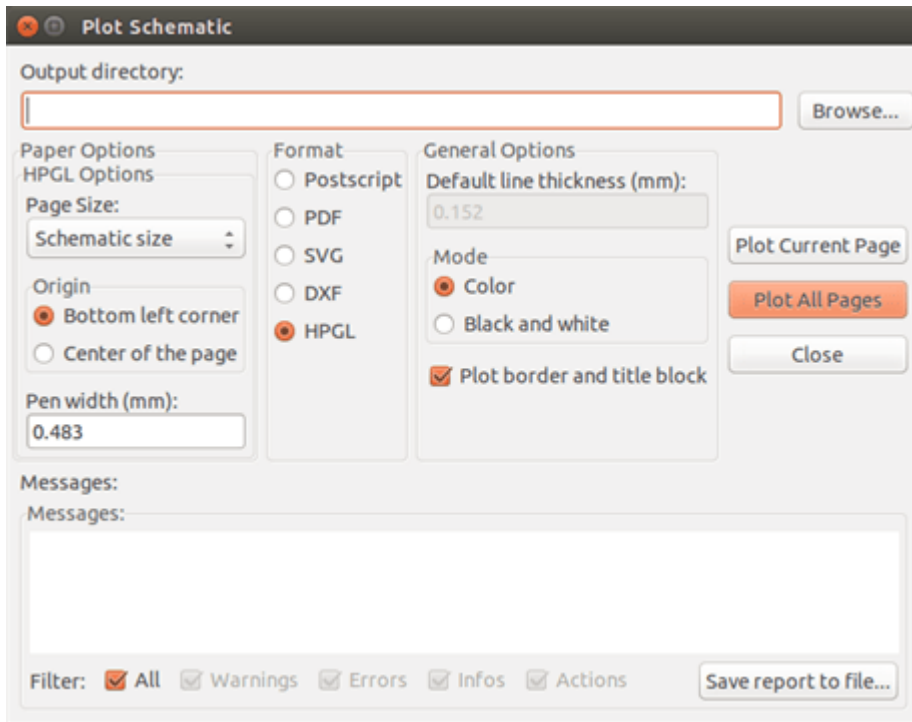
## 在 HPGL 中

此命令允许您创建 HPGL 文件。在这种格式中，您可以定义：

- 页面大小。
- 原点。
- 笔宽(mm)。



□□□□置□□框窗口如下所示：



□出文件名将是工作表名称加上□展名 .plt。

## □□尺寸□□

通常□□□□尺寸。在□种情况下，将使用□□□菜□中定义的□□尺寸，并且所□的比例将□1.如果□□了不同的□□尺寸（A4 □ A0，或 A □ E□□□自□□整比例以填充□面。

## 偏移□整

□于所有□准尺寸，您可以□整偏移以尽可能准确地使□形居中。由于□□□在工作表的中心或左下角有原点，因此必□能□引入偏移以便正确□□□

一般来□□

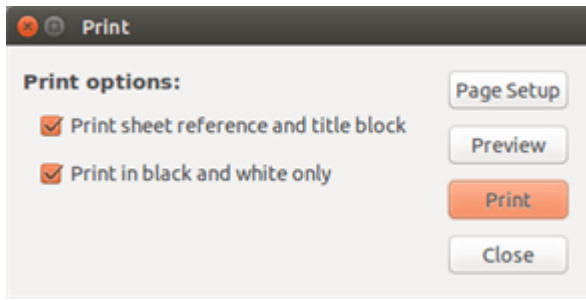
- □于原点位于□□中心的□□□□偏移量必□□□□并□置□□□尺寸的一半。
- □于原点位于□□左下角的□□□□偏移量必□□置□□□

要□置偏移量：

- □□□□尺寸。
- □置偏移量 X 和偏移量 Y.
- □□接受偏移量。

## 在□上打印

此命令可通□□□□得，允□您可□化并生成□准打印机的□计文件。



“打印表格参考和标题” 可启用或禁用表格参考和标题

“黑白打印” 置为色打印。如果使用黑白激光打印机，通常需要此选项因色打印成半色通常不太可能

# 符号管理器

## 关于符号的一般信息

符号是一个示意元素，包含图形表示，电气接口和定义符号的字段。原理图中使用的符号存储在符号库中。Eeschema 提供了一个符号库工具，允许您在库中创建、添加、删除或重命名符号，将符号导出到文件以及从文件导入符号。该工具提供了一种管理符号库文件的方法。

## 符号概述

符号由一个或多个符号组成。通常，符号按功能，类型和/或制造商进行分类。

符号由以下部分组成：

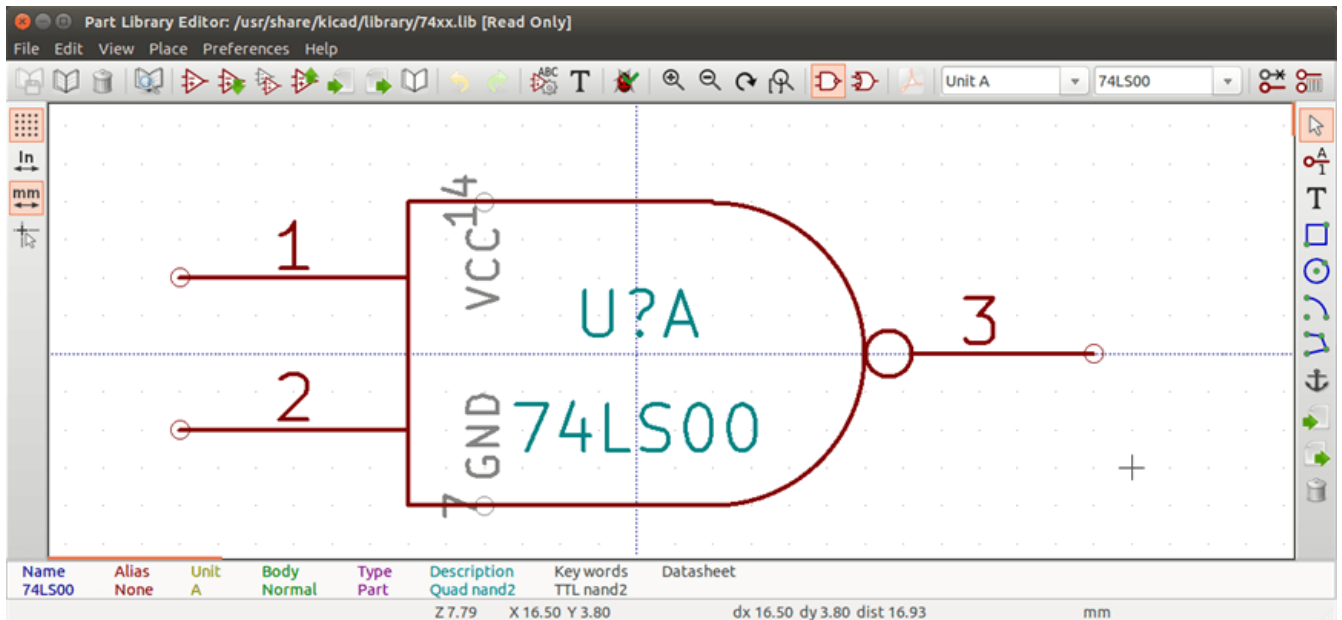
- 提供符号定义的图形元素（弧，文本等）。
- 具有图形属性（例如反低电平有效等）和电气属性（ERC）工具使用的电气属性（输入，输出，双向等）的引脚。
- 例如参考，PCB 设计的封装名称等字段等。
- 名称用于将常用符号（如 7400）与其所有衍生物（如 74LS00、74HC00 和 7437）相关。所有名称共享相同的符号。

正确的符号设计需要：

- 定义符号是否由一个或多个单元组成。
- 定义符号是否具有替代的体型，也称为 De Morgan 表示。
- 使用线条、矩形、圆形、多边形和文本设计其符号表示。
- 通过定义每个引脚的图形元素，名称、编号和电气属性（输入，输出，三态源端口等）来添加引脚。
- 如果其他符号具有相同的设计，添加名称，如果已从其他符号创建符号，将其固定或删除。
- 添加可选项字段，例如 PCB 设计件使用的封装名称和/或定义其可选项性。
- 通过添加描述字符串和数据表链接等来丰富符号。
- 将其保存在所需的库中。

## 符号管理器概述

符号管理器主窗口如下所示。它由三个工具组成，可快速访问常用功能和符号查看/编辑区域。并非所有命令都可在工具栏上使用，但可以使用菜单。



## 主工具

通常位于主窗口的主工具包括管理工具，撤消/重做命令，放命令和符号属性框。



	保存当前定的如果没有，按将被禁用 当前已中或当前未更改 已制作完成。
	要
	从当前定的或任何中除符号 如果当前没有由目定义。
	打开符号器以和符号
	建一个新符号。
	从当前定的中加符号以行
	从当前加的符号建新符号。
	将当前符号更改保存在内存中。文件不是 改
	从文件中入一个符号。
	将当前符号出到文件。
	建包含当前符号的新文件。注意：新的不会自添加到目中。
	撤消上次
	重做最后一次撤消。
	当前符号属性。

	显示当前符号的字段。
	显示当前符号是否存在设计错误
	放大。
	缩小。
	刷新显示。
	缩放以适合显示中的符号。
	显示正常的体型。如果当前，该按钮被禁用 符号没有替代的体型规格。
	显示用体型。如果当前，该按钮被禁用 符号没有替代的体型规格。
	显示相关文档。如果没有，该按钮将被禁用 文档是当前符号定义的。
	显示要显示的位。如果，将禁用下拉控件 当前符号不是从多个单元派生的。
	显示名称。如果是，下拉控件将被禁用 当前符号没有任何名称。
	引脚具有多个位和替代符号的符号。
	显示引脚表。

## 元素工具

垂直工具通常位于主窗口的右侧，允许您放置设计符号所需的所有元素。下表定义了每个工具按钮

	选择工具。使用选择工具右单击可打开上下文菜单于光标下的对象。使用选择工具左单击在消息中指示光下对象的属性。主窗口底部的面板。双击左单击工具将打开对象下的属性面板。
	引脚工具。左单击以添加新引脚。
	文本工具。左单击以添加新的文本。
	矩形工具。左单击以开始绘制第一个角。再次左单击以放置角。矩形。
	圆形工具。左单击以开始绘制新的圆形。中心。再次左单击以定义圆的半径。
	弧工具。左单击以开始从中绘制新的弧形。中央。再次左单击以定义第一个弧点。左单击再次定义第二个弧点。
	多边形工具。左单击以开始绘制新的多边形。在当前的符号。左单击每个添加多边形。双击左单击以完成多边形。
	点工具。左单击以设置符号的点位置。
	从文件导入符号。
	将当前符号导出到文件。
	删除工具。左单击以从当前符号中删除对象。

## 工具

垂直工具通常位于主窗口的左允许您配置一些设备。下表定义了每个工具按钮。

	打开和关闭网格可见性。
	将单位设置为英寸。
	将单位设置为毫米。
	打开和关闭全屏模式。

## 与

可以通过当前它会显示所有可用并允许您添加或保存符号。它将被放入此中。符号的名称是其 value 字段的内容。

### NOTE

- 您必须将添加到 Eeschema 中才能其内容。
- 通过主工具上的，可以在修改后保存当前的内容。
- 通像可以从任何中删除符号：。


## 并保存符号

当您符号您上并不是在其中的符号上工作，而是在计算机内存中的符号上复制它。任何操作都可以松撤消。可以从本地或有符号加符号。



### 符号

主工具上的  将示可从当前所中中和加的可用符号列表。

#### NOTE


如果通名了符号，加的符号的名称将示在窗口上，而不是示在定的名上。符号名列表始随每个符号一起加并且可以行。您可以通过从像中当前符号的名来建新符号：。名列表中的第一是符号的根名称。

#### NOTE

或者， 可以加先前由像保存的符号：。

## 保存符号

修改后，符号可以保存在当前中，新中，也可以出到份文件中。

要将修改后的符号保存在当前中，。注意，更新命令将符号更改保存在本地内存中。您可以在恢复之前做出决定。


要将符号更改永久保存到文件，，它将覆盖有文件并更改符号。

如果要建包含当前符号的新，。系将要求您入新的名称。

#### NOTE






新不会自添加到当前目中。

您必使用《manage-sym-lib-table, Symbol Library Table框》将您希望在原理中使用的任何新添加到 Eeschema 中的目列表中。

 以建包含当前符号的文件。文件将是一个准文件，它只包含一个符号。此文件可用于将符号入另一个上，建新的命令和出命令基本相同。

## 将符号移到另一个

您可以使用以下命令很容易地将符号从源复制到目中：

- 像源。
- 通像加要符号：。符号将示在区域中。
- 像目。
- 像将当前符号保存到本地内存中的新。
- 像将符号保存在当前本地文件中：


## 弃符号化

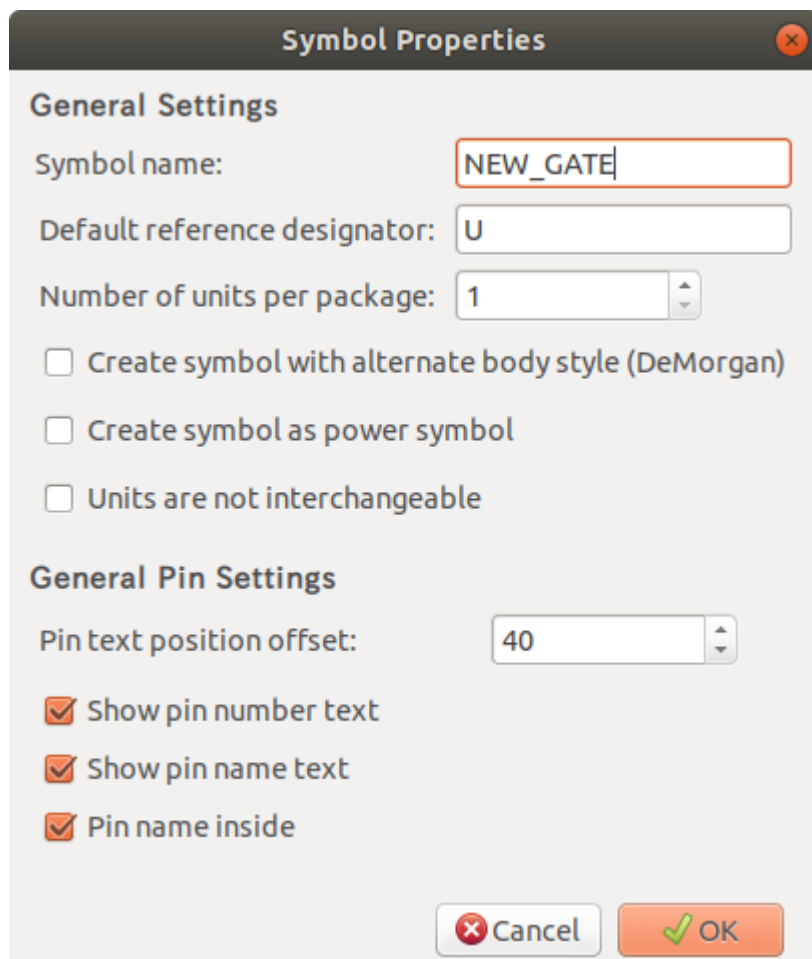
理符号的符号只是其中符号的工作副本。意味着只要你没有保存它，你可以重新加它以弃所做的所有更改。如果您已在本地内存中更新它并且尚未将其保存到文件，可以随退出并重新启 Eeschema 将撤消所有更

改。

## 创建符号

### 创建一个新符号

您现在可以创建新符号： 。系统将要求您输入符号名称（此名称用作原理图中字段的默认参考符号（U, IC, R ...），每个包装的位数（例如一个 7400 由每个包装4个单元组成）并且如果需要替代的体型（有称 DeMorgan）。如果参考指示符字段为空，默认 U。稍后可以更改些属性，但最好在创建符号时正确设置它们



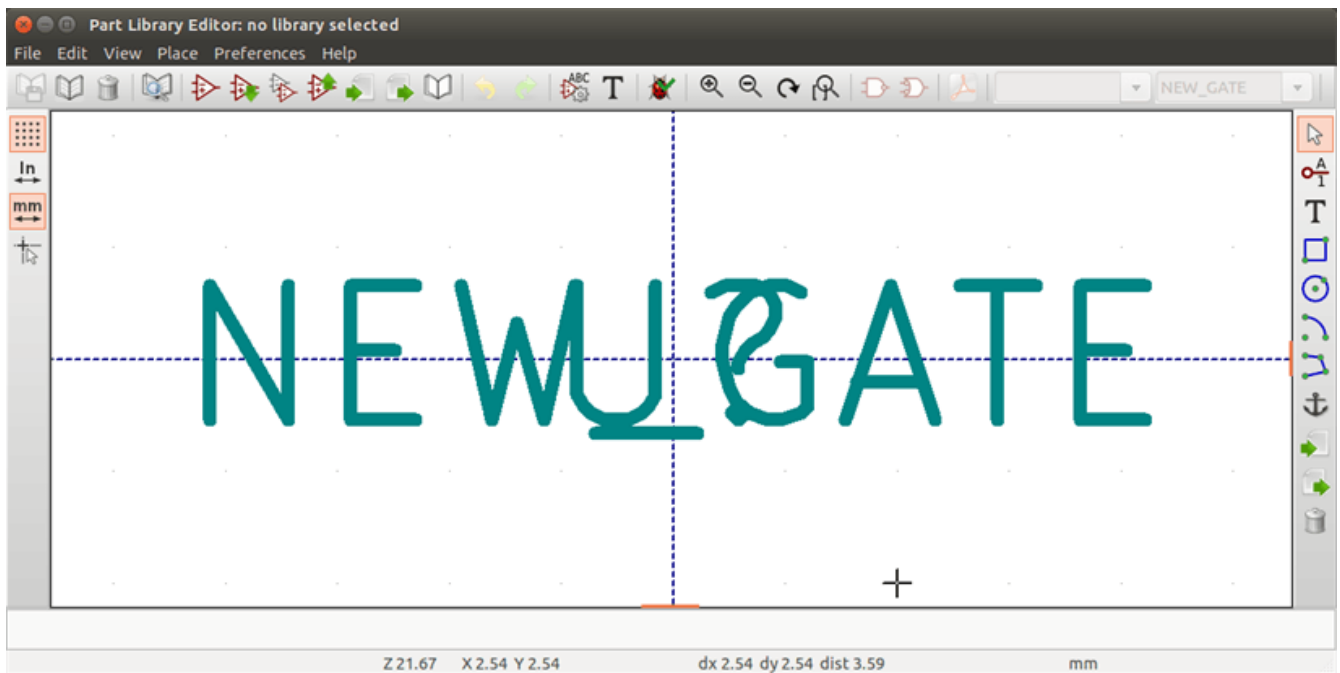
The image shows a 'Symbol Properties' dialog box with the following settings:

