PCB Editor

The KiCad Team

Table of Contents

Manual de referencia

NOTE	This manual is in the process of being revised to cover the latest stable release version of KiCad. It contains some sections that have not yet been completed. We ask for your patience while our volunteer technical writers work on this task, and we welcome new contributors who would like to help make KiCad's documentation better than ever.

Contribuidores

Jean-Pierre Charras, Fabrizio Tappero, Wayne Stambaugh, Cirilo Bernardo, Jon Evans, Graham Keeth

Traducción

Antonio Morales <antonio1010.mr@gmail.com>, 2015-2016 Iñigo Figuero <ifs@elektroquark.com>, 2016

Realimentacion

The KiCad project welcomes feedback, bug reports, and suggestions related to the software or its documentation. For more information on how to submit feedback or report an issue, please see the instructions at https://www.kicad.org/help/report-an-issue/

Introduction to the KiCad PCB Editor

Initial configuration

When the PCB Editor is run for the first time, if the global footprint table file `fp-lib-table` is not found in the KiCad configuration folder then the KiCad will ask how to create this file:

The first option is recommended (**Copy default global footprint library table (recommended)**). The default footprint library table includes all of the standard footprint libraries that are installed as part of KiCad.

If this option is disabled, KiCad was unable to find the default global footprint library table. This probably means you did not install the standard footprint libraries with KiCad, or they are not installed where KiCad expects to find them. On some systems the KiCad libraries are installed as a separate package.

- If you have installed the standard KiCad footprint libraries and want to use them, but the first option is disabled, select the second option and browse to the `fp-lib-table` file in the directory where the KiCad libraries were installed.

- If you already have a custom footprint library table that you would like to use, select the second option and browse to your `fp-lib-table` file.

- If you want to construct a new footprint library table from scratch, select the third option.

Footprint library management is described in more detail later.

The PCB Editor user interface

The main PCB Editor user interface is shown above, with some key elements indicated:

1. Top toolbars (file management, zoom tools, editing tools)

2. Left toolbar (display options)

3. Message panel and status bar

4. Right toolbar (drawing and design tools)

5. Appearance panel

6. Selection filter panel

Navigating the editing canvas

The editing canvas is a view onto the board being designed. You can pan and zoom to different areas of the board, and also flip the view to show the board from the bottom.

By default, dragging with the middle or right mouse button will pan the canvas view and scrolling the mouse wheel will zoom the view in or out. You can change this behavior in the Mouse and Touchpad section of the preferences (see Configuration and Customization for details).

Several other zoom tools are available in the top toolbar:

- zooms in on the center of the viewport.

- zooms out from the center of the viewport.

- zooms to fit the frame around the drawing sheet.

- zooms to fit the items within the drawing sheet.

-

allows you to draw a box to determine the zoomed area.

The cursor's current position is displayed at the bottom of the window (X and Y), along with the current zoom factor (Z), the cursor's relative position (dx, dy, and dist), the grid setting, and the display units.

The relative coordinates can be reset to zero by pressing `Space`. This is useful for measuring distance between two points or aligning objects.

Hotkeys

The `Ctrl`+`F1` shortcut displays the current hotkey list. The default hotkey list is included in the [Actions Reference](#) section of the manual.

The hotkeys described in this manual use the key labels that appear on a standard PC keyboard. On an Apple keyboard layout, use the `Cmd` key in place of `Ctrl`, and the `Option` key in place of `Alt`.

Many actions do not have hotkeys assigned by default, but hotkeys can be assigned or redefined using the hotkey editor (**Preferences → Preferences… → Hotkeys**).

NOTE	Many of the actions available through hotkeys are also available in context menus. To access the context menu, right-click in the editing canvas. Different actions will be available depending on what is selected or what tool is active.

Hotkeys are stored in the file `user.hotkeys` in KiCad's configuration directory. The location is platform-specific:

- Windows: `%APPDATA%\kicad\8.0\user.hotkeys`

- Linux: `~/.config/kicad/8.0/user.hotkeys`

- macOS: `~/Library/Preferences/kicad/8.0/user.hotkeys`

KiCad can import hotkey settings from a `user.hotkeys` file using the **Import Hotkeys** button in the hotkey editor.

Display and selection controls

Board layers

Layers in the PCB Editor represent physical copper layers on a board, as well as graphical layers used for defining things such as silkscreen, solder mask, and the board edge. There is always one layer that is active in the editor. The active layer is drawn on top of other layers and will be the layer assigned to newly-created objects. The active layer is indicated in the layer selector drop-down box in the top toolbar and is also highlighted in the appearance panel. To change the active layer, you can left-click a layer name in the appearance panel, use the drop-down layer selector in the top toolbar, or use a hotkey. Layers can be hidden to simplify the board view. You can hide a layer even if it is the active layer.

Display order for board layers

The display order for board layers is dynamic and depends on which layer is selected as the active layer. The active layer is always drawn on top of other layers. In addition, layers that are related to the active layer are drawn on top of layers that are unrelated. For example, if you make B.Silkscreen the active layer, then all of the other back layers (B.Cu, B.Adhesive, B.Paste, B.Mask, B.Fab, and B.Courtyard) will be drawn on top of the front, user, and inner copper layers, with B.Silkscreen topmost. If you make Edge.Cuts active, then it will be drawn on top, and the User.* layers and Margin will also be be brought to the front.

> **NOTE** Selected objects are always drawn on top, even if they are not on the active layer.

The appearance panel

The appearance panel provides controls to manage the visibility, color, and opacity of objects in the PCB Editor's drawing canvas. It has three tabs: the Layers tab contains controls for the board layers, the Objects tab contains controls for different types of graphical objects, and the Nets tab contains controls for the appearance of the ratsnest and copper items.

Layer controls

In the Layers tab of the appearance panel, each board layer is shown with its color and visibility state. The active layer is shown highlighted with an arrow indicator to the left of the color swatch. Left-click on a layer to choose it as the active layer. Left-click on the corresponding visibility icon to toggle the layer between visible and hidden. Double-click or middle-click on the color swatch to change the layer's color.

> **NOTE** You must first create a custom color theme in Preferences before you can change layer colors in the appearance panel.

Below the list of layers is an expandable panel that contains layer display options. The first setting controls how non-active layers are displayed: normal, dimmed, or hidden. The layer display mode can be used to simplify the view and focus on a single layer. Items on inactive layers cannot be selected when the non-active layer display mode is "Dim" or "Hide". You can use the hotkey `Ctrl` + `H` to cycle through these display modes quickly.

Flip board view will show the board as if you are looking from the bottom (that is, mirrored around the Y-axis). This option is also available in the View menu.

Object controls

The Objects tab of the appearance panel is similar to the Layers tab. The main differences are that some objects have no color setting and that four types of objects (tracks, vias, pads, and zones) have opacity control sliders. The opacity setting here will be multiplied with any opacity set in the layer colors. By default, all objects are fully opaque except for zones, which are set to translucent in order to make it easier to see objects through filled zone areas.

Layer presets

Layer presets store which layers and objects are visible and hidden for easy recall. There are several built-in layer presets and you can save your own custom presets. Custom presets are stored in the project settings for a board, as presets may be specific to a certain board stackup.

To load a preset, choose it from the Presets drop-down menu at the bottom of the appearance panel or use the quick switcher by holding down `Ctrl` and pressing `Tab`. Once the quick switcher window appears, you can press `Tab` and `Shift`+`Tab` to cycle through the available presets. When you let go of the `Ctrl` key, the highlighted preset will be loaded.

To save a custom preset, first use the visibility controls to choose which layers you want visible, then choose **Save preset...** from the Presets drop-down menu. Give your preset a name and it will now be available via the drop-down menu and the quick switcher. To modify a custom preset, follow the same process and save the modified version with the same name to overwrite the existing version. To delete a custom preset, choose the **Delete preset...** option from the drop-down menu and select the preset to be deleted from the list.

Viewports

Viewports store the current view location and zoom level so you can quickly switch back to it later, or switch between several saved views.

To load a viewport, choose it from the Viewports drop-down menu at the bottom of the appearance panel or use the quick switcher by holding down `Shift` and pressing `Tab`. Once the quick switcher window appears, you can press `Tab` to cycle through the stored viewports. When you let go of the `Shift` key, the highlighted viewport will be loaded.

To save a new viewport, scroll and zoom to show the desired area of the board, then choose **Save viewport...** from the Viewports drop-down menu. Give your viewport a name and it will now be available via the drop-down menu and the quick switcher. To modify an existing viewport, save a new viewport with the same name to overwrite the existing version. To delete a viewport, choose the **Delete viewport...** option from the drop-down menu and select the preset to be deleted from the list.

Net and net class controls

The Nets tab of the appearance panel shows a list of all nets and net classes in the board. Each net has a visibility control that controls the visibility of that net in the ratsnest. Hiding nets in the ratsnest does not change the connectivity of the board and will not impact the design rule checker; it only is intended to make the ratsnest easier to understand.

Each net and net class can also have a color assigned. By default, this color applies to the ratsnest lines for the net (or for all the nets in the net class). Nets have no color by default; this is indicated by a checkerboard pattern in the color swatch. Double-click or right-click a net or net class color swatch to set the color.

NOTE	The Default net class cannot have a color assigned, as nets in this class will just use the default ratsnest color defined by the color theme.

You can also select and highlight nets and net classes via the appearance panel: right-click on a net or net class to show these options in a menu.

Below the list of net classes is an expandable panel that contains net display options. The first option controls how net colors are applied. When "All" is selected, all copper items (pads, tracks, vias, and zones) belonging to a net or net class will take on the chosen color. When "Ratsnest" is selected (the default value), only the ratsnest is affected by net and net class colors. When "None" is selected, net and net class colors are ignored.

The second option controls how ratsnest lines are drawn. "All layers" means that ratsnest lines will be drawn between all unconnected items. "Visible layers" means that no ratsnest lines will be drawn to items that are on hidden layers, even when those items are unconnected.

NOTE	You can configure the thickness of ratsnest lines in the PCB Editor Editing Options section of the Preferences dialog, to make the ratsnest more or less visible.

Selection and the selection filter

Selecting items in the editing canvas is done with the left mouse button. Single-clicking on an object will select it and dragging will perform a box selection. A box selection from left to right will only s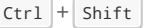elect items that are fully inside the box. A box selection from right to left will select any items that touch the box. A left-to-right selection box is drawn in yellow, with a cursor that indicates exclusive selection, and a right-to-left selection box is drawn in blue with a cursor that indicates inclusive selection.

The selection action can be modified by holding modifier keys while clicking or dragging. The following modifier keys apply when clicking to select single items:

Modifier Keys (Windows)	Modifier Keys (Linux)	Modifier Keys (macOS)	Selection Effect
`Ctrl`	`Ctrl`	`Cmd`	Toggle selection. Note: `Ctrl`+click can be remapped to highlight net in **Preferences → PCB Editor → Editing Options**.
`Shift`	`Shift`	`Shift`	Add the item to the existing selection.
`Ctrl` + `Shift`	`Ctrl` + `Shift`	`Cmd` + `Shift`	Remove the item from the existing selection.
long click	long click or `Alt`	long click or `Option`	Clarify selection from a pop-up menu.

The following modifier keys apply when dragging to perform a box selection:

Modifier Keys (Windows)	Modifier Keys (Linux)	Modifier Keys (macOS)	Selection Effect
`Ctrl`	`Ctrl`	`Cmd`	Toggle selection.
`Shift`	`Shift`	`Shift`	Add item(s) to the existing selection.
`Ctrl` + `Shift`	`Ctrl` + `Shift`	`Cmd` + `Shift`	Remove item(s) from the existing selection.

The selection filter panel in the lower right corner of the PCB Editor window controls which types of objects can be selected with the mouse. Turning off selection of unwanted object types makes it easier to select items in a dense board. The "All items" checkbox is a shortcut to turn the other items on and off. The "Locked items" checkbox is independent of the rest, and controls whether or not items that have been locked can be selected. You can right-click any object type in the selection filter to quickly change the filter to only allow selecting that type of object.

When a connected copper item is selected, you can expand the selection to other copper items of the same net using the Expand Selection command in the right-click context menu or with the hotkey `U`. The first time you run this command, the selection will be expanded to the nearest pad. The second time, the selection will be expanded to all connected items on all layers.

Selecting an object displays information about the object in the message panel at the bottom of the window. Double-clicking an object opens a window to edit the object's properties.

Pressing `Esc` will always cancel the current tool or operation and return to the selection tool. Pressing `Esc` while the selection tool is active will clear the current selection.

Net highlighting

An electrical net (or set of nets) can be highlighted in the PCB editor to visualize how the net is routed across the PCB. Net highlighting can be activated by selecting the net to highlight in the PCB editor or by selecting the corresponding net in the schematic editor when cross-probe highlighting is enabled (see below). When net highlighting is active, the highlighted net or nets will be shown in a brighter color and all other items will be shown in a dimmer color than normal.

There are three ways to select a net or nets to highlight in the PCB editor: by using the hotkey `` ` `` after selecting a copper object, by using the context menu of any copper object, and by using the context menu of the Nets tab of the Appearance panel. When you press the Highlight Net hotkey, the nets of any selected copper items will be highlighted. If no copper items are selected, the net of the copper item under the editor cursor will be highlighted.

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

When a net or nets have been selected for highlighting, the Toggle Net Highlighting action becomes enabled on the left toolbar (also accessible by hotkey, `Ctrl` + `` ` ``). This action will turn the highlighting display on or off without choosing a new net to highlight.

Cross-probing from the schematic

KiCad allows bi-directional cross-probing between the schematic and the PCB. There are several different types of cross-probing.

Selection cross-probing allows you to select a symbol or pin in the schematic to select the corresponding footprint or pad in the PCB (if one exists) and vice-versa. By default, cross-probing will result in the display centering on the cross-probed item and zooming to fit. You can disable the centering and zooming behavior, or disable selection cross-probing entirely, in the Display Options section of the Preferences dialog. Even when selection cross-probing is disabled, you can manually cross-probe from the schematic to the PCB by right-clicking an object and selecting **Select on PCB**, or from the PCB to the schematic by right-clicking an object and choosing **Select → Select on Schematic**.

Highlight cross-probing allows you to highlight a net in the schematic and PCB at the same time. If the option "Highlight cross-probed nets" is enabled in the Display Options section of the Preferences dialog, highlighting a net or bus in the schematic editor will cause the corresponding net or nets to be highlighted in the PCB editor.

Left toolbar display controls

The left toolbar provides options to change the display of items in the PCB Editor.

	Turns grid display on/off. **Note:** by default, hiding the grid does not disable grid snapping. This behavior can be changed in the Display Options section of Preferences.
	Turns item-specific grid overrides on/off.
	Switch between polar and Cartesian coordinate display in the status bar.
	Display/entry of coordinates and dimensions in inches, mils, or millimeters.
	Switches between full-screen and small editing cursor (crosshairs).
	Switches between free angle and 45 degree mode for placement of new tracks, zones, graphical shapes, dimensions, and other objects. You can also toggle between free angle and 45 degree mode using `Shift` + `Space`.
	Turns the ratsnest display on/off.
	Switches between straight and curved ratsnest lines.
	Switches the non-active layer display mode between Normal and Dim. **Note:** this button will be highlighted when the non-active layer display mode is either Dim or Hide. In both cases, pressing the button will change the layer display mode to Normal. The Hide mode can only be accessed via the controls in the Appearance Panel or via the hotkey `Ctrl` + `H`.
	When a net has been selected for highlighting, switches the highlighting on or off. **Note:** this button will be disabled when no net has been highlighted. To highlight a net, use the hotkey `` ` ``, right-click any copper object in the net and choose Highlight Net from the Net Tools menu, or right-click the net in the list in the Nets tab of the Appearance panel.
	Show zone filled areas.
	Show zone outlines only.
	Switches display of pads between filled and outline mode.
	Switches display of vias between filled and outline mode.
	Switches display of tracks between filled and outline mode.
	Shows or hides the Appearance and Selection Filter panels on the right side of the editor.
	Shows or hides the Properties Manager panel on the left side of the editor.

Creating a PCB

Basic PCB concepts

A printed circuit board in KiCad is generally made up of **footprints** representing electronic components and their pads, **nets** defining how those pads connect to each other, **tracks**, **vias**, and **filled zones** that form the copper connections between pads in each net, and various graphic shapes defining the board edge, silkscreen markings, and any other desired information.

KiCad normally keeps the information about nets on a PCB synchronized with an associated schematic, but nets can also be created and edited directly within the PCB editor.

Capabilities

KiCad is capable of creating printed circuit boards with up to 32 copper layers, 14 technical layers (silkscreen, solder mask, component adhesive, solder paste, etc), and 13 general-purpose drawing layers.

The internal measurement resolution of all objects in KiCad is 1 nanometer, and measurements are stored as 32-bit integers. This means it is possible to create boards up to approximately 4 meters by 4 meters.

KiCad currently supports one board file per project / schematic.

Starting from a schematic

Creating a board from a schematic is the recommended workflow for KiCad. When you create a new project, KiCad will generate an empty board file with the same name as the project. To start designing the board after you have created a schematic, simply open the board file. You can do this either from the KiCad project manager, or by clicking the "Open PCB in board editor" button in the schematic editor. To import the schematic design information into the board editor, including footprints and net connections, use the **Tools → Update PCB from Schematic…** action (F8). You can also use the icon in the top toolbar.

NOTE	Update PCB from Schematic is the preferred way to transfer design information from the schematic to the PCB. In older versions of KiCad, the equivalent process was to export a netlist from the Schematic Editor and import it into the Board Editor. It is no longer necessary to use a netlist file.

For more information about the Update Schematic from PCB tool, see the forward annotation section of the manual.

Starting from scratch

It is also possible to create a board with no matching schematic, although this workflow has some limitations and is not recommended for most users. To do this, you must start the PCB editor standalone (not from the KiCad project manager). Before beginning your design, it is a good idea to save the board file, which will also create a project file to store board settings. Use "Save As..." from the File menu to choose where to save your board file. A project file with the same name will be created in the same location you choose to save the board file in.

Board setup

Before beginning your board design, use the Board Setup dialog to configure the basic parameters of the board. To open Board Setup, click the [icon] icon in the top toolbar or choose "Board Setup..." from the File menu.

Configuring board stackup and physical parameters

There are two sections of Board Setup used to configure the stackup and layers of the board. The Board Editor Layers section is used to enable or disable technical (non-copper) layers, and give custom names to layers if desired. The Physical Stackup section is used to configure the number of copper layers, as well as the physical parameters of the copper and dielectric layers such as thickness and material type. Dielectric,

soldermask, and silkscreen layers can have colors assigned to them, which affects the board's appearance in the 3D viewer.

To configure the board stackup, start on the Physical Stackup section:

Set the number of copper layers in the upper left corner and then enter the physical parameters of the stackup if desired. These parameters may be left at their default values, but note that the board thickness value will be used when exporting a 3D model of the board, and layer thicknesses will be included in net length calculations for any nets that include vias. If you plan to use these features, it is a good idea to ensure that the stackup thickness is correct.

> **NOTE**   KiCad currently only supports stackups with an even number of copper layers. To create designs with an odd number of layers (for example, flexible printed circuits or metal-core printed circuits), simply choose the next highest even number and ignore the extra layer.

Next, if desired, use the Board Editor Layers section to rename layers or hide non-copper layers that you will not be using in the design. For example, if you will not use a back silkscreen on the design, uncheck the box next to the `B.Silkscreen` layer.

	F.Courtyard	Off-board, testing
☑	F.Fab	Off-board, manufacturing
☑	F.Adhesive	On-board, non-copper
☑	F.Paste	On-board, non-copper
☑	F.Silkscreen	On-board, non-copper
☑	F.Mask	On-board, non-copper
☑	F.Cu	signal
☑	B.Cu	signal
☑	B.Mask	On-board, non-copper
☑	B.Silkscreen	On-board, non-copper
☑	B.Paste	On-board, non-copper
☑	B.Adhesive	On-board, non-copper
☑	B.Fab	Off-board, manufacturing
	B.Courtyard	Off-board, testing
	Edge.Cuts	Board contour
	Margin	Edge_Cuts setback
☑	User.Eco1	Auxiliary
☑	User.Eco2	Auxiliary
☑	User.Comments	Auxiliary

NOTE	Copper layers can be designated as signal, power plane, mixed, or jumper in the Board Editor Layers section. This designation is intended as a guide for the user only. Tracks and zones can be routed on any copper layer, no matter what the type is configured to in this dialog.

Some additional board stackup settings are found on the Board Finish and Solder Mask/Paste sections of the Board Setup dialog. The Board Finish section has settings for defining the copper finish and special features such as castellations or edge plating. Note that these settings only impact the board attributes output as part of Gerber job files at this time.

The Solder Mask/Paste section allow global adjustment of the clearance (positive or negative) between the copper shapes and solder mask / solder paste shapes of pads on the board. These values will be added to any clearance overrides set on individual footprints or pads. Positive clearance values will result in the shape of the solder mask or paste opening being *larger* than the copper shape. Negative clearance values will result in the opening being *smaller* than the copper shape.

WARNING	Most commercial PCB fabricators expect these values to be zero and make their own adjustments to solder mask and paste openings as part of their CAM process. It is usually best to leave these values at their default of zero unless you are making the PCB yourself or have specific advice from your fabricator to use different values.

Configuring default text and graphic settings

The Text & Graphics Defaults section of the Board Setup dialog can be used to configure the properties that will be used for new text and graphic shapes that are placed on the board.
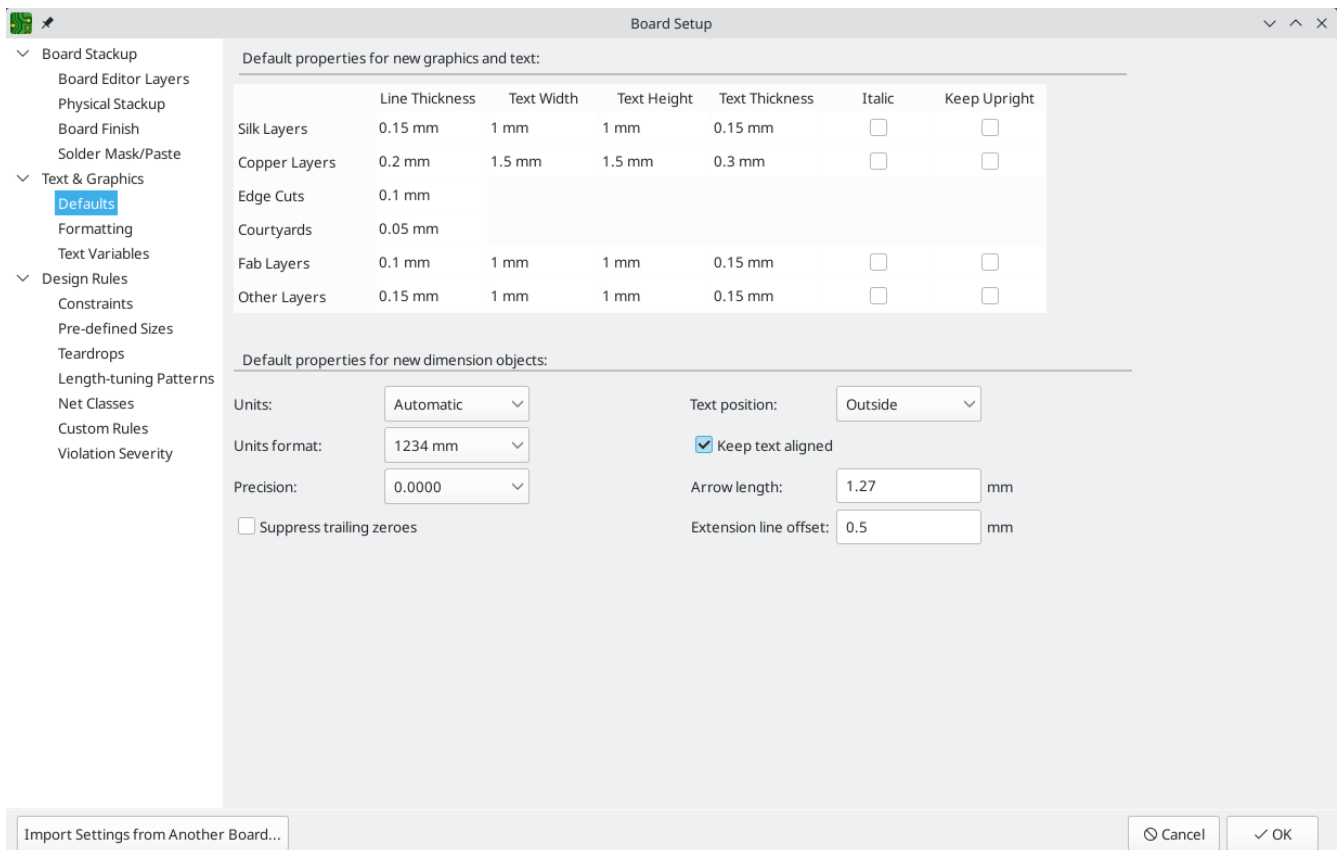
Line thickness, text size, and text appearance can be configured for the six different categories of layers shown in the dialog. Additionally, the properties for dimension objects can be configured for all layers. For more details about dimension properties, see the Dimensions section below.

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

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

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

Text variables can also be created in Schematic Setup. Text variables are project-wide; variables created in the schematic editor are also available in the board editor, and vice versa.

There are also a number of built-in system text variables.

Configuring design rules

Design rules control the behavior of the interactive router, the filling of copper zones, and the design rule checker. Design rules can be modified at any time, but we recommend that you establish all known design rules at the beginning of the board design process.

Constraints

Basic design rules are configured in the Constraints section of the Board Setup dialog. Constraints in this section apply to the entire board and should be set to the values recommended by your board manufacturer. Any minimum value set here is an *absolute* minimum and cannot be overridden with a more specific design rule. For example, if you need the copper clearance on part of a board to be 0.2mm and in the rest 0.3mm, you must enter 0.2mm for the minimum copper clearance in the Constraints section and use a netclass or custom rule to set the larger 0.3mm clearance.

In addition to setting minimum clearances, a number of features can be configured here:

Setting	Description
Arc/circle approximated by segments	In some situations, KiCad must use a series of straight line segments to approximate round shapes such as those of arcs and circles. This setting controls the maximum error allowed by this approximation: in other words, the maximum distance between a point on one of these line segments and the true shape of the arc or circle. Setting this to a lower number than the default value of 0.005mm will result in smoother shapes, but can be very slow on larger boards. The default value typically results in arc approximation error that is not detectable in the manufactured board due to manufacturing tolerances.
Allow fillets outside zone outline	Zones can have fillets (rounded corners) added in the Zone Properties dialog. By default, no zone copper, including fillets, is allowed outside the zone outline. This effectively means that inside corners of the zone outline will not be filleted even when a fillet is configured. By enabling this setting, inside corners of the zone outline will be filleted even though this results in copper from the zone extending outside the zone outline.
Minimum thermal relief spoke count	This sets the minimum acceptable number of thermal relief spokes connecting a pad to a zone. A DRC violation will be generated if this constraint is violated.
Include stackup height in track length calculations	By default, the length tuner uses the height of the stackup to calculate the additional length of a track that travels through vias from one layer to another. This calculation relies on the board stackup height being correctly configured. In some situations, it is preferable to ignore the height of vias and just calculate the track length assuming that vias add no length. Disabling this setting will exclude via length from length tuner track length calculations.

Pre-defined Sizes

The pre-defined sizes section allows you to define the track and via dimensions you want to have available while routing tracks. Net classes can be used to define the default dimensions for tracks and vias in different nets (see below) but defining a list of sizes in this section will allow you to step through these sizes while routing. For example, you may want the default track width on a board to be 0.2 mm, but use 0.3 mm for some sections that carry more current, and 0.15 mm for some sections where space is limited. You can define each of these track widths in the Board Setup dialog and then switch between them when routing traces.
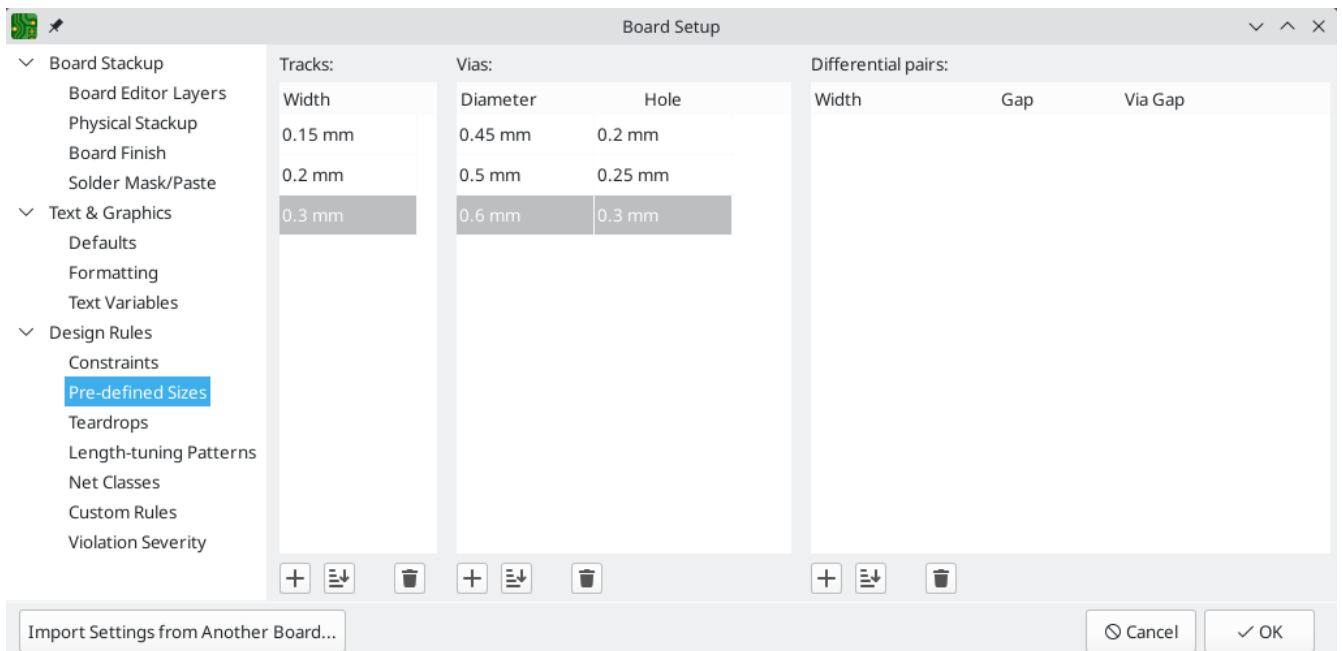
Teardrops

The teardrops section lets you set default parameters for various types of teardrops. There are different settings for teardrop connections to round objects, rectangular objects, and teardrop connections between tracks. The default teardrop parameters can be overridden when teardrops are added, and also changed in the properties for individual connected items. See the teardrops documentation for more information.
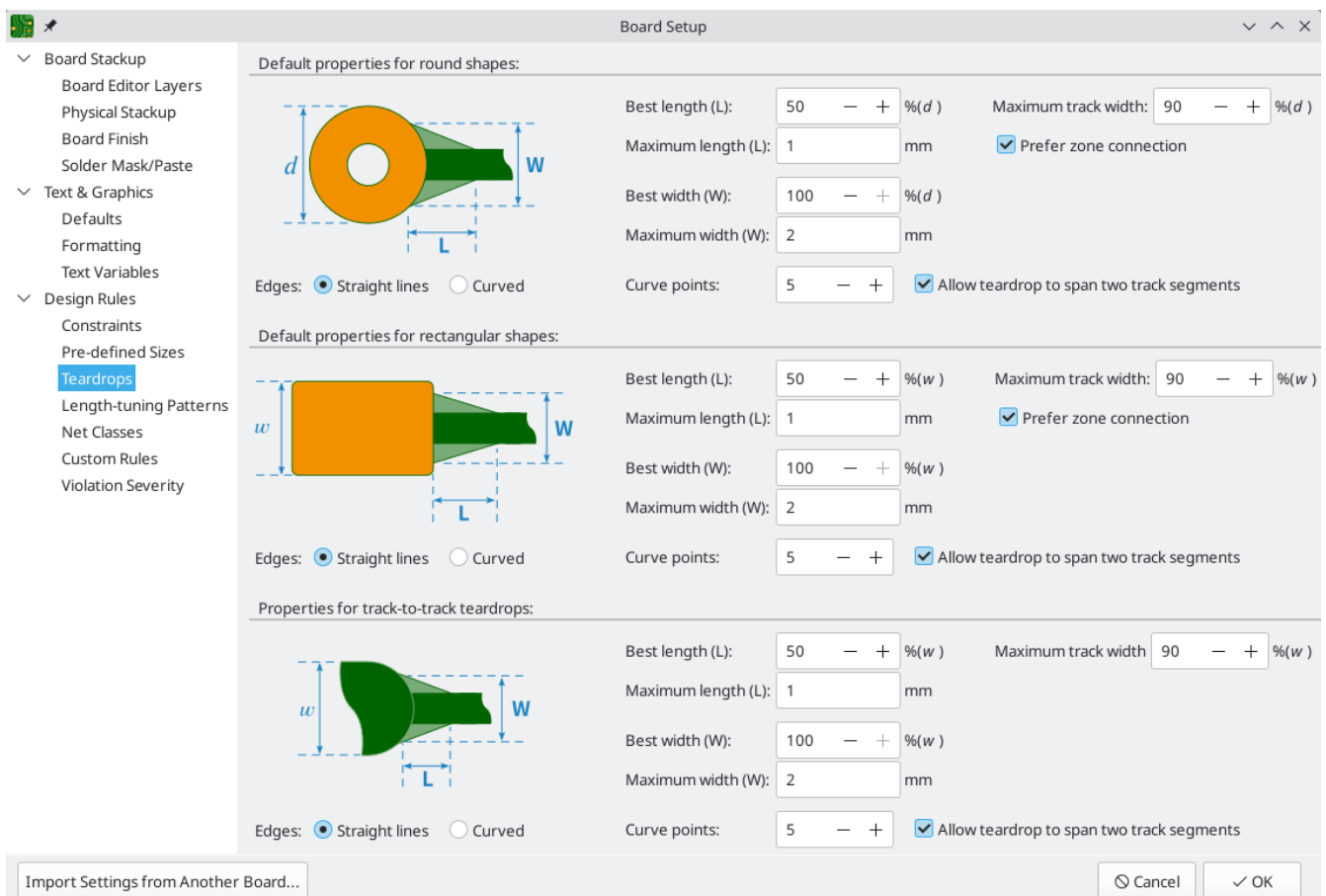
Length-tuning patterns

The length-tuning patterns section lets you set default parameters for each type of length-tuning pattern (single-track length, differential-pair length, and differential-pair skew). These defaults can be overridden in the properties of each tuning pattern added to the board. See the length tuning documentation for more ifnormation.

Net Classes

The Net Classes section allows you to configure routing and clearance rules for different classes of nets. In KiCad, each net is part of exactly one net class. If you do not add a net to a specific class, it will be part of the Default class, which always exists. Net classes may be created and edited in either the Schematic or Board Setup dialogs.

The upper portion of the Net Classes section contains a table showing the net classes in the design and the design rules that apply to each net class. Every class has values for copper clearance, track width, via sizes, and differential pair sizes. These values will be used when creating tracks and vias unless a more specific rule overrides them (see Custom Rules below).

NOTE	No rule may override the minimum values set in the Constraints section of Board Setup. For example, if you set a net class clearance to `0.1 mm`, but the Minimum Clearance in the Constraints section is set to `0.2 mm`, nets in that class will have a clearance of `0.2 mm`.

The track widths and via sizes defined for each net class are used when the track width and via size controls are set to "use netclass values" in the PCB editor. These widths and sizes are considered the default, or optimal, sizes for that netclass. They are not minimum or maximum values. Manually changing the track width or via size to a different value from that defined in the Net Classes section will not result in a DRC violation. To restrict track width or via size to specific values, use Custom Rules.

The lower portion of the Net Classes section lists pattern-based netclass assignments. Working with pattern-based netclass assignments is explained in the Schematic Editor documentation; pattern-based assignments can be edited in either the Board or Schematic Setup windows.

Note that pattern-based assignments can be created directly from the PCB editing canvas by right clicking a copper track or zone and clicking **Assign netclass…**. Netclasses can also be assigned in the schematic using netclass directives or labels instead of pattern-based assignments.

Custom Rules

The Custom Rules section contains a text editor for creating design rules using the custom rules language. Custom rules are used to create specific design rule checks that are not covered by the basic constraints or net class settings.

Custom rules will only be applied if there are no errors in the custom rules definitions. Use the Check Rule Syntax button to test the definitions and fix any problems before closing Board Setup.

See Custom Design Rules in the Advanced Topics chapter for more information on the custom rules language as well as example rules.

Violation Severity

The Violation Severity section allows you to configure the severity of each type of design rule check. Each rule may be set to create an error marker, a warning marker, or no marker (ignored).

NOTE	Individual rule violations may be ignored in the Design Rule Checker. Setting a rule to Ignore in the Violation Severity section will completely disable the corresponding design rule check. Use this setting with caution.

Importing settings

You can import part or all of the board setup from an existing board. This technique can be used to create a "template" board that has the settings you want to use on multiple designs, and then importing these settings from the template board into each new board rather than entering them manually.
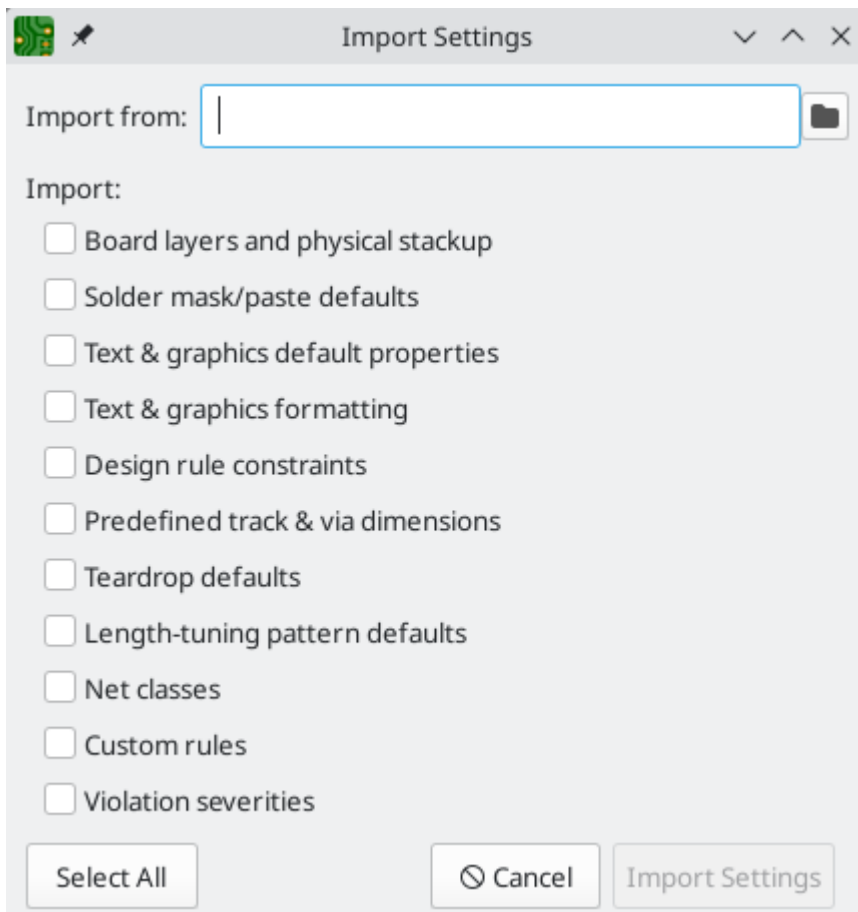
To import settings, click the Import Settings from Another Board… button at the bottom of the Board Setup dialog and then choose the `kicad_pcb` file you want to import from. Select which settings you want to import and the current settings will be overwritten with the values from the chosen board.

Editing a board

Placement and drawing operations

Placement and drawing tools are located in the right toolbar. When a tool is activated, it stays active until a different tool is selected or the tool is canceled with the `Esc` key. The selection tool is always activated when any other tool is canceled.

Some toolbar buttons have more than one tool available in a palette. These tools are indicated with a small arrow in the lower-right corner of the button:

To show the palette, you can click and hold the mouse button on the tool or click and drag the mouse. The palette will show the most recently used tool when it is closed.

	Selection tool (the default tool).
	Local ratsnest tool: when the board ratsnest is hidden, selecting footprints with this tool will show the ratsnest for the selected footprint only. Selecting the same footprint again will hide its ratsnest. The local ratsnest setting for each footprint will remain in effect even after the local ratsnest tool is no longer active.
	Footprint placement tool: click on the board to open the footprint chooser, then click again after choosing a footprint to confirm its location.
	Route tracks / route differential pairs: These tools activate the interactive router and allow placing tracks and vias. The interactive router is described in more detail in the Routing Tracks section below.
	Tune length: These tools allow you to tune the length of single tracks or the length or skew of differential pairs, after they have been routed. See the Routing Tracks section for details.
	Add vias: allows placing vias without routing tracks. Vias placed on top of tracks using this tool will take on the net of the closest track segment and will become part of that track (the via net will be updated if the pads connected to the tracks are updated). Vias placed anywhere else will take on the net of a copper zone at that location, if one exists. These vias will not automatically take on a new net if the net of the copper zone is changed.
	Add filled zone: Click to set the start point of a zone, then configure its properties before drawing the rest of the zone outline. Zone properties are described in more detail below.
	Add rule area: Rule areas, formerly known as keepouts, can restrict the placement of items and the filling of zones and can also define named areas to apply specific custom design rules to.
	Draw lines. **Note:** Lines are graphical objects and are not the same as tracks placed with the Route Tracks tool.

⌒	Draw arcs: pick the center point of the arc, then the start and end points. By right clicking this button, you can change the arc editing mode between a mode that maintains the existing arc center and a mode that maintains the arc radius.
▢	Draw rectangles. Rectangles can be filled or outlines.
◯	Draw circles. Circles can be filled or outlines.
⟩	Draw graphical polygons. Polygons can be filled or outlined. **Note:** Filled graphical polygons are not the same as filled zones: graphical polygons cannot be assigned to a net and will not keep clearance from other items.
🖼	Add bitmap image for reference. Reference images are not included in fabrication outputs.
T	Add text.
▤	Add a textbox.
⟋	Add dimensions. Dimension types are described in more detail below.
↳	
✛	
✛	
⟍ⓝ	
▸	Deletion tool: click objects to delete them.
⦂⦂	Set grid origin or drill/place origin (used for fabrication outputs).
↳	
⟋	Interactively measure the distance between two points.

Grids and snapping

When moving, dragging, and drawing board elements, the grid, pads, and other elements can have snapping points depending upon the settings in the user preferences. In complex designs, snap points can be so close together that it makes the current tool action difficult. Both grid and object snapping can be disabled while moving the mouse by using the modifier keys in the table below.

NOTE	On Apple keyboards, use the `Cmd` key instead of `Ctrl`.

Modifier Key	Effect
`Ctrl`	Disable grid snapping.
`Shift`	Disable object snapping.

Tools only snap to objects on visible layers. You can reduce unwanted snapping points by hiding unneeded layers or using the single-layer view mode. Additionally, you can toggle between snapping to objects on all layers or only snapping to objects on the current layer by pressing `Shift` + `S`.

Snapping to different types of objects (pads, tracks, and graphics) can be configured in the Editing Options section of the PCB Editor preferences.

You can adjust the grid size using the grid dropdown in the top toolbar or by right-clicking and selecting a new grid from the list in the **Grid** submenu. Pressing the `n` or `N` hotkeys will cycle to the next and previous grid in the list, respectively.

You can also select a new grid or edit the available grids in the **Grids** pane of the preferences dialog. As a shortcut to reach this dialog, right click the ⣿ button on the left toolbar and select **Edit Grids…**.
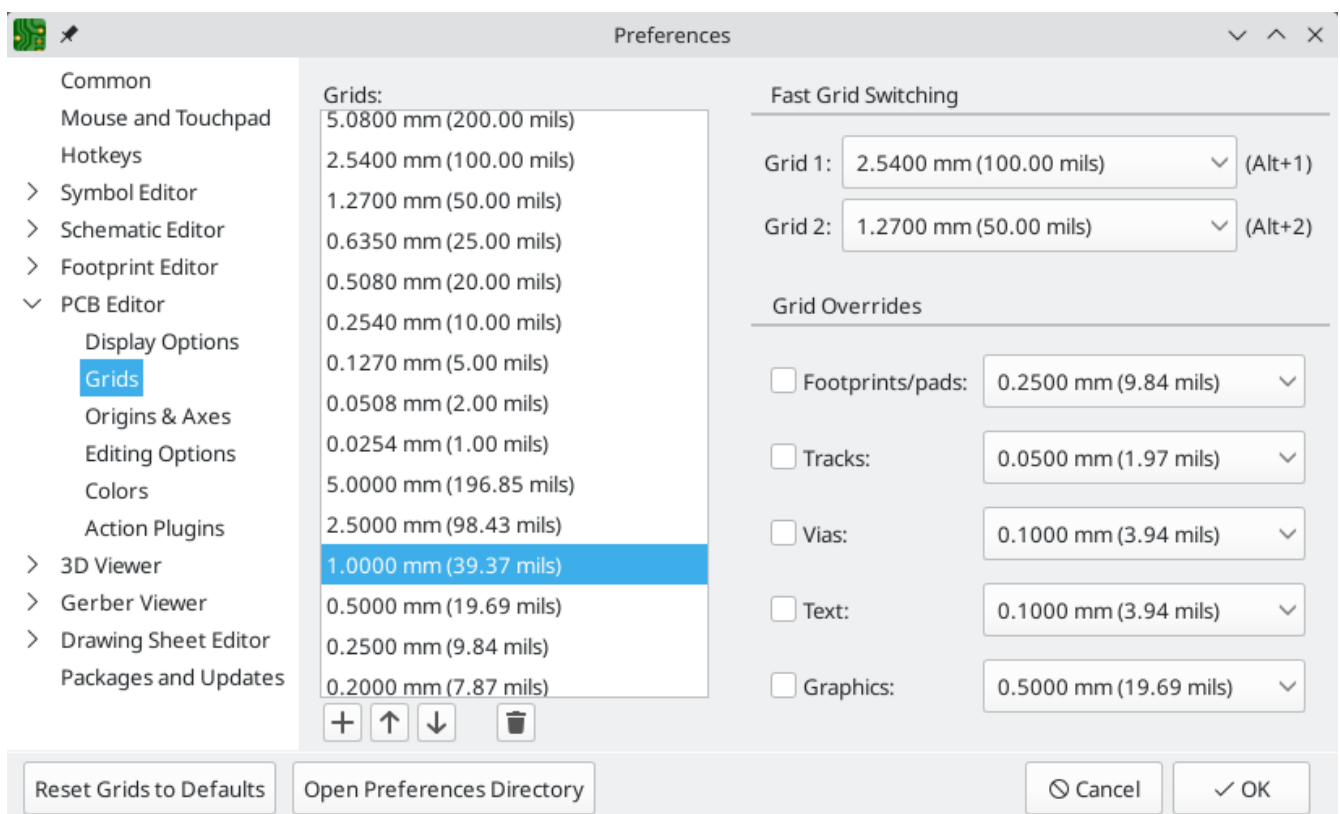
In this dialog you can select an active grid from the list of grids, reorder the list of grids, and add or remove grids. Grids defined in this dialog can have unequal X and Y spacing as well as an optional name.

This dialog also lets you designate two grids from the list as "Fast Grids", which can be quickly selected using `Alt` + `1` and `Alt` + `2`.

Finally, you can configure grid overrides for different types of objects. Grid overrides let you set particular grid sizes for different types of objects which will be used instead of the default grid when working with those objects. For example, you can set a 100 mil grid for footprints and pads while using smaller grids to finely position tracks, vias, and text. Grid overrides can be individually enabled and disabled in this dialog, or globally enabled and disabled using the 🔒 button on the left toolbar (`Ctrl` + `Shift` + `G`).

To change the origin (zero point) of the grid, use **Place → Grid Origin** and click to place the origin in the canvas. This function is also available with the ⣿ button in the right toolbar. Alternatively, you can enter explicit coordinates for the grid origin with **Edit → Grid Origin…**.

The visual appearance of the grid can also be customized in several ways. You can change the thickness of the grid markings, switch their shape (dots, lines, or crosses), and set the minimum displayed spacing in the **Display Options** page of the preferences dialog, and you can change the grid color in the **Colors** page of the preferences dialog.

The grid can be shown or hidden using the ⣿ button on the left-hand toolbar. By default the grid is still active even if it is hidden, but this is configurable in the **Display Options** preferences page. There you can set the grid to be disabled when it is hidden or even disable the grid entirely.

Editing object properties

All objects have properties that are editable in a dialog. Use the hotkey E or select **Properties** from the right-click context menu to edit the properties of selected item(s). You can only open the properties dialog if all the items you have selected are of the same type.

You can also view and edit item properties using the properties manager. The properties manager is a docked panel that displays the properties of the selected item or items for editing. If multiple types of items are selected at once, the properties panel displays only the properties shared by all of the selected item types.

Properties	
Footprint	

∨ Basic Properties

Position X	127 mm
Position Y	82.55 mm
Locked	☐
Layer	🟥 top_copper
Orientation	0°

∨ Fields

Reference	U1
Value	74LS125
Library Link	Package_DIP:DIP-14_W7.62mm_LongPads
Library Description	14-lead though-hole mounted DIP package
Keywords	THT DIP DIL PDIP 2.54mm 7.62mm 300mi

∨ Attributes

Not in Schematic	☐
Exclude From Position Files	☐
Exclude From Bill of Materials	☐
Do not Populate	☐

∨ Overrides

Exempt From Courtyard Requirement	☐
Clearance Override	0 mm
Solderpaste Margin Override	0 mm
Solderpaste Margin Ratio Override	0
Zone Connection Style	Inherited

Editing a property in the properties manager immediately applies the change. When multiple items are selected, property modifications are applied to each selected item individually, not to the whole selection as a group. For example, when changing the orientation of multiple items, each item is individually rotated around its own origin, not the group's origin.

Show the properties manager with **View → Show Properties Manager** or the ⚒ button on the left toolbar.

Several tools are available for editing properties of specific types of objects in bulk. For text and graphical items, you can use the Edit Text and Graphics Properties tool. Tracks and vias can be bulk-edited using the Edit Track and Via Properties tool. Teardrop properties can be edited with the Edit Teardrops tool.

In properties dialogs and many other dialogs, any field that contains a numeric value can also accept a basic math expression that results in a numeric value.

For example, a dimension may be entered as `2 * 2mm`, resulting in a value of `4mm`. Basic arithmetic operators as well as parentheses for defining order of operations are supported.

Board outlines (Edge Cuts)

KiCad uses graphical objects on the `Edge.Cuts` layer to define the board outline. The outline must be a continuous (closed) shape, but can be made up of different types of graphical object such as lines and arcs, or be a single object such as a rectangle or polygon. If no board outline is defined, or the board outline is invalid, some functions such as the 3D viewer and some design rule checks will not be functional.

Working with footprints

Adding footprints to the board

Footprints are automatically added to the board when the PCB is updated from the schematic. The footprint associated with each schematic symbol is added to the board if it is not already present, and each footprint pad is associated with the corresponding symbol pin's net. Symbol pins are matched to footprint pads by pin/pad number.

When footprints are added to the board after an update from the schematic, they are grouped by schematic sheet and by geographical location in the schematic. They are initially attached to the cursor; you can place them by clicking in the desired location.

You can also add footprints to the board manually using the Add Footprint tool (A or the button).

NOTE	Footprints added in this way will not be automatically associated with a symbol or have nets assigned to their pads, and subsequent updates from the schematic will remove these unassociated footprints unless the footprint is locked or the **Delete footprints with no symbols** option is unchecked in the Update PCB From Schematic dialog. For these reasons, it is usually recommended to avoid manually adding footprints to the board. Manually adding footprints is necessary for PCB-only workflows, and can also be useful for adding logos or other footprints that do not need a corresponding schematic symbol.

Placing and moving footprints

Once footprints have been added to the board, you can reposition them in many ways.

The Move command (M) moves a footprint or a selection of footprints, ignoring any connected track segments that are not selected. No DRC checking is done when moving footprints with the Move command, although any footprint courtyards that collide with the moved footprint's courtyard will be highlighted.

There is a reference point for the move operation, which is the point in the footprint which attaches to the cursor and therefore the point in the footprint that snaps to the grid and to other objects. The reference point during a move is determined by the location of the cursor when the Move command is initiated. If the cursor is over a pad, the pad's center will be used as the reference point. If the cursor is not over a pad, the footprint's anchor (coordinate origin point) will be used. To select an arbitrary snapping point, you can use the Move With Reference command instead of the regular Move command (right click → **Positioning Tools** → **Move with Reference**). After initiating the command, click on the desired reference point; KiCad will then begin the move with that point as the reference.

You can also use the Drag command (D) to move the selected footprint using the interactive router, maintaining all track connections to the footprint. Dragging footprints behaves like the Highlight Collisions router mode: obstacles will not be avoided or shoved, only highlighted. Ordinarily the router will prevent you from dragging a footprint into a position that violates DRC: when you click to commit a drag in a

position that violates DRC, the footprint will return to its original position. To force a drag to be committed even if it violates DRC, `Ctrl`-click to commit the drag. Like the Move command, colliding courtyards are highlighted.
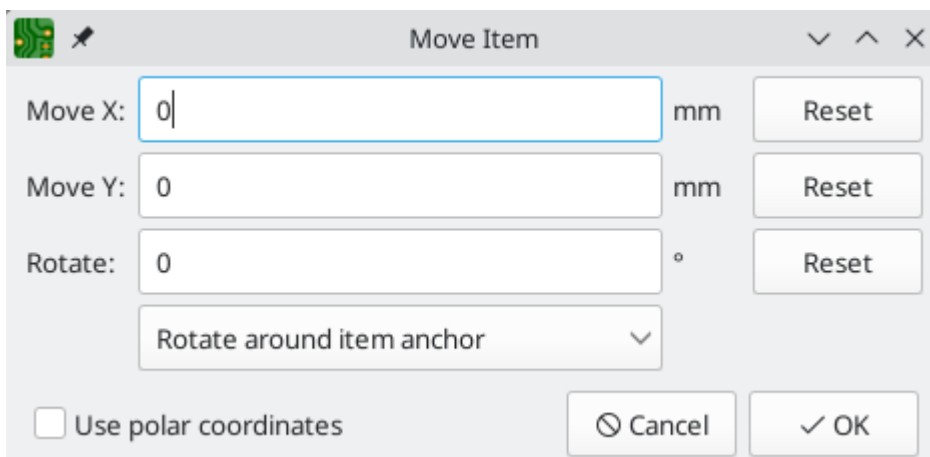
NOTE	Only tracks that end at the origin of the footprint's pads will be dragged. Tracks that simply pass through the pad or that end on the pad at a location other than the origin will not be dragged.

You can move a footprint to the opposite side of the board with the Flip command (`F`). Any parts of the footprint on a front layer will be swapped to the corresponding back layer, and vice versa.

Footprints can be rotated counter-clockwise using the `R` hotkey, or clockwise using `Shift`+`R`. By default, footprints are rotated by 90 degrees every time the rotate command is used, but you can configure the rotation angle step in **Preferences → PCB Editor → Editing Options**.

You can directly set a footprint's exact absolute position, rotation angle, and PCB side using either the Footprint Properties dialog or the Properties panel.

To reposition a footprint relative to its current position, use the Move Exactly tool (`Shift`+`M`). The dialog lets you specify an X and Y translation, as well as a rotation, that will be applied to the footprint. The rotation can be performed relative to either the footprint's anchor, the local coordinate origin, or the drill/place origin. You can also use polar coordinates instead of Cartesian coordinates.

To position a footprint relative to another object, you can use the Position Relative tool (`Shift`+`P`). With this tool, you select a reference point for the move, which can be the local origin, the grid origin, or another arbitrary point, such as a pad in another footprint. The selected footprint is moved to the specified offset from the reference point.
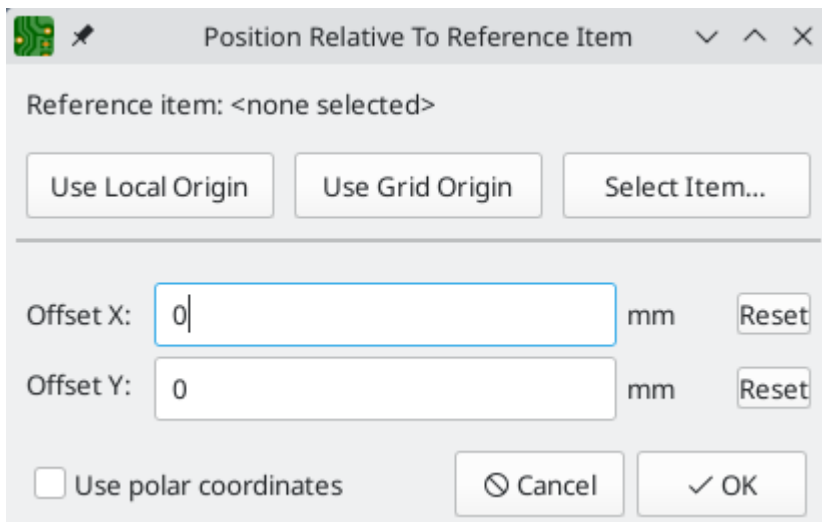
You can swap the position of two selected footprints using the Swap command (`S`). The first footprint is assigned the location, rotation, and board side of the second footprint, and vice versa. If there are more than two footprints selected, the locations are cycled: the last footprint gets the position of the first footprint, the first footprint gets the location of the second, and so on.

There are several convenience features that make it easier to find, select, and move specific footprints or footprints related to another footprint.

The Get and Move Footprint command (`T`) prompts you to choose a footprint from a list or by typing a reference designator. KiCad then attaches the chosen footprint to your cursor for a move operation.

There are two commands to select other footprints that need to be connected to the selected footprint but don't yet have routed connections. The Select All Unconnected Footprints command (`O`) selects all footprints that have ratsnest lines to the currently selected footprints. The command can be executed repeatedly to further expand the selection based on the newly selected items. The Grab Nearest Unconnected Footprint command (`Shift` + `O`) selects the closest footprint with ratsnest lines to the currently selected footprint, and additionally begins to move it. If there are multiple footprints initially selected, the command will act like the Move Individually command described below, individually moving the closest unconnected footprint for each of the initially selected footprints.

You can select footprints based on their schematic sheet using the right click → **Select** → **Items in Same Hierarchical Sheet** command, which selects all other footprints that are in the same schematic sheet as the originally selected footprint.

If you want to move multiple selected footprints in sequence, use the Move Individually command (`Ctrl` + `M`). After triggering the command, KiCad will begin moving the first selected footprint. After you click to place the footprint, KiCad will immediately start moving the next footprint, in the same order that you selected the footprints. You can skip moving a footprint by pressing `Tab` , commit the current move and skip any remaining moves by double-clicking, or cancel all moves (including those already completed) by pressing `Esc` .

If you want to move a collection of footprints at once into one area, the Pack and Move Footprints command (`P`) closely packs the selected footprints together and moves them as a block.

Finally, KiCad can automatically place footprints onto the board. The auto-place function attempts to optimally place footprints to simplify ratsnest connections to other footprints. You can auto-place the selected footprints with **Place → Auto-Place Footprints → Place Selected Footprints**, or auto-place all footprints outside of the board outline with **Place → Auto-Place Footprints → Place Off-Board Footprints**.