- General Settings**
  - Symbol name: NEW\_GATE
  - Default reference designator: U
  - Number of units per package: 1
  - ☐ Create symbol with alternate body style (DeMorgan)
  - ☐ Create symbol as power symbol
  - ☐ Units are not interchangeable
- General Pin Settings**
  - Pin text position offset: 40
  - ☒ Show pin number text
  - ☒ Show pin name text
  - ☒ Pin name inside

Buttons: Cancel, OK






将使用上面的属性创建一个新符号，它将出现在图中，如下所示。






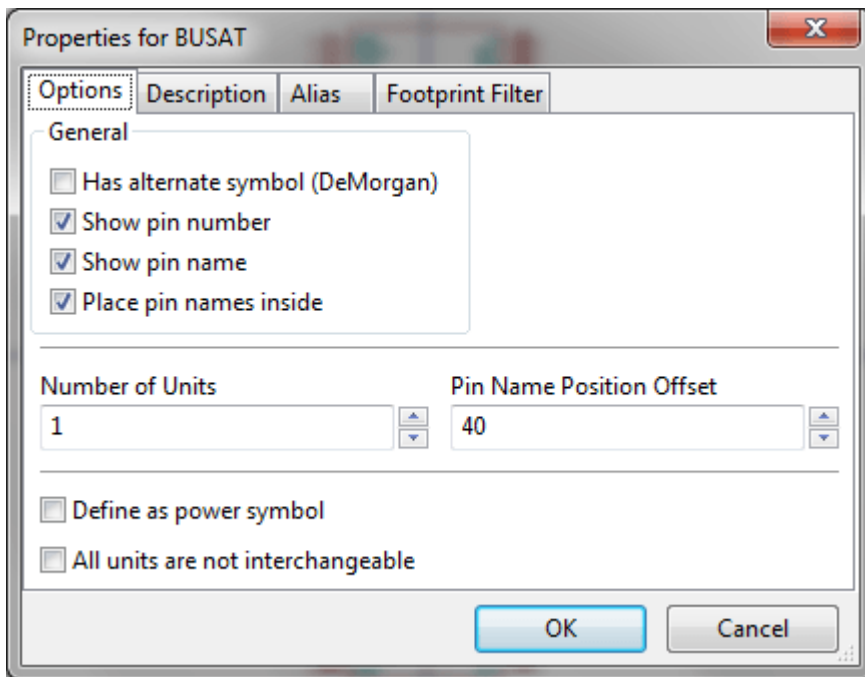
## 从另一个符号□建符号

通常，您要制作的符号□似于符号□中已有的符号。在□种情况下，很容易加□和修改□有符号。

- 加□将用作起点的符号。
- □□  或通□右□□□□字段并□□文本来修改其名称。如果您□□复制当前符号，系□将提示您□入新的符号名称。
- 如果模型符号具有□名，系□将提示您从新符号中□除与当前□冲突的□名。如果答案□否，□将中止新的符号□建。符号□不能包含任何重复的名称或□名。
- 根据需要□□新符号。
- □□□像更新当前□中的新符号：  或□□□像保存到新□□  或者 如果要将此新符号保存在其他□有□中，□□□□像□□其他□□  并保存新符号。
- □□□像将当前□文件保存到磁□□ .

## 符号属性

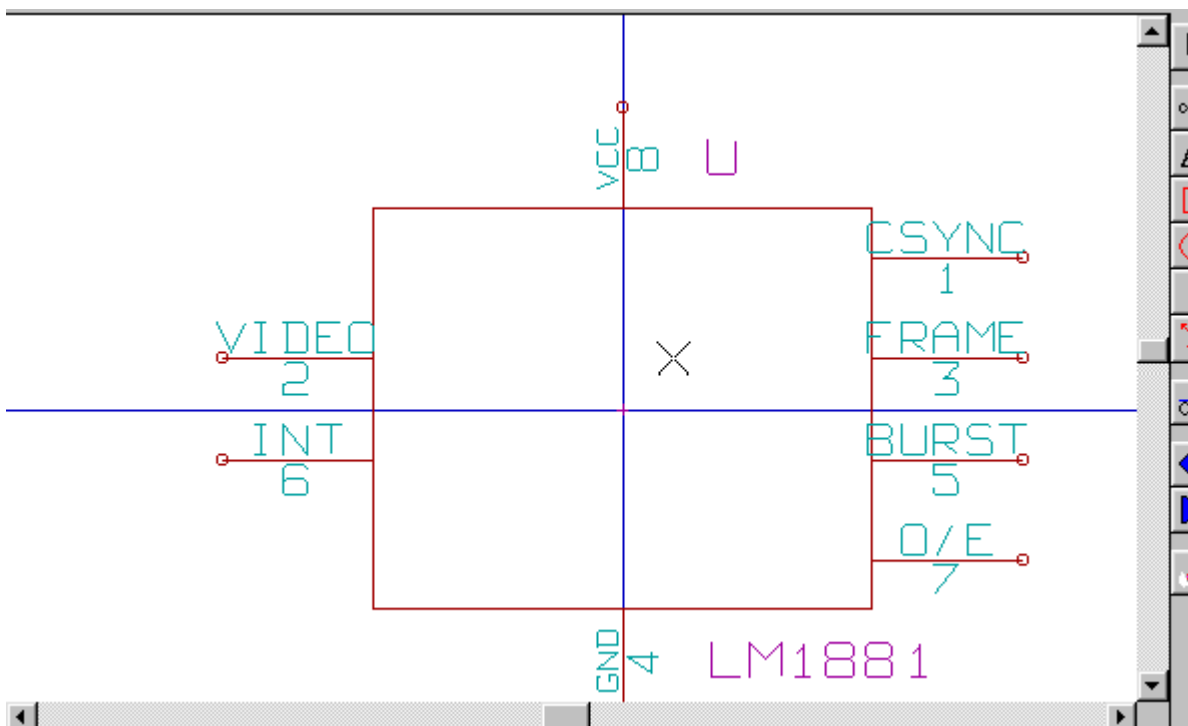
□在符号□建期□仔□□置符号属性，或者从复制的符号□承符号属性。要更改符号属性，□□□  以□示下面的□□框。




正确设置每个封装的单元数和用符号表示（如果已启用）非常重要，因为在PCB或原理图中每个单元的相同引脚都会受到影响。如果在创建和放置引脚后更改每个封装的单元数，您可能需要额外的时间来添加新的单元引脚和符号。然而，可以随时修改这些属性。


形状指示引脚号和指示引脚名称定义了引脚号和引脚名称文本的可变性。如果在表中相同的文本将引脚名称放在内部定义了相对于引脚体的引脚名称位置。如果在表中相同的文本将指示在符号轮廓内。在这种情况下，引脚名称位置偏移属性定义文本从引脚的主体端移开。30 到 40（1/1000 英寸）的偏移是合理的。

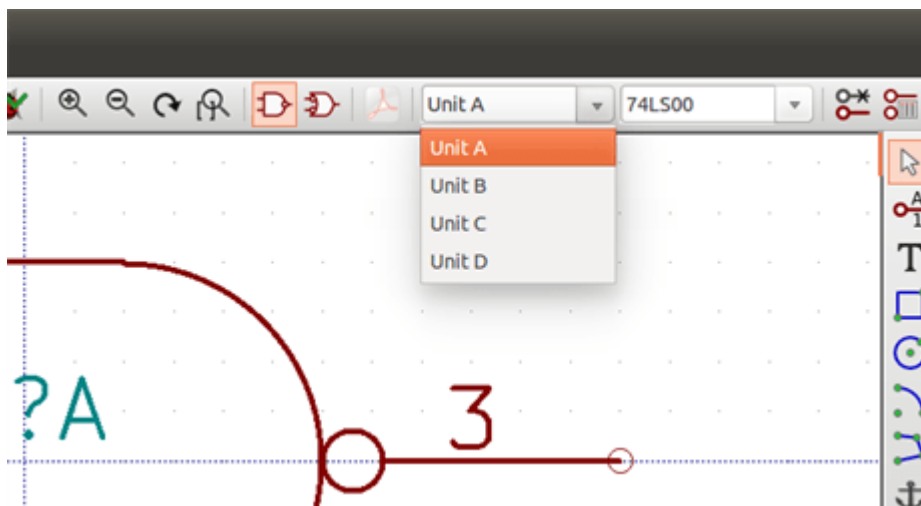
下面的示例显示了未选中“放置引脚名称”时的符号。注意名称和引脚号的位置。



## 有替代符号表示的符号

如果符号具有多个符号再存必须有一个表示来它要常表示，.

要□□□用表示，□□□。使用下面□示的 74LS00 □□要□□的□位。



## □形元素

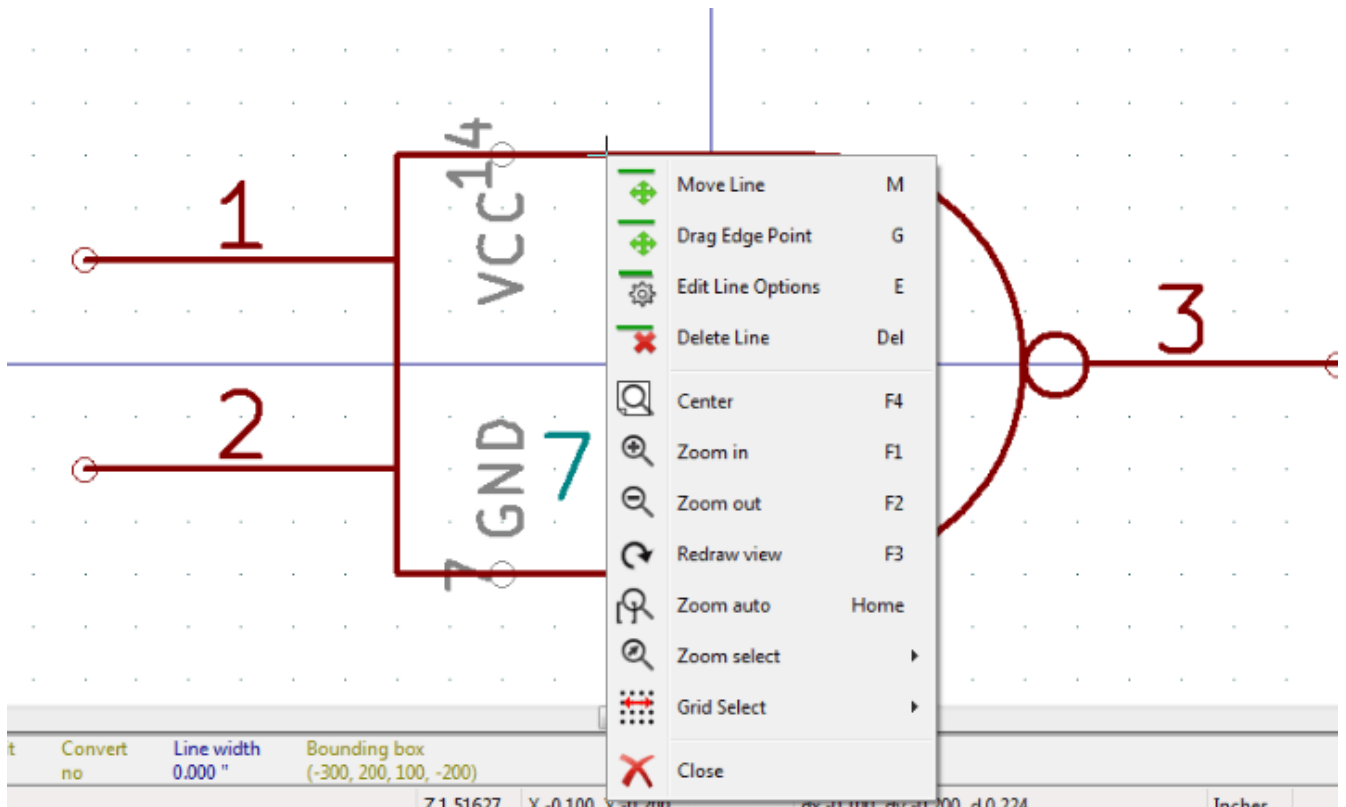
□形元素□建符号的表示，不包含□气□接信息。使用以下工具可以□计它□□

- 由起点和□点定义的□和多□形。
- 由两个□角□定义的矩形。
- 由中心和半径定义的□□
- 由弧的起点和□点及其中心定义的弧。弧度从0°到180°。

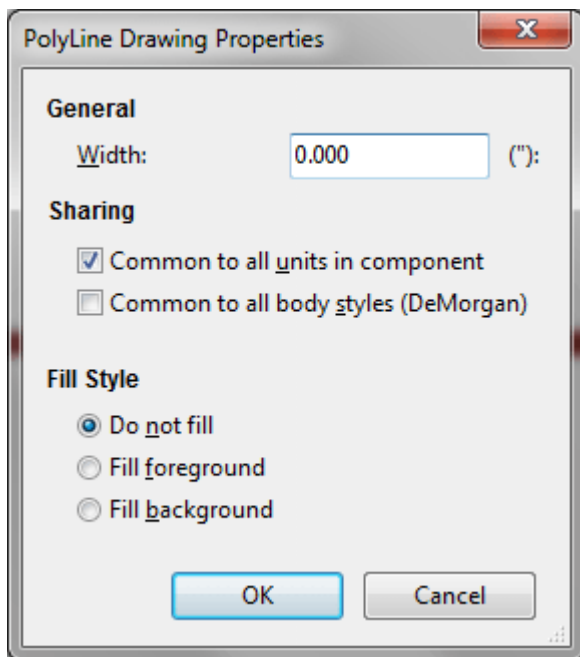
主窗口右□的垂直工具□允□您放置□计符号表示所需的所有□形元素。

## □形元素成□□格

每个□形元素（□□弧，□等）可以被定义□□于所有□元和/或主体□式是共同的或者□于□定□元和/或主体□式是特定的。右□□□元素可以快速□□元素□□□以□示所□元素的上下文菜□□ 下面是□元素的上下文菜□□



您可以通过双击元素以修改其属性。下面是多边形元素的属性对话框。



多边形元素的属性是：

- 宽度用于定义当前多边形位中元素线条的宽度。
- 符号中所有位共同位置定义是否每个包含多个位的符号控制多边形元素，或者是否当前位控制多边形元素。
- “所有主体样式共同（DeMorgan）”位置定义是否具有替代主体样式的符号中的每个符号表示控制多边形元素，或者是否当前主体样式控制多边形元素。
- 填充样式位置确定多边形元素定义的符号是否要控制未填充，背景填充或前景填充。

## □形文本元素

□个 **T** 允□建□形文本。即使□像符号，□形文本也始□可□□注意，□形文本□不是字段。

## 每个符号多个□位和替代体型□式

符号可以具有两个符号表示（□准符号和□用符号，通常称□*DeMorgan*）和/或每个包具有多于一个□元（例如□□□□□某些符号每个封装可以有多个□元，每个□元具有不同的符号和引脚配置。

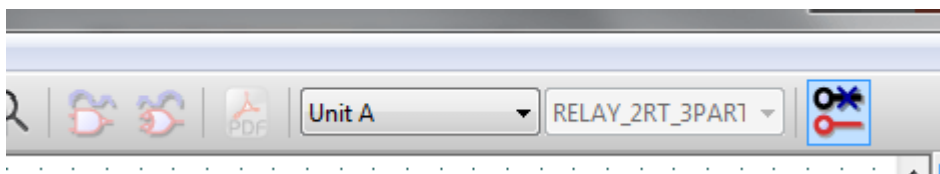
例如，考□具有两个开关的□□器，其可以被□计□具有三个不同□元的符号：□圈，开关1和开关2。□计每个封装具有多个□元和/或交替的主体□式的符号是非常灵活的。引脚或体型符号□□于所有□元可以是共同的或者□于□定□元是特定的，或者它□□于两个符号表示可以是共同的，因此特定于□定符号表示。

默认情况下，引脚特定于每个□元的每个符号表示，因□引脚号特定于□元，并且形状取决于符号表示。当引脚□于每个□元或每个符号表示是公共的□□您需要□所有□元和所有符号表示□建一个引脚（□通常是□源引脚的情况）。□也是身体□式□形形状和文本的情况，每个□元可能是共同的（但通常特定于每个符号表示）。