Editing Footprints

Footprints in the board can be individually edited. Editing a footprint in the board only affects that particular instance of the footprint; it does not affect any other copies of that footprint in the board, and it does not affect the library footprint.

To edit a footprint in the board, open its properties dialog (E)

The majority of the settings in this dialog are the same as in the [footprint editor](). You can edit the footprint's fields, attributes, clearance and zone connection settings, and 3D models, as in the footprint editor. However, here you can also set the footprint's position, orientation, and side. You can also update the footprint from the library, exchange it for a different footprint, or edit the footprint itself in the footprint editor.
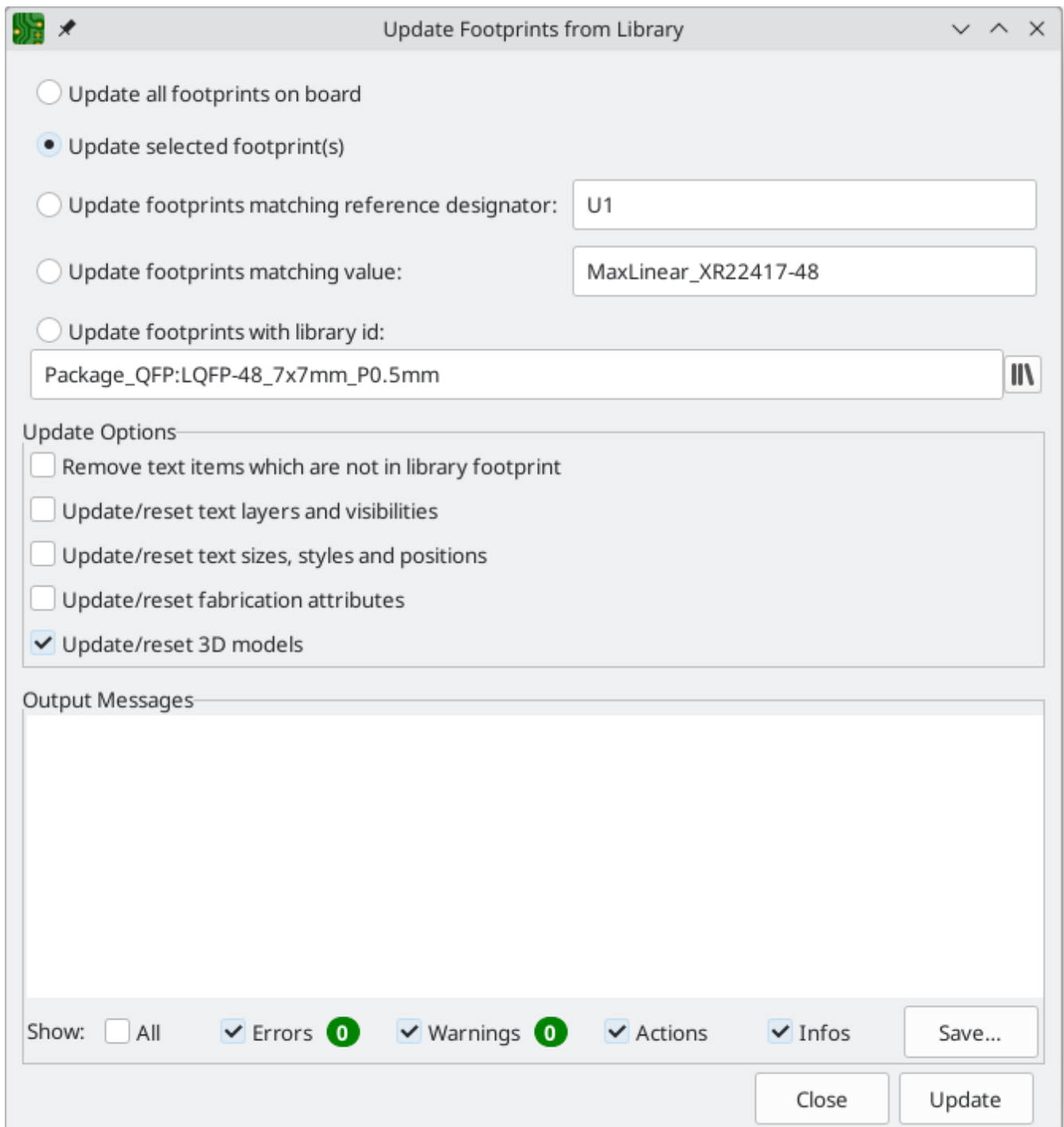
There are two options for editing the footprint in the footprint editor.

- **Edit Footprint...** will open the specific instance of the footprint in the footprint editor. Editing this footprint will only affect this one instance of the footprint in the board. It will not affect other instances of the footprint in the board, and it will not affect the library copy of the footprint.

- **Edit Library Footprint...** will open the library copy of the footprint in the footprint editor. Editing the library copy of the footprint will edit the footprint in the footprint library, but will not immediately affect any instances of that footprint in the board. To update footprints in the board with changes to the library footprint, use the **Update Footprint from Library...** tool.

Updating and exchanging footprints

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

To update footprints in the board to match the corresponding library footprint, use **Tools → Update Footprints from Library…**, or right click a footprint and select **Update Footprint…**. You can also access the tool from the footprint properties dialog.

The top of the dialog has options to choose which footprints will be updated. You can update all footprints on the board, update only the selected footprints, or update only the footprints that match a specific

reference designator, value, or library identifier. The reference designator and value fields support wildcards: `*` matches any number of any characters, including none, and `?` matches any single character.

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

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

To change an existing footprint to a different footprint, use **Edit → Change Footprints…**, or right click an existing footprint and select **Change Footprint…**. This dialog is also accessible from the footprint properties dialog.

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

Editing footprint fields

An individual symbol text field can be edited directly with the `E` hotkey (with a field selected instead of a footprint) or by double-clicking on the field.

The options in this dialog are the same as those in the full Footprint Properties dialog, but are specific to a single field.

Only footprint fields can be edited this way in the board editor. Unlike fields, Footprint text is a graphic object that can only be edited or moved in the footprint editor.

NOTE	In versions of KiCad before version 8.0, footprint fields did not exist. Instead, footprint text could be edited directly in the board editor. In KiCad 8.0, footprint text is not editable in the board editor and can only be edited in the footprint editor.

Working with pads

The properties of each individual pad of a footprint can be inspected and edited after placing the footprint on the board. In other words, it is possible to override the design of an individual footprint pad in a specific instance of the footprint on the board, if the footprint design in the library is not appropriate. For example, you may wish to remove the solder paste aperture for a pad that needs to remain unsoldered in a specific design, or you may wish to move the location of a through-hole pad for an axial-lead resistor in order to fit a specific design.

NOTE	By default, the position of all footprint pads are locked, so it is possible to edit the pad properties but not move the pad's location relative to the rest of the footprint. Pads may be unlocked to allow free movement, which can be useful for certain applications (such as through-hole footprints with varying lead positions) but is generally never recommended for surface-mount footprints.

The pad properties dialog is opened through the context menu or default hotkey `E` when a pad is selected. Note that KiCad assumes that if you click near a pad, you are probably trying to select the entire footprint rather than a single pad. To select a single pad, make sure to click inside the pad area, or turn off the

Footprints setting in the selection filter (and make sure the Pads setting is turned on) to prevent accidental selection of the entire footprint rather than a specific pad.
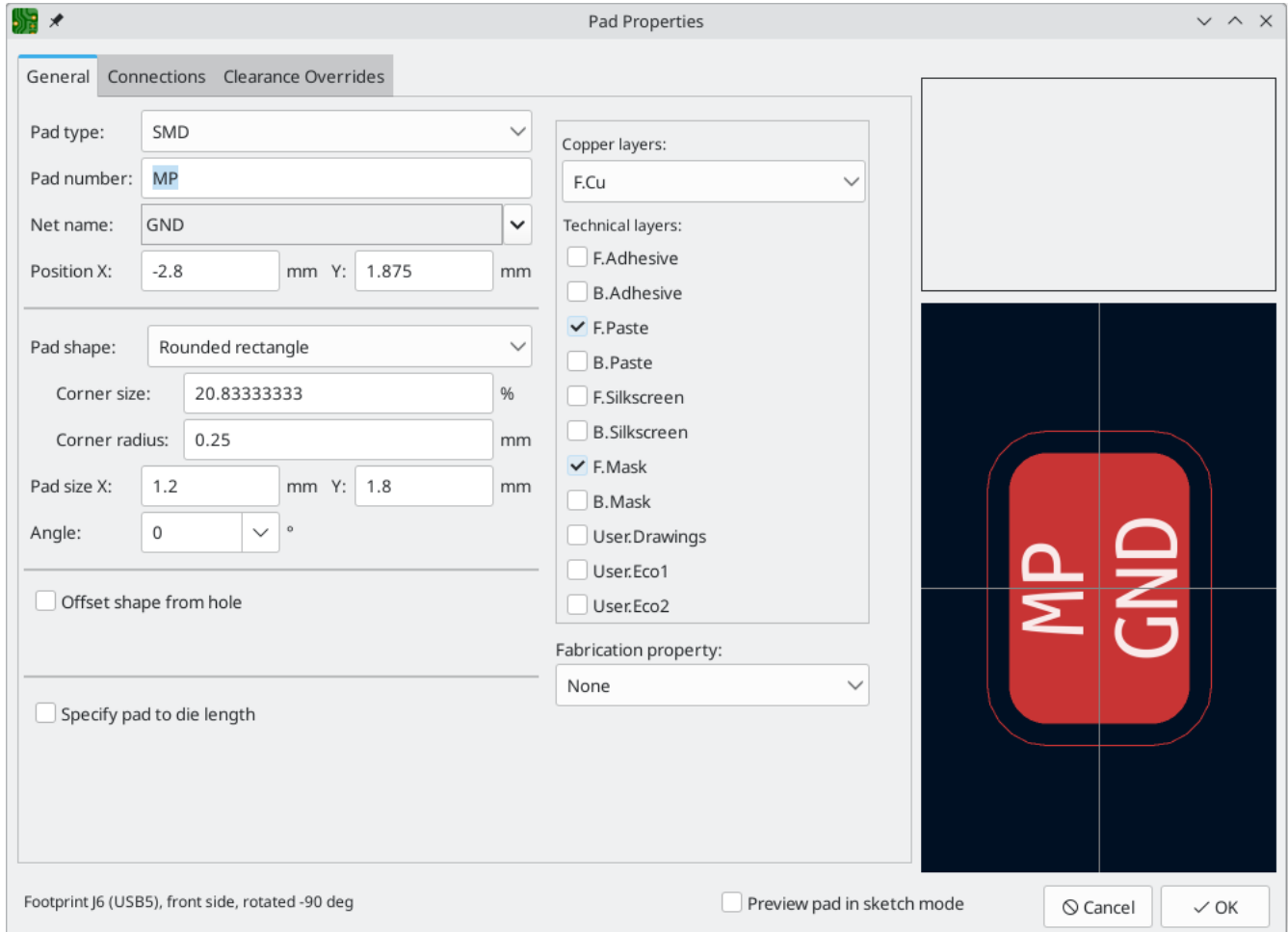
This dialog lets you edit the physical properties of the pad, including size and shape. You can also modify how the pad connects to other objects on the board, including clearance properties, teardrops, and thermal reliefs.

This dialog is the same as the pad properties dialog in the footprint editor, except that here you can also manually assign a net to a pad using the **net name** selector. The remaining options are explained in the Footprint Editor documentation.

NOTE	While you can manually assign nets to pads in the PCB editor, this is not a typical workflow. Usually net-to-pad connections are defined by the schematic and then transferred to the PCB editor.

Working with zones

Copper zones, also sometimes called copper pours or fills by other EDA tools, are solid or hatched areas of copper assigned to a particular net that automatically keep clearance from other copper objects. Zones are commonly used to fill in all free space on a board layer (or a portion of a layer) in order to create ground and power planes, carry high currents, or to provide shielding.

NOTE	Some EDA tools have separate tools for creating "plane layers" and for creating copper zones on signal layers. In KiCad, the Copper Zone tool is used for both these applications.

Zones are defined by a polygonal **outline** that defines the maximum extent of the filled copper area. This outline does not represent physical copper and will not appear in exported manufacturing data. The actual copper areas of the zone must be **filled** each time the outline, or any objects inside the outline, are modified. The filling process may be run on a single zone, or on all zones in a board (default hotkey `B`). Zones may be **unfilled** (default hotkey `Ctrl` + `B`) to improve performance and reduce visual clutter while editing large boards.

NOTE	By default, zone filling is a manual process rather than occurring every time an object changes that would result in a change to the zone copper. This is because zone filling can be a slow process on older computers or very large designs. It is important to make sure zone fills are up-to-date before generating outputs. KiCad will check that zones have been updated and warn you before generating outputs or running DRC when zones have not yet been refilled. You can optionally enable automatic zone-filling in the Preferences dialog (**PCB Editor → Editing Options → Miscellaneous → Automatically refill zones**).

To draw a zone, click the Add Filled Zone tool ( ) on the right toolbar, or use default hotkey `Ctrl` + `Shift` + `Z`. Click to choose the first point of the zone outline. The Zone Properties dialog will appear, allowing you to choose the zone net and other properties. These properties may be edited at any time, so it is not critical to choose them all correctly at first. Accept the dialog and continue placing points to define the zone outline. To finish the zone, double-click to set the last point. Zone outline points may be modified like graphic polygons, by dragging the square handles to move a corner or dragging the circular handles to move an edge. To edit the zone's properties, use hotkey `E` or select Properties from the context menu.

Layer: A single zone object can create filled copper on one or more copper layers. Check the box next to each copper layer that this zone outline should fill on. The copper on each layer will be filled independently, but all layers will share the same net.

Net: Select the electrical net that the zone copper should be connected to. It is possible to create zones with no net assignment. Zones with no net will keep clearance from any copper objects on any net.

Zone name can be used to assign a specific name to a zone. This name can be used to refer to the zone in custom DRC rules.

Zone priority level determines the order in which multiple zones on a single layer are filled. The highest priority level zone on a given layer will be filled first. Lower-priority zones will keep clearance to the filled areas of higher-priority zones. Two zones on the same layer with the same priority level will overlap (short-circuit) with each other, unless they are assigned different nets. When two zone outlines with the same priority and different nets touch, one zone will maintain clearance to the other so that they don't short.

Locked controls whether or not the zone outline object is locked. Locked objects may not be manipulated or moved, and cannot be selected unless the **Locked Items** option is enabled in the Selection Filter panel.

Outline display controls how the zone outline is drawn on screen. In **Line** mode, only the border lines of the outline are drawn. In **Hatched** mode, hatch lines are drawn on the inside of the outline border for a

short distance, to make the zone outline more apparent. In **Fully Hatched** mode, hatch lines are drawn across the entire inside of the zone outline.

Corner smoothing controls the behavior of the filled copper areas at corners of the outline. Corners can be smoothed by a chamfer or fillet, or can extend all the way to the outline corner if smoothing is disabled. The chamfer or fillet size is configurable when those modes are selected.

NOTE	By default, chamfers and fillets are not added to **inside corners** of the zone outline, because this would result in filled copper extending *outside* the outline. If smooth inside corners are desired, enable the **Allow fillets outside zone outline** option in the Constraints section of the Board Setup dialog.

Clearance controls the minimum clearance the filled areas of this zone will keep from other copper objects. Note that if two clearance values are in conflict, the larger clearance value will be used. For example, if a zone is set to use 0.2mm clearance but its netclass is set to use 0.3mm clearance, the result will be an 0.3mm clearance.

Minimum width controls the minimum size of narrow necks of copper created inside the zone. Any copper areas that would be below this minimum width are removed during the filling process.

Pad connection controls the way that the filled zone areas will connect to footprint pads on the same net. **Solid** connections will result in the copper completely overlapping the pads. **Thermal reliefs** will result in small copper spokes connecting the pad to the rest of the copper zone, increasing the thermal resistance between the pad and the rest of the zone. This can be useful for hand soldering. **Reliefs for PTH** will apply thermal reliefs to plated through-hole pads and use solid connections for surface mount pads. **None** will result in the zone not connecting to any pads on the same net.

Thermal relief gap controls the distance maintained between any pad and the copper zone when the pad connection mode is set to generate thermal reliefs.

Thermal spoke width controls the width of the "spokes", or short copper segments connecting the pad to the rest of the copper zone.

Fill type controls how the copper zone is filled: the default is **solid fill**, which will result in copper filling in all available space within the zone outline. The zone can also be set to fill a **hatch pattern**, which will fill the area with a pattern that contains less copper. This can be useful for flexible printed circuits and other specialty applications.

Orientation controls the angle of the hatch pattern lines. An orientation of 0 degrees will result in the hatch pattern using horizontal and vertical lines.

Hatch width controls the width of each line in the hatch pattern.

Hatch gap controls the distance between each line in the hatch pattern.

Smoothing effort controls the style of smoothing applied to the hatch pattern. A value of 0 will result in no smoothing, and a value of 3 will result in the finest smoothing. Higher values will result in longer processing time and larger Gerber files.

Smoothing amount is a ratio that controls the size of the smoothing chamfers or fillets that are generated when **smoothing effort** is set to a value other than 0. An amount of 0.0 results in no smoothing, and a value

of 1.0 results in maximum smoothing (in other words, a chamfer or fillet equal to half of the hatch gap).

Remove islands controls the behavior of isolated copper areas, also called islands, after the initial zone fill. When this is set to **always**, isolated areas inside the zone are removed. When set to **never**, isolated areas are left alone, and will result in copper areas that are not connected to the rest of the net. When set to **below area limit**, a **minimum island size** can be specified, and islands below this threshold will be removed.

NOTE	Regardless of the **remove islands** setting, islands are never removed from zones that are electrically unconnected. In other words, islands are only removed from zones that have at least one electrical connection.

Routing tracks

KiCad features an interactive router that:

- Allows manual or guided (semi-automatic) routing of single tracks and differential pairs

- Enables modifications of existing designs by:

    - Re-routing existing tracks when they are dragged

    - Re-routing tracks attached to footprint pads when the footprint is dragged

- Allows tuning of track lengths and differential pair skew (phase) by inserting serpentine tuning shapes for designs with tight timing requirements

By default, the router respects the configured design rules when placing tracks: the size (width) of new tracks will be taken from the design rules and the router will respect the copper clearance set in the design rules when determining where new tracks and vias can be placed. It is possible to disable this behavior if desired by using the Highlight Collisions router mode and turning on the Allow DRC Violations option in the router settings (see below).

The router has three modes that can be selected at any time. The router mode is used for routing new tracks, but also when dragging existing tracks using the Drag (hotkey D) command. These modes are:

- **Highlight Collisions**: in this mode, most of the router features are disabled and routing is fully manual. When routing, *collisions* (clearance violations) will be highlighted in green and the newly-routed tracks cannot be fixed in place if there is a collision unless the Allow DRC Violations option is turned on. In this mode, up to two track segments may be placed at a time (for example, one horizontal and one diagonal segment).

- **Shove**: in this mode, the track being routed will walk around obstacles that cannot be moved (for example, pads and locked tracks/vias) and *shove* obstacles that can be moved out of the way. The router prevents DRC violations in this mode: if there is no way to route to the cursor position that does not violate DRC, no new tracks will be created.

- **Walk Around**: in this mode, the router behaves the same as in Shove mode, except no obstacles will be moved out of the way.

Which mode to use is a matter of preference. For most users, we recommend using Shove mode for the most efficient routing experience or Walk Around mode if you do not want the router to modify tracks that are not being routed. Note that Shove and Walk Around modes always create horizontal, vertical, and 45-degree

(H/V/45) track segments. If you need to route tracks with angles other than H/V/45, you must use Highlight Collisions mode and enable the Free Angle Mode option in the Interactive Router Settings dialog.
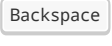
There are four main routing functions: Route Single Track, Route Differential Pair, Tune length of a single track, and Tune skew of a differential pair. All of these are present in both the Route menu dropdown (individually) on the top toolbar and the drawing toolbar in two overloaded icons on the drawing toolbar on the right. The use of the overloaded icons is described above. One is for the two Route functions and one is for the two Tune functions. In addition, the Route menu allows the selection of Set Layer Pair and Interactive Router Settings.

To route tracks, click the Route Tracks ⌐ icon (from the drawing toolbar or from the top toolbar under **Route**) or use the hotkey `X`. Click on a starting location to select which net to route and begin routing. The net being routed will automatically be highlighted and the allowable clearance for the net will be indicated with a gray outline around the tracks being routed. The clearance outline can be disabled by changing the Clearance Outlines setting in the Display Options section of the Preferences dialog.

NOTE	The clearance outline shows the maximum clearance from the routed net to any other copper on the PCB. It is possible to use custom design rules to specify different clearances for a net to different objects. These clearances will be respected by the router, but only the largest clearance value will be shown visually.

When the router is active, new track segments will be drawn from the routing start point to the editor cursor. These tracks are *unfixed* temporary objects that show what tracks will be created when you use a left-click or the `Enter` key to *fix* the route. The unfixed track segments are shown in a brighter color than the fixed track segments. When you exit the router using the `Esc` key or by selecting another tool, only the fixed track segments will be saved. The Finish Route action (hotkey `End`) will fix all tracks and exit the router.

While you are routing, you can use the Undo Last Segment command (hotkey `Backspace`) to unfix the tracks you most recently fixed. You can use this command repeatedly to step back through the route that you have already fixed.

In previous versions of KiCad, using the left mouse button or `Enter` to fix the routed segments would fix all segments up to but *not including* the segment ending at the mouse cursor location. In KiCad 6 and later, this behavior is optional, and by default, all segments *including* the one ending at the mouse cursor location will be fixed. The old behavior can be restored by disabling the "Fix all segments on click" option in the Interactive Router Settings dialog.

While routing, you can hold the `Ctrl` key to disable grid snapping, and hold the `Shift` key to disable snapping to objects such as pads and vias.

NOTE	Snapping to objects can also be disabled by changing the Magnetic Points preferences in the Editing Options section of the Preferences dialog. We recommend that you leave object snapping enabled in general, so that you do not accidentally end tracks slightly off-center on a pad or via.

Track posture

When routing in H/V/45 mode, the *posture* refers to how a set of two track segments connect two points that cannot be reached by a single H/V/45-degree segment. In such a case, the points will be connected by one

horizontal or vertical segment and one diagonal (45-degree) segment. The posture refers to the order of these segments: whether the horizontal/vertical segment or the diagonal segment comes first.
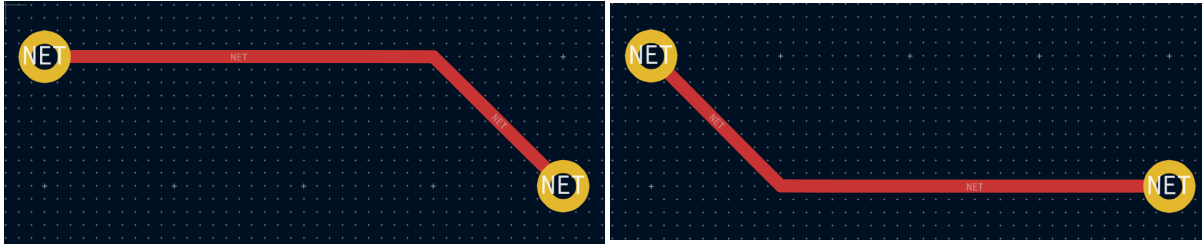
KiCad's router attempts to pick the best posture automatically based on a number of factors. In general, the router will attempt to minimize the number of corners in a route, and will avoid "bad" corners such as acute angles whenever possible. When routing from or to a pad, KiCad will choose the posture that lines up the route with the longest edge of the pad.

In some cases, KiCad cannot guess the posture you intend correctly. To switch the posture of the track while routing, use the Switch Track Posture command (hotkey `/`).

In situations where there is no obvious "best" posture (for example, when starting a route from a via), KiCad will use the movement of your mouse cursor to select the posture. If you would like the route to begin with a straight (horizontal or vertical) segment, move the mouse away from the starting location in a mostly horizontal or vertical direction. If you would like the route to begin diagonally, move in a diagonal direction. Once the cursor is a sufficient distance away from the routing start location, the posture is "locked" and will no longer change unless the cursor is brought back to the starting location. Detection of posture from the movement of the mouse cursor can be disabled in the Interactive Router Settings dialog as described below.

NOTE	If you use the Switch Track Posture command to override the posture chosen by KiCad, the automatic detection of posture from mouse movement will be disabled for the remainder of the current routing operation.

Track corner mode

KiCad's router can place tracks with either sharp or rounded (arc) corners when routing in H/V/45 mode. To switch between sharp and rounded corners, use the Track Corner Mode command (hotkey `Ctrl` + `/`). When routing with rounded corners, each routing step will place either a straight segment, a single arc, or both a straight segment and an arc. The track posture determines whether the arc or the straight segment will be placed first.

Track corners can also be rounded after routing by using the Fillet Tracks command after selecting the tracks on either side of the corner to be filleted. If a contiguous track selection contains multiple corners, they will all be filleted.

NOTE	Dragging of tracks with arcs is not supported. Arcs are treated as immovable by the shove router.

Ancho de pista

The width of the track being routed is determined in one of three ways: if the routing start point is the end of an existing track and the ⌷=⌷ button on the top toolbar is enabled, the width will be set to the width of the existing track. Otherwise, if the track width dropdown in the top toolbar is set to "use netclass width", the width will be taken from the netclass of the net being routed (or from any custom design rules that

specify a different width for the net, such as inside a neckdown area). Finally, if the track width dropdown is set to one of the pre-defined track sizes configured in the Board Setup dialog, this width will be used.

NOTE	The track width can never be lower than the minimum track width configured in the Constraints section of the Board Setup dialog. If a pre-defined width is added that is lower than this minimum constraint, the minimum constraint value will be used instead.

KiCad's router supports a single track width for the active route. In other words, to change widths in the middle of a track, you must end the route and then restart a new route from the end of the previous route. To change the width of the active route, use the hotkeys `W` and `Shift`+`W` to step through the track widths configured in the Board Setup dialog.

Placing vias

While routing tracks, switching layers will insert a through via at the end of the current (unfixed) track. Once you place the via, routing will continue on the new layer. There are several ways to select a new layer and insert a via:

- By using the hotkey to select a specific layer, such as `PgUp` to select `F.Cu` or `PgDn` to select `B.Cu`.

- By using the Next Layer or Previous Layer hotkeys (`+` and `-`).

- By using the Place Via hotkey (`V`), which will switch to the next layer in the active layer pair.

- By using the Select Layer and Place Through Via action (hotkey `<`), which will open a dialog to select the target layer.

After using any of the above methods to add a via and change layer, but before clicking to fix the via and commit the current trace segment, you can cancel placing the via by pressing `V`. The via will be removed and routing will continue on the original layer.

You can place a via and end the current trace, without changing layers, by pressing `V` and then double-clicking or `Shift`-clicking to place the via.

The size of the via will be taken from the active Via Size setting, accessible from the drop-down in the top toolbar or the Increase Via Size (`'`) and Decrease Via Size (`\`) hotkeys. Much like track width, when the via size is set to "use netclass sizes", the via sizes configured in the Net Classes section of the Board Setup will be used (unless overridden by a custom design rule).

You can also place microvias and blind/buried vias while routing. Use the hotkey `Ctrl`+`V` to place a microvia and `Alt`+`Shift`+`V` to place a blind/buried via. Microvias may only be placed such that they connect one of the outer copper layers to an adjacent layer. Blind/buried vias may be placed on any layer.

Vias placed by the router are considered to be part of a routed track. This means that the via net can be updated automatically (just like track nets can), for example when updating the PCB from the schematic changes the net name of the track. In some cases this may not be desired, such as when creating stitching vias. The automatic update of via nets can be disabled for specific vias by turning off the "automatically update via nets" checkbox in the via properties dialog. Vias placed with the Add Free-standing Vias tool are created with this setting disabled.

Routing Convenience Functions

KiCad offers several functions to make certain routing operations more convenient.

If you need to route a number of traces from a set of pads, you can use the Route Selected tool to quickly route from each pad in sequence. Select the pads you want to use as starting points, then press `Shift` + `X` to route from each pad in sequence. The router will begin a trace from the first selected pad, which you can route as you would any other trace. When you complete the trace, the router will automatically begin a new trace from the next pad in the selection, in the same order that you selected the pads. Pads that already have traces attached are skipped. You can also skip routing the current trace and move on to the next pad by pressing `Esc`. You can also select footprints instead of pads; all unrouted pads in the selected footprints will be used as starting points.

If you want to route a number of traces *to* a set of pads, instead of *from* the pads, you can use the Route Selected From Other End tool (`Shift` + `E`). This tool works the same way as the Route Selected tool, except it uses each selected pad as an end point rather than a starting point. The starting point for each trace is the other end of the ratsnest line for each selected pad.

Routing from the other end is also possible while routing individual traces: press `Ctrl` + `E` while routing a trace to commit the current segment and begin routing from the other end of the in-progress trace's ratsnest line.

Finally, you can quickly unroute traces connected to an object (footprint, pad, or trace) by selecting the object, right-clicking, and selecting **Unroute Selected**. Any traces connected to the selected object will be removed, starting at the selected object and continuing until another pad is encountered.

Automatically completing traces

KiCad's router can automatically route individual traces, based on the connections defined in the schematic. This can be thought of as a limited form of auto-routing that considers a single trace at a time. The router will only use the current layer; it will not use vias or change layers.

While routing, press the `F` key to have the router attempt to automatically finish the current trace. The trace will be automatically routed from the end of the last fixed trace segment to the closest ratsnest anchor. If the router can't automatically finish the trace, it will allow you to complete the trace manually. This action can also be performed by clicking **Attempt Finish** in the context menu while routing.

When the router is not the active tool, you can automatically route multiple traces by selecting footprints, pads, and traces to route from and pressing `Shift` + `F`. You do not need to select both ends of a desired connection; the router will route from the selected item to its nearest ratsnest anchor. If multiple items were selected, each item will be routed in sequence, in the order that they were selected. If a connection cannot be automatically completed, the tool will pause with the router active so that you can complete the trace manually. With the automatic completion paused for a manual connection, you can press `Esc` to skip routing the current trace. After manually completing the trace or skipping the connection, the tool will continue attempting to route the remaining connections.

Routing differential pairs

Differential pairs in KiCad are defined as nets with a common *base name* and a positive and negative suffix. KiCad supports using `+` and `-`, or `P` and `N` as the suffix. For example, the nets `USB+` and `USB-` form a differential pair, as do the nets `USB_P` and `USB_N`. In the first example, the base name is `USB`, and `USB_` in the second. The suffix styles cannot be mixed: the nets `USB+` and `USB_N` do not form a differential pair. Make sure you name your differential pair nets accordingly in the schematic in order to allow use of the differential pair router in the PCB editor.

To route a differential pair, click the Route Differential Pairs ⤒ icon (from the drawing toolbar or from the top toolbar under **Route**) or use the hotkey `6`. Click on a pad, via, or the end of an existing differential pair track to start routing. You can start routing from either the positive or negative net of a differential pair.

The differential pair router will attempt to route the pair of tracks with a gap taken from the design rules (differential pair gap can be configured in the Net Classes section of the Board Setup dialog, or by using custom design rules). If the starting or ending location of the route is a different distance apart from the configured gap, the router will create a short "fan out" section to minimize the length of track where the differential pair is not coupled.

When switching layers or using the Place Via (`V`) action, the differential pair router will create two vias next to each other. These vias will be placed as close as possible to each other while respecting the design rules for copper and hole-to-hole clearance.

Modifying tracks

After tracks have been routed, they can be modified by moving or dragging, or deleted and re-routed. When a single track segment is selected, the hotkey `U` can be used to expand the selection to all connected track segments. The first press of `U` will select track segments between the nearest junctions with pads or vias. The second press of `U` will expand the selection again to include all track segments connected to the selected track on all layers. Selecting tracks with this technique can be used to quickly delete an entire routed net.

There are two different drag commands that can be used to modify a track segment. The Drag (45-degree mode) command, hotkey `D`, is used to drag tracks using the router. If the router mode is set to Shove, dragging with this command will shove nearby tracks. If the router mode is set to Walk Around, dragging with this command will walk around or stop at obstacles. The Drag Free Angle command, hotkey `G`, is used to split a track segment into two and drag the new corner to any location. Drag Free Angle behaves like the Highlight Collisions router mode: obstacles will not be avoided or shoved, only highlighted.

NOTE	Dragging of tracks containing arcs is not yet possible. Attempting to drag these tracks will result in the arcs being removed in some cases. It is possible to resize a particular arc by selecting it and using the drag command (`D`). When resizing an arc using this command, no DRC checking is performed.

The Move command (hotkey `M`) can also be used on track segments. This command will pick up the selected track segments, ignoring any attached track segments or vias that are not selected. No DRC checking is done when moving tracks using the Move command.

It is possible to re-route tracks attached to footprints while moving the footprints. To do so, use the drag command (`D`) with a footprint selected. Any tracks that end at one of the footprint's pads will be dragged along with the footprint. This feature has some limitations: it only operates in Highlight Collisions mode, so the tracks attached to footprints will not walk around obstacles or shove nearby tracks out of the way. Additionally, only tracks that end at the origin of the footprint's pads will be dragged. Tracks that simply pass through the pad or that end on the pad at a location other than the origin will not be dragged.

You can modify the width of tracks and the size of vias, without re-routing them, in the properties of the track or via. You can choose a predefined size or set the size to an arbitrary value. You can also change the position of tracks and vias or change a via's start and end layers.

To modify tracks and vias in bulk you can use the **Edit Track and Via Properties** dialog (**Edit → Edit Track & Via Properties…**)..

Scope settings restrict the tool to editing only tracks, vias, or both. If no scopes are selected, nothing will be edited.

Filter Items restricts the tool to editing particular objects in the selected scope. Objects will only be modified if they match all enabled and relevant filters (some filters do not apply to certain types of objects.

For example, via size filters do not apply to tracks). If no filters are enabled, all objects in the selected scope will be modified. For filters with a text box, wildcards are supported: `*` matches any characters, and `?` matches any single character.

Filter items by net filters to items assigned the specified net. **Filter items by netclass** filters to items assigned to the specified netclass.

Filter items by layer filters to items on the specified board layer.

Filter tracks by width filters to tracks with the specified track width. **Filter vias by size** filters to vias with the specified track width.

Selected items only filters to the current selection.

Properties for filtered objects can be set to new values in the bottom part of the dialog. Properties can be set to arbitrary values by selecting **set to specified values** or set to the default value from the net class (or custom rule) by selecting **set to net class / custom rule values**.

Drop-down lists can be set to `-- leave unchanged --` to preserve existing values, or set to a pre-defined track or via size to change the filtered objects' size. You can also change the filtered objects' layer.

Length tuning

The length tuning tools can be used to add serpentine tuning shapes to tracks after routing. Length tuning shapes are persistent objects that can be modified after they are created. To tune the length of a track, first pick the appropriate tool.

- The single-track length tuning tool (icon or hotkey 7) will add serpentine shapes to bring the length of a single track up to the target value.

- The differential pair length tuning tool (icon or hotkey 8) will do the same for a differential pair.

- The differential pair skew tuning tool (icon or hotkey 9) will add length to the shorter member of a differential pair in order to eliminate skew (phase difference) between the positive and negative sides of the pair.

As with the Routing icons, the Tuning icons are found in both the **Route** menu dropdown from the top toolbar and the drawing toolbar on the right.

When a tuning tool is active, you can hover over traces in the board to show a status window that displays their current length or skew as well as the target values. Click on the desired trace to start tuning it. As you move the mouse cursor along the track, meander shapes will be added interactively. If a target length has been set, meanders will stop being added when the target length is reached. You can set a target length with custom DRC rules or in the tuning shape properties; both methods are explained below. The popup window next to the cursor shows a live measure of the length or skew compared to the design targets. You can adjust the spacing (1 to increase and 2 to decrease) and amplitude (3 to increase and 4 to decrease) while you tune. When you are done, click again to commit the tuned shape. The tuned trace doesn't need to be perfect because you can adjust the shape after committing it. You can also place multiple tuning shapes on the same track.

NOTE	The length tuning tools only support tuning the length of point-to-point nets between two pads. Tuning the length of nets with different topologies is not supported.

NOTE	Differential pair length tuning can only be applied to the coupled portions of differential pairs. To apply length tuning to the uncoupled portions of differential pairs, you must use single-track length tuner.

Editing tuning patterns

After a tuning pattern has been added, it can be selected, modified, and moved. While it is selected, the target length and routed length are shown in the message panel at the bottom left of the window.
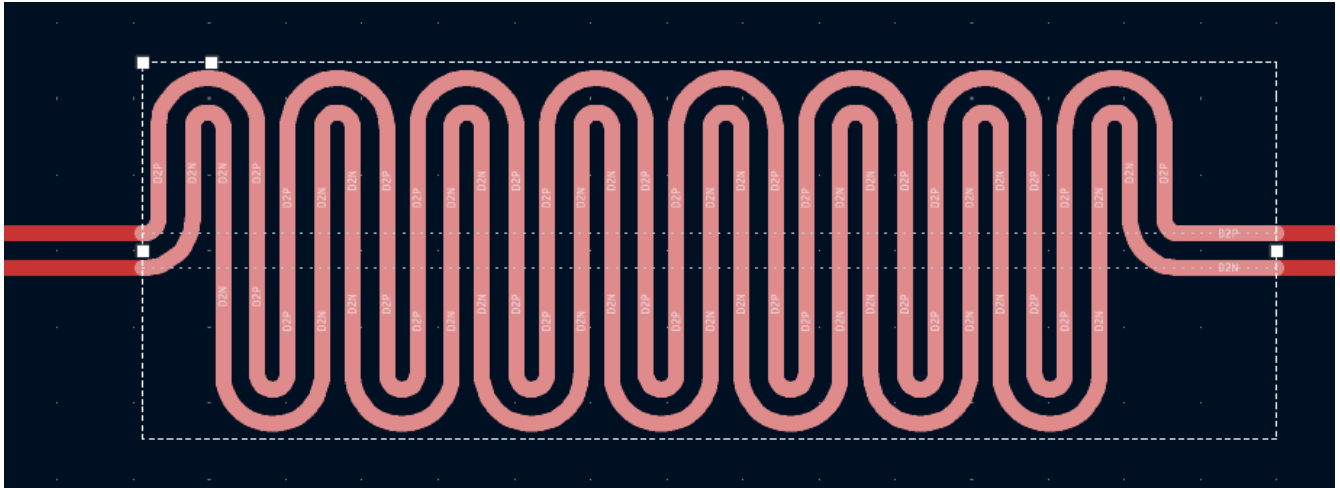
When a pattern is selected, editing handles appear, which let you adjust the pattern geometry.

- Dragging the handles at the ends of the pattern will expand or contract the pattern along the trace.

- Dragging the corner handle towards or away from the trace will respectively decrease or increase the maximum meander amplitude.

- The final handle controls the meander spacing; dragging it towards the corner handle will increase the spacing, while dragging it away from the corner handle will increase the spacing.

The selection box and editing handles represent the maximum allowable extents of the tuning pattern. Making the box smaller will reduce the size of the tuning pattern, even if this results in the tuned trace being shorter than the target length. When the box is enlarged, the tuning pattern will expand to fill the box until the target length is reached.

You can move a tuning pattern along its track by selecting it and dragging with the mouse, or using the Move tool (M). Deleting a tuning pattern (Del) removes the tuning pattern and restores the original untuned tracks. You can also ungroup the tuning pattern, which will decompose it into its component tracks. The basic tracks have the same shape as the tuning pattern but can be edited individually. Once ungrouped into tracks, a tuning pattern cannot be regrouped.

Another way to edit a tuning pattern is through its properties dialog. The properties dialog exposes several additional parameters that can't be modified using the on-canvas interactive editor. These properties can also be edited in the properties manager.
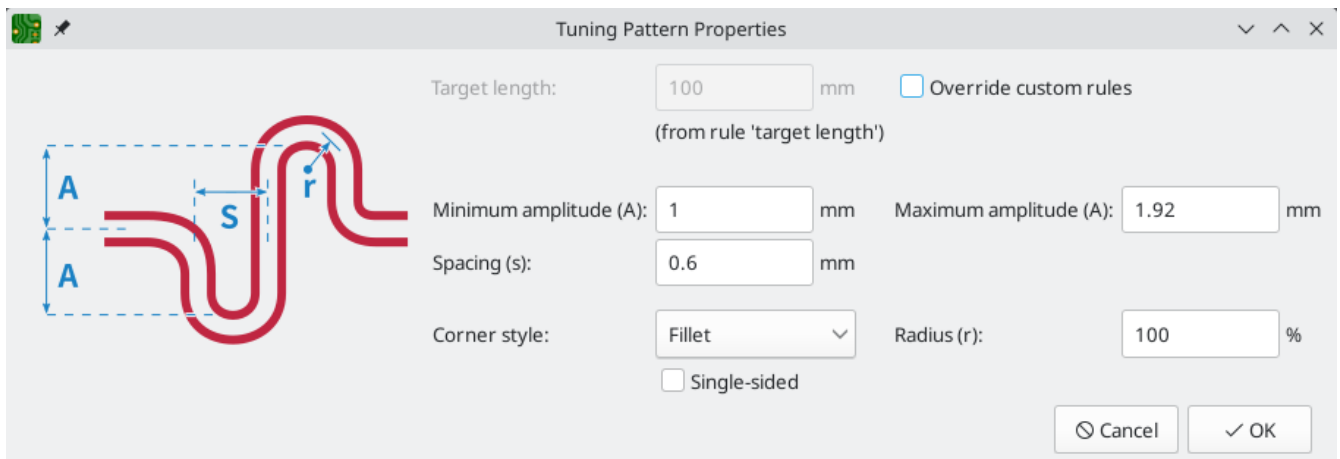
As with the interactive editor, you can set a maximum amplitude for the tuning pattern and a spacing between meanders, but here you can set a minimum amplitude and configure the corner style. Corners can be **filleted** (rounded) or **chamfered**. In each case you can set the **radius** as a percentage of the maximum possible radius for the spacing and amplitude. You can also configure the tuning pattern to be **single-sided**, which restricts it to one side of the baseline, as opposed to the default style which positions meanders on both sides of the baseline.

You can set default values for these properties in the **Design Rules** → **Length-tuning Patterns** page of the Board Setup dialog. Each type of tuning pattern (single track length, differential pair length, and differential pair skew) can have its own defaults.

Finally, the tuning pattern properties dialog is one of two ways to set the target length or skew for a tuning pattern. Setting length targets is explained below.

Setting target length and skew

There are two ways to set a target length or skew for a net:

- In the properties dialog for a tuning pattern that has already been added to a track.

- Using a custom DRC rule with the `length` and/or `skew` constraints.

The first method is to specify a target in the **target length** or **target skew** field of the tuning pattern's properties dialog. This target will only apply to the selected tuning pattern. Therefore, length targets set in this way must be set separately for each tuning pattern in the design. The properties dialog for a tuning pattern is only accessible after the pattern is initially created, so changing a target length or skew in this way may require the pattern to be adjusted to meet the new target value, if the pattern's geometric constraints do not allow sufficient space to meet the new target.

You can also set a target length and/or skew using custom design rules. If custom rules are used, they will override any targets set in tuning pattern properties, unless the **override custom rules** checkbox is enabled in the tuning pattern properties.

Using a custom rule allows you to set a net's target length and/or skew up front, before a pattern is created. With custom rules you can set different length and skew targets based on specific criteria, such as netclass or net name. You will also result in a DRC violation if the net's length or skew is out of bounds.

When target length or skew is adjusted in a custom DRC rule after a pattern is created, the pattern geometry will not be automatically updated to achieve the new target. You can use **Edit** → **Update All Tuning Patterns**

to recalculate all tuning patterns to meet the new targets.

The following example custom rule sets a target length and skew for nets in the `high_speed` netclass. The target length is 100mm, and a DRC error will be raised if it is below 95mm or above 105mm. The target skew is at most 0.1mm.

```
(rule "target length and skew"
     (condition "A.NetClass == 'high_speed'")
     (constraint length (min 95mm) (opt 100mm) (max 105mm))
     (constraint skew (max 0.1mm)))
```

See the custom rule documentation for more details of how to create rules that only apply to certain nets.

Length tuning pitfalls and tips

The length tuner only tunes nets with a point-to-point topology; branching nets are not supported. When the length tuner encounters a branch, it stops at the branch and only considers the length of the net up to that branch.

Sometimes you may end up with leftover stub tracks somewhere in your design. These can turn what appears to be a point-to-point net into a branched topology, which will prevent length tuning from working as expected. It may be easier to find such stub tracks when you switch footprints, vias, and tracks to outline mode (, , and buttons, respectively). You can also use the track cleanup tool (**Tools → Cleanup Tracks and Vias…**) to remove many of these stubs automatically.

By default, the length tuner includes vias in its length calculations. Only the layer-to-layer length of the via is used, which may be shorter than the full top-to-bottom via height if the tuned path is not exclusively on the board top and bottom. The accuracy of this calculation depends on the board stackup being accurately configured. Via length can be ignored in length tuner calculations by deselecting **include stackup height in track length calculations** in the **Constraints** page of the Board Setup dialog.

The length tuner is optimized for adjusting the effective electrical distance between two points, and therefore it calculates net length in a slightly different way than other tools, such as the Net Inspector. In addition to discounting net branches and unused portions of vias, the length tuner also optimizes paths through pads to use the shortest possible path in its calculations. In comparison, the Net Inspector reports a simple summation of copper segment lengths. Both calculations are accurate, but they are optimized for different purposes. These differences are discussed in more detail in the Net Inspector documentation.

Teardrops

Teardrops are areas of extra copper that smooth the transition between track and pads, vias, or other tracks. Teardrops are added to increase the mechanical robustness of a trace connection. They also reduce the risk of a misaligned drill hole disconnecting a trace from a drilled pad or via.
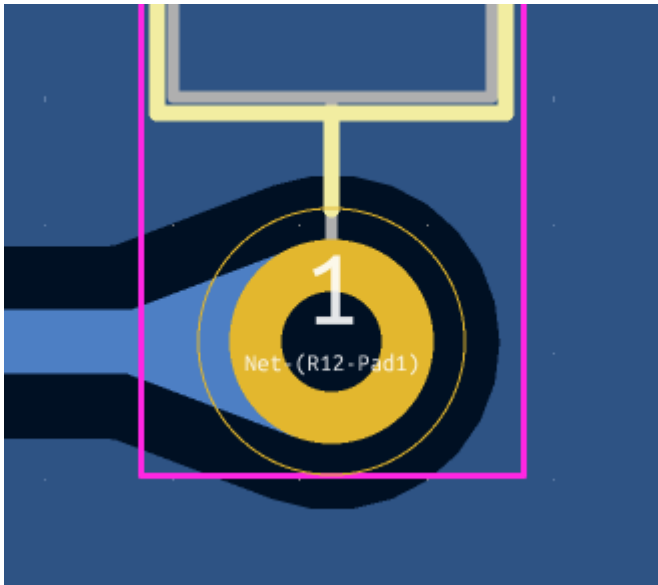
You can add teardrops to your design in bulk using the **Edit → Edit Teardrops...** dialog. This dialog has controls for filtering which objects are affected and settings for configuring the shape of the new teardrops. It also lets you edit or remove existing teardrops.

The **Scope** section controls which types of objects will be affected: PTH pads, SMD pads, vias, and/or track-to-track connections. The **Filter Items** section lets you filter objects by other criteria; you can filter items by net, net class, and layer, or choose to act only on round pads, pre-existing teardrops, or the objects in your selection.

The **Action** section controls whether to add or remove teardrops, as well as the size and shape of the new teardrops. Adding a teardrop to an object that already has a teardrop will update the existing teardrop with the new settings. When adding teardrops, you can choose to use the default teardrop settings from the Board Setup dialog, or choose specific values for the new teardrops.

Prefer zone connection: if selected, a teardrop will not be created if the object is also connected to a zone. **Allow teardrops to span 2 track segments:** if selected, the teardrop will be able to spread over a second track segment if the first segment is too short to support a full teardrop. **Maximum track width:** a teardrop will not be created for a track connection that is wider than this percentage of the pad width (minimum pad dimension). **Best length:** the ideal length of the teardrop, as a percentage of the width (smallest dimension) of the attached object. **Maximum length:** the maximum length of the teardrop, as an absolute length. **Best width:** the ideal width of the teardrop, as a percentage of the width (smallest dimension) of the attached object. **Maximum width:** the maximum width of the teardrop, as an absolute width. **Curved edges:** if selected, the teardrop edges will be curved instead of a straight line. If curved, **points** controls the number of points in the curve; more points will result in a smoother curve.
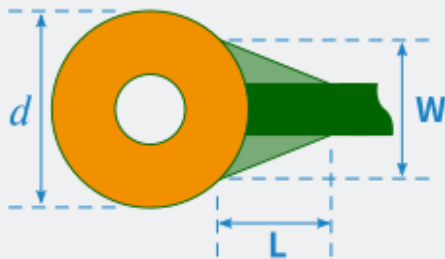
Default properties for teardrops can be configured in the Board Setup dialog. These defaults will be used in the Edit Teardrops dialog when **add teardrops with default values for shape** is selected in that dialog. The defaults are configured separately for teardrops connecting to round shapes, rectangular shapes, or between tracks. The available options for each type of teardrop are the same as in the Edit Teardrops dialog.
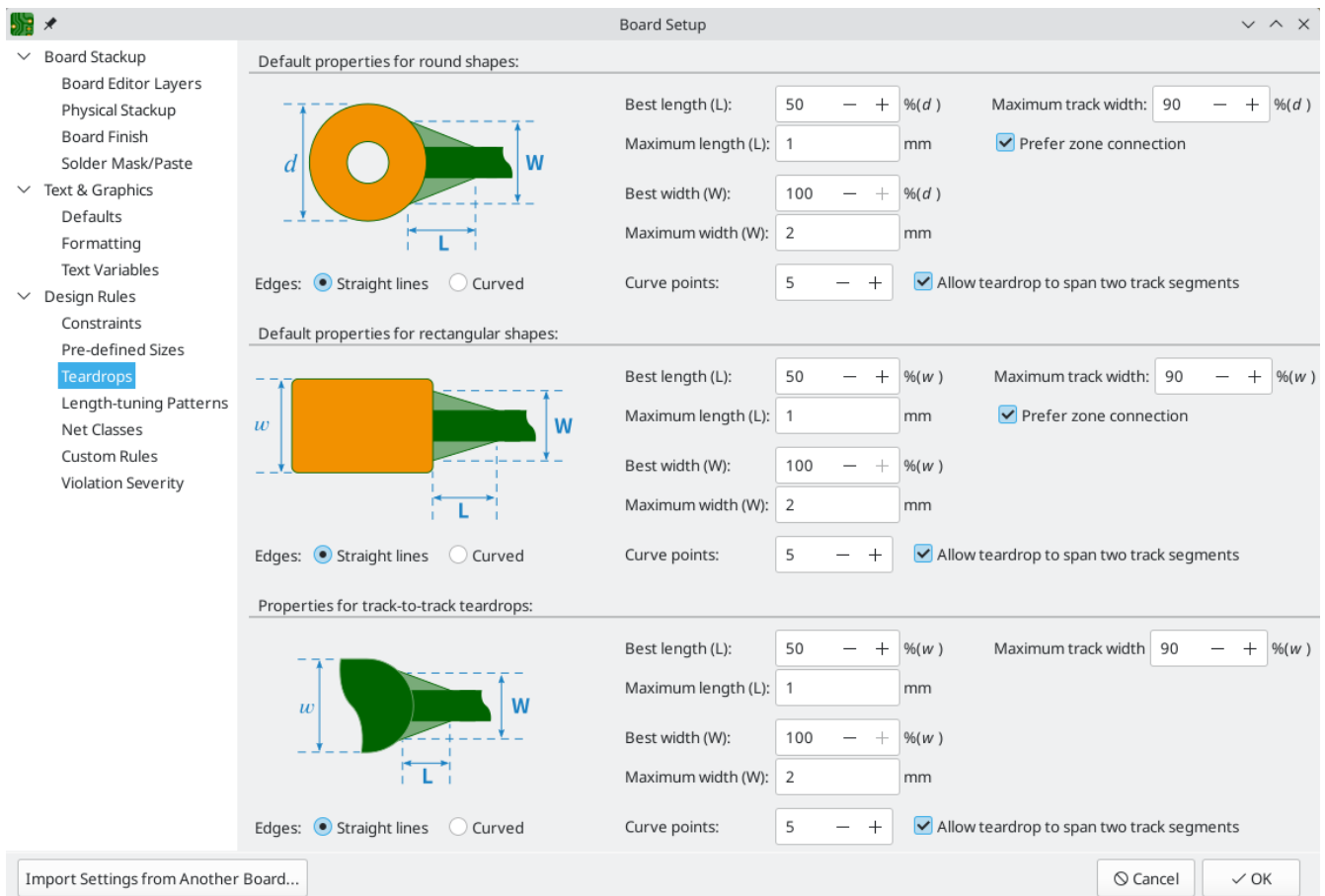
Rather than in bulk, you can add or edit teardrops for individual vias in the properties dialog for that via, or for individual pads in the **Connections** tab of the pad's properties dialog. The settings in the properties dialogs are the same as in the Edit Teardrops dialog. You can also edit teardrops for individual pads and vias with the Properties Manager.
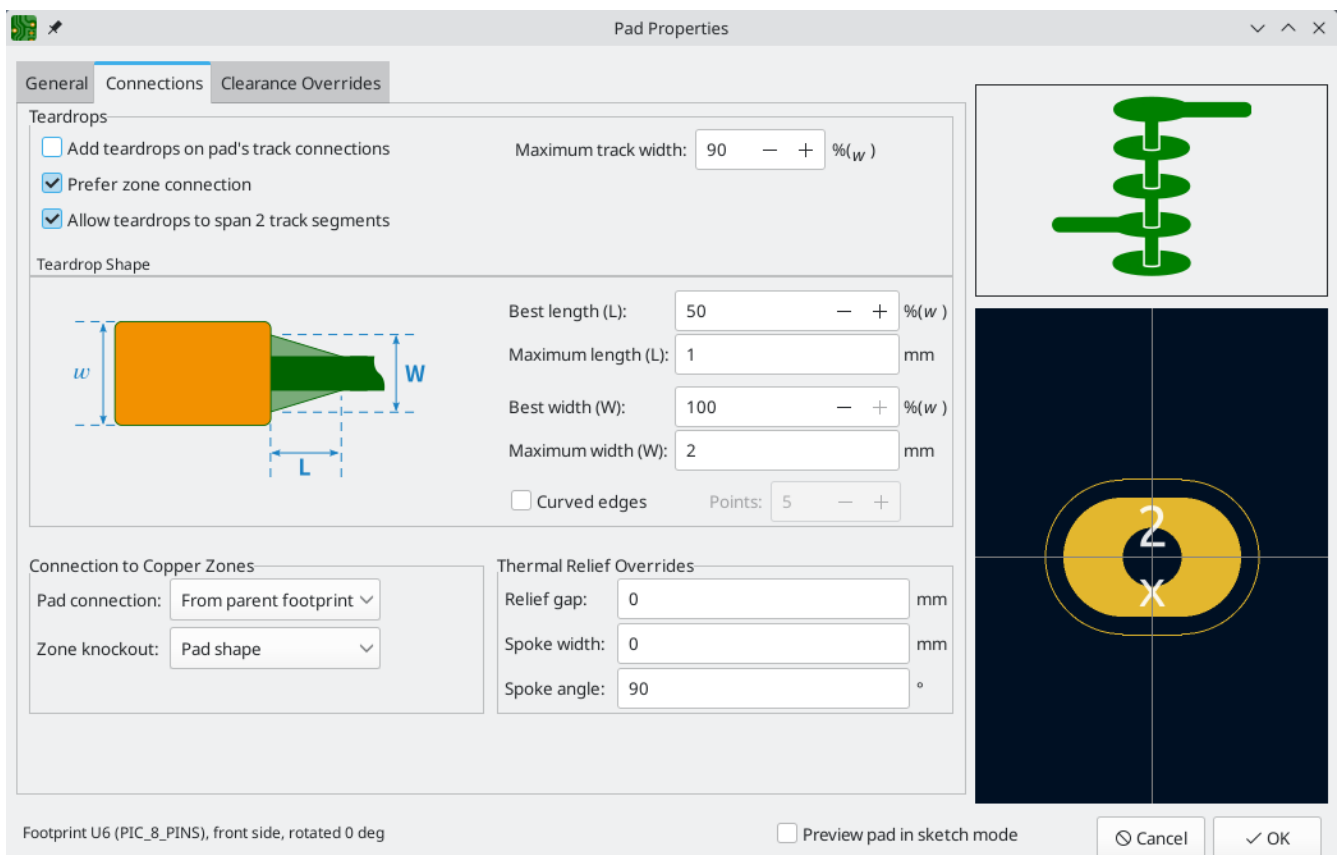
Teardrops in KiCad are small zones, meaning that when they refill they avoid shorting to copper objects on other nets. They are initally filled when they are added, but they are unfilled and refilled with other zones on the board: when using the Unfill All Zones and Refill All Zones commands, running DRC, generating fabrication outputs, etc. Teardrops can be shown in filled or outline mode using the zone display controls in the left toolbar.

Interactive router settings

The interactive router settings can be accessed through the **Route** menu, or by right-clicking on the button in the toolbar. These settings control the router behavior when routing tracks as well as when dragging existing tracks.

Setting	Description
Mode	Sets the operating mode of the router for creating new tracks and dragging existing tracks. [See above] for more information.
Free angle mode	Allows routing tracks at any angle, instead of just at 45-degree increments. This option is only available if the router mode is set to Highlight collisions.
Allow DRC violations	Allow placing tracks and vias that violate DRC rules. This option is only available if the router mode is set to Highlight collisions.
Shove vias	Allow the router to shove vias along with tracks. When this is disabled, vias cannot be shoved. This option is only available if the router mode is set to Shove.
Jump over obstacles	Allow the router to attempt to move colliding tracks behind solid obstacles (such as pads). This option is only available if the router mode is set to Shove.
Remove redundant tracks	Automatically removes loops created in the currently-routed track, keeping only the most recently routed section of the loop.
Optimize pad connections	When this setting is enabled, the router attempts to avoid acute angles and other undesirable routing when exiting pads and vias.
Smooth dragged segments	When dragging tracks, attempts to combine track segments together to minimize direction changes.
Optimize entire track being dragged	When enabled, dragging a track segment will result in KiCad optimizing the rest of the track that is visible on the screen. The optimization process removes unnecessary corners, avoids acute angles, and generally tries to find the shortest path for the track. When disabled, no optimizations are performed to the track outside of the immediate section being dragged.
Use mouse path to set track posture	Attempts to pick the track posture based on the mouse path from the routing start location.
Fix all segments on click	When enabled, clicking while routing will fix the position of all the track segments that have been routed, including the segment that ends at the mouse cursor. A new segment will be started from the mouse cursor location. When disabled, the last segment (the one that ends at the mouse cursor) will not be fixed in place and can be adjusted by further mouse movement.

Graphical objects

Graphical objects (lines, arcs, rectangles, circles, polygons, text, and dimensions) can exist on any layer. Unlike zones, the shape of a graphical object is exactly defined by its own properties, and is not affected by other objects. Shape properties include size, position, line width, and fill.

Graphical objects on copper layers can be assigned nets and make connections to other copper objects, just like tracks and zones.