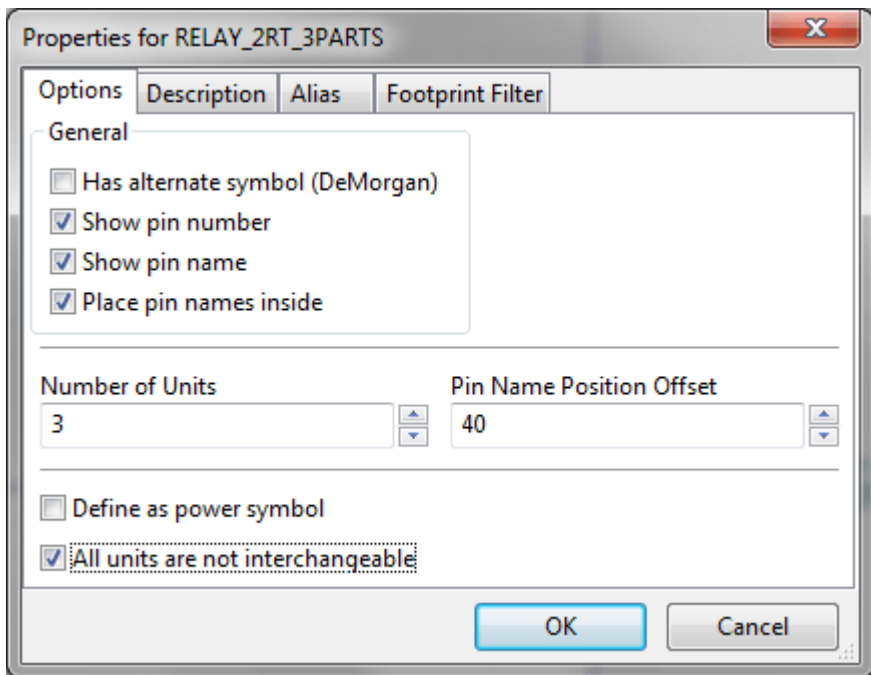
## 具有不同符号的多个□元的符号示例：

□是一个□□器的例子，每个包装有三个□元，开关1，开关2和□圈：

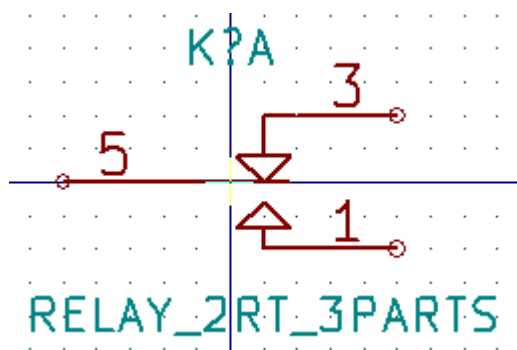
□□□引脚未□接。可以□每个□元添加或□□引脚，而无需与其他□元的引脚耦合。



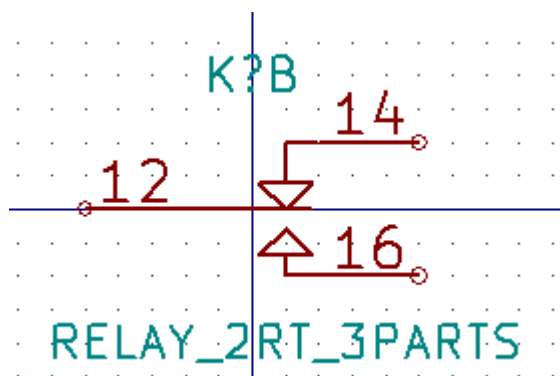
必□□□所有不可互□的□元。



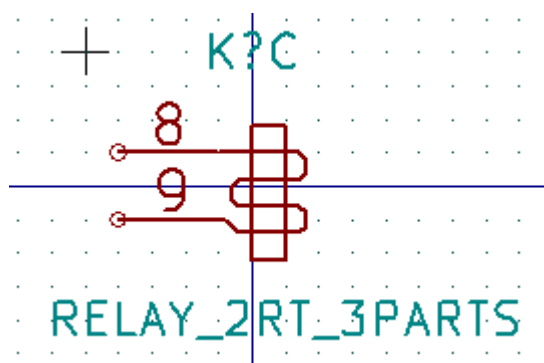
□元1



元2



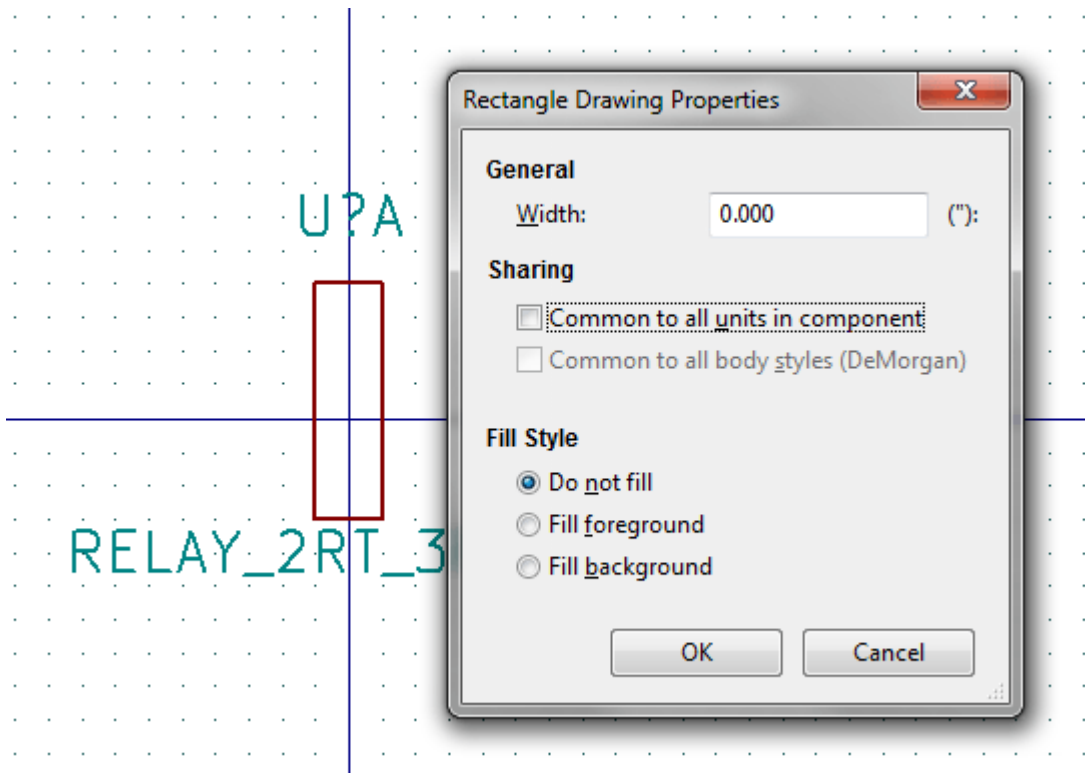
元3




它没有相同的符号和引脚布局，因此不能与元1和2互

## 形符号元素

下面示的是形主体元素的属性。从上面的器示例中，三个元具有不同的符号表示。因此，每个元都是独建的，并且形主体元素必禁用“符号中所有元的通用”。



## 引脚创建和编辑

您可以通过  来创建和插入引脚。通过双击引脚或右键单击引脚以打开引脚上下文菜单可以编辑所有引脚属性。必须仔细创建引脚，因为任何错误都会对 PCB 设计产生影响。可以删除和/或移动已放置的任何引脚。

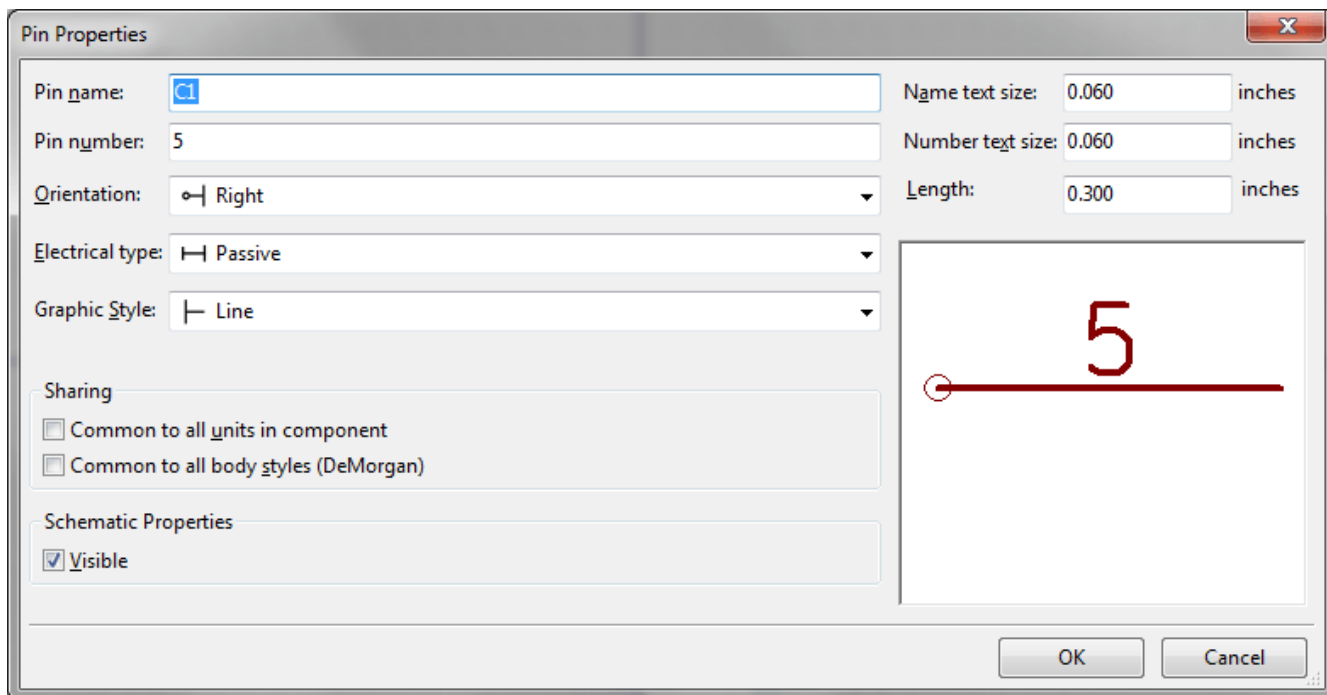
## 引脚概述

引脚由其形状表示，名称和“编号”定义。引脚的“数字”由一到四个字母和/或编号定义。要使电气规则检查（ERC）工具有用，必须正确定义引脚的“电气”类型（输入，输出，三态……）。如果未正确定义此类型，原理图 ERC 结果可能无效。

### 重要笔记

- 不要在引脚名称和数字中使用空格。
- To define a pin name with an inverted signal (overline) use the ~ (tilde) character followed by the text to invert in braces. For example ~{FO}O would display  $\overline{FO}$  O.
- 如果引脚名称减少一个符号，引脚被未命名。
- 以“#”开头的引脚名称保留用于源端口符号。
- 引脚“编号”由1到4个字母和/或数字组成。1,2, ... 9999是有效数字。A1, B3, Anod, Gnd, Wire 等也有效。
- 符号中不能存在重复的引脚“编号”。

## 引脚属性

The image shows a 'Pin Properties' dialog box with various configuration options. On the left, there are fields for 'Pin name' (containing 'C1'), 'Pin number' (containing '5'), 'Orientation' (set to 'Right'), 'Electrical type' (set to 'Passive'), and 'Graphic style' (set to 'Line'). To the right of these are input fields for 'Name text size', 'Number text size', and 'Length', all set to '0.060 inches', '0.060 inches', and '0.300 inches' respectively. Below these are two sections: 'Sharing' with checkboxes for 'Common to all units in component' and 'Common to all body styles (DeMorgan)', and 'Schematic Properties' with a checked 'Visible' checkbox. On the right side of the dialog is a preview window showing a red horizontal line with a circle at the left end and the number '5' above it. At the bottom right are 'OK' and 'Cancel' buttons.

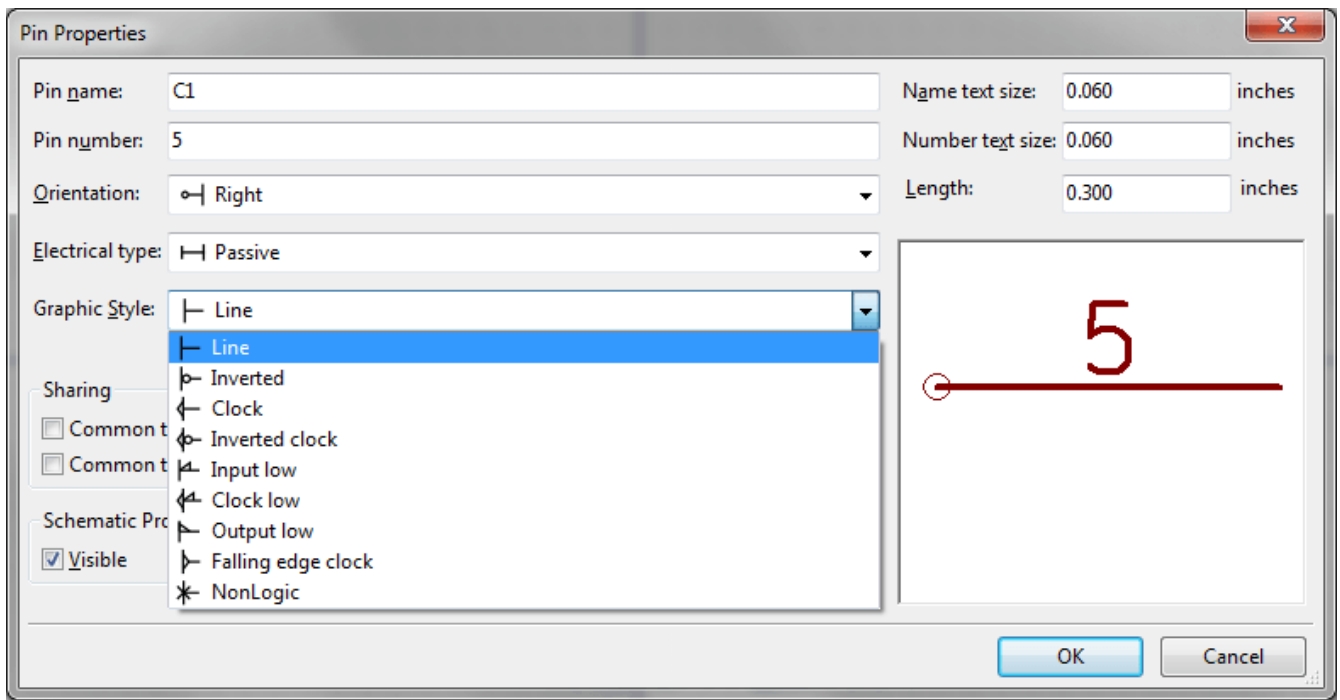
引脚属性对话框允许您配置引脚的所有特性。新建引脚或双端有引脚时会自出此对话框。此对话框允许您修改：

- 名称和名称的文字大小。
- 数字和数字的文字大小。
- 角度。
- 电气和图形型。
- 位和替代代表成格式。
- 可见性。

## 引脚图形式

下图示了不同的图形引脚式。图形式的引脚的电气型没有任何影响。





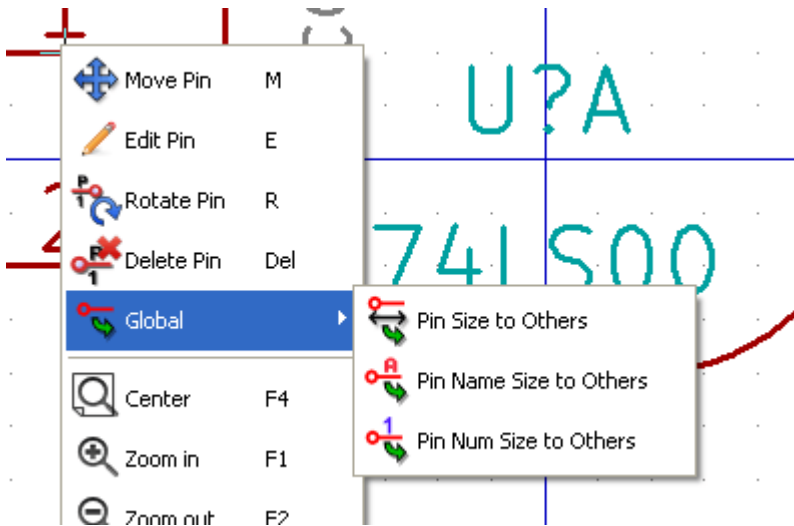
## 引脚电气型

正确的电气型对于原理图 ERC 工具很重要。定义的电气型是：

- 双向指示输入和输出之可交的双向引脚（例如微处理器数据总线）
- 三态是通常的三态输出。
- 无源用于无源符号引脚，电阻，连接器等。
- 当 ERC 无关紧要可以使用未指定的。
- 电源输入用于符号的电源引脚。电源引脚自接到具有相同名称的其他电源输入引脚。
- 功率输出用于驱动器输出。
- 开路集电极和开路集电极型可用于如此定义的三态输出。
- 当符号具有没有内部连接的引脚时使用未连接。