Graphical shapes

The buttons on the right toolbar can be used to create:

- Lines (╱ , default hotkey `Ctrl` + `Shift` + `L`)

- Arcs (⌒ , default hotkey `Ctrl` + `Shift` + `A`)

- Rectangles (▢)

- Circles (⬤ , default hotkey `Ctrl` + `Shift` + `C`)

- Polygons (⬠ , default hotkey `Ctrl` + `Shift` + `P`)

Arcs have two editing modes, which are selectable in **Preferences → PCB Editor → Editing Options** or by right clicking the ⌒ button on the right toolbar. The first mode (**keep arc center, adjust radius**) maintains the position of the arc center as as the arc endpoints or midpoint are dragged, changing the radius as necessary. The second mode (**keep arc endpoints or direction of starting point**) maintains the position of the arc endpoints and the arc's direction of curvature as the midpoint or center are dragged.

Just like with tracks, you can expand a selection from one graphic line to include all other contiguous graphic lines by pressing `U` .

The properties of a graphic shape can be adjusted in the shape's properties dialog or with the properties manager. Rectangles, circles, and polygons can be filled shapes or outlines. The **line width** option controls the width of the outline, even for filled objects. The outline width extends on both sides of the "ideal" shape of the graphic object. For example, a graphic circle that is defined to have 2mm radius and 0.2mm line width will consist of a torus with an outer radius of 2.2mm and inner radius of 1.8mm. If the **filled shape** option is enabled and the line width is set to 0, the shape will be a filled circle with 2mm radius. Several line styles are available: solid, dashed, dotted, dash-dot, and dash-dot-dot.

You can customize the default style of newly-created graphical shapes in the Text & Graphics Defaults section of the Board Setup dialog.

Graphical shapes on copper layers can have a net assigned in their properties dialog. Copper shapes with a net make connections like tracks or zones. Unlike zones, copper graphical objects always maintain their shape and do not keep clearance to other copper objects.

Shape modification tools

KiCad has several tools for modifying combinations of graphic shapes in useful ways, such as chamfering two lines or combining two polygons. These tools are used by selecting the shapes you want to modify, right clicking, and then choosing the relevant tool in the **Shape Modification** submenu. Different tools are available for different combinations of selected shapes.

Heal shapes fixes a discontinuity between two lines or arcs. A new line segment is added to connect the ends of each shape together, up to a specified tolerance.

Fillet lines adds an arc to round the corner between two connected lines with a specified radius. The two original lines are shortened to meet the endpoints of the arc.

Chamfer lines adds a line segment to create a new edge between two connected lines with a specified setback. The two original lines are shortened to meet the endpoints of the new segment.

Extend lines to meet lengthens two selected lines until they intersect each other. The two lines will share a coincident endpoint.

Merge polygons combines two or more selected polygons into one new polygon that is the union of the original shapes.

Subtract polygons subtracts one or more polygons from another polygon, resulting in a new polygon that is the difference of the original shapes. The first-selected polygon(s) are subtracted from the last-selected polygon.

Intersect polygons results in a new polygon that is the shape of the overlapping area between two or more selected polygons.

Converting objects to and from graphic shapes

KiCad provides tools to convert graphic objects to other types of objects, other types of objects to graphic objects, and graphic objects to other kinds of graphic objects. These tools are used by selecting the shapes you want to convert, right clicking, and then choosing the desired result object from the **Create From Selection** submenu. Most types of object conversions have several conversion options that are presented in a settings dialog. The exact options differ based on the target object type.

When converting to a graphic polygon, rule area, or zone, there are several options for how to convert the source objects into a polygonal outline.

- If **copy line width of first object** is selected, an unfilled polygon will be created that has its line width taken from the line width of the first selected source object. This option is only available when converting to a graphic polygon, and the source object must be a closed shape.

- If **use centerlines** is selected, an object with zero line width will be created, with its outline placed at the centerlines of the source objects. The source object must be a closed shape. If the target object is a graphic polygon, it will be filled.

- If **create bounding hull** is selected, an object will be created with the specified **line width**. The object's outline will be offset from the outermost extents of the source object by the specified **gap**. The source object does not need to be a closed shape when a bounding hull is created.

Most conversions provide a **delete source objects after conversion** option, which will result in the original object being deleted during the conversion, only leaving the new object in place. If this option is not selected, the conversion will leave the original object in place in addition to the new object. The original object will be selected following the conversion so that it can be manually deleted by pressing `Delete`.
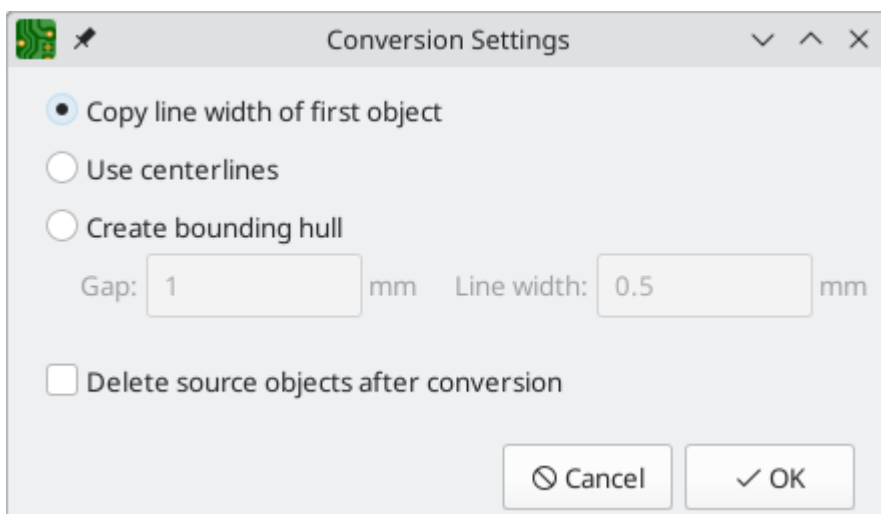
The following conversion types are available:

-

Create Polygon From Selection converts a graphic shape, text, zone, rule area, or track into a polygon. This can be used to convert separate graphic shapes, such as lines and arcs, into a unified shape. It can also be used to convert a text object into a shape that can have its outline manipulated graphically.

- **Create Zone From Selection** converts a graphic shape, text, zone, rule area, or track into a zone. In addition to the conversion settings, the conversion dialog also shows options for configuring the resulting zone. This can be used to create zone outlines with complex shapes, such as curves, that would otherwise be difficult to create using the zone tool.

- **Create Rule Area From Selection** converts a graphic shape, text, zone, rule area, or track into a rule area. In addition to the conversion settings, the conversion dialog also shows options for configuring the resulting rule area. This can be used to create rule area outlines with complex shapes, such as curves, that would otherwise be difficult to create using the rule area tool.

- **Create Lines From Selection** converts a graphic polygon or rectangle into graphic lines that follow the source shape's outline. This can be used to convert a unified shape into its constituent outline segments.

- **Create Tracks From Selection** converts a graphic shape, zone, or rule area into tracks that follow the source shape's outline. If the source object is not on a copper layer, a dialog will be presented to specify the target copper layer. The source object is not removed following conversion, but remains selected so that it can be easily deleted if desired.

- **Create Arc From Selection** converts a graphic line segment or track segment into a graphic arc. The arc's endpoints are placed at the endpoints of the source segment and its thickness is taken from the source object's line thickness. The source segment is not removed following conversion, but remains selected so that it can be easily deleted if desired.

Text objects

Graphical text may be placed by using the ⊤ button in the right toolbar or by keyboard shortcut `Ctrl` + `Shift` + `T`. Activating the tool brings up a text properties dialog. After configuring the text and its properties and accepting the dialog, you can click in the canvas to place the text.

You can also add text boxes, which are similar to regular text except that they have an optional border and they automatically reflow text within that border. Text boxes are placed with the ▤ button, and require clicking twice to specify the top left and bottom right corners of the box.

Locked controls whether or not the text object is locked. Locked objects may not be manipulated or moved, and cannot be selected unless the **Locked Items** option is enabled in the Selection Filter panel.

Layer controls the text's layer. Text may be placed on any layer, but note that text on copper layers cannot be associated with a net and cannot form connections to tracks or pads. Copper zones will fill around the rectangular bounding box of text objects.

There are several formatting options: text can be bolded, italicized, left/right/center aligned, and reversed. Regular text objects (not text boxes) can also have their vertical alignment adjusted. The **knockout** option, which is also limited to regular text objects, adds a solid rectangle surrounding the text and makes the text itself a negative cutout.

The text itself can use any TTF font available on your system, or the built-in KiCad stroke font.

NOTE	User fonts are not embedded in the project. If the project is opened on another computer that does not have the selected font installed, a different font will be substituted. For maximum compatibility, use the KiCad font. Also consider converting text objects to polygons before sharing a project (right click a text object → **Create from Selection** → **Create Polygon from Selection…**). Text converted to polygons is not editable as text, but will render identically on any computer.

You can adjust the text size with the **text width** and **text height** controls. When you are using the KiCad font, you can also adjust the stroke width with the **thickness** control.

Position X and **position Y** control the text object's location. These properties are not available for text boxes.

Orientation is the rotation angle of the text object. You can select an angle in 90 degree increments from the dropdown, or type in an arbitrary angle.

Text boxes additionally have options controlling their border. The **border** checkbox makes the border visible or invisible. For visible borders, you can adjust the border's thickness with the **border width** control and the line style with the **border style** control (solid, dashed, dotted, dash-dot, or dash-dot-dot).

NOTE	You can customize the default style of newly-created text objects in the Text & Graphics Defaults section of the Board Setup dialog.

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

| Feature	Markup Syntax	Result
Superscript	`text^{superscript}`	text^{superscript}
Subscript	`text_{subscript}`	text$_{subscript}$
Overbar	`~{text}`	$\overline{text}$
Variables	`${variable}`	*variable_value*
Symbol Fields	`${refdes:field}`	*field_value* of symbol *refdes*

| NOTE	Variables must be defined in Board Setup before they can be used. There are also a number of built-in text variables.

Dimensions

Dimensions are graphical objects used to show a measurement or other marking on a board design. They may be added on any drawing layer, but are normally added to one of the User layers. KiCad currently supports five different types of dimension: aligned, orthogonal, center, radial, and leader.

Aligned dimensions (⬈) show a measurement of distance between two points. The measurement axis is the line that connects those two points, and the dimension graphics are kept parallel to that axis.

Orthogonal dimensions (⬑) also measure the distance between two points, but the measurement axis is either the X or Y axis. In other words, these dimensions show the horizontal or vertical component of the distance between two points. When creating orthogonal dimensions, you can select which axis to use as the measurement axis based on where you place the dimension after selecting the two points to measure.

Center dimensions (⊹) create a cross mark to indicate a point or the center of a circle or arc.

Radial dimensions (⊹) show a measurement between a center point and the outside of a circle or arc. The center point is indicated by a cross.

Leader dimensions (⬉) create an arrow with a leader line connected to a text field. This text field can contain any text, and an optional circular or rectangular frame around the text. This type of dimension is often used to call attention to parts of the design for reference in fabrication notes.

After creating a dimension, its properties may be edited (hotkey `E`) to change the format of the displayed number and the style of the text and graphic lines.

> **NOTE** You can customize the default style of newly-created dimension objects in the Text & Graphics Defaults section of the Board Setup dialog.

Dimension format options

Override value: When enabled, you may enter a measurement value directly into the **Value** field that will be used instead of the actual measured value.

Prefix: Any text entered here will be shown before the measurement value.

Suffix: Any text entered here will be shown after the measurement value.

Layer: Selects which layer the dimension object exists on.

Units: Selects which units to display the measured value in. **Automatic** units will result in the dimension units changing when the display units of the board editor are changed.

Units format: Select from several built-in styles of unit display.

Precision: Select how many digits of precision to display.

Dimension text options

Most of the dimension text options are identical to those options available for other graphical text objects (see the Graphical Objects section above). Some specific options for dimension text are also available:

Position mode: Choose whether to position the dimension text manually, or to automatically keep it aligned with the dimension measurement lines.

Keep aligned with dimension: When enabled, the orientation of the dimension text will be adjusted automatically to keep the text parallel with the measurement axis.
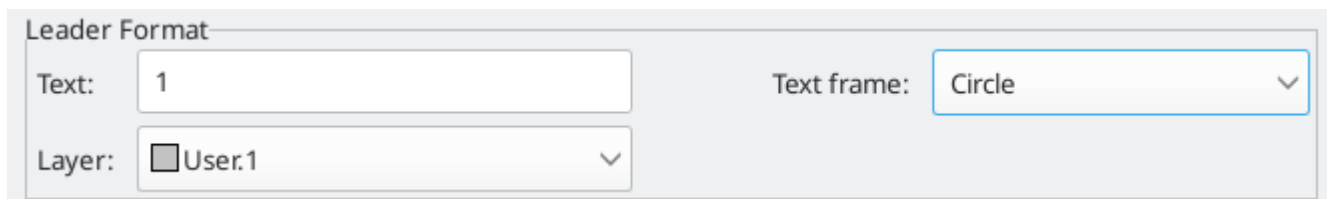
Dimension line options

Line thickness: Sets the thickness of the graphical lines that make up a dimension's shape.

Arrow length: Sets the length of the arrow segments of the dimension's shape. A negative arrow length reverses the arrow direction.

Extension line offset: Sets the distance from the measurement point to the start of the extension lines.

Extension line overshoot: Sets the distance from the dimension's line to the end of the extension lines.

Leader options

Value: Enter the text to show at the end of the leader line.

Text frame: Select the desired border around the text (circle, rectangle, or none).

Bulk editing text and graphics

Properties of text and graphics, including dimensions, can be edited in bulk using the **Edit Text and Graphics Properties** dialog (**Edit → Edit Text & Graphic Properties…**).

Scope and Filters

Scope settings restrict the tool to editing only certain types of objects. If no scopes are selected, nothing will be edited.

Filters restrict the tool to editing particular objects in the selected scope. Objects will only be modified if they match all enabled and relevant filters (some filters do not apply to certain types of objects. For example, parent footprint filters do not apply to graphic items and are ignored for the purpose of changing graphic properties). If no filters are enabled, all objects in the selected scope will be modified. For filters with a text box, wildcards are supported: `*` matches any characters, and `?` matches any single character.

Filter items by layer filters to items on the specified board layer.

Filter items by parent reference designator filters to fields in the footprint with the specified reference designator. **Filter items by parent footprint library id** filters to fields in footprint with the specified library identifier.

Selected items only filters to the current selection.

Action

Properties for filtered objects can be set to new values in the bottom part of the dialog. Properties can be set to arbitrary values by selecting **set to specified values** or reset to their layer's default value by selecting **set to layer default values**.

Drop-down lists and text boxes can be set to `-- leave unchanged --` to preserve existing values. Checkboxes can be checked or unchecked to enable or disable a change, but can also be toggled to a third "leave unchanged" state.

All items can have their **layer** set.

Graphic items can have their **line thickness** modified.

Text properties that can be modified are **font**, **text width**, **text height**, **text thickness** (KiCad font only), emphasis (**bold** and **italic**), orientation (**keep upright**), and alignment (**center on footprint**). Footprint text can also have its **visibility** set.

Cleaning up graphics

There is a dedicated tool for performing common cleanup operations on graphics, which is run via **Tools → Cleanup Graphics…**.
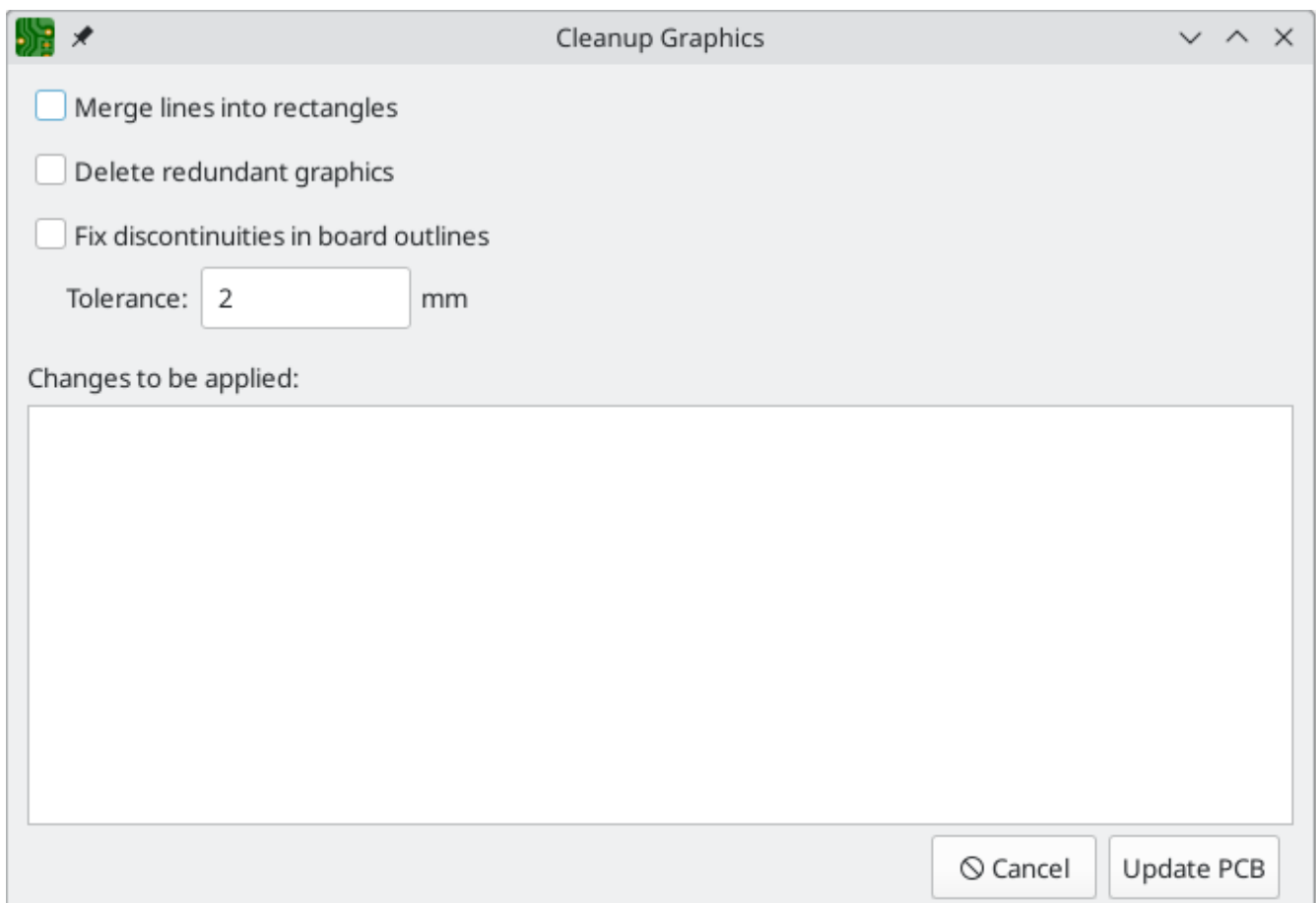
The following cleanup actions are available and will be performed when selected:

Merge lines into rectangles: combines individual graphic lines that together form a rectangle into a single rectangle shape object.

Delete redundant graphics: deletes graphics objects that are duplicated or degenerate.

Fix discontinuities in board outlines: modifies the existing board outline to fix any discontinuities that are within the specified tolerance.

Any changes that will be applied to the board are displayed at the bottom of the dialog. They are not applied until you press the **Update PCB** button.

Rule areas (keepouts)

Rule areas, also known as keepouts, are board regions that can have specific DRC rules defined for them. Some basic rules are available that will raise DRC errors if certain types of objects are within the bounds of the rule area, but rule areas can also be used together with custom DRC rules to define complex DRC behavior that only applies within the rule area.

You can add a rule area by clicking the ▧ button on the right toolbar (`Ctrl` + `Shift` + `K`). Click on the canvas to place the first corner, which will show the Rule Area Properties dialog. After configuring the rule area appropriately, press **OK** to continue placing corners of the rule area. The rule area shape can be an arbitrary polygon; click on the starting corner or double click to finish placing the rule area.
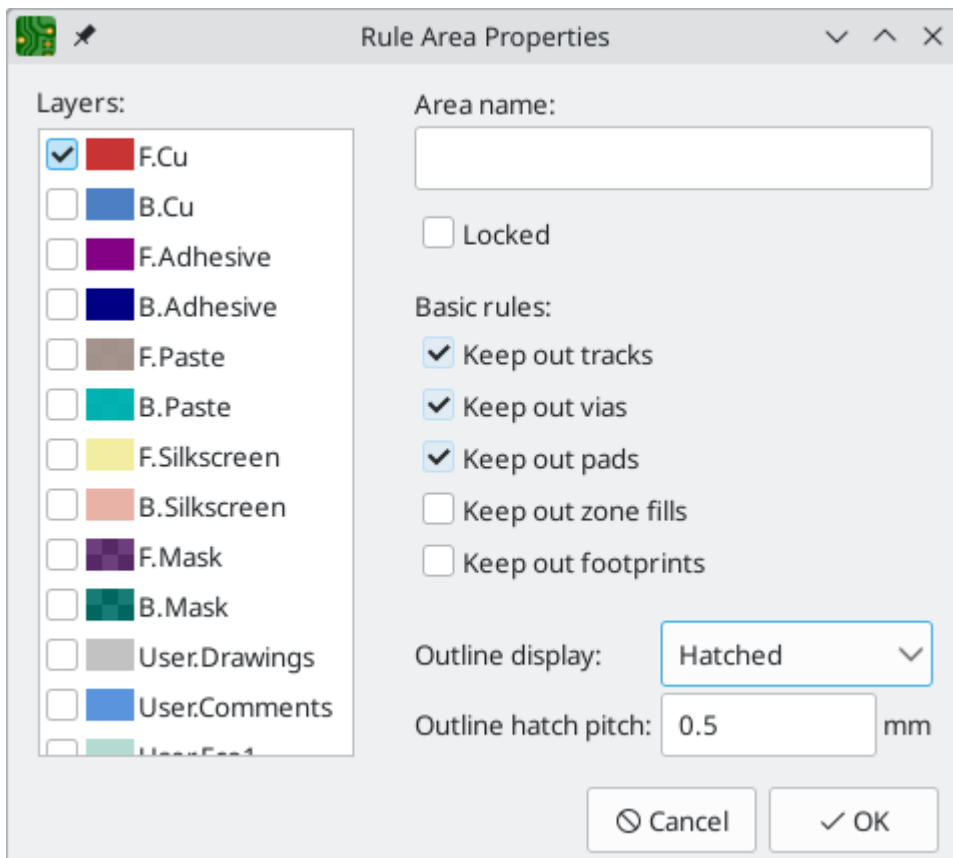
The Rule Area Properties dialog has the following options:

The **layers** list determines which layers the rule area applies to. The area only appears on these layers and the selected keepout rules only apply on these layers. At least one layer must be selected. By default, the active layer in the editing canvas is preselected in the rule area layer list.

The **area name** field is optional and provides an identifier for the rule area. If it is provided, it is included in DRC violation messages to make them clearer. It can also be used in custom DRC rules to identify a particular rule area.

The **locked** checkbox determines if the rule area should be locked. As with other objects, rule areas can also be locked or unlocked after they are created.

Several **basic rules** are available to keep out various types of objects. The basic rules can be configured to keep out tracks, vias, pads, zone fills, and/or footprints. If an object of one of the selected types is within the rule area, a DRC error will be raised. Additionally, zone fills will automatically avoid a rule area if the rule area is configured to keep out zones.

NOTE	Even with no basic rules selected, rule areas can still be used to define specific areas in which to apply custom DRC rules.

There are a few options for the **outline display** of the rule area. The area can be shown with a hatched outline, fully hatched throughout the area, or with just the outline with no hatching. The **outline hatch pitch** is also adjustable.

Locking

Most objects can be locked through their properties dialogs, by using the right-click context menu, or by using the Toggle Lock hotkey (L). Locked objects cannot be selected unless the "Locked items" checkbox is enabled in the selection filter. Attempting to move locked items will result in a warning dialog:
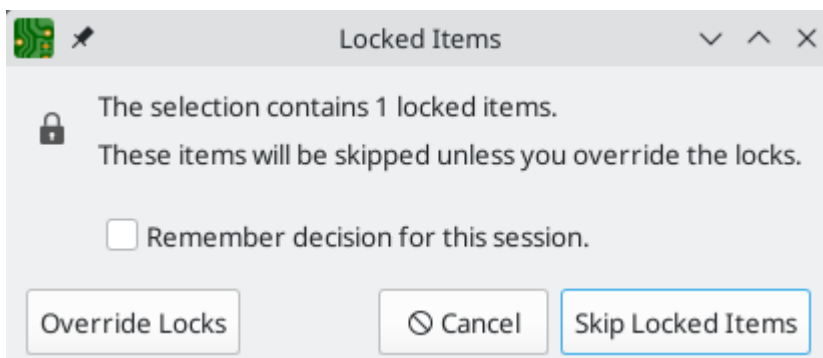
Selecting "Override Locks" in this dialog will allow moving the locked items. Selecting "OK" will allow you to move any unlocked items in the selection; leaving the locked items behind. Selecting "Do not show again" will remember your choice for the rest of your session.

Locked items are displayed with a colored shadow around them. This can be customized in your color scheme.

Groups

Groups let you treat multiple objects as a single object for the purposes of moving or rotating them. Each object in the group will maintain its position relative to the other objects in the group. Groups can also have a name, which is displayed in the editing canvas when the group is selected.
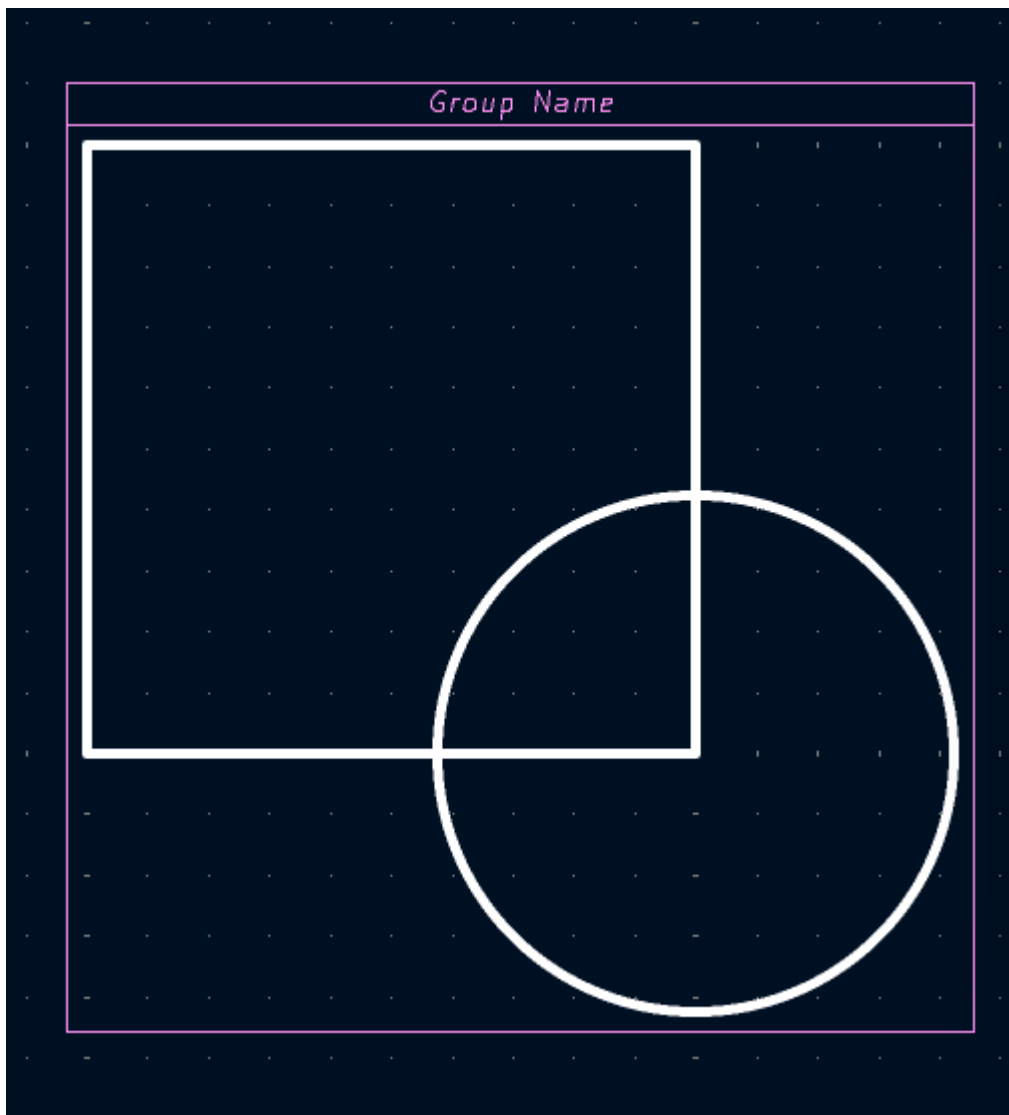
Most types of objects in the Board Editor can be grouped: footprints, tracks, zones, graphic items, and even other groups. Groups can contain multiple different types of objects at once.

To add objects to a group, select them, then right click and choose **Grouping → Group Items**. To remove all items from a group, select the group, right click, and choose **Grouping → Ungroup Items**.

Once objects have been added to a group, selecting any of the objects will select the group as a whole instead of the constituent objects. To edit a specific object within a group, first select the group, the right click and choose **Enter Group**. Double clicking on a group also enters the group. When a group has been entered, objects within the group can be selected and edited individually without affecting the other objects in the group. To leave the group and stop editing its members individually, right click and select **Leave Group**, select an object outside the group, or use `Esc`.