## 引脚全局属性

您可以使用引脚上下文菜单的全局命令条目修改所有引脚的名称和/或引脚的宽度或文本大小。要修改的参数，然后输入新值然后将值用于所有当前符号的引脚。



## 多个元和用符号表示定义引脚

在建和引脚具有多个元和/或形表示的符号尤其成大多数引脚特定于每个元（因它的引脚号特定于每个元）和每个符号表示（因它的形式和位置特定于每个符号表示）。于每个封装具有多个元的符号和替代符号表示，引脚的建和可能是有的。符号器允同建引脚。默认情况下，多个位符号的所有位以及具有替代符号表示的符号的两个表示都行引脚所做的更改。

唯一的例外是引脚的形型和名称。建立此依关系以便在大多数情况下更容易建和引脚。可以通过主工具上的来禁用此依关系。将允您完全独立地每个元和表示建引脚。

符号可以具有两个符号表示（表示“Demorgan”），并且可以由多个元成，如具有的情况。于某些符号，您可能需要几个不同的形元素和引脚。与“前一部分具有多个不同符号的符号的示例”，中所示的器本似，器可以由三个不同的元表示：圈，开关触点1，并切开关触点2。

具有多个元的符号的管理和具有替代符号表示的符号是灵活的。引脚可以是通用的或特定于不同的元。引脚也可以是符号表示或每个符号表示特有的。

默认情况下，引脚特定于每个元的每个表示，因它的数量因每个元而不同，并且它的计于每个符号表示是不同的。当一个引脚所有元都是通用的它只需要制一次，例如在源引脚的情况下。

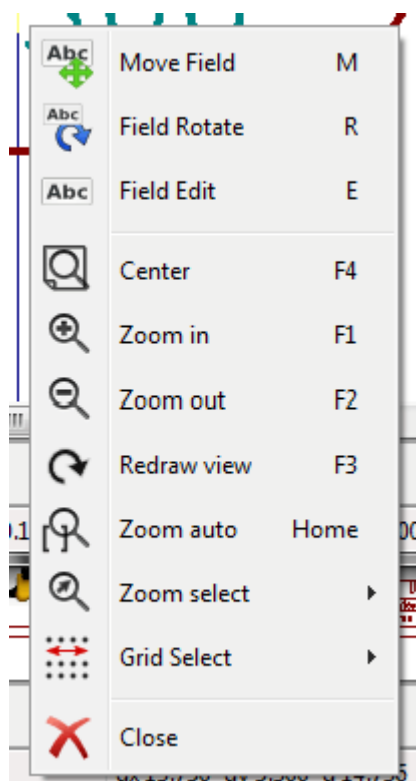
一个例子是出引脚 7400 四路双入与非由于有四个元和两个符号表示，因此在符号定义中定义了八个独的出引脚。建新的 7400 符号正常符号表示的元A将示在器中。要在用符号表示中引脚式，必首先通工具上的按钮启用它。要每个元的引脚号，使用以下像相的元： 下拉控件。

## 符号字段

所有符号都定义了四个默认字段。无论何建或复制符号，都会建参考指示符，占用空分配和文档文件接字段。需要参考指示符和字段。于有字段，可以通过右引脚来使用上下文菜单命令。中定义的符号通常使用四个默认字段定义。如供商，部件号，位成本等附加字段可以添加到符号中，但通常是在原理器中完成的，因此附加字段可以用于原理中的所有符号。

## 符号字段

要有符号字段，右字段文本以示下面示的字段上下文菜单



要添加未定义的字段，添加新字段或从主工具上的可定义字段 **T**，以打开下面所示的字段属性对话框。

Field Properties

Name	Va...
Reference	U
Value	74LS00
Footprint	
Datasheet	

Add Field
Delete Field
Move Up

Horiz. Justify

☐ Left
☒ Center
☐ Right

Vert. Justify

☐ Bottom
☒ Center
☐ Top

Visibility

☒ Show
☐ Rotate

Style:

☒ Normal
☐ Italic
☐ Bold
☐ Bold Italic

Field Name

Reference

Field Value

U

Show in Browser

Size

0.060

in

PosX

0.000

in

PosY

-0.050

in

OK

Cancel

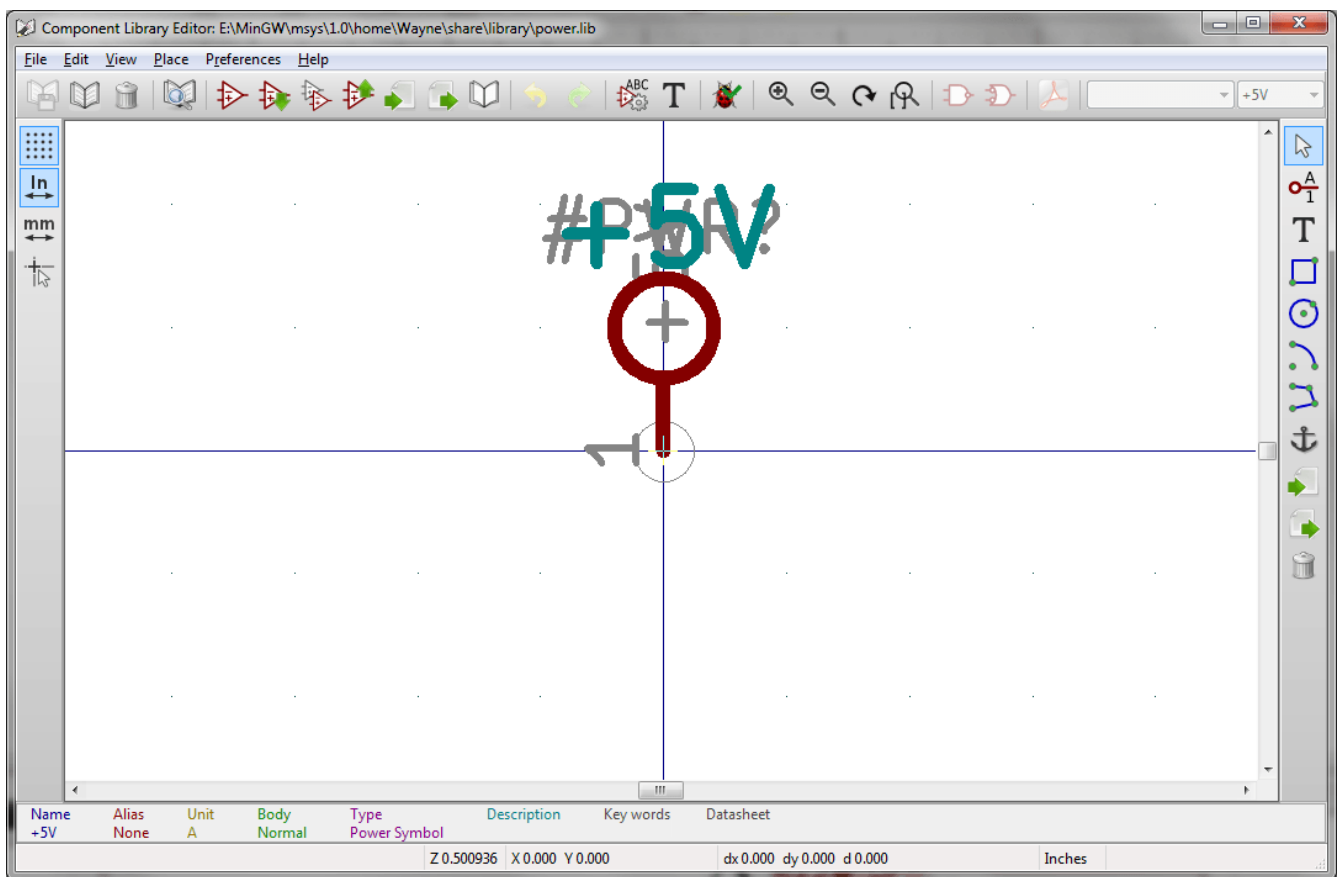
字段是与符号关联的文本部分。不要将它与属于此符号形表示的文本混淆。

### 重要笔记

- 修改字段有效地使用当前符号作新符号的起点新建符号。将新符号保存到当前定义的库中将包含在字段中。
- 上面的字段框必须用于空的字段或启用了不可编辑属性。
- 封装定义使用 LIBNAME:FPNAME 格式的封装，其中 LIBNAME 是封装表中定义的封装的名称（参看 Pcbnew 参考手册中的 封装表 部分），FPNAME 是 LIBNAME 中的封装名称。

## 源符号

功率符号的创建方式与普通符号相同。将它放在库（如 power.lib）中可能很有用。源符号由圆形符号和源/藏型的引脚组成。原理图捕获元件可以像处理任何其他符号一样处理源端口符号。一些防护措施至关重要。以下是源 +5V 符号的示例。



要创建源符号，请使用以下步骤：

- 添加名为 +5V 的“源/入”型的引脚（重要的是因此名称将建立与网 +5V 的连接），引脚号 1 不重要，长度 0 以及“路”“形”格。
- 如图所示，将一个小圈和一个从引脚到圈的部分放置。
- 符号的点在引脚上。
- 符号为“+5V”。
- 符号引用是“#+5V”。除了必须“#”的第一个字符表示符号是源符号外，参考文本并不重要。按照惯例，引用字段以“#”开头的每个符号都不会出现在符号列表或网表中，并且引用被声明不可编辑。

新建的源端口符号的更好方法是使用另一个符号作模型：

- 加有的源符号。
- 使用新源符号的名称引脚名称。
- 如果要示源端口, 将字段与引脚相同的名称。
- 保存新符号。

# LibEdit - 符号

## 概述

符号由以下元素组成

- 图形表示（几何形状，文本）。
- 引脚。
- 后置处理器使用的字段或关键字文本：网表，符号列表。

要初始化两个字段：引用和与符号相关的字段的名称以及关键字的足迹的名称，其他字段是空字段，它们通常可以保持为空，并且可以在原理图捕捉期填充。

但是，管理与任何符号相关的文档有助于您的研究，使用和以下相关文档包括

- 一行注释
- 一系列关键字，如 TTL CMOS NAND2，由空格分隔。
- 附加文件名（例如应用程序注释或 pdf 文件）。

附加文件的默认目录

`kicad/share/library/doc`

如果没有找到：

`kicad/library/doc`

在 linux 下：

`/usr/local/kicad/share/library/doc`

`/usr/share/kicad/library/doc`

`/usr/local/share/kicad/library/doc`

关键字允许您根据各种标准准确地搜索符号。注释和关键字显示在各种菜单中，特别是从符号库

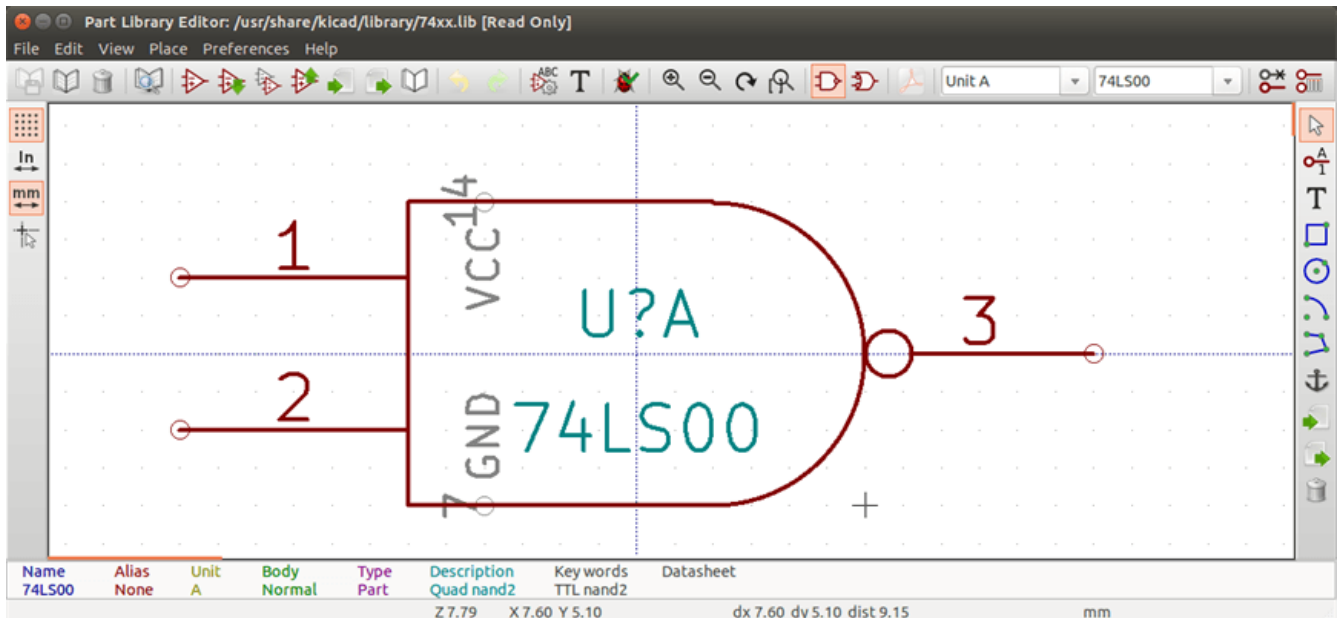
符号也有一个点。相对于点旋转或镜像，并且在放置期间点用作参考位置。因此精确定位是有用的。

符号可以具有别名，即等效名称。这允许您减少需要建立的符号数量（例如，74LS00 可以具有别名，例如 74000,74HC00,74HCT00 .....）。

最后，符号分布在库中（按主键或制造商分类）以便于管理。

## 定位符号点

点位于坐标 0,0 并由屏幕上显示的颜色指示。



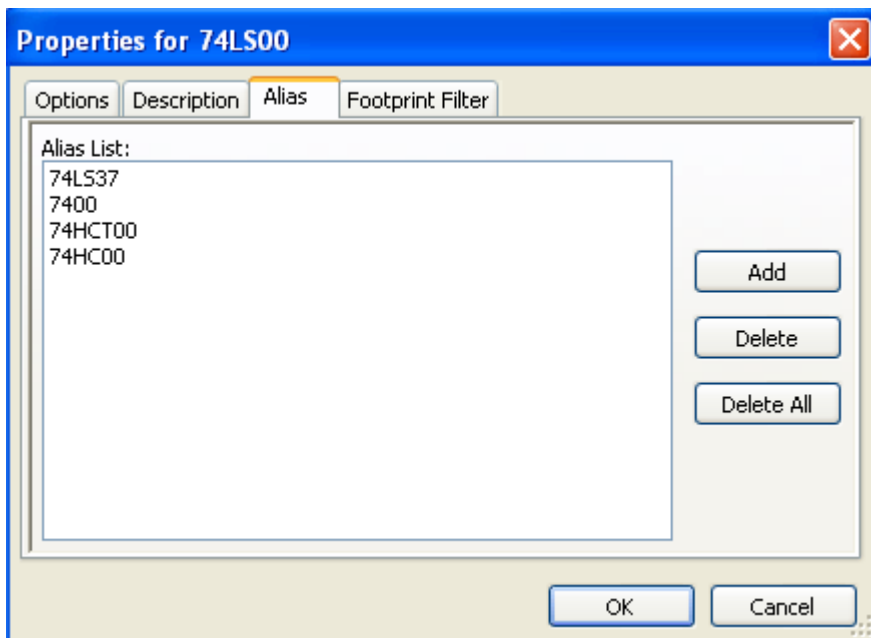
可以通过  并新的所需点位置来重新定位点。 将自重新定位在新点上。

## 符号名

名是与中相同符号的另一个名称。具有似引脚和表示的符号然后可以由一个符号表示，具有若干名（例如，具有名 74LS00,74HC00,74LS37 的 7400）。

使用名可以快速构建完整的 此外，些更加凑，可以松加 KiCad。

要修改名列表，必通  主窗口，然后名文件



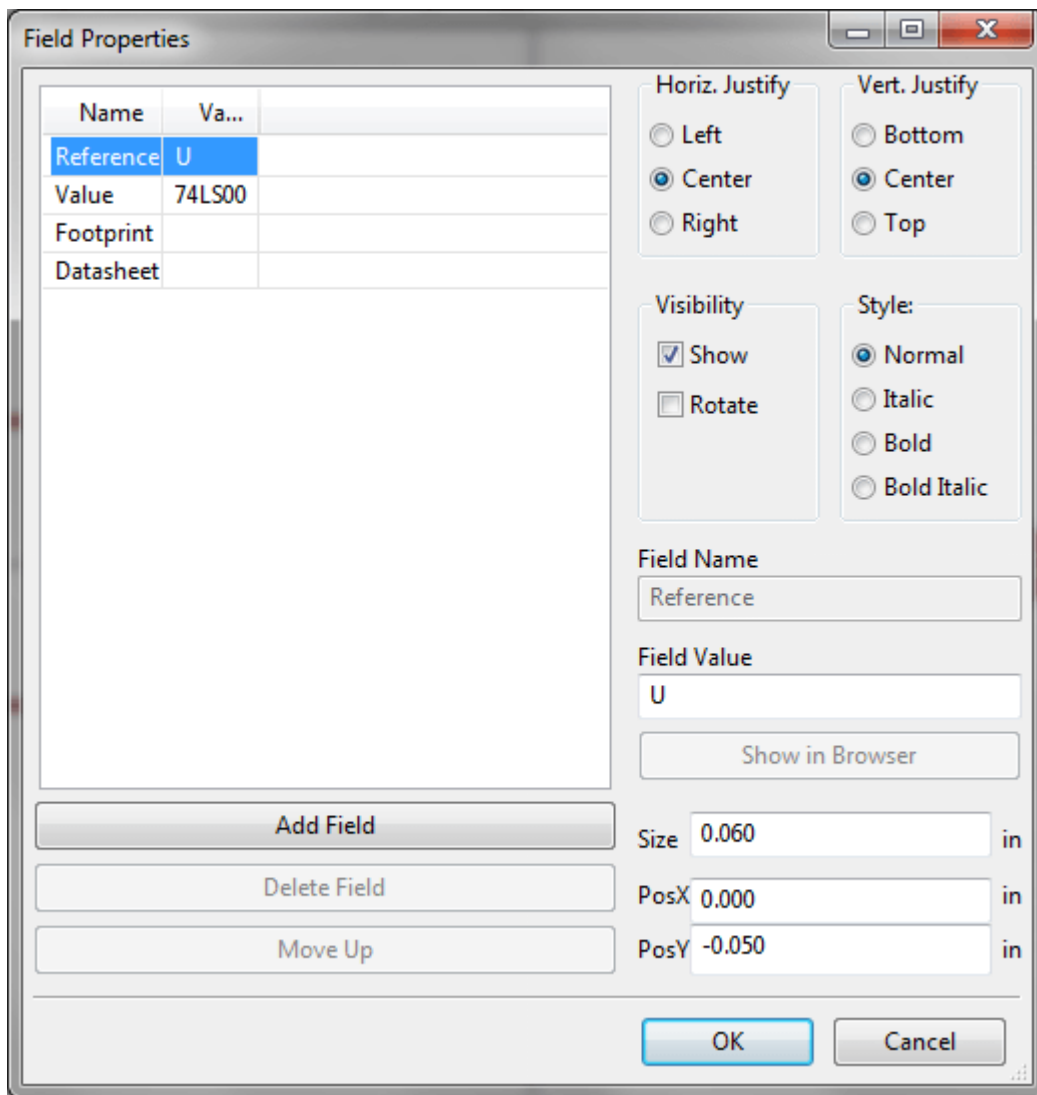
因此，您可以添加或删除所需的。由于了当前的，因此然无法将其除。

要除所有，首先要根符号。主工具窗口中名列表中的第一个符号。

## 符号字段

字段器通用： **T** 。


有四个特殊字段（符号附加的文本）和可配置的用□字段



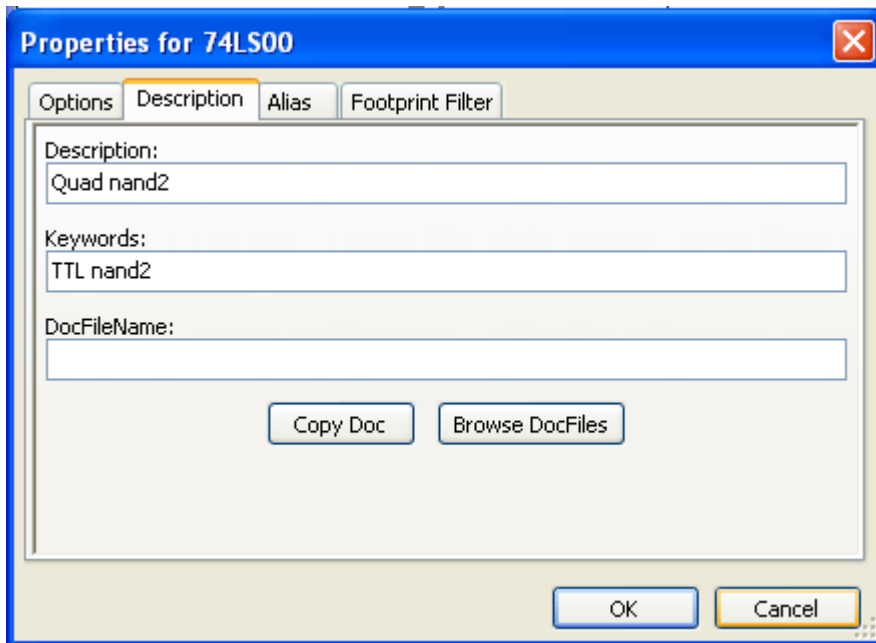
## 特殊字段

- 参考。
- 它是中的符号名称和原理中的默认字段。
- 封装。它是用于路板的封装名称。使用 CvPcb 置封装列表不是很有用，但如果不使用 CvPcb 必使用。
- 表。它是一个保留字段, 在写不使用。

# 符号文档

要文档信息，需要通  用符号的主窗口，并文档文件





必须正确的名或根符号，因此文档是名之唯一不同的特征。“复制文档”按钮允许您将文档信息从根符号复制到当前符号的名。

## 符号关键字

关键字允许您根据特定的标准（功能，技术系列等）以的方式搜索符号

Eeschema 研究工具不区分大小写。中使用的最新关键字是

- 用于系列的 CMOS TTL
- AND2 NOR3 XOR2 INV ...用于AND2 = 2 输入 AND NOR3 = 3 输入 NOR
- JKFF DFF ... 用于 JK 或 D 触发器。
- ADC, DAC, MUX...
- 具有开路集电极输出的 OpenCol。因此，如果在原理图中，您搜索符号：通关键字 NAND2 OpenCol Eeschema 将显示具有两个关键字的符号列表。