There are several ways to modify which objects belong to a group. To remove objects from an existing group, enter the group, then select the objects you want to remove, right click, and choose **Grouping → Remove Items**. To add items to a group, first ungroup all the items from the group. This will leave the group's former members selected. Then add the new item to the selection and group the selection. Note that without first ungrouping, this process would create a nested group: a new group containing the new item and the entire original group, not just the items in the original group.

You can also add or remove objects from a group in the group's properties dialog. To open a group's properties dialog, press E or right click and click **Properties....** The properties dialog lists the objects contained in the group. To add an additional object to the group, click the + button, then click on the desired object in the editing canvas. The object you click on will be added to the group. To remove an object, select it in the list, then click the 🗑 button.
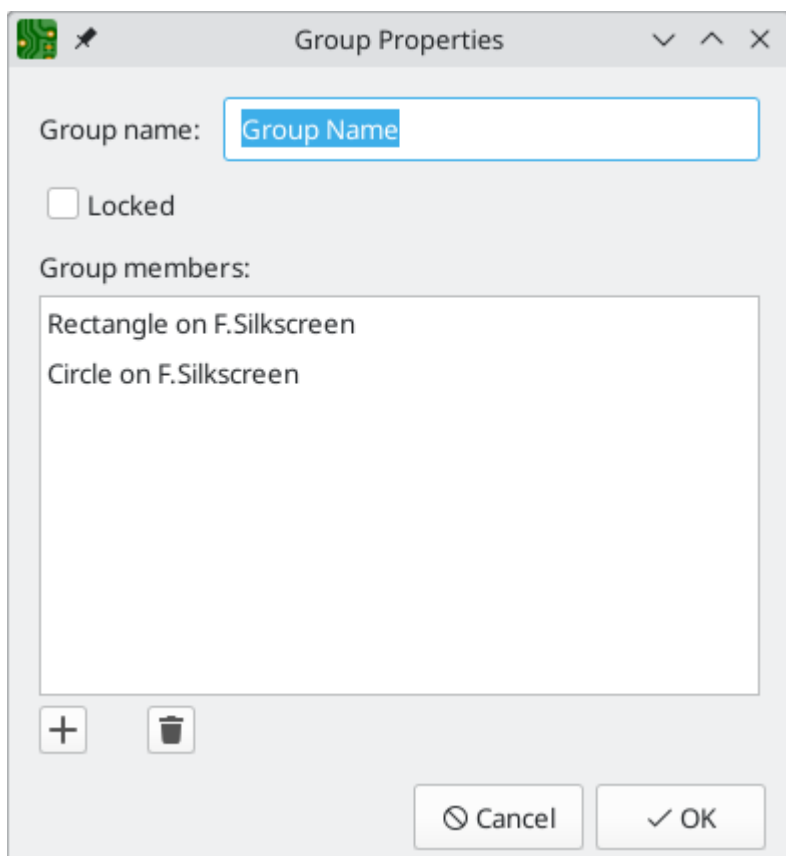
The group properties dialog also lets you specify a name for the group or lock the group. Groups can also be named or locked using the properties manager.

Arrays

KiCad has an array tool to create rectangular or circular arrays of objects (footprints, vias, graphical objects, etc.). Two types of array are possible: **Grid** and **Circular**.

Grid Arrays are rectangular and are described by a **horizontal count** and a **vertical count**, which set the number of columns and rows in the array, respectively. The **horizontal** and **vertical spacing** settings describe the distance between columns and rows, while the **horizontal** and **vertical offset** settings describe a shift applied to each row/column compared to the previous row/column.

You can create a repeating staggered pattern by choosing a **stagger** setting, which controls the number of rows or columns that are offset before the pattern repeats. You can stagger by **row** or by **column**. For example, if two staggered rows are selected, each row will be horizontally offset from the previous row by half of the array's horizontal spacing setting. Every other row will be placed at the original spacing and offset. If three staggered columns are selected, each column will be vertically offset by a third of the array's vertical spacing setting. Every third column will be placed at the original spacing and offset. Offsets from the stagger settings are added to the previous horizontal and vertical offset settings.

Circular Arrays are described by a center point, an angular spacing, and a number of arrayed items. If **set center by position** is selected, the center point of the array will be defined by the absolute X/Y position you enter in **center pos X** and **center pos Y**. If **set center by radius** is selected, the center point of the array will be defined by the **array radius** and **array angle** parameters, which describe the position of the array center relative to the source object's position. The source object's position is displayed for reference as **ref point pos X** and **ref point pos Y**.

The **item count** field determines the number of objects in the array, including the source object. The **angle** field determines the angular spacing between items, with the center point at the center of the array. Positive angles result in a counter-clockwise rotation relative to the center point and the source item, while negative angles result in a clockwise rotation. An angle of `0` will result in a complete circle with objects evenly spaced to provide the specified number of objects. If the item count is too small to create a full circle with the

specified spacing between objects, the array will not be a complete circle. When **rotate items** is selected, objects will be rotated around their origins as array sweeps around the center point. Otherwise, objects will maintain the same orientation as the source item.

When creating an array of footprints, whether rectangular or circular, the **Footprint Annotation** settings control how the reference designators will be set in the new footprints. This affects the linkage of the new footprints to the schematic. If **keep existing reference designators** is selected, the new footprints in the array will have the same reference designators as the source footprints, resulting in duplicated reference designators in the board. If **assign unique reference designators** is selected, each new footprint created in the array will have a unique reference designator automatically assigned.

NOTE	Creating an array of footprints will result in multiple copies of the source footprint(s). If you are using a schematic-based workflow, this will result in footprints that are not represented in the schematic, so careful syncing between the board and the schematic will be needed.

After creating an array, the newly added objects remain selected (not including the original source object), which allows you to easily delete the array if the parameters need to be adjusted.

Cleanup tools

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

Importing vector drawings

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

Using reference images

KiCad supports displaying reference images in the canvas. These are background images that you can use to help you lay out a board; they are purely for reference during the design process and are not included in any fabrication outputs.

To add a reference image, use the button on the right toolbar and select the desired reference image file.

Once the image has been added to the canvas, you can scale it by dragging the editing canvas or open its properties dialog (E) and set the scale explicitly in the **Image** tab. Here you can also **Convert to Greyscale** if you wish.
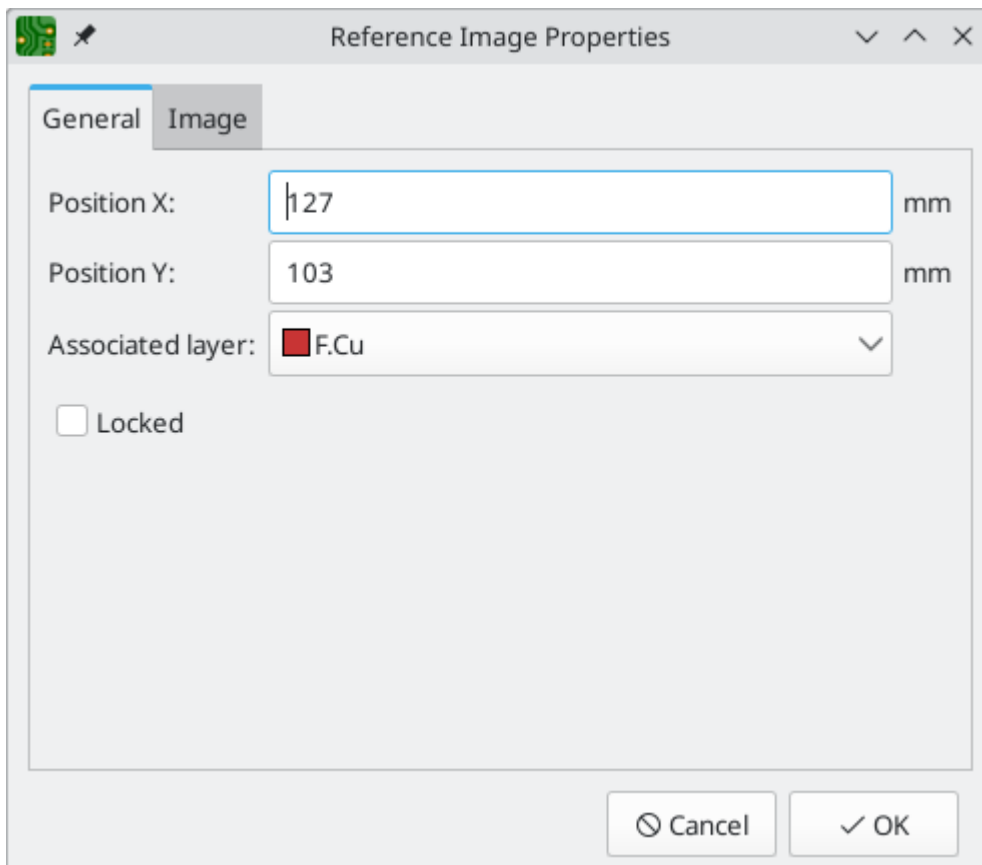
Reference images have an associated layer; they are shown and hidden along with this layer. The layer initially associated with a reference image is the layer that was active when the image was added. You can change the associated layer in the image's properties.

Another way to hide reference images is with the appearance manager. You can show or hide all reference images by toggling the visibility of **Image** objects in the **Objects** tab (button). You can also adjust the opacity of reference images here.

Forward and back annotation

Update PCB From Schematic (forward annotation)

Use the Update PCB from Schematic tool to sync design information from the Schematic Editor to the Board Editor. The tool can be accessed with **Tools → Update PCB from Schematic** (F8) in both the schematic and board editors. You can also use the icon in the top toolbar of the Board Editor. This process is often called forward annotation.

NOTE	Update PCB from Schematic is the preferred way to transfer design information from the schematic to the PCB. In older versions of KiCad, the equivalent process was to export a netlist from the Schematic Editor and import it into the Board Editor. It is no longer necessary to use a netlist file.

The tool adds the footprint for each symbol to the board and transfers updated schematic information to the board. In particular, the board's net connections are updated to match the schematic.

The changes that will be made to the PCB are listed in the *Changes To Be Applied* pane. The PCB is not modified until you click the **Update PCB** button.

You can show or hide different types of messages using the checkboxes at the bottom of the window. A report of the changes can be saved to a file using the **Save…** button.

Options

The tool has several options to control its behavior.

Option	Description
Re-link footprints to schematic symbols based on their reference designators	Footprints are normally linked to schematic symbols via a unique identifier created when the symbol is added to the schematic. A symbol's unique identifier cannot be changed.
	If checked, each footprint in the PCB will be re-linked to the symbol that has the same reference designator as the footprint.
	If unchecked, footprints and symbols will be linked by unique identifier as usual, rather than by reference designator. Each footprint's reference designator will be updated to match the reference designator of its linked symbol.
	This option should generally be left unchecked. It is useful for specific workflows that rely on changing the links between schematic symbols and footprints, such as refactoring a schematic for easier layout or replicating layout between identical channels of a design.
Delete footprints with no symbols	If checked, any footprint in the PCB without a corresponding symbol in the schematic will be deleted from the PCB. Footprints with the "Not in schematic" attribute will be unaffected.
	If unchecked, footprints without a corresponding symbol will not be deleted.
Replace footprints with those specified in the schematic	If checked, footprints in the PCB will be replaced with the footprint that is specified in the corresponding schematic symbol.
	If unchecked, footprints that are already in the PCB will not be changed, even if the schematic symbol is updated to specify a different footprint.

Update Schematic from PCB (back annotation)

The typical workflow in KiCad is to make changes in the schematic and then sync the changes to the board using the Update PCB From Schematic tool. However, the reverse process is also possible: design changes can be made in the board and then synced back to the schematic using **Tools → Update Schematic From PCB** in either the schematic or board editors. This process is often called back annotation.

The tool syncs changes in reference designators, values, footprint assignments, and net names from the board to the schematic. Each type of change can be individually enabled or disabled.

The changes that will be made to the schematic are listed in the *Changes To Be Applied* pane. The schematic is not modified until you click the **Update Schematic** button.

You can show or hide different types of messages using the checkboxes at the bottom of the window. A report of the changes can be saved to a file using the **Save...** button.

Options

The tool has several options to control its behavior.

Option	Description
Re-link footprints to schematic symbols based on their reference designators	If checked, each footprint in the PCB will be re-linked to the symbol that has the same reference designator as the footprint. This option is incompatible with updating symbol reference designators. If unchecked, footprints and symbols will be linked by unique identifier as usual, rather than by reference designator.
Reference designators	If checked, symbol reference designators will be updated to match the reference designators of the linked footprints. If unchecked, symbol reference designators will not be updated.
Values	If checked, symbol values will be updated to match the values of the linked footprints. If unchecked, symbol values will not be updated.
Footprint assignments	If checked, footprint assignments will be updated for symbols which have had their footprints changed or replaced in the board. If unchecked, symbol footprint assignments will not be updated.
Net names	If checked, the schematic will be updated with any net name changes that have been made in the board. Net labels will be updated or added to the schematic as necessary to match the board. If unchecked, net names will not be updated in the schematic.

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

Back annotation with CMP files

Select changes can also be synced from the PCB back to the schematic by exporting a CMP file from the PCB editor (**File → Export → Footprint Association (.cmp) File…**) and importing it in the Schematic Editor (**File → Import → Footprint Assignments…**).

NOTE
This method can only sync changes made to footprint assignments and footprint fields. It is recommended to use the Update Schematic from PCB tool instead.

Geographical re-annotation

The Geographical Reannotation tool lets you automatically set the reference designators of footprints based on their physical location on the board.

To run the Geographical Reannotation tool, use **Tools → Geographical Reannotate…**. This opens the geographical reannotation dialog with options for how to perform the reannotation.
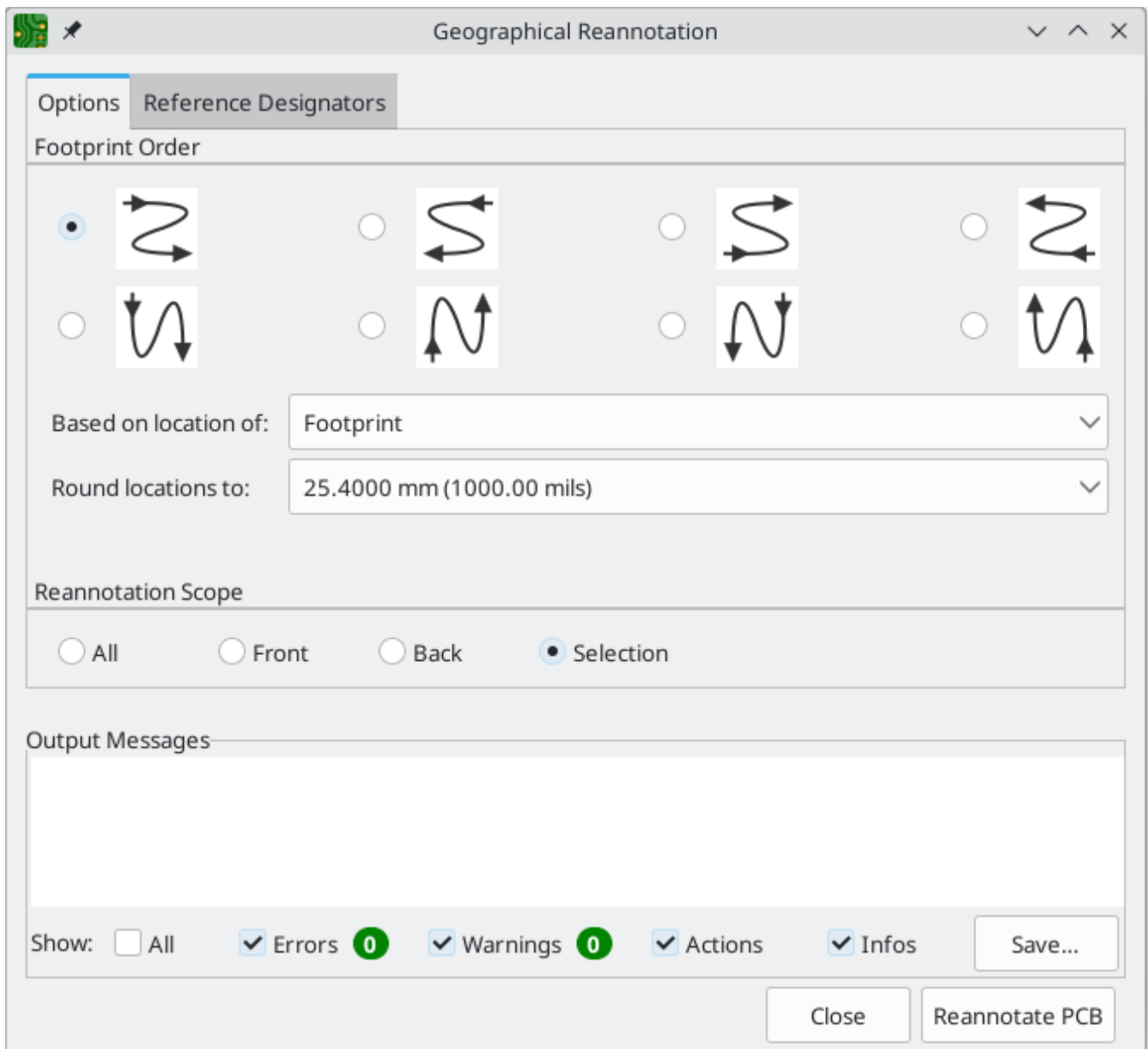
The **Options** tab contains settings for how footprint locations affect reannotation. The arrow diagrams indicate which geographical ordering to use when reannotating. You can reannotate from left-to-right, right-to-left, top-to-bottom, or bottom-to-top, and you can select whether to use a column-major order (go through all footprints in the same column before moving to the next column) or row-major order (go through all footprints in the same row before moving to the next row).

Geographical reannotation can either use the location of the footprint itself or the location of the footprint's reference designator. You can also select how much to round footprint locations before determining which footprints are at the same X or Y position. Rounding to a finer coordinate resolution will result in fewer footprints considered to be in the same row or column.

Finally, you can select which footprints to reannotate. You can reannotate all footprints on the board, all footprints on the front or back of the board, or all footprints in your selection.

The **Reference Designators** tab contains options for how to allocate new reference designators. There are separate settings for footprints on the front and back of the board.

Reference start controls the number for the first new reference designator on each side of the board. If no start value is given for the back of the board, back side footprints will be annotated starting at one higher than the last front side reference designator.

Prefix specifies a prefix string to insert at the beginning of each newly assigned reference designator. This prefix will be inserted before any prefix that is already present. If the **remove prefix** option is selected, footprints with the specified prefix will instead have that prefix removed instead of added. Footprints without that prefix will not have not have any prefix added or removed.

If **exclude locked footprints** is checked, locked footprints will not be reannotated. You can also avoid reannotating specific footprints by entering their reference designators as a comma-separated list in the **exclude references** box.

When you click the **Reannotate PCB** button, footprints will be reannotated according to the selected settings.

The Geographical Reannotation tool updates reference designators in the board, but not in the schematic. After geographically reannotating the board, be sure to sync the updated reference designators to the schematic by running the Update Schematic from PCB tool with the **re-link footprints to schematic symbols based on their reference designators** option disabled. If the schematic is not updated, reference designators in the board will not match those in the schematic.

Inspecting a board

Measurement tool

The measurement tool allows you to make distance and angle measurements between points on the PCB. To activate the tool, click the ![ruler icon] icon in the right toolbar, or use the hotkey `Ctrl` + `Shift` + `M` . Once the tool is active, click once to set the measurement start point, then click again to finish a measurement.

NOTE	The measurement tool is used for quick measurements that do not need to be displayed permanently. Any measurement you make will only be shown while the tool is active. To create permanent dimensions that will appear in printouts and plots, use the Dimension tools.

Design rule checking

The design rule checker is used to verify that the PCB meets all the requirements established in the Board Setup dialog and that all pads are connected according to the netlist or schematic. KiCad can automatically prevent some design rule violations while routing tracks, but many others cannot be prevented automatically. This means it is important to use the design rule checker before creating manufacturing files for a PCB.

To use the design rule checker, click the ![checklist icon] icon in the top toolbar, or select **Design Rules Checker** from the **Inspect** menu.
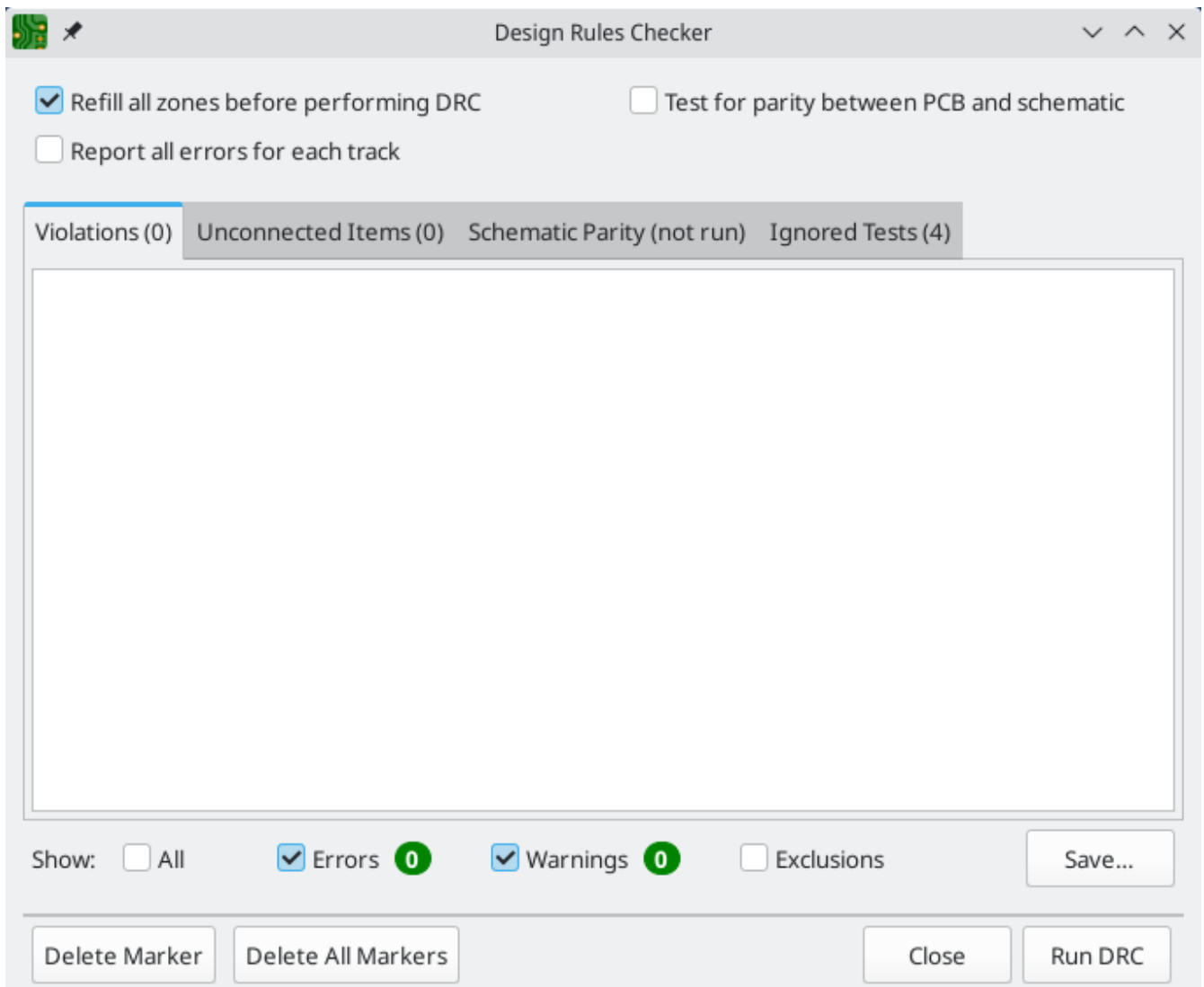
The top section of the DRC Control window contains some options that control the design rule checker:

Refill all zones before performing DRC: when enabled, zones will be refilled every time the design rule checker is run. Disabling this option may result in incorrect DRC results if zones have not been refilled manually.

Report all errors for each track: when enabled, all clearance errors will be reported for each track segment. When disabled, only the first error will be reported. Enabling this option will result in the design rule checker running more slowly.

Test for parity between PCB and schematic: when enabled, the design rule checker will test for differences between the schematic and PCB in addition to testing the PCB design rules. This option has no effect when running the PCB editor in standalone mode.

After running DRC, any violations will be shown in the center part of the DRC Control window. Rule violations, unconnected items, and differences between the schematic and the PCB are shown in three different tabs. The controls below the list of violations can be used to show or hide violations depending on their severity. A report file in plain text format can be created after running DRC using the Save button.
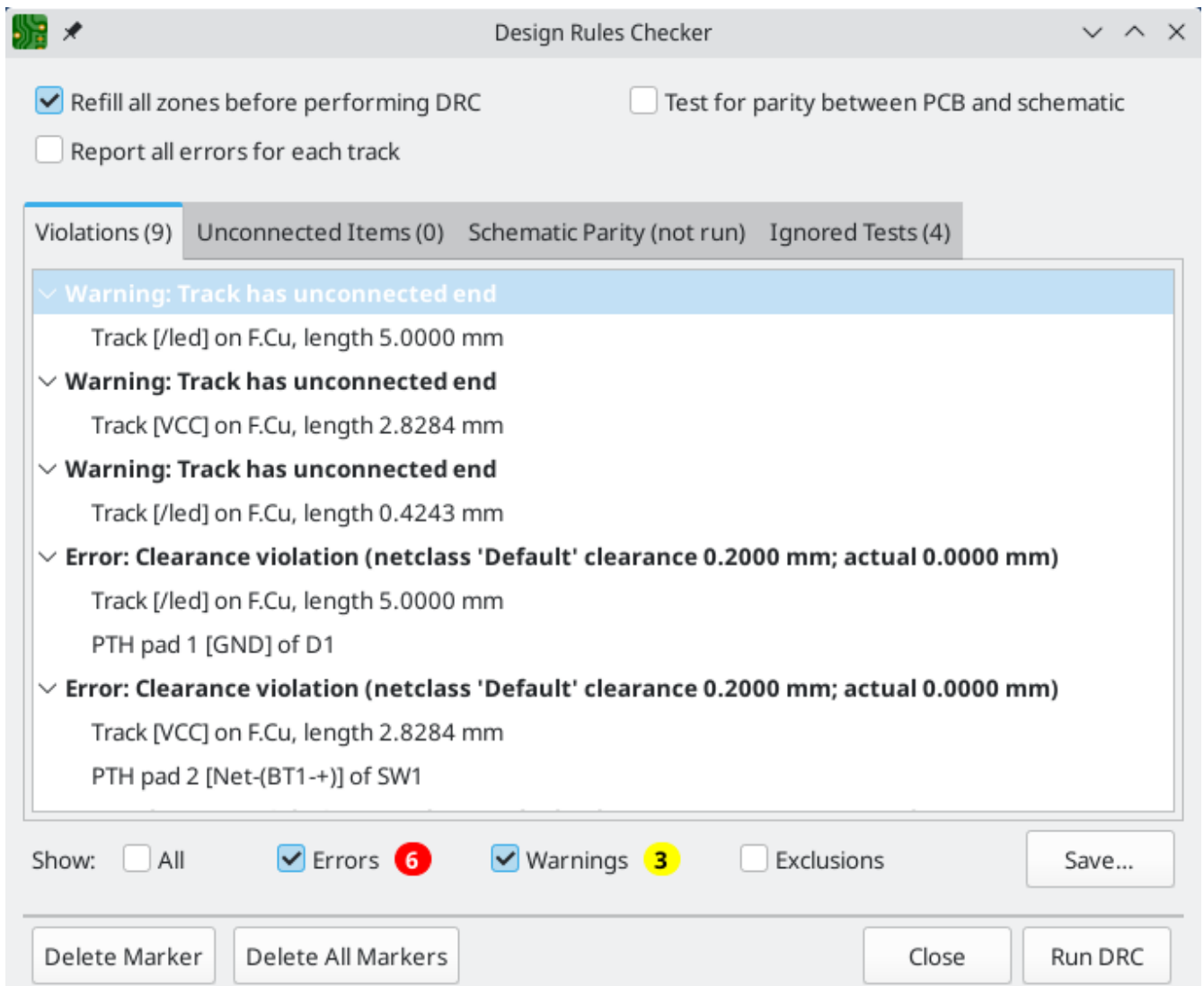
Each violation involves one or more objects on the PCB. In the list of violations, the objects involved are listed below the violation. Clicking on the violation in the list view will move the PCB Editor view so that the affected area is centered. Clicking on one of the objects involved in a violation will highlight the object.

The numbers at the bottom of the window show the number of errors, warnings, and exclusions. Each type of violation can be filtered from the list using the respective checkboxes. Clicking **Delete Markers** will clear all violations until DRC is run again.

Violations can be right-clicked in the dialog to ignore them or change their severity:

- **Exclude this violation:** ignores this particular violation, but does not affect any other violations.

- **Change severity:** changes a type of violation from warning to error, or error to warning. This affects all violations of a given type.

- **Ignore all:** ignores all violations of a given type. This test will now appear in the **Ignored Tests** tab rather than the **Violations** tab.

Excluded and ignored violations are remembered between runs of the design rule checker.

Clearance and constraint resolution

The clearance and constraint resolution tools allow you to inspect which clearance and design constraint rules apply to selected items. These tools can help when designing PCBs with complex design rules where it

is not always clear which rules apply to an object.

To inspect the clearance rules that apply between two objects, select both objects and choose **Clearance Resolution** from the **Inspect** menu. The Clearance Report dialog will show the clearance required between the objects on each copper layer, as well as the design rules that resulted in that clearance.
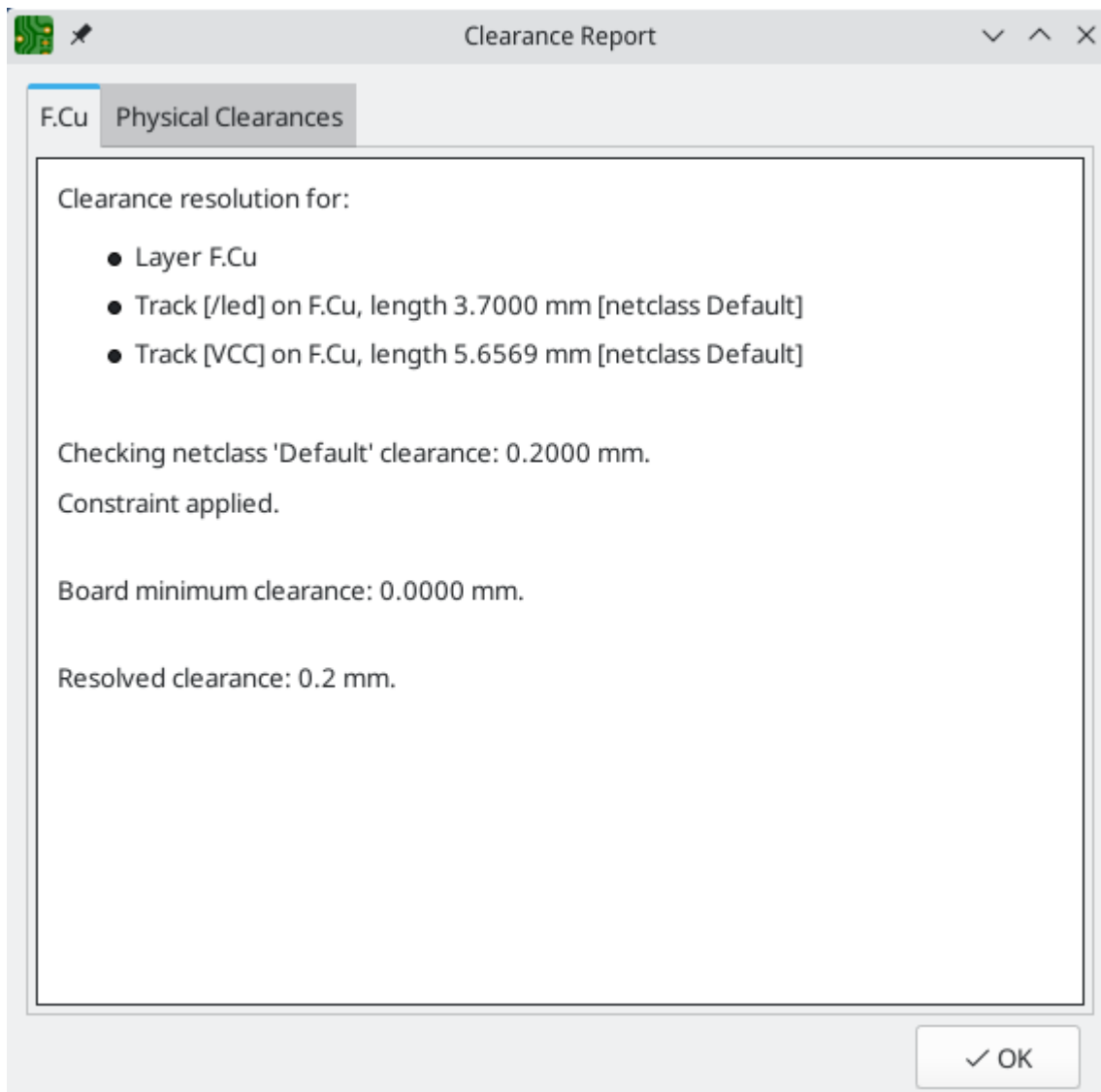
To inspect the design constraints that apply to an object, select it and choose **Constraints Resolution** from the **Inspect** menu. The Constraints Report dialog will show any constraints that apply to the object.
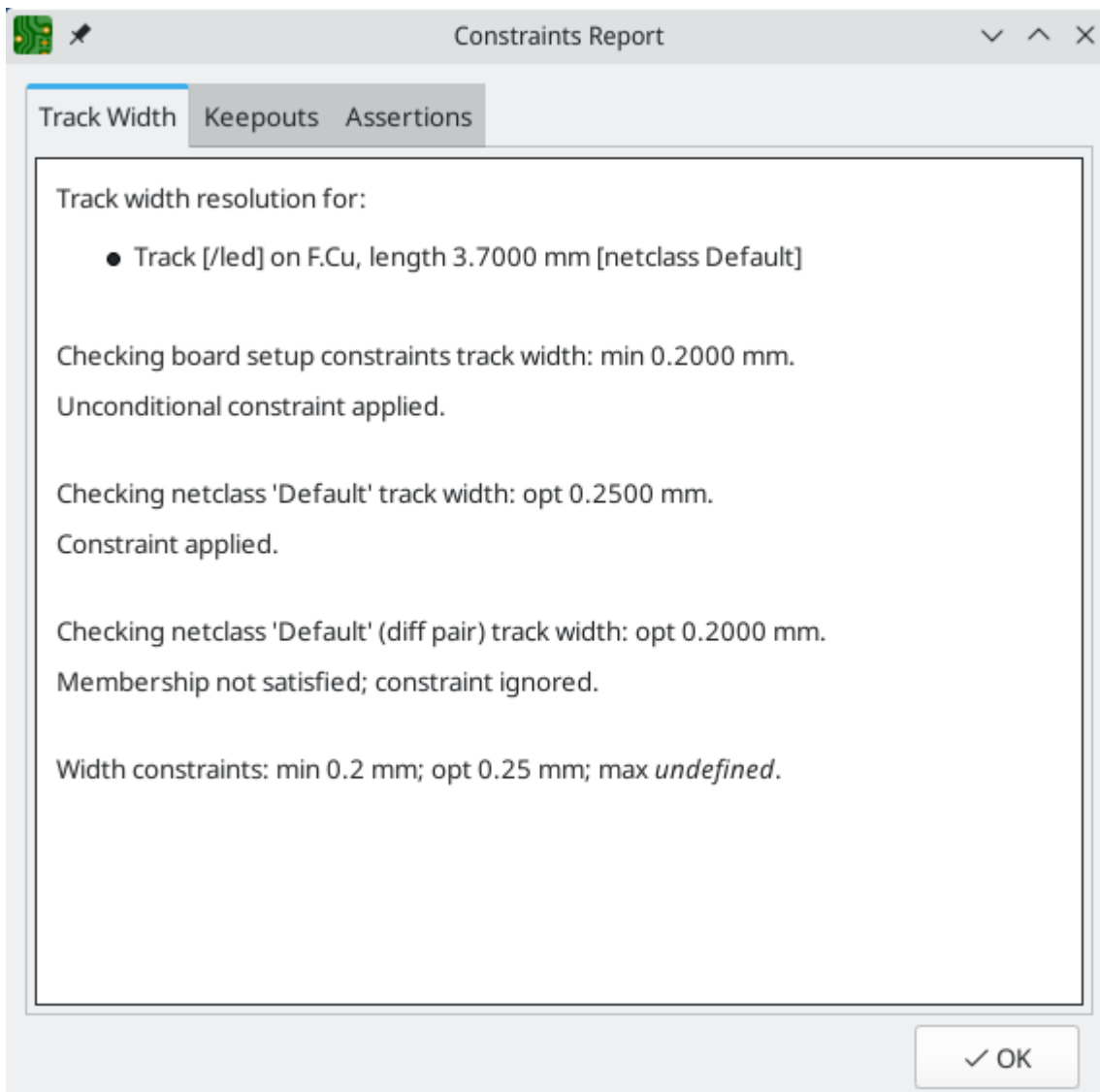
Board Statistics

The Board Statistics dialog shows a summary of the board's contents, including the number of components, pads and vias; each by their own types as well as the overall board size.

Find tool

The Find tool searches for text in the PCB, including reference designators, footprint fields, and graphic text. When the tool finds a match, the canvas is zoomed and centered on the match and the text is highlighted. Launch the tool using the (🔍Ⓐ) button in the top toolbar.

The Find tool has several options:

Match case: Selects whether the search is case-sensitive.

Words: When selected, the search will only match the search term with complete words in the PCB. When unselected, the search will match if the search term is part of a larger word in the PCB.

Wildcards: When selected, wildcards can be used in the search terms. `?` matches any single character, and `*` matches any number of characters. Note that when this option is selected, partial matches are not returned: searching for `abc*` will match the string `abcd`, but searching for `abc` will not.

Wrap: When selected, search results will return to the first hit after reaching the last hit.

Search footprint reference designators: Selects whether the search should apply to footprint reference designators.

**Search footprint values:** Selects whether the search should apply to footprint value fields.

Search other text items: Selects whether the search should apply to other text items, including graphical text and footprint fields other than value and reference.

Search DRC markers: Selects whether the search should apply to the violation descriptions of DRC markers shown on the board.

Search net names: Selects whether the search should apply to the names of nets in the board.

Search panel

The search panel is a docked panel that lists information about footprints, zones, nets, ratsnest lines (unrouted segments), and text from the PCB. You can optionally filter the list based on a search string. When no filter is used, all items in the design are listed in the corresponding tab.
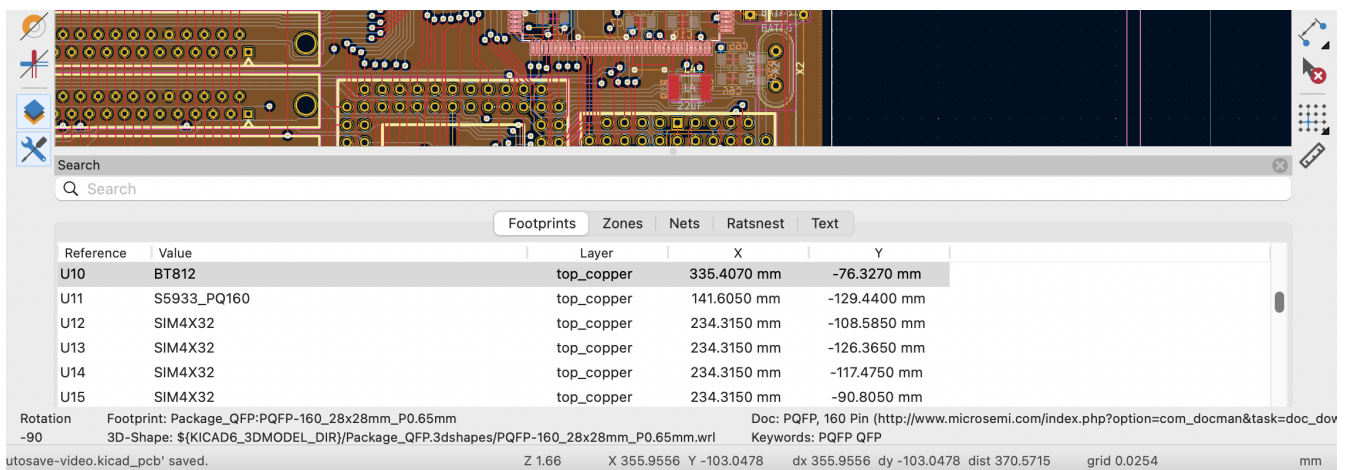
Items are filtered based on their properties: footprints are filtered by their reference designator and value, zones by the zone name, net and ratsnest items by the net name, and text (text, textboxes, and dimensions) by the text content. You can sort the filtered results in ascending or descending order of the value in a particular column by clicking on that column header.

Filters support wildcards: `*` matches any characters, and `?` matches any single character. You can also use regular expressions, such as `/footprint value/`.

The displayed information depends on the item type. In addition to the item's name and/or value, physical items (footprints, zones, and text) list their layer and X/Y location. Text also displays the type of text object

(text, textbox, or dimension.) Net and ratsnest items list their net name and net class.

When you click an item in the search panel, the item is selected in the editing canvas. Double-clicking an item in the search panel opens its properties dialog (for net and ratsnest items, the net classes dialog is opened instead).

Show or hide the search panel with **View** → **Show Search Panel** or use the `Ctrl` + `G` shortcut.

Visor 3D

The 3D Viewer shows a 3-dimensional view of the board and the components on the board. You can view the board from different perspectives, show or hide different types of components, cross-probe from the PCB Editor to the 3D viewer, and generate raytraced renders of the board. Show the 3D Viewer with **View** → **3D Viewer** or use the `Alt` + `3` shortcut.
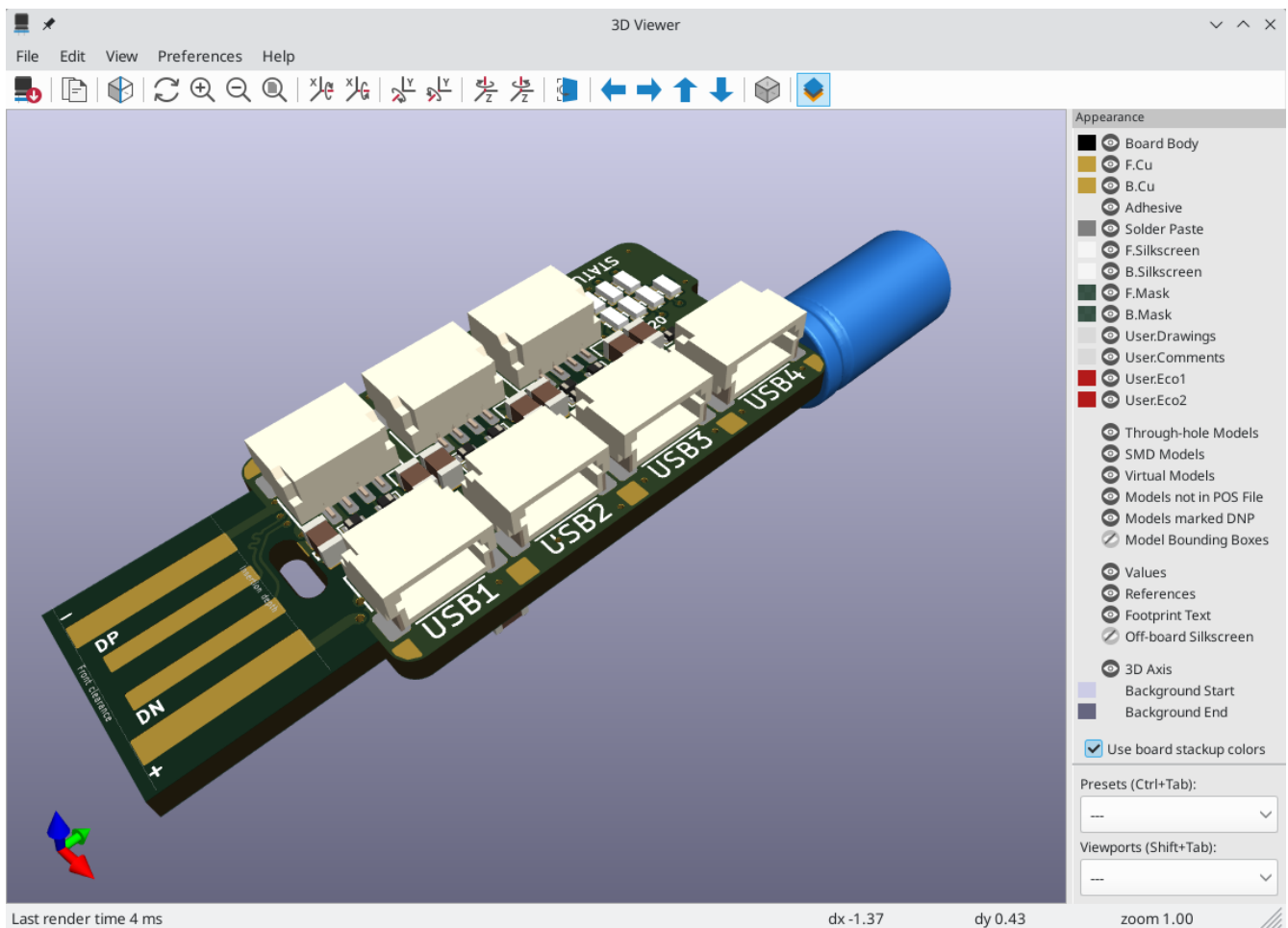

NOTE	The 3D model for a component will only appear if the 3D model file exists and has been assigned to the footprint.

NOTE	Many footprints in KiCad's standard library do not yet have model files created for them. However, these footprints may contain a path to a 3D model that does not yet exist, in anticipation of the 3D model being created in the future.

Navigating the 3D view

Dragging with the left mouse button will orbit the 3D view. By default this is the centroid of the board, but the pivot point can be reset to a new point on the board by moving the cursor over the desired point and

pressing `Space` . Scrolling the mouse wheel will zoom the view in or out. Scrolling while holding `Ctrl` pans the view left and right, and scrolling while holding `Shift` pans up and down. Dragging with the middle mouse button also pans the view.

Different sized 3D grids can be set using the **View → 3D Grid** menu. Bounding boxes for each component can be enabled with **Preferences → Show Model Bounding Boxes**.

When the PCB Editor and the 3D Viewer are both open, selecting a footprint in the PCB Editor will also highlight the component in the 3D Viewer. The highlight color is adjustable in **Preferences → Preferences… → 3D Viewer → Realtime Renderer → Selection Color**.

Appearance manager

The appearance manager is a panel at the right of the viewer which provides controls to manage the visibility, color, and opacity of different types of objects and board layers in the 3D view.

Each layer or type of object in the list can be individually shown or hidden by clicking its corresponding visibility icon. PCB layers can have their colors customized; double-click on the color swatch next to the item type to edit the item's color and opacity. To use the colors selected in the Board Setup dialog's Physical Stackup editor, enable the **use board stackup colors** option.

You can save an appearance configuration as a preset, or load a configuration from a preset, using the **Preset** selector at the bottom. The `Ctrl`+`Tab` hotkey cycles through presets; press `Tab` repeatedly while holding `Ctrl` to cycle through multiple presets. Several built-in presets are available: "Follow PCB Editor" matches the visibility settings in the PCB editor, "Follow PCB Plot Settings" matches the visibility settings selected in the Plot dialog, and "legacy colors" matches the default 3D Viewer color settings from older versions of KiCad.

Finally, you can save a viewport for later retrieval using the **Viewports** selector at the bottom. You can quickly cycle between saved viewports using `Shift`+`Tab` ; pressing `Tab` repeatedly while holding `Shift` will cycle through multiple viewports.

Generating images with the 3D Viewer

The current 3D view can be saved to an image with **File → Export Current View as PNG…** or **Export Current View as JPG…**, depending on the desired image format. The current view can also be copied to the clipboard using the button, or **Edit → Copy 3D Image**.

The 3D Viewer has a raytracing rendering mode which displays the board using a more physically accurate rendering model than the default rendering mode. Raytracing is slower than the default rendering mode, but it can be used when the most visually attractive results are desired. Raytracing mode is enabled with the button, or with **Preferences → Raytracing**. The 3D grid and selection highlights are not shown in raytracing mode.

Colors and other rendering options, for both raytraced and non-raytraced modes, can be adjusted in **Preferences → Preferences… → 3D Viewer**.
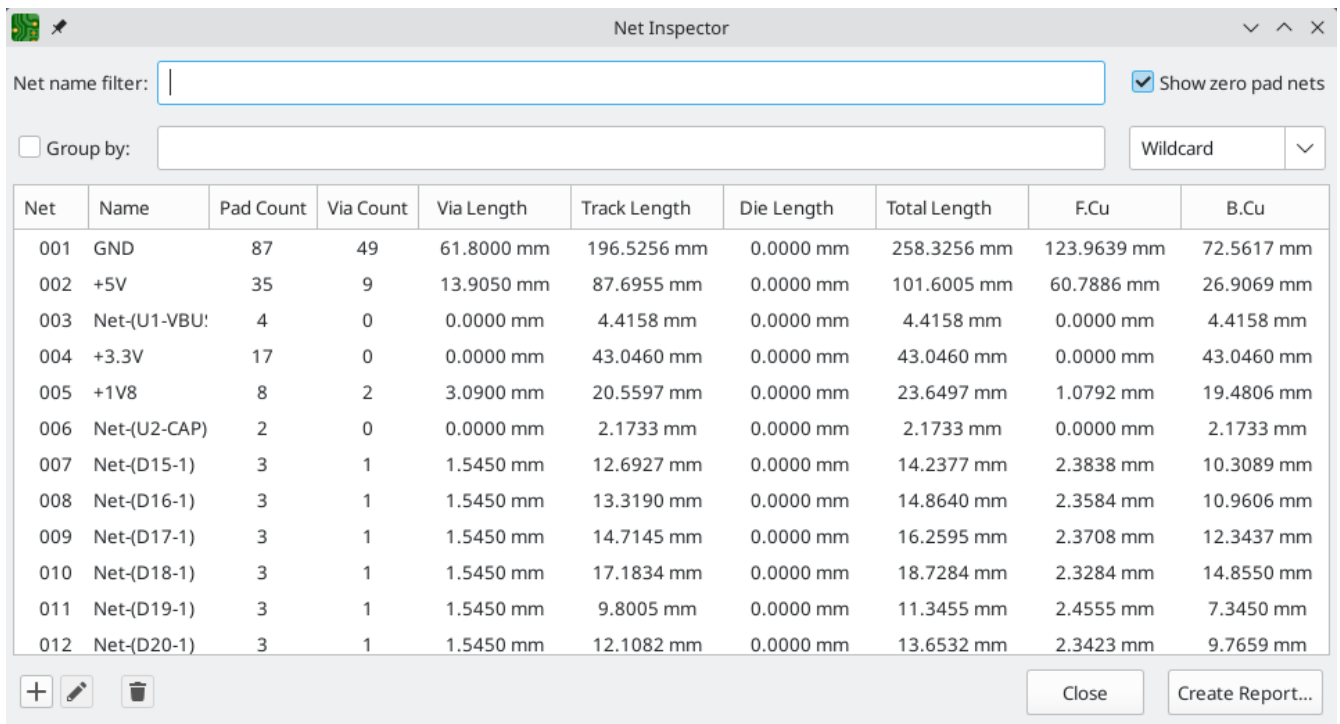
3D viewer controls

Many viewing options are controlled with the top toolbar.

	Reload the 3D model
	Copy 3D image to clipboard
	Render current view using raytracing
	Redraw
	Zoom in
	Zoom out
	Fit drawing in display area
	Rotate X clockwise
	Rotate X counterclockwise
	Rotate Y clockwise
	Rotate Y counterclockwise
	Rotate Z clockwise
	Rotate Z counterclockwise
	Flip board view
	Pan board left
	Pan board right
	Pan board up
	Pan board down
	Enable/disable orthographic projection
	Show/hide the appearance manager

Net inspector

The Net Inspector allows you to view statistics about all the nets in a board. To open the inspector, click the
icon at the top of the Nets section of the Appearance panel, or select **Net Inspector** from the **Inspect**
menu.

Clicking a net in the list will highlight that net on the board. Clicking a column title allows you to sort the list of nets by that column.

The **Group By** field allows you to combine different nets together and view the total length of the combined nets. For example, if you have two nets named `DATA0` and `DATA0_EXT`, using a Group By value of `DATA0*` will create a group containing both nets. More complicated groupings can be created by changing the Group By mode from Wildcard to RegEx (regular expressions). The substring (Substr) variants of the Group By mode will create groups for each set of nets that matches the pattern differently.

For example, if you have the nets `U1D+`, `U1D-`, `U2D+`, and `U2D-`, the grouping pattern `U*D` will match all four nets in Wildcard mode, creating a single group `U*D`. In Wildcard Substr mode, it will match all four nets, but create two different groups: `U1D` and `U2D`.

Pad Count and **Via Count** show the number of pads (surface mount and through hole) and vias on a net. **Via Length** shows the total height of each via (not accounting for which copper layers the via connects to). In other words, Via Length is equal to Via Count multiplied by the stackup height of the board. **Track Length** shows the total length of all track segments in a net, not accounting for topology. **Die length** shows the total of all Pad to Die Length values set for pads on the net.

Differences between Net Inspector and Length Tuner

The Net Inspector may report different net lengths than the length tuner, because the two tools have different purposes and calculate track/net lengths differently. In short, the Net Inspector sums up the total length of each track segment and via on a net, while the length tuner calculates the effective electrical length of a path between two points on a net. The specific differences are as follows:

- The Net Inspector reports track length as a simple sum of the length of each track segment on a net. The length tuner calculates an effective electrical length of a net, which includes optimizing paths through pads to calculate the shortest possible path.

- If a routed net has a branching topology, the Net Inspector total includes the length of each branch in the total. The length tuner calculates a point-to-point length; if there are any branches, the length tuner will

stop at the closest branch and report the length up to the branch.

- The Net Inspector always includes the effective via height in its via length and total length calculations. If a via connects to traces on both the top and bottom layers, the full via height is included in the length calculation. Otherwise, only the stackup height between the connected layers is included. The length tuner calculates effective via height in the same way as the Net Inspector, but via height is only included in the length calculation when the **use stackup height** setting is enabled board constraint settings. If the setting is disabled, the length tuner will not include vias in its calculations at all.

Generating outputs

KiCad can generate and export files in a number of different formats useful for manufacturing PCBs and interfacing with external software. This functionality is available in the File menu in a few different sections. The Fabrication Outputs section contains the most common operations needed to prepare a PCB for fabrication. The Export section contains tools for generating files that can be read by external software. The Plot function allows you to export 2D line drawings of the PCB in various formats. The Print function allows you to send a view of the PCB to a 2D printer.

Fabrication outputs and plotting

KiCad uses Gerber files as its primary plotting format for PCB manufacturing. To create Gerber files, open the **Plot…** dialog from the **File** menu, or select **Gerbers (.gbr)…** from the **Fabrication Outputs** section of the **File** menu. The Plot dialog will open, allowing you to configure and generate Gerber files.

Plotting options

Include Layers: Check that every layer used on your board is enabled in the list. Disabled layers will not be plotted.

Plot on All Layers: Selected layers will be included in the plot for each layer selected in the **include layers** list. The additional layers are plotted on top of the base layer. You can reorder these layers using the arrow

buttons at the bottom; items that are lower in the list are plotted after (on top of) items that are higher in the list.

Output directory: Specify the location to save plotted files. If this is a relative path, it is created relative to the project directory.

Plot drawing sheet: If enabled, the drawing sheet border and title block will be plotted on each layer. This should usually be disabled when plotting Gerber files.

Plot footprint values: If enabled, the Value field of each footprint will be plotted on whatever layer it exists on (unless the field visibility is disabled for a specific footprint).

Plot reference designators: If enabled, the Reference Designator field of each footprint will be plotted on whatever layer it exists on (unless the field visibility is disabled for a specific footprint).

Plot footprint text: If enabled, text fields in footprints will be plotted on whatever layers they exist on (unless the field visibility is disabled for a specific footprint). Disabling this option also disables the **plot footprint values** and **plot reference designators** options.

Force plotting of invisible values / refs: If enabled, all footprint values and reference designators will be plotted, even if the field visibility is disabled for some of these fields.

Plot Edge.Cuts on all layers: If enabled, the Edge.Cuts (board outline) layer will be added to all other layers. Check with your manufacturer to see what the correct value of this setting is for their manufacturing process.

Sketch pads on fabrication layers: If enabled, footprint pads on fabrication (F.Fab, B.Fab) layers will be drawn as unfilled outlines rather than filled shapes.

Check zone fills before plotting: When enabled, zone fills will be checked (and refilled if outdated) before generating outputs. Plot outputs may be incorrect if this option is disabled!

Drill marks: For plot formats other than Gerber, marks may be plotted at the location of all drilled holes. Drill marks may be created at the actual size (diameter) of the finished hole, or at a smaller size.

Scaling: For plot formats that support scaling other than 1:1, the plot scale may be set. The Auto scaling setting will scale the plot to fit the specified page size.

Plot mode: For some plot formats, filled shapes may be plotted as outlines only (sketch mode).

Use drill/place file origin: When enabled, the coordinate origin for plotted files will be the drill/place file origin set in the board editor. When disabled, the coordinate origin will be the absolute origin (top left corner of the worksheet).

Mirrored plot: For some plot formats, the output may be mirrored horizontally when this option is set.

Negative plot: For some plot formats, the output may be set to negative mode. In this mode, shapes will be drawn for the empty space inside the board outline, and empty space will be left where objects are present in the PCB.

Do not tent vias: If enabled, vias will be left uncovered on the solder mask layers (F.Mask, B.Mask). If disabled, vias will be covered by solder mask (tented).

NOTE	KiCad does not support tenting or uncovering specific vias. Tenting may only be controlled globally (for all vias on a board).

Gerber options

Use Protel filename extensions: When enabled, the plotted Gerber files will be named with file extensions based on Protel (`.GBL` , `.GTL` , etc). When disabled, the files will have the `.gbr` extension.

Generate Gerber job file: When enabled, a Gerber job file (`.gbrjob`) will be generated along with any Gerber files. The Gerber job file is an extension to the Gerber format that includes information about the PCB stackup, materials, and finish. More information about Gerber job files is available at the Ucamco website.

Subtract soldermask from silkscreen: When enabled, silkscreen will be automatically removed from board areas that aren't covered by soldermask.

Coordinate format: Configure how coordinates will be stored in the plotted Gerber files. Check with your manufacturer for their recommended setting for this option.

Use extended X2 format: When enabled, the plotted Gerber files will use the X2 format, which includes information about the netlist and other extended attributes. This format may not be compatible with older CAM software used by some manufacturers.

Include netlist attributes: When enabled, the plotted Gerber files will include netlist information that can be used for checking the design in CAM software. When X2 format mode is disabled, this information is included as comments in the Gerber files.

Disable aperture macros: When enabled, all shapes will be plotted as primitives rather than by using aperture macros. This setting should only be used for compatibility with old or buggy CAM software when requested by your manufacturer.

PostScript options

Scale factor: Controls how coordinates in the board file will be scaled to coordinates in the PostScript file. Using a different value for X and Y scale factors will result in a stretched / distorted output. These factors may be used to correct for scaling in the PostScript output device to achieve an exact-scale output.