## 符号文档 (Doc)

注释行（和关键字）显示在各种菜单中，尤其是在的符号列表和 ViewLib 菜单中符号

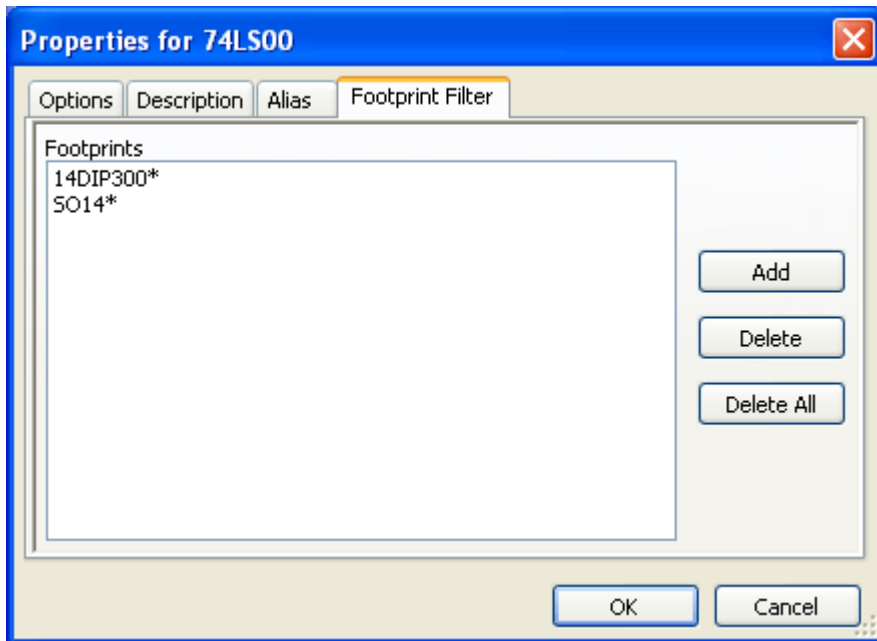
如果个 Doc. 文件存在，也可以在原理图中，通过右符号显示的出菜单中它。

## 相关文档文件 (DocFileName)

表示可用的附件（文档，用原理pdf 文件，原理等）。

## CvPcb 的封装

您可以输入符号允许的覆盖区列表。此列表充当 CvPcb 用于示允许的覆盖区的器。无效列表不会任何内容。



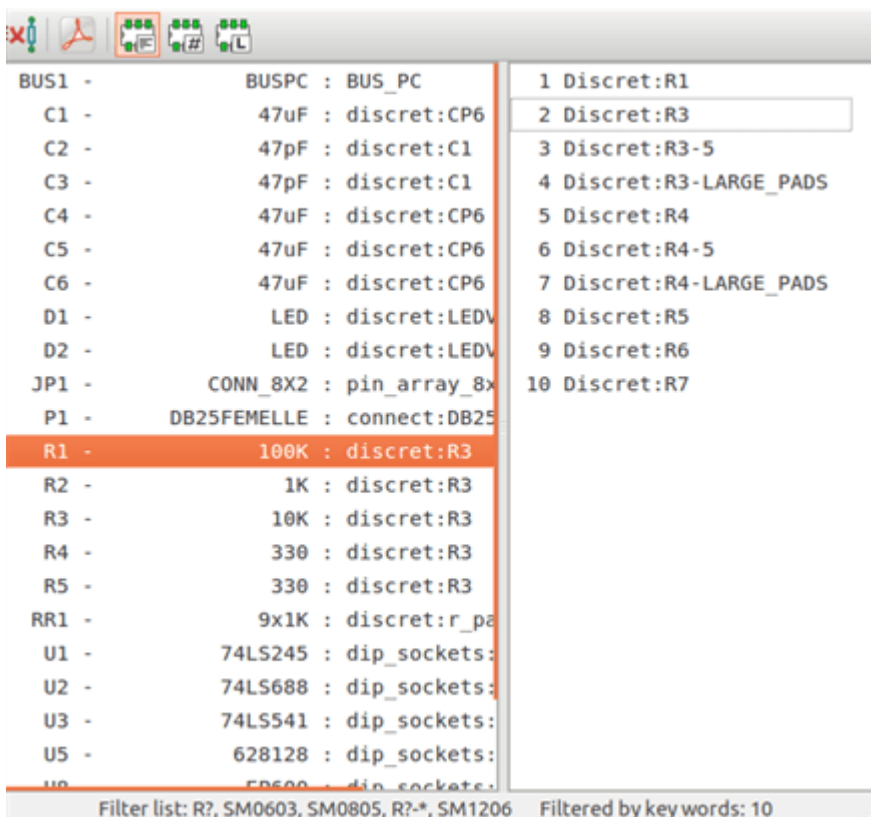
允许使用通配符。

SO14\* 允许 CvPcb 显示名称以 SO14 开头的封装。

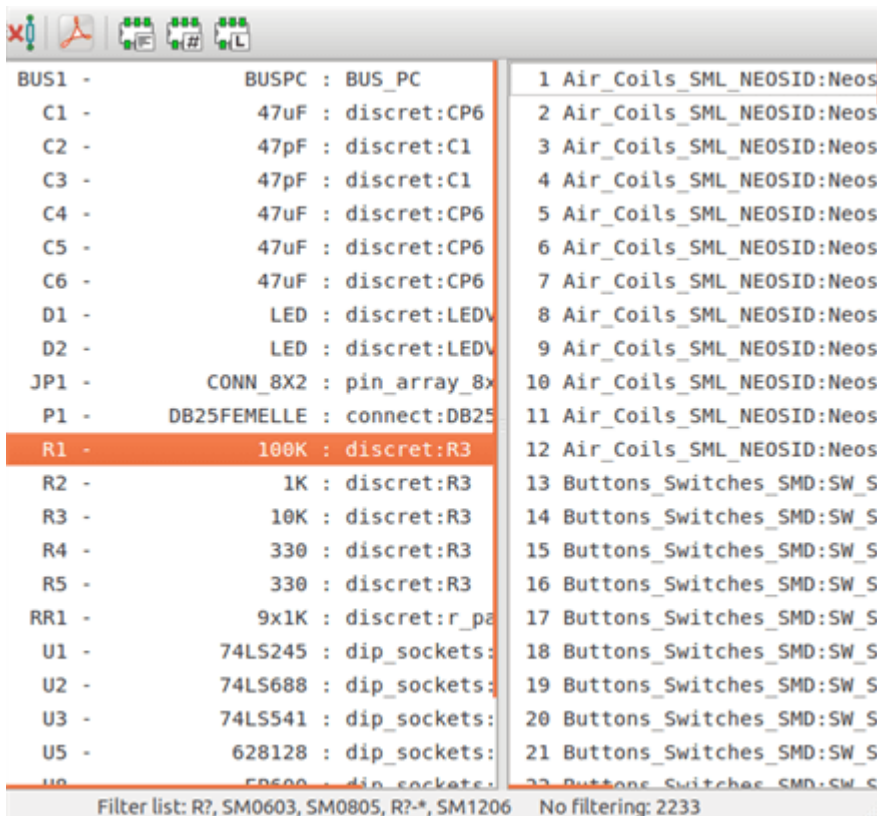
电阻器，R 显示所有有以 R 开头的 2 个字母名称的封装。

以下是本：有和没有

有



没有




## 符号

您可以轻松包含常用符号的符号文件。可以用于创建符号（三角形，与，或，异或等的形状）以用于保存和随后的重复使用。

些文件默认存储在目录中，并具有“.sym”扩展名。些符号不像普通符号那样收集在库中，因为它通常不是那么多。

### 出或创建符号


可以使用按钮像出符号： 。您通常只能创建一个符号，删除所有引脚（如果存在）也是一个好主意。

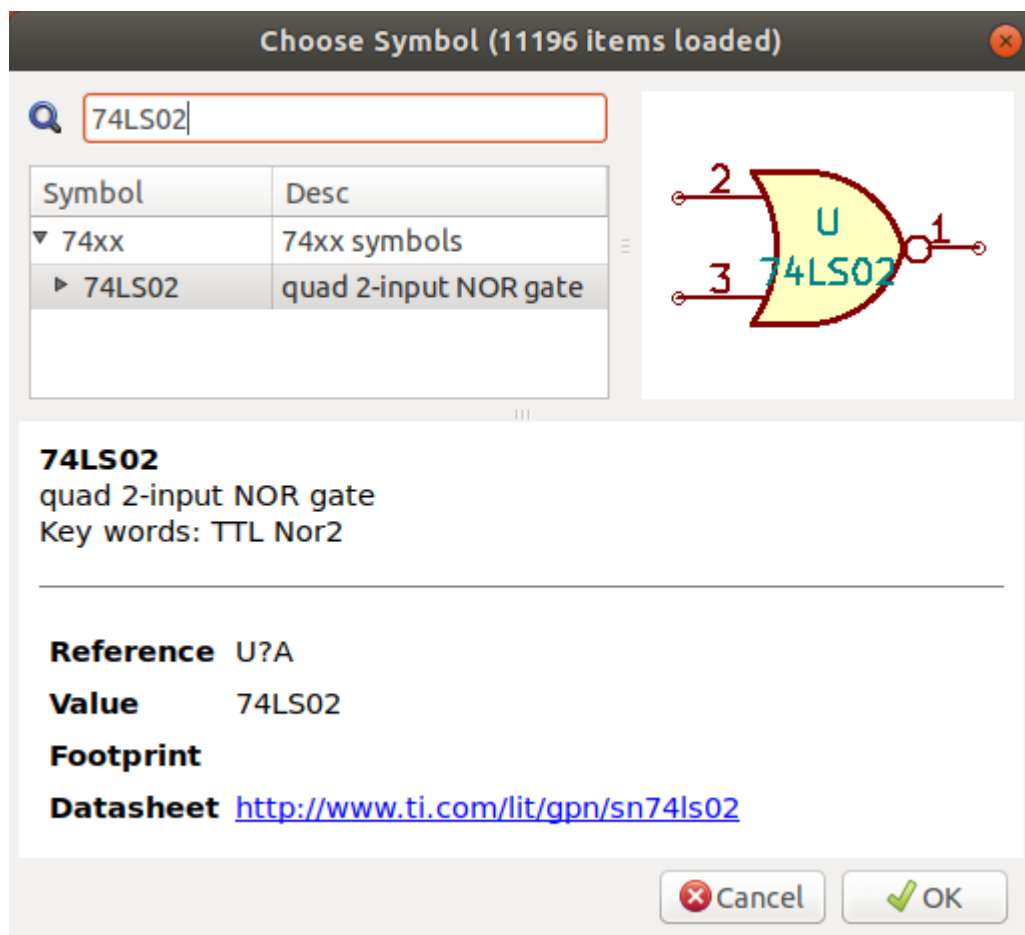
### 入符号

入允许您将符号添加到正在编辑的符号中。使用按钮像入符号： 。入的符号在现有符号中创建添加。

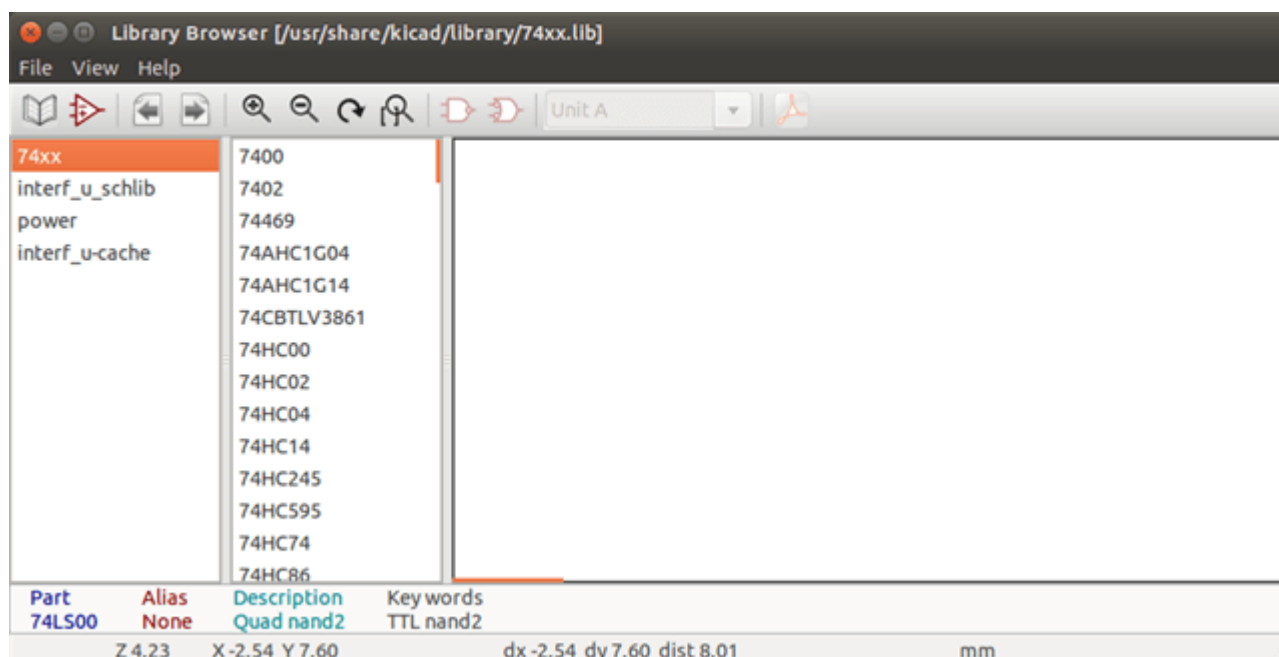
# 符号库器

## 简介

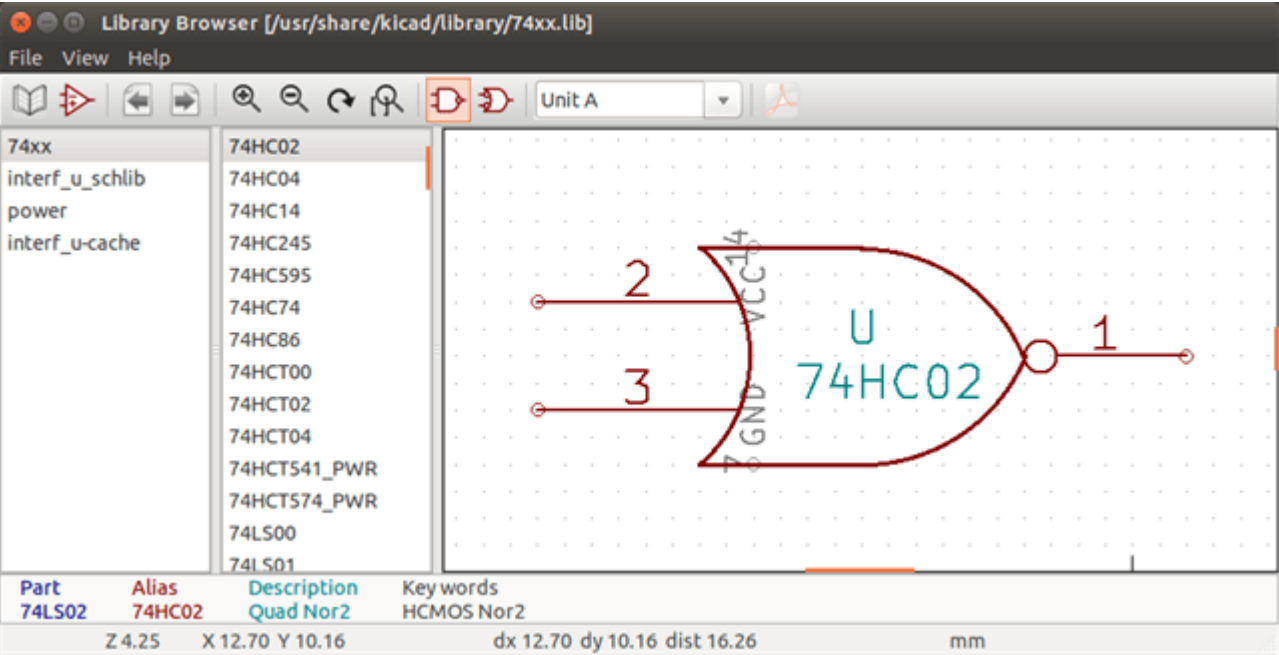
符号库器允许您快速查看符号的内容。可以通过主工具上的  在 菜 单 中 器 条 目 或 双 符 号 上 的 符 号 像 来 看 符 号 器 窗 口。



## 主屏幕



要显示的内容，从左窗格的列表中选择一个，所有符号都将显示在第二个窗格中。符号名称以查看符号。



符号器部工具

符号器中的部工具如下所示。



可用的命令是：

	所需的也可以在中 列表。
	可在 中的符号 名
	上一个符号。
	下一个符号。
	放工具。
	如果存在，表示（正常或）
	包含多个位的符号的位。
	如果存在， 示 的文档。 在被叫 存在 来自 Eeschema 的地方符号框。
	关 器并将所 符号放在 Eeschema 中。 只有在从 Eeschema 用 器 才会 示此 双 在元件器中的符号上）。

# 创建自定义网表和 BOM 文件

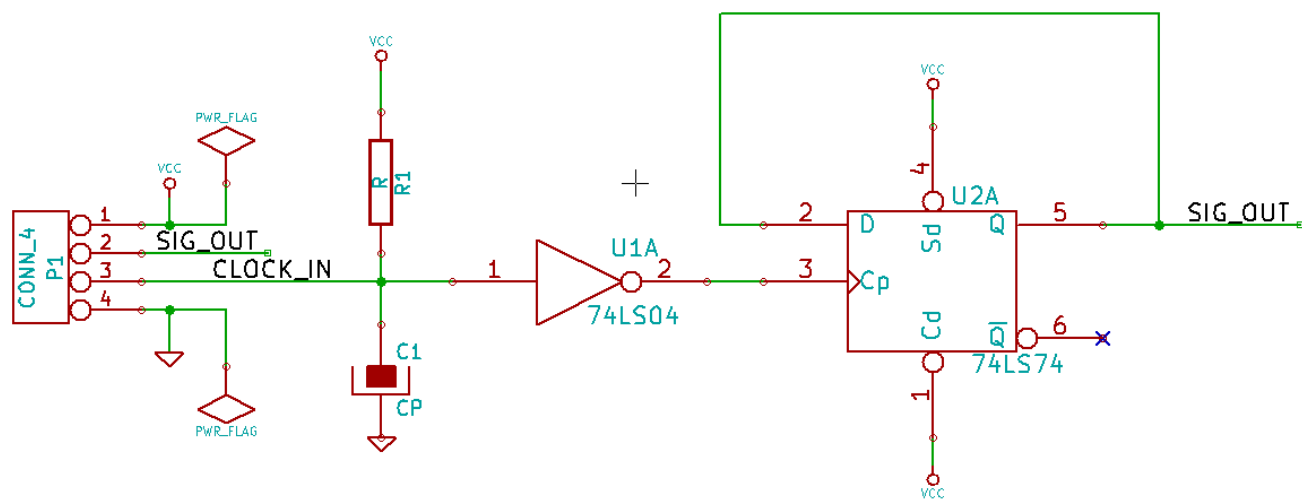
## 中间网表文件格式

BOM 文件和网表文件可以从 Eeschema 创建的中间网表文件

此文件使用 XML 方法，称中间网表。中间网表包含有关您的电路板的大量数据，因此，它可以与后处理一起用于创建 BOM 或其他报告。

根据输出（BOM 或网表），将在后处理中使用完整的中间网表文件的不同子集。

## 原理图



## 中间网表文件示例

上述电路的中间网表 (使用 XML 方法) 如下所示。

```

<?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 20:35:21</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="/" />
      <tstamp>4C6E2141</tstamp>
    </comp>
    <comp ref="U2">
      <value>74LS74</value>
      <libsource lib="74xx" part="74LS74"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E20BA</tstamp>
    </comp>
    <comp ref="U1">
      <value>74LS04</value>
      <libsource lib="74xx" part="74LS04"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E20A6</tstamp>
    </comp>
    <comp ref="C1">
      <value>CP</value>
      <libsource lib="device" part="CP"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E2094</tstamp>
    </comp>
    <comp ref="R1">
      <value>R</value>
      <libsource lib="device" part="R"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E208A</tstamp>
    </comp>
  </components>
  <libparts>
    <libpart lib="device" part="C">
      <description>Condensateur non polarise</description>
      <footprints>
        <fp>SM*</fp>
        <fp>C?</fp>
        <fp>C1-1</fp>
      </footprints>
      <fields>
        <field name="Reference">C</field>
        <field name="Value">C</field>
      </fields>
      <pins>
        <pin num="1" name="~" type="passive"/>
        <pin num="2" name="~" type="passive"/>
      </pins>
    </libpart>
    <libpart lib="device" part="R">
      <description>Resistance</description>
      <footprints>
        <fp>R?</fp>
        <fp>SM0603</fp>
        <fp>SM0805</fp>
      </footprints>
    </libpart>
  </libparts>
</export>

```

## 新的网表格式

通常将后处理器用于中间网表文件，您可以生成外部网表文件以及 BOM 文件。由于此是文本到文本因此可以使用 Python, XSLT 或任何其他能将 XML 作为输入的工具来写此处理器。

XSLT 本身是一种非常适合 XML 的 XML 语言。有一个名为 *xsltproc* 的免费程序，您可以下载并安装。xsltproc 程序可用于取中间 XML 网表输入文件，用样式表来输入，并将结果保存在输出文件中。使用 xsltproc 需要使用 XSLT 定义的样式表文件。完成该程序由 Eeschema 处理，在配置一次后以特定方式运行 xsltproc。

## XSLT 方法

描述 XSL (即 XSLT) 的文档可在此获得：

<http://www.w3.org/TR/xslt>

## 创建 Pads-Pcb 网表文件

“pads-pcb” 的格式由两部分组成。

- 封装列表。
- 网表: 按网络引用行分

接下来是样式表，它将中间网表文件转换为 pad-pcb 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.

如何使用:
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 -->
<xsl:template match="node">
  <xsl:text> </xsl:text>

```

□是运行 xsltproc 后的 pads-pcb □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*
```

□行此□□的命令是：

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

## □建一个 Cadstar 网表文件

Cadstar 格式由两个部分□成。

- 封装列表。
- 网表: 按网□□□□引用□行分□□

以下是□行此特定□□的□式表文件：

```

<?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 -->
      <xsl:if test="count(node)>1">
            <xsl:variable name="netname">

```

是 Cadstar 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
```

## 创建 OrcadPCB2 网表文件

此格式只有一个部分是封装列表。每个封装包括其参考网的列表。

以下是此特定表的式表：

```

<?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.

如何使用:
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 $noname U1 74LS125
    ... pin list ...
  )
  and calls "create_pin_list" template to build the pin list
-->
<xsl:template match="comp">
  <xsl:text> ( </xsl:text>

```

是 OrcadPCB2 出文件。

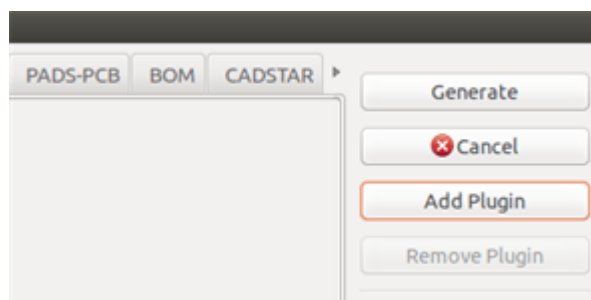
```
( { Eeschema Netlist Version 1.1 29/08/2010 21:07:51
eeschema (2010-08-28 BZR 2458)-unstable}
( 4C6E2141 $noname P1 CONN_4
( 1 VCC )
( 2 /SIG_OUT )
( 3 /CLOCK_IN )
( 4 GND )
)
( 4C6E20BA $noname U2 74LS74
( 1 VCC )
( 2 /SIG_OUT )
( 3 N-04 )
( 4 VCC )
( 5 /SIG_OUT )
( 6 ? )
( 7 GND )
( 14 VCC )
)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

## Eeschema 插件界面

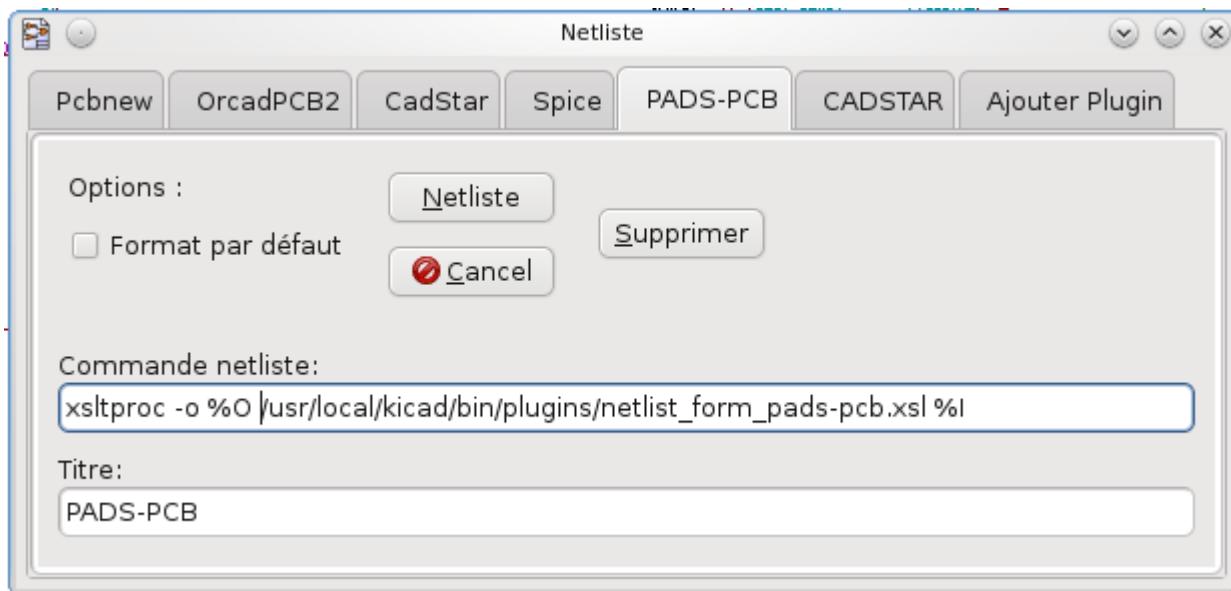
中网表器可以在 Eeschema 中自启

### 初始化窗口

可以通过 添加插件 按钮添加新的网表插件用界面卡。



以下是 PadsPcb 卡的配置数据：



## 插件配置参数

Eeschema 插件配置对话框需要以下信息：

- 例如，网表格式的名称。
- 用于后处理器的命令行。

网表按钮后，将产生以下情况：

1. Eeschema 建了一个中间网表文件 \*.xml，例如 test.xml。
2. Eeschema 通过取 test.xml 来运行插件，并建 test.net。

## 使用命令行生成网表列表文件

假如我使用程序 `xsltproc.exe` 将工作表格式用于中间文件，使用以下命令行 `xsltproc.exe`：

`xsltproc.exe -o <output filename> < style-sheet filename> <input XML file to convert>`

在 Windows 下的 KiCad 中，命令行如下：

`f:/kicad/bin/xsltproc.exe -o "%O" f:/kicad/bin/plugins/netlist_form_pads-pcb.xml "%I"`

在 Linux 下，命令如下：

`xsltproc -o "%O" /usr/local/kicad/bin/plugins/netlist_form_pads-pcb.xml "%I"`

`netlist_form_pads-pcb.xml` 是您要用的样式表。不要忘记文件名周围的双引号，允许它在 Eeschema 替换后有空格。

命令行格式接受文件名的参数：

支持的格式参数是。

- %B = 基本文件名和所输出文件的路径，减去路径和扩展名。
- %I = 完整的文件名和输入文件的路径（中间网表文件）。
- %O = 完整的文件名和用输出的输出文件的路径。

%I 将被[]的中[]文件名替[]

%O 将替[]出文件名。

## 命令行格式：xsltproc 的示例

xsltproc 的命令行格式如下：

```
<path of xsltproc> xsltproc <xsltproc parameters>
```

在 Windows 下：

```
f:/kicad/bin/xsltproc.exe -o "%O" f:/kicad/bin/plugins/netlist_form_pads-pcb.xml "%I"
```

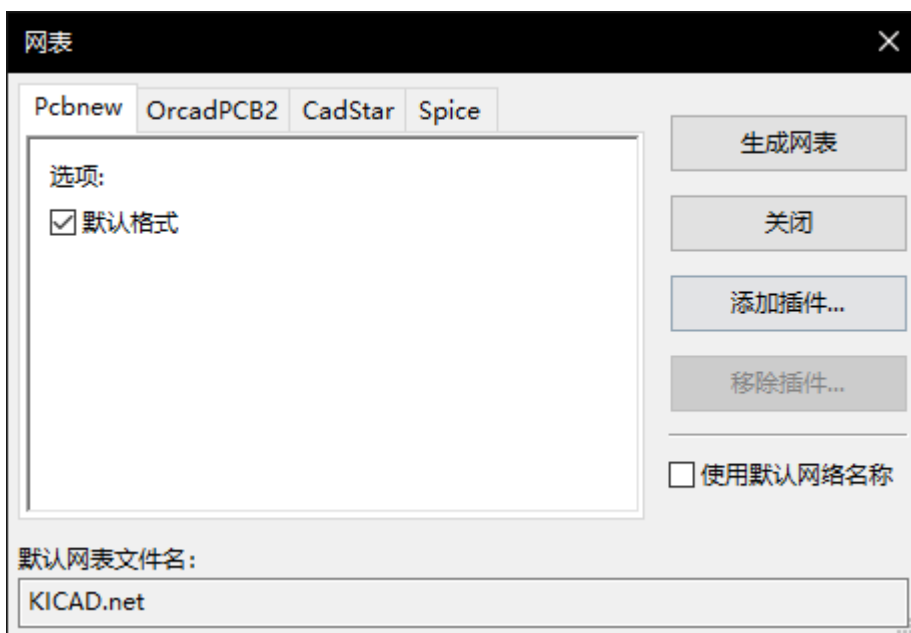
在 Linux 下：

```
xsltproc -o "%O" /usr/local/kicad/bin/plugins/netlist_form_pads-pcb.xml "%I"
```

上面的示例假[] xsltproc 安装在 Windows 下的 PC 上，所有文件都位于 kicad/bin 中。

## 物料清[] (BOM) 生成

由于中[]网表文件包含有关已使用元件的所有信息，因此可以从中提取 BOM。 以下是用于[]建自定义物料清[] (BOM) 文件的插件[]置窗口（在 Linux 上）：



[]式表 bom2csv.xml 的路径取决于系[] 目前用于 BOM 生成的最佳 XSLT []式表称[] bom2csv.xml。 您可以根据自己的需要自由修改它，如果您开[]了一些非常有用的[]西，[]让它成[] KiCad []目的一部分。

## 命令行格式：python 脚本的示例

python 的命令行格式如下：

```
python <脚本文件名> <[]入文件名> <[]出文件名>
```

在 Windows 下：

```
python *.exe f:/kicad/python/my_python_script.py "%I" "%O"
```



在 Linux 下：

```
python /usr/local/kicad/python/my_python_script.py "%I" "%O"
```

假如你的 PC 上安装了 python。

## **中网表构**

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="/" />
      <tstamp>4C6E2141</tstamp>
    </comp>
    <comp ref="U2">
      <value>74LS74</value>
      <libsource lib="74xx" part="74LS74"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E20BA</tstamp>
    </comp>
    <comp ref="U1">
      <value>74LS04</value>
      <libsource lib="74xx" part="74LS04"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E20A6</tstamp>
    </comp>
    <comp ref="C1">
      <value>CP</value>
      <libsource lib="device" part="CP"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E2094</tstamp>
    <comp ref="R1">
      <value>R</value>
      <libsource lib="device" part="R"/>
      <sheetpath names="/" tstamps="/" />
      <tstamp>4C6E208A</tstamp>
    </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"/>
      <node ref="U2" pin="3"/>
    </net>
  </nets>

```

## 一般网表文件结构

中网表占五个部分。

- “” 部分。
- “元件” 部分。
- “元件” 部分。
- “” 部分。
- “网” 部分。

文件内容具有分隔符 <export>

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

### “” 部分

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>
```

此部分可被批注部分。

### “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="/">
<tstamp>4C6E2141</tstamp>
</comp>
</components>
```

本包含原理中的元件列表。每个元件都是描述的：

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

<b>libsource</b>	找到此元件的库的名称。
<b>part</b>	此库中的元件名称。
<b>sheetpath</b>	本次结构中工作表的路径：工作表 在完整的原理图本次结构中。
<b>tstamps (time stamps)</b>	原理图文件的戳。
<b>tstamp (time stamp)</b>	元件的戳。

关于元件的戳的注意事项

要网表中的元件，从而板上，戳参考每个元件都是唯一的。然而，KiCad 提供了一种助方法来元件，元件是路板上相的占位面 允重新批注原理图中的元件，并且不会失元件与其占用空之的接。

戳是原理图中每个元件或工作表的唯一符。但是, 在复的本次结构中, 同一工作表多次使用, 因此此工作表包含具有相同戳的元件。

复本次结构中的定工作表具有唯一符：其 sheetpath。定元件（在复本次结构内）具有唯一符：sheetpath + 其 tstamp

“部件” 部分

部件部分具有分隔符 <libparts>, 并且此部分的内容在原理图中定义。部件部分包含

- 允的封装名称(名称使用通配符)以 <fp> 分隔符。
- 分隔符中定义的字段 <fields>。
- 引脚列表分隔 <pins>。

```
<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>
```

似 <pin num="1" type="passive"/> 的路也出了气引脚型。可能的引脚型有

Input	入引脚
Output	出引脚
Bidirectional	入或出
Tri-state	入/出
Passive	无源元件的束
Unspecified	未知气型
Power input	元件源入引脚
Power output	源出引脚作器出
Open collector	模比器中常的开路集极出
Open emitter	有在找到开放射器出。
Not connected	必在原理中保持未接状

“” 部分

部分具有分隔符<libraries>。本包含目中使用的原理列表。

```
<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” 部分

“网” 部分具有分隔符 <nets>。本部分包含原理图的“连接”。

```
<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>
```

本部分列出了原理图中的所有网表

可能的网表包含以下内容。

```
<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>
```

net code	是此网表的内部标识符
name	是此网表的名称
node	指出一个连接到网表的引脚引用

有关 xsltproc 的更多信息

参见页面：<http://xmlsoft.org/XSLT/xsltproc.html>

## 简介

xsltproc 是一个命令行工具，用于将 XSLT 样式表用于 XML 文档。虽然它是作 GNOME 项目的一部分开发的，但可以独立于 GNOME 桌面运行。

从命令行用 xsltproc，其中包含要使用的样式表的名称，后跟要用样式表的文件的名称。如果提供的文件名是 -，它将使用标准输入。

如果样式表包含在具有样式表处理指令的 XML 文档中，不需要在命令行中命名样式表。xsltproc 将自动包含的样式表并使用它。默认情况下，输出到 *stdout*。您可以使用 -o 指定要输出的文件。

## 简介

```
xsltproc [[-V] | [-v] | [-o *file* ] | [--timing] | [--repeat] |
[--debug] | [--novalid] | [--noout] | [--maxdepth *val* ] | [--html] |
[--param *name* *value* ] | [--stringparam *name* *value* ] | [--nonet] |
[--path *paths* ] | [--load-trace] | [--catalogs] | [--xinclude] |
[--profile] | [--dumpextensions] | [--nowrite] | [--nomkdir] |
[--writesubtree] | [--noddattr]] [ *stylesheet* ] [ *file1* ] [ *file2* ]
[ *...* ]
```

## 命令行选项

*-V o --version*

显示使用的 libxml 和 libxslt 的版本。

*-v o --verbose*

输出 xsltproc 在处理样式表和文档时采取的步骤

*-o o --output file*

直接输出到名为 *file* 的文件。对于多个输出，也称 *chunking*，-o *directory/* 将输出文件定向到指定的目录。目录必须已存在。

*--timing*

显示用于解析样式表，解析文档和用样式表并保存结果的总时间以毫秒显示。

*--repeat*

运行 20 次。用于定时

*--debug*

输出处理后文档的 XML 以行形式

*--novalid*

跳过加载文档的 DTD。

*--noout*

不输出结果。

*--maxdepth value*

在 libxslt 断定它于无限循环之前，整模板堆栈的最大深度。默认 500。

*--html*

输入文档是 HTML 文件。

*--param name value*

将名称 *name* 和 *value* 的参数以键值对形式表。您可以指定多个名称/值对，最多 32。如果 *value* 是字符串而不是点号，则改用 *--stringparam*。

*--stringparam name value*

名称 *name* 和 *value* 的参数，其中 *value* 是字符串而不是点号。（注意：字符串必须是 utf-8。）

*--nonet*

不要使用互联网来检索 DTD 实体或文档。

*--path paths*

使用 *paths* 指定的文件系统路径的列表（由空格或列分隔）来加载 DTD 实体或文档。

*--load-trace*

向 stderr 显示加载的所有文档。

*--catalogs*

使用 SGML\_CATALOG\_FILES 中指定的 SGML 目录来解析外部实体的位置。默认情况下，xsltproc 查找 XML\_CATALOG\_FILES 中指定的目录。如果未指定，则使用 /etc/xml/catalog。

*--xinclude*

使用 Xinclude 规范处理输入文档。有关该方面的更多信息，请参阅 Xinclude 规范：  
<http://www.w3.org/TR/xinclude/> [<http://www.w3.org/TR/xinclude/>]

*--profile --norman*

输出分析信息，详细说明每个部分所花费的时间在优化表达式性能时很有用。

*--dumpextensions*

将所有已注册扩展名的列表输出到 stdout。

*--nowrite*

拒绝写入任何文件或源。

*--nomkdir*

拒绝创建目录。

*--writesubtree path*



□允□在 *path* 子□内写入文件。

`--nodtdattr`

不要从文档的 DTD □用默认属性。

## Xsltproc 返回□

xsltproc 返回一个状□号，在脚本中□用它□非常有用。

0: 正常

1: 无参数

2: 参数太多

3: 未知□□

4 : 无法解析□式表

5: □式表中的□□

6 : 其中一个文件出□

7: 不支持的 xsl: □出方法

8 : 字符串参数包含引号和双引号

9: 内部□理□□

10 : 通□□止消息停止□理

11 : 无法将□果写入□出文件

## 有关 xsltproc 的更多信息

libxml 网□□<http://www.xmlsoft.org/>[\[http://www.xmlsoft.org/\]](http://www.xmlsoft.org/)

W3C XSLT □面 : <http://www.w3.org/TR/xslt>[\[http://www.w3.org/TR/xslt\]](http://www.w3.org/TR/xslt)

# 仿真器

Eeschema 使用 [ngspice](#) 作为模型引擎提供嵌入式电路仿真器。

使用模型器时您可能会发现官方的 *pspice* 很有用。它包含用于模型的公共符号，如电压源或晶体管，其引脚编号与 ngspice 点序号范相匹配。

有一些演示目录来表明模型器的功能。您将在 *demos/simulation* 目录中找到它们。

## 分配模型

在启模型之前，元件需要分配 Spice 模型。

即使元件由多个元件组成，每个元件也只能分配一个模型。在这种情况下，第一个元件具有指定的模型。

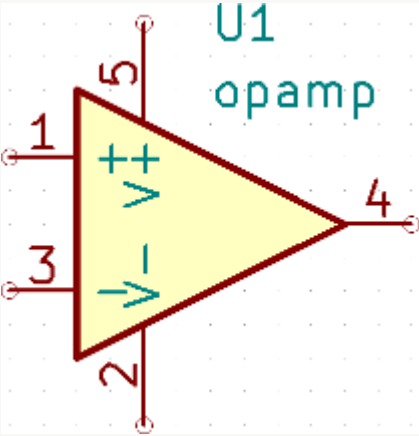
“无源模型”参考匹配 Spice 表示法中的器件类型的无源元件（ $R^*$  表示电阻器， $C^*$  表示电容器， $L^*$  表示电感器）将公式分配模型并使用 `model` 字段确定他的属性。

### NOTE

请记住，在 Spice 表示法中，“M”代表 milli，“Meg”代表 mega。如果您更喜欢使用“M”来表示超前您可以在（模型位置，模型位置框）中请求这样做。

Spice 模型信息作为文本存储在符号字段中，因此您可以在符号器或原理图中定义它。打开符号属性框，然后单击 *Spice 模型* 按钮以打开 Spice 模型器框。

Spice 模型器框有三个用于不同模型类型的卡。所有模型类型共有两个

禁用模型的符号	模型中元件将从模型中排除。
模型点序列	<p>允许模型将符号引脚覆盖模型点映射。要定义不同的映射，模型按模型期的顺序指定引脚号。</p> <p>例子：'+</p> <p>“* 接：”+ “* 1: 非反相输入”</p> <p>“* 2: 反相输入”</p> <p>“* 3: 正电源”</p> <p>“* 4: 负电源”</p> <p>“* 5: 输出”</p> <p>“子电路 tl071 1 2 3 4 5”</p>  <p>要将符号引脚与上面所示的 Spice 模型点相匹配，需要使用具有模型的模型点序列“1 3 5 2 4”。它是与 Spice 模型点顺序的引脚号列表。</p>

## 无源

无源模型卡允许模型将无源器件模型（电阻，电容或电感）分配给元件。它是一个很少使用的模型，因为它通常被元件的模型分配了模型无源模型，模型形，除非元件引用与模型模型不匹配。

### NOTE

明确定义的被模型模型先于模型分配的模型。这意味着一旦分配了被模型模型，在模型期不会考虑参考和模型字段。当指定的模型与原理图上所示的模型不匹配可能会导致混乱的情况。

Spice Model Editor

Passive

Model

Source

Type:Resistor

Passive type

Value:1k

Spice value in simulation

In Spice values,the decimal separator is the point.  
Values can use Spice unit symbols.

Spice unit symbols in values (case insensitive):

f	femto	1e-15
p	pico	1e-12
n	nano	1e-9
u	micro	1e-6
m	milli	1e-3
k	kilo	1e3
meg	mega	1e6
g	giga	1e9
t	tera	1e12

☐ Disable symbol for simulation

☐ Alternate node sequence:

Cancel

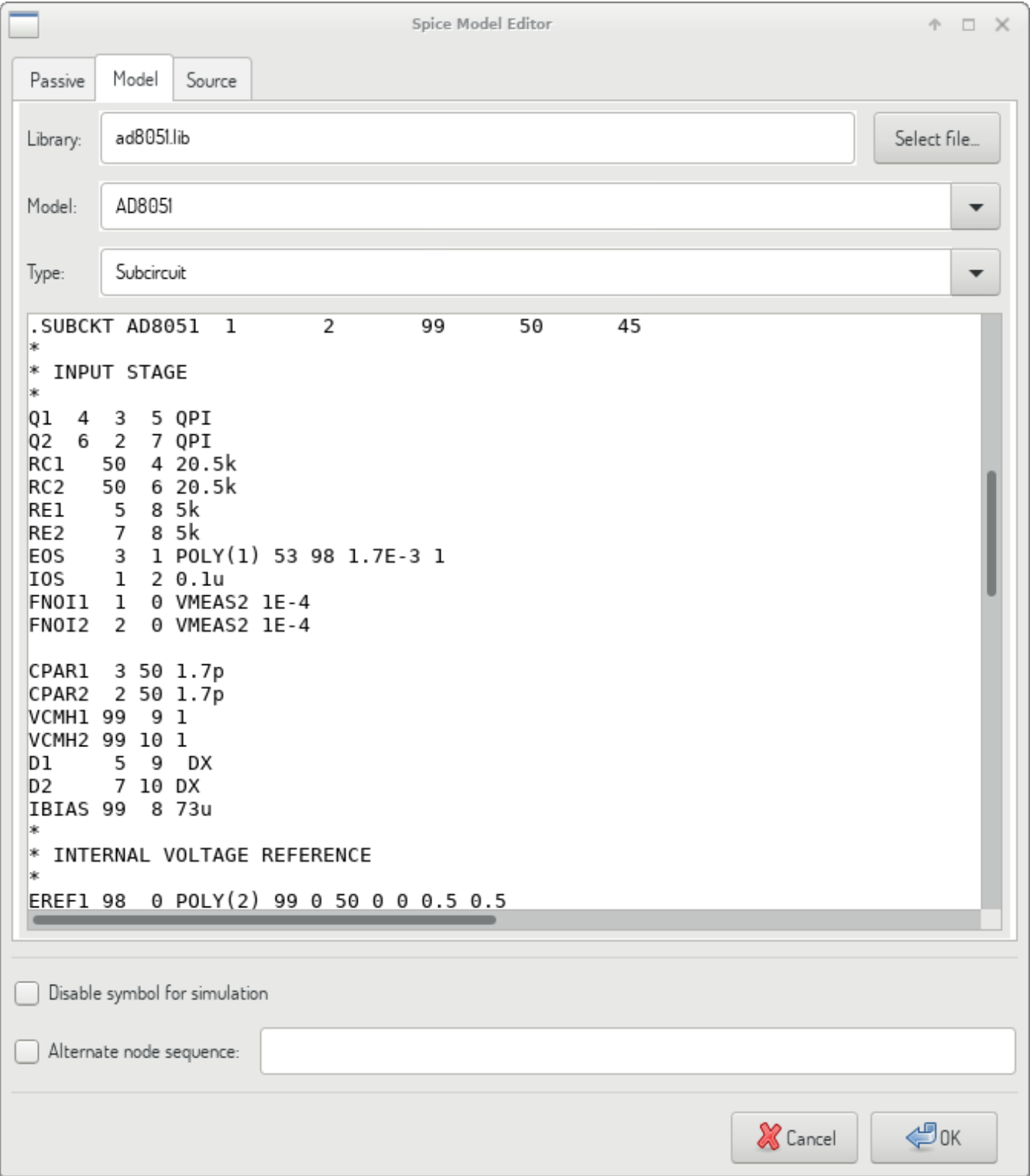
OK

□型	□□器件□型 (□阻, □容或□感)。
□	定义器件属性 (□阻, □容或□感)。□ 可以使用常□的 Spice □元前□□如文本□入字段下方列出的) 和 □□使用点作□小数点分隔符。□注意, Spice 不正确 解□在□中交□的前□□例如 1k5)。

模型

Model □□卡用于分配外部□文件中定义的半□体或复□模型。Spice 模型□通常由□□制造商提供。

主文本小部件显示所的文件内容。将模型描述放在文件中是常见的做法，包括点顺序。

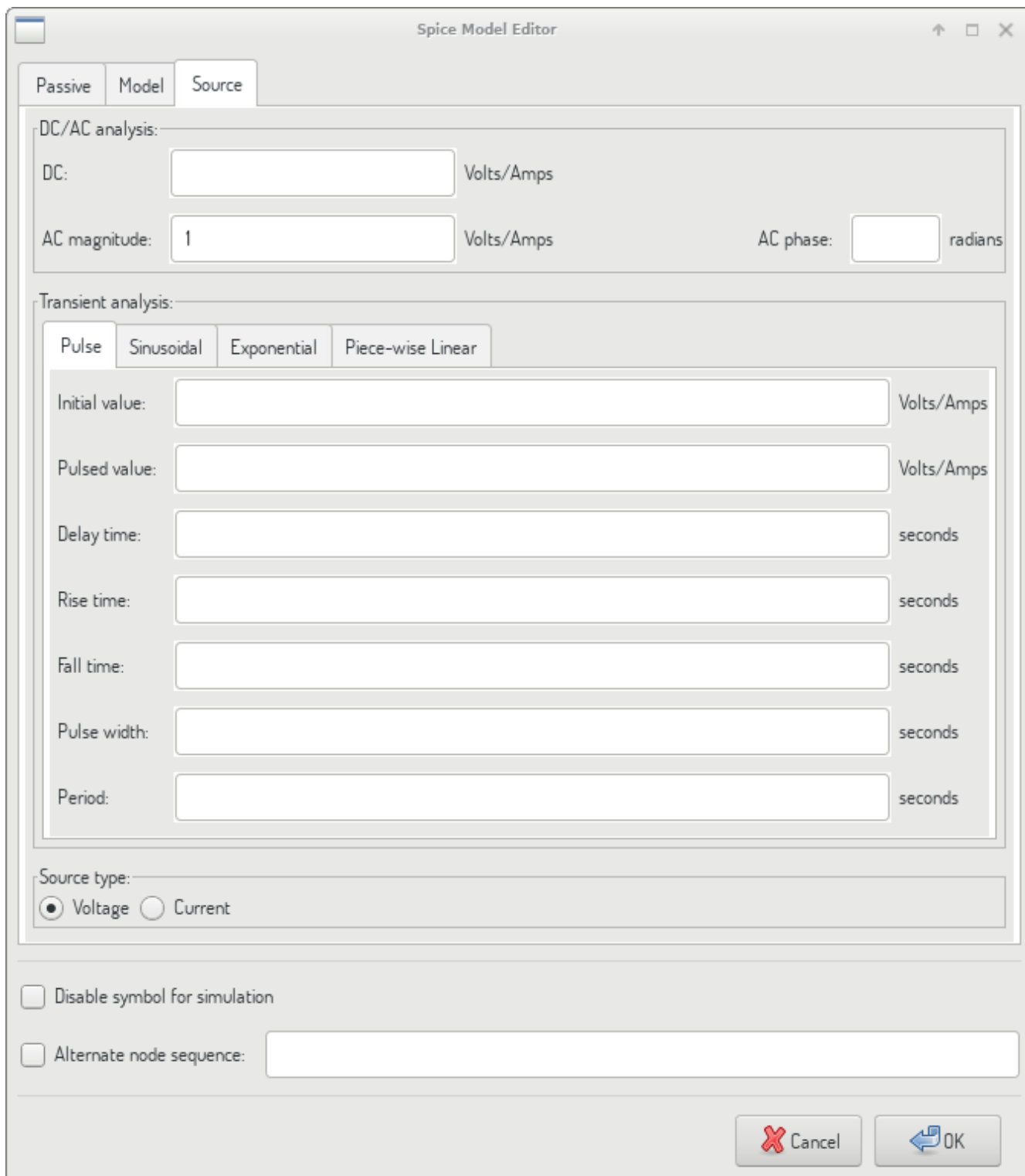


文件	Spice 文件的路径。文件将由模拟器使用，因为它使用 <code>.include</code> 指令添加的。
型号	所型号。文件后，列表将填充可用模型可供
型	型号型（子路，BJT，MOSFET 或二极管）。通常是定的模型自

## 源

Source 卡用于分配源或信号源模型。有两个部分：“DC/AC 分析”和“瞬态分析”。每个都定义了相应模型类型的源参数。

Source type 适用于所有模型。



The image shows the 'Spice Model Editor' dialog box with the 'Source' tab selected. The dialog is divided into two main sections: 'DC/AC analysis' and 'Transient analysis'.

**DC/AC analysis:**

- DC: [ ] Volts/Amps
- AC magnitude: 1 [ ] Volts/Amps
- AC phase: [ ] radians

**Transient analysis:**

Sub-tabs: Pulse (selected), Sinusoidal, Exponential, Piece-wise Linear

- Initial value: [ ] Volts/Amps
- Pulsed value: [ ] Volts/Amps
- Delay time: [ ] seconds
- Rise time: [ ] seconds
- Fall time: [ ] seconds
- Pulse width: [ ] seconds
- Period: [ ] seconds

**Source type:**

- ☒ Voltage ☐ Current

**Options:**

- ☐ Disable symbol for simulation
- ☐ Alternate node sequence: [ ]

**Buttons:** Cancel, OK

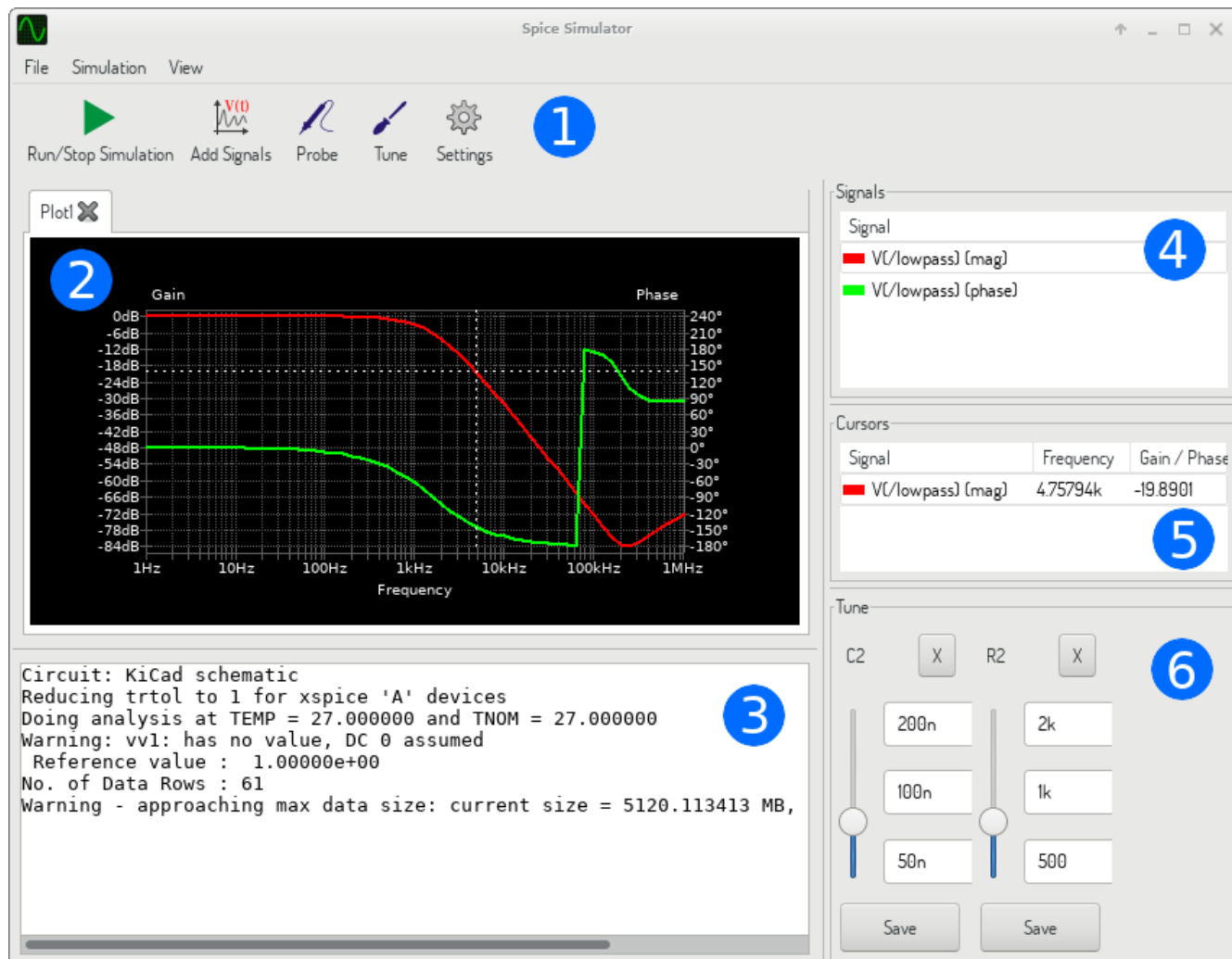
有关源的更多信息，请参考 [ngspice文档](#)，第4章（直流和交流源）。

## Spice 指令

可以通过将 Spice 指令放在原理图工作表的文本字段中来添加它。此方法便于定义默认模型。此功能仅限于以点开  
的 Spice 指令（例如“.tran 10n 1m”），无法使用文本字段放置其他元件。

## 仿真

要启动模型在原理图器窗口中，菜单 **工具** → **仿真** 打开 Spice 仿真框。



该框分为几个部分：

- 《模型-工具》工具
- 《模型面板, 信号面板》
- 《模型输出控制台, 信号输出控制台》
- 《模型信号列表, 信号列表》
- 《模型游标列表, 游标列表》
- 《模型面板, 信号面板》

## 菜单

### 文件

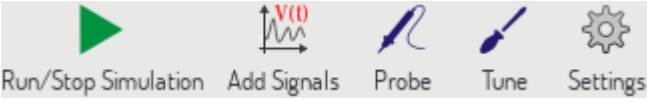
新制	在面板中新建一个新卡。
打开工作簿	打开制信号列表。
保存工作簿	保存制信号列表。
另存像	将活出 .png 文件。
另存 .csv 文件	将活原始数据点出到 .csv 文件。
退出模	关框。

仿真

运行模	使用当前置行模
添加信号.....	打开一个框以要制的信号。
原理探	后原理“模探工具，探”工具。
整元件	后“模工具，”工具。
示 SPICE 网表...	打开一个框，示生成的网表模路。
置...	打开“模置，模置框”。

小	小活
适合屏幕	整放置以示所有
示网格	切网格可性。
示度	切表例可性。

工具



部工具提供最常行的操作的

运行/停止模	后或停止模
添加信号	打开一个框以要制的信号。
探	后原理“模探工具，探”工具。
	后”模工具，“工具。
置	打开“模置，模置框”。



## 面板

将模型可视化可以在单独的卡中打开多个但只有在并行模式才会更新活动。这样就可以比较不同运行的模型。

可以使用“模型菜单位置”菜单切换网格和示例可塑性来自定义。当示例可以拖拽它来改变其位置。

面板交互：

- 鼠标放大/缩小
- 右键打开上下文菜单以调整
- 绘制矩形以放大所区域
- 拖拽光以更改其坐

## 出控制台

出控制台示来自模型器的消息。建立控制台出以确认没有或警告。

## 信号列表

示活中示的信号列表。

信号列表交互：

- 右键打开上下文菜单以隐藏信号或切光
- 双以隐藏信号

## 游列表

示游列表及其坐。每个信号可以示一个光。使用“模型信号列表，信号”列表置游可性。

## 面板

示使用“模型工具，”工具取的元件。面板允许快速修改元件并察它模型果的影响 - 每次更改元件都会重新运行模型并更新形。

于每个元件，有一些控件关

- 部文本字段置最大元件
- 中文本字段置的元件
- 底部文本字段置最小元件
- 滑允用以平滑的方式修改元件
- Save 按钮将原理上的元件修改使用滑的元件
- X 按钮从面板中除元件并恢复其原始

三个文本字段 Spice 元前

## 工具

器工具允许要整的元件。

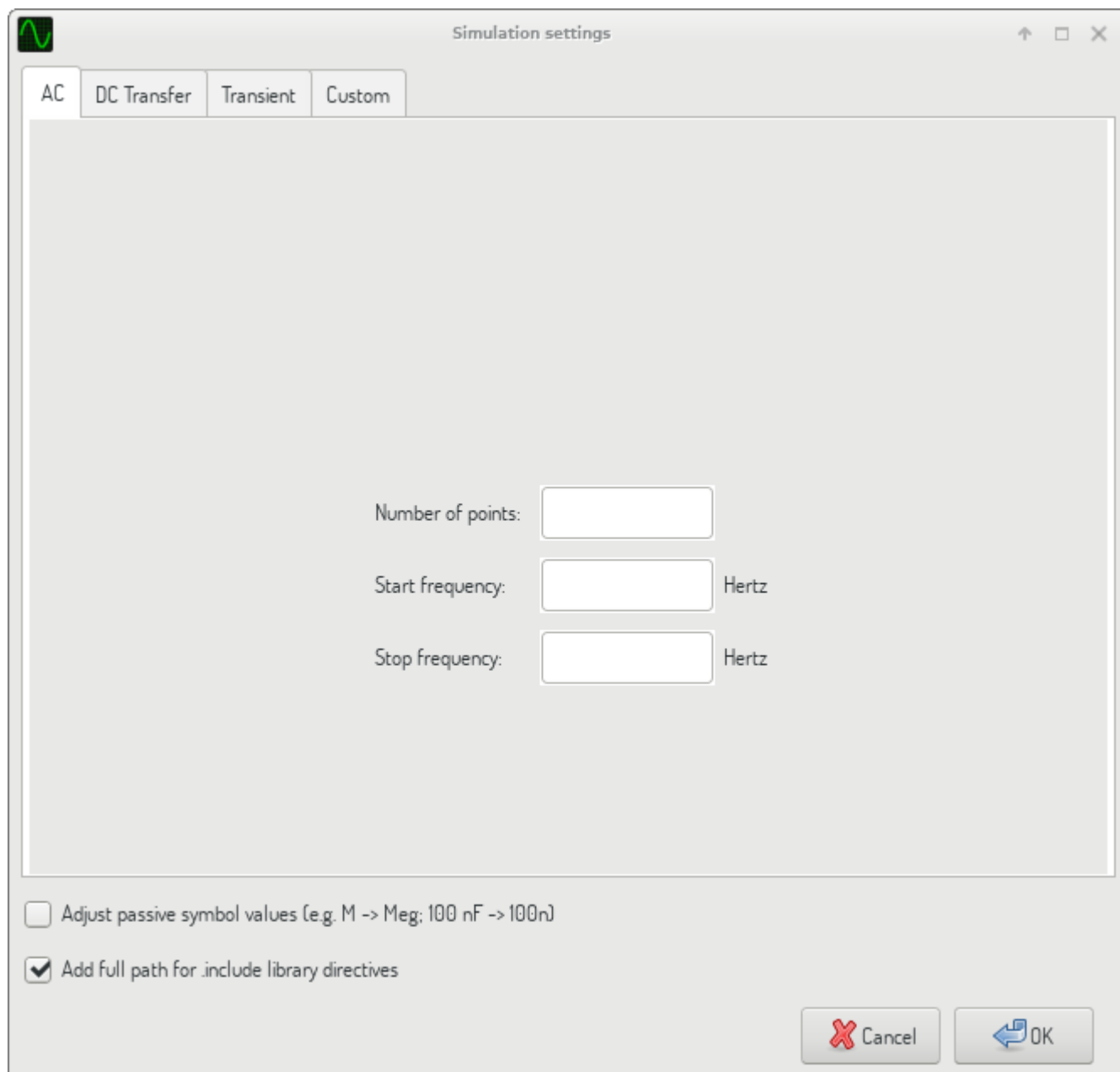
要调整元件，在工具栏中激活原理图器中的一个元件。所选元件将出现在“模型工具，”面板中。只能调整被选元件。

## 探针工具

探针工具提供了一种用户友好的方式来用于信号的。

要向添加信号，在工具栏中激活原理图器中的相应

## 仿真设置

The image shows a "Simulation settings" dialog box. At the top, there is a green waveform icon and the title "Simulation settings". Below the title bar are four tabs: "AC", "DC Transfer", "Transient", and "Custom". The "AC" tab is currently selected. The main area of the dialog is a large, empty rectangular box. Below this box, there are three input fields with labels: "Number of points:" followed by an empty text box; "Start frequency:" followed by an empty text box and the word "Hertz"; and "Stop frequency:" followed by an empty text box and the word "Hertz". At the bottom of the dialog, there are two checkboxes: the first is "Adjust passive symbol values (e.g. M -> Meg; 100 nF -> 100n)" and is unchecked; the second is "Add full path for .include library directives" and is checked. In the bottom right corner, there are two buttons: "Cancel" with a red X icon and "OK" with a blue arrow icon.

模型设置框允许用设置模型型和参数。有四个卡：

- 交流
- 直流
- 短的
- 自定义

前三个卡提供可以指定模型参数的表 最后一个卡允许输入自定义 Spice 指令以配置模型 有关仿真模型和参数的更多信息，参 [ngspice文档](#)，第1.2章。

配置模型的另一种方法是在原理图上的文本字段中输入“模型指令，Spice 指令”。与模型相关的任何文本字段指令都会被框中指令覆盖。意味着一旦开始使用模型框，框将覆盖原理图指令，直到重新打开模型器。

所有模型共有两个

被符号	替被符号以常元件符号表示 Spice 表示法。
.include 指令添加完整路径	Prepend Spice 模型文件名完整路径。通常，ngspice 需要完整路径才能文件。