Track width correction: A global factor that is added (or subtracted, if negative) from the size of tracks, vias, and pads when plotting a PostScript file. This factor may be used to correct for errors in the PostScript output device to achieve an exact-scale output.

Force A4 output: When enabled, the generated PostScript file will be A4 size even if the KiCad board file is a different size.

SVG options

Precision: Controls how many significant digits will be used to store coordinates.

Output mode: Controls whether the generated SVG file is in color or black and white.

DXF options

Plot graphic items using their contours: Graphic shapes in DXF files have no width. This option controls how graphic shapes with a width (thickness) in a KiCad board are plotted to a DXF file. When this option is enabled, the outer contour of the shape will be plotted. When this option is disabled, the centerline of the shape will be plotted (and the shape's thickness will not be visible in the resulting DXF file).

Use KiCad font to plot text: When enabled, text in the KiCad design will be plotted as graphic shapes using the KiCad font. When disabled, text will be plotted as DXF text objects, which will use a different font and will not appear in exactly the same position and size as shown in the KiCad board editor.

Export units: Controls the units that will be used in the DXF file. Since the DXF format has no specified units system, you must export using the same units setting that you want to use for importing into other software.

HPGL options

Default pen size: Controls the plotter pen size used to create graphics.

PDF options

Output mode: Controls whether the generated PDF file is in color or black and white.

Generate property popups for front footprints: When enabled, interactive popups will be added to the generated PDF containing part information for each footprint on the front of the board.

Generate property popups for back footprints: When enabled, interactive popups will be added to the generated PDF containing part information for each footprint on the back of the board.

Drill files

KiCad can generate CNC drilling files required by most PCB manufacturing processes in either Excellon or Gerber X2 format. KiCad can also generate a drill map: a graphical plot of the board showing drill locations. Select the **Drill Files (.drl)…** option from the **Fabrication Outputs** section of the **File** menu to open the dialog:

Output folder: Choose the folder to save generated drill and map files to. If a relative path is entered, it will be relative to the project directory.

Drill file format: Choose whether to generate Excellon drill files (required by most PCB manufacturers) or Gerber X2 files.

Mirror Y axis: For Excellon files, choose whether or not to mirror the Y-axis coordinate. This option should in general not be used when having PCBs manufactured by a third party, and is provided for convenience for users who are making PCBs themselves.

Minimal header: For Excellon files, choose whether to output a minimal header rather than a full file header. This option should not be enabled unless requested by your manufacturer.

PTH and NPTH in single file: By default, plated holes and non-plated holes will be generated in two different Excellon files. With this option enabled, both will be merged into a single file. This option should not be enabled unless requested by your manufacturer.

Oval holes drill mode: Controls how oval holes are represented in an Excellon drill file. The default setting, **Use route command**, is correct for most manufacturers. Only choose the **Use alternate drill mode** setting if

requested by your manufacturer.

Map file format: Choose the output format for plotting a drill map.
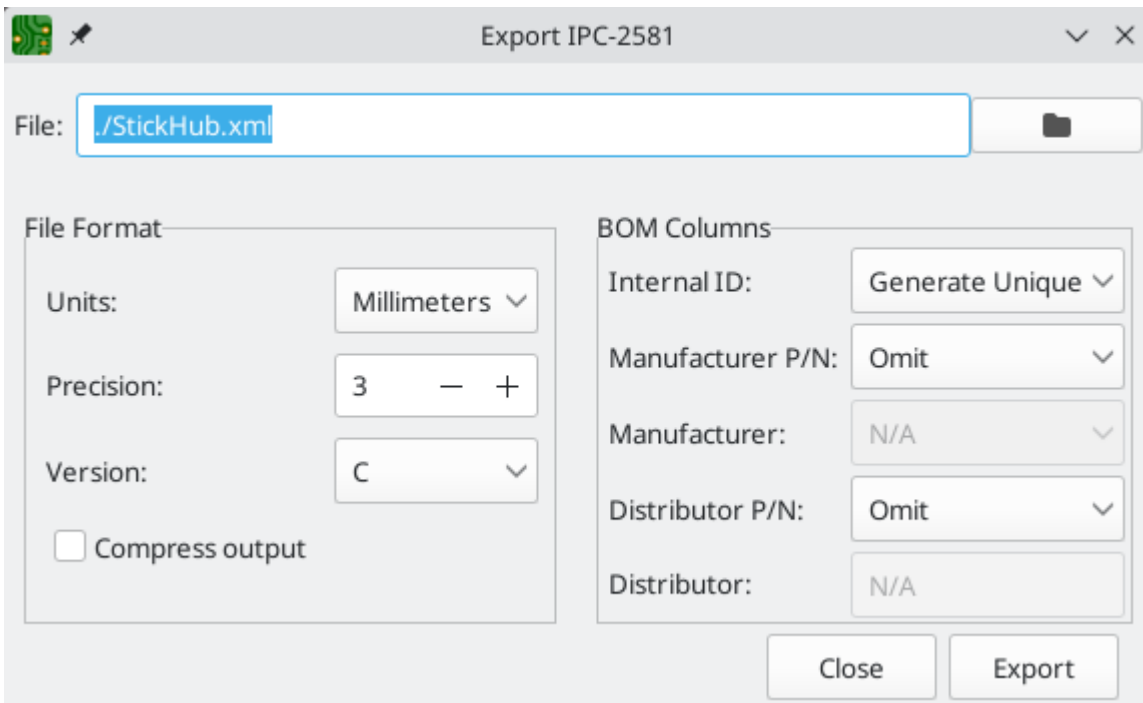
Drill origin: Choose the coordinate origin for drill files. **Absolute** will use the page origin at the top left corner. **Drill/place file origin** will use the origin specified in the board design.

Drill units: Choose the units for drill coordinates and sizes.

Zeros format: Controls how numbers are formatted in an Excellon drill file. Select an option here based on your manufacturer's recommendations.

IPC-2581 files

IPC-2581 files are XML files that contain complete fabrication and assembly data for a board design. If your manufacturer accepts IPC-2581 files, these can replace Gerber files, drill files, and component placement files. To create an IPC-2581 file, select **IPC-2581 File (.xml)…** from the **Fabrication Outputs** section of the **File** menu.

Units: Choose the units for the generated file.

Precision: Choose the number of digits after the decimal point for numbers in the generated file.

Version: Choose the IPC-2581 standard version (B or C).

Compress output: If enabled, the generated file will be compressed as a ZIP file.

Internal ID: Choose the footprint field to use for the BOM's internal ID column. This can be a generated unique ID or set to any footprint field in the design.

Manufacturer PN: Choose the footprint field to use for the BOM's manufacturer part number column. This can be omitted or set to any footprint field in the design.

Manufacturer: Choose the footprint field to use for the BOM's manufacturer column. This can be omitted or set to any footprint field in the design.

Distributor PN: Choose the footprint field to use for the BOM's distributor part number column. This can be omitted or set to any footprint field in the design.

Distributor: Choose the footprint field to use for the BOM's distributor column. This can be omitted or set to any footprint field in the design.

Component placement files

Component placement files are text files that list each component (footprint) on the board along with its center position and orientation. These files are usually used for programming pick-and-place machines, and may be required by your manufacturer if you are ordering fully-assembled PCBs. To create placement files, select **Component Placement (.pos, .gbr)...** from the **Fabrication Outputs** section of the **File** menu.

NOTE	A footprint will not appear in generated placement files if the "Exclude from position files" option is enabled for that footprint. This may be used for excluding certain footprints that do not represent physical components to be assembled. You can also optionally exclude DNP components, depending on your manufacturer's requirements.

Format: Choose between generating a plain text (ASCII), comma-separated text (CSV), or Gerber X3 placement file format.

Units: Choose the units for component locations in the placement file.

Files: Choose whether to generate separate files for footprints on the front and back of the board or a single file combining both sides.

Include only SMD footprints: When enabled, only footprints with the SMD fabrication attribute will be included. Check with your manufacturer to determine if non-SMD footprints should be included or excluded from the position file.

Exclude all footprints with through hole pads: When enabled, footprints will be excluded from the placement file if they contain any through-hole pads, even if their fabrication type is set to SMD.

Exclude all footprints with the Do Not Populate flag set: When enabled, footprints will be excluded from the placement file if they have the Do Not Populate attribute set. Check with your manufacturer to determine if DNP components should be included or excluded from the position file.

Include board edge layer: For Gerber placement files, controls whether or not the board outline is included with the footprint placement data.

Use drill/place file origin: When enabled, component positions will be relative to the drill/place file origin set in the board design. When disabled, the positions will be relative to the page origin (upper left corner).

Use negative X coordinates for footprints on bottom layer: When enabled, the X coordinates will be flipped (negated) for footprints on the bottom layer.

Additional fabrication outputs

KiCad can also generate footprint report files, IPC-D-356 netlist files, and a bill of materials (BOM) from the board design. These output formats have no configurable options.

Printing

KiCad can print the board view to a standard printer using the Print action in the File menu.
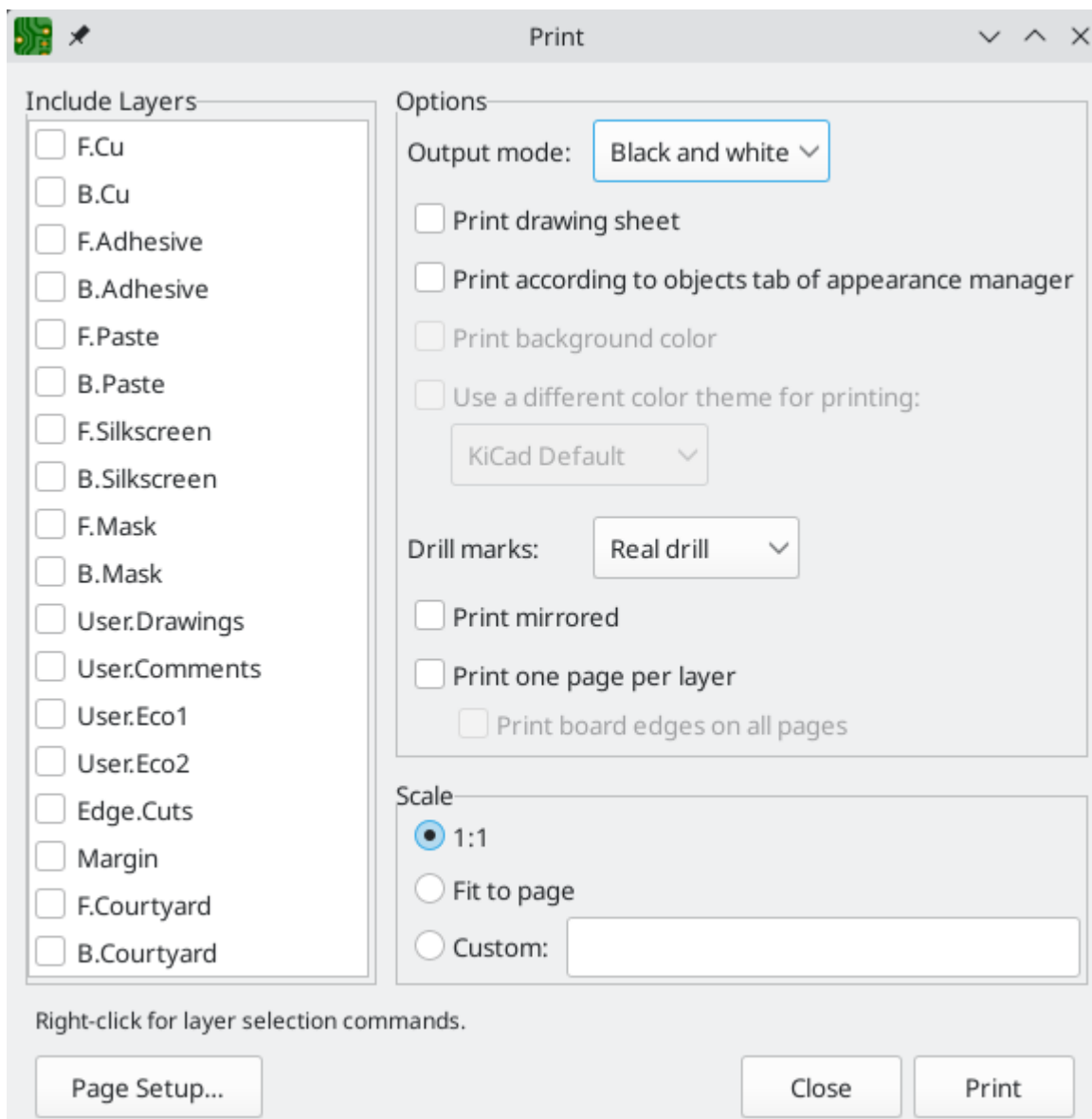
Include layers: Select the layers to include in the printout. Unselected layers will be invisible. Right-click the list for layer selection commands.

Output mode: Choose whether to print in black and white or full color.

Print drawing sheet: When enabled, the page border and title block will be printed.

Print according to objects tab of appearance manager: When enabled, any objects that have been hidden in the Objects tab of the Appearance panel will be hidden in the printout. When disabled, these objects will be printed if the layer they appear on is selected in the Included Layers area.

Print background color: When printing in full color, this option controls whether or not the view background color will be printed.

Use a different color theme for printing: When printing in full color, this option allows a different color theme to be used for printing. When disabled, the color theme used by the board editor will be used for printing.

Drill marks: Controls whether to show drilled holes at their actual size, at a small size, or hide them from the printout.

Print mirrored: When enabled, the printout will be mirrored horizontally.

Print one page per layer: When enabled, each layer selected in the Included Layers area will be printed to an individual page. If this option is enabled, the **Print board edges on all pages** option controls whether to add the Edge.Cuts layer to each printed page.

Scale: controls the scale of the printout relative to the page size configured in Page Setup.

Exporting files

KiCad can export a board design to various third-party formats for use with external software. These functions are found in the **Export** section of the **File** menu.

Specctra DSN exporter

The Specctra DSN exporter creates a file suitable for importing into certain third-party autorouter software. This exporter has no configurable options.

GenCAD exporter

The GenCAD exporter creates a GenCAD file for fabrication, testing, or importing into other software.
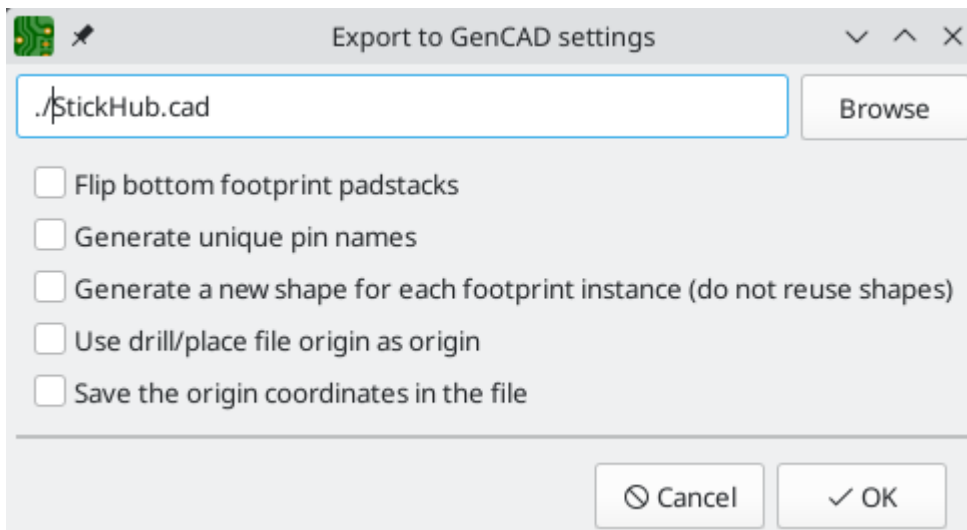
The GenCAD exporter has several options.

Flip bottom footprint padstacks: If enabled, separate flipped padstack definitions will be added for bottom-side footprints. This may be necessary for importing into some third-party software.

Generate unique pin names: If enabled, a suffix will be added to each pin name so that no footprint in the generated file will have two pins with the same name.
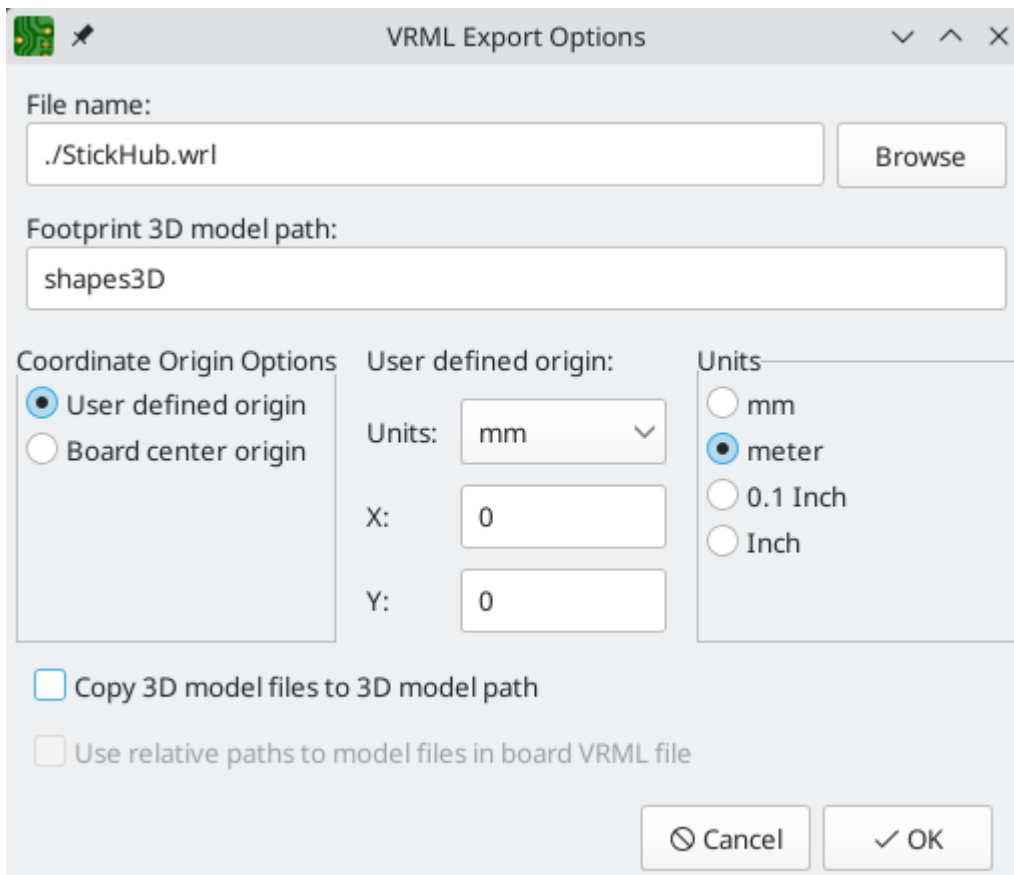
Generate a new shape for each footprint instance: If enabled, a unique footprint will be output for every footprint instance, even if two footprints are identical.

Use drill/place file origin as origin: If enabled, coordinates in the generated file will be relative to the drill/place file origin.

Save the origin coordinates in the file: If enabled, the selected origin coordinates will be included in the generated file. If not enabled, the origin in the generated file will be set to (0,0).

VRML exporter

The VRML exporter creates a VRML (`.wrl`) 3D model file containing the PCB and any VRML files specified in footprints. VRML models are suitable for use in applications where visual appearance is important and dimensional accuracy is not critical.

The VRML exporter has several options.

Coordinate origin options: Selects the origin for the generated model. If **user defined origin** is selected, you can manually specify the origin point.

VRML units for output files: Selects the unit system for the generated model. Dimensions in the generated model will be scaled appropriately.

Copy 3D model files to 3D model path: If enabled, VRML files referenced in footprints will be copied into a subdirectory of the directory containing the generated board VRML model, and the generated model will reference the copied files. The subdirectory name is set by the **footprint 3D model path** field. If disabled, VRML files referenced in footprints will be embedded in the generated VRML files.

Use relative paths to model files in board VRML file: If enabled, references to external models will use paths relative to the generated board VRML file. If disabled, the references will use absolute paths. This option is only available when the **copy 3D model files to 3D model path** option is enabled.

IDF exporter

The IDF exporter exports an [IDFv3](#) compliant board (`.emn`) and library (`.emp`) file for communicating mechanical dimensions to a mechanical CAD package. The exporter exports the board outline and cutouts, all pad and mounting through holes including slotted holes, and component outlines; this is the most basic

set of mechanical data required for interaction with mechanical designers. All other entities described in the IDFv3 specification are currently not exported.

NOTE	You must attach IDF component models to your design's footprints before they will be included in the exported model. For more information on attaching models to footprints, see the footprint documentation. Some IDF-specific guidance is included in the Advanced Topics documentation.

NOTE	For more information on creating IDF component models, including descriptions of the IDF utility tools included with KiCad, see the Advanced Topics documentation.

Once models have been specified for all desired components, the model of the board can be exported. In the PCB Editor, select **File → Export → IDFv3…**.
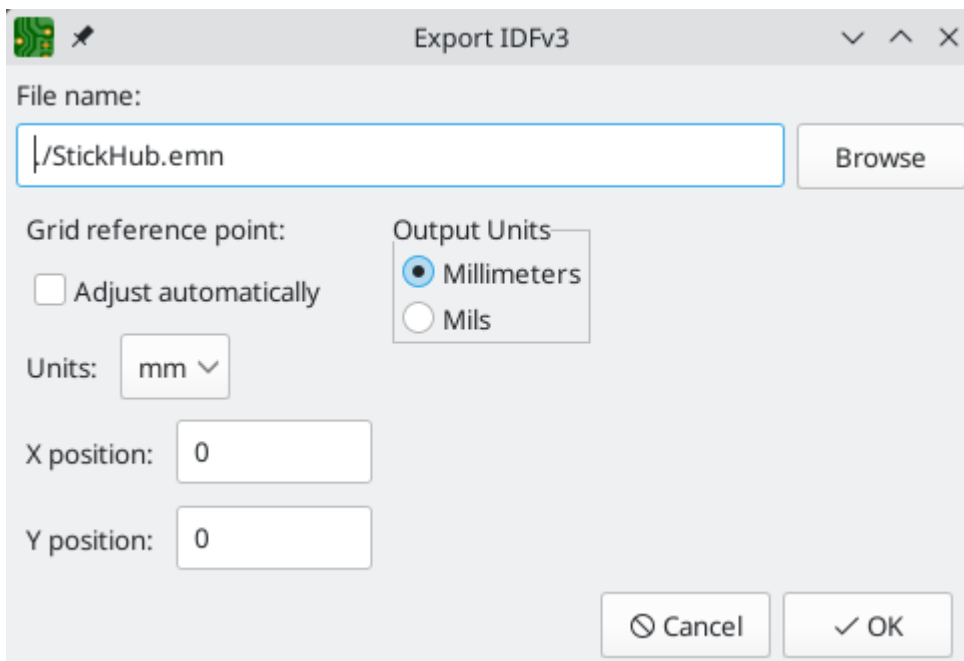
Grid reference point: Choose where the exported model's reference point should be. If the **Adjust automatically** option is selected, KiCad will set the reference point to the centroid of the PCB. Otherwise, the reference point is set relative to the display origin.

Output units: Choose whether the exported model's units are millimeters or mils.

The outputs can be viewed directly in a mechanical CAD application or converted to VRML using the `idf2vrml` tool.

STEP exporter

The STEP exporter creates a STEP (`.step`) 3D model file containing the PCB and any STEP files specified in footprints. STEP models are suitable for use in mechanical CAD applications.

The STEP exporter can also export a binary GLTF (`.glb`) model file by changing the output file's format to Binary GLTF.

NOTE	KiCad's footprint library includes both STEP and VRML (`.wrl`) versions of each model. However, footprints in KiCad's library only reference the VRML versions of the models. VRML models are not included in STEP exports, but the STEP exporter will instead include the corresponding STEP version of the model if the **subsitute similarly named models** option is enabled.
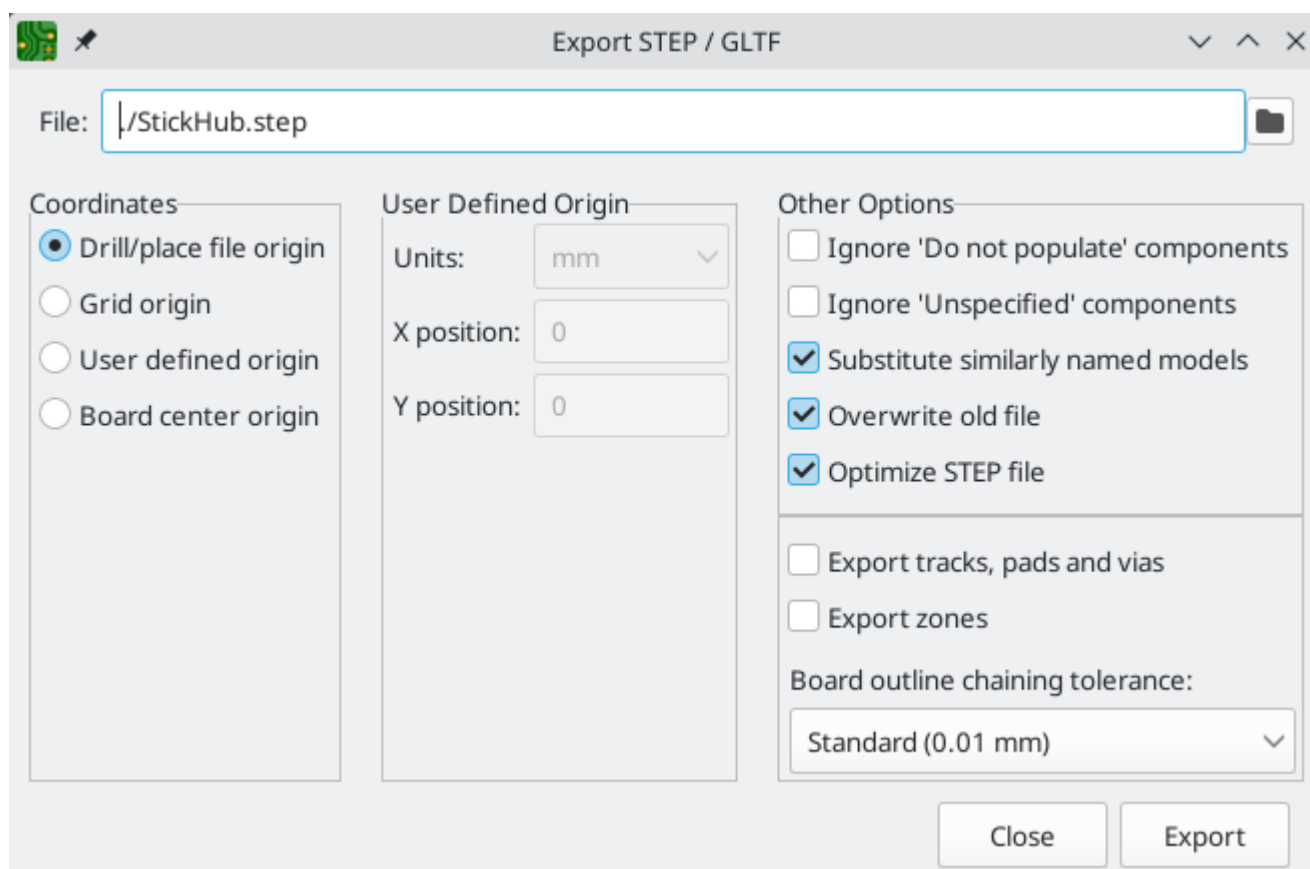
Coordinates: Selects the origin for the generated model. If **user defined origin** is selected, you can manually specify the origin point.

Ignore 'Do not populate' components: If enabled, components with the DNP attribute set will not be included in the exported STEP model.

Ignore 'Unspecified' components: If enabled, components with the Unspecified footprint type will not be included in the exported STEP model.

Substitute similarly named models: VRML models cannot be used for STEP exports, but if this option is enabled the exporter will look for an identically named STEP model to include in the export instead of a footprint's specified VRML model. Note that footprints in KiCad's footprint library specify VRML models, but suitably named STEP models are included for each VRML model. Therefore this option must be enabled in order to export STEP models for footprints from KiCad's library.

Overwrite old file: If enabled, the exported STEP model will overwrite an existing file with the same name.

Optimize STEP file: If enabled, parametric curves will be disabled in the exported STEP model. This reduces the file size, but may reduce compatibility with some software.

Export tracks, pads and vias: If enabled, tracks, pads, and vias on outer layers will be modeled in the exported STEP model. This option may increase the export time.

Export zones: If enabled, zones on outer layers will be modeled in the exported STEP model. This option may increase the export time.

Board outline chaining tolerance: Controls the minimum distance between two points for the points to be considered coincident. If the board outline in the exported STEP model is not contiguous, try increasing this tolerance.

SVG exporter

The SVG exporter creates a vector graphics image of the board.

Layers: The selected layers will be included in the generated SVG.

Print mode: Controls whether the generated SVG file is in color or black and white.

Color theme: Controls the color theme used for the generated SVG file. If the **use current board theme** option is selected, the theme that is selected in the board editor will be used.

SVG page size: Controls the size of the generated SVG drawing. If **page with frame and title block** is selected, the drawing will match the board's sheet size and will include the drawing sheet and title block. If **current page size** is selected, the drawing will match the board's sheet size but will not include the drawing sheet. If **board area only** is selected, the drawing will be just big enough to fit the board itself.

Print mirrored: When selected, layers will be horizontally mirrored.

Print one page per layer: When selected, a separate SVG file will be generated for each selected layer. If the **print board edges on all pages** option is selected, the `Edge.Cuts` layer will be included in all generated SVGs, even if it is not selected as a layer.

Footprint association (CMP) exporter

CMP files are used to sync footprint assignments and some other footprint fields between the PCB and the schematic. You can import CMP files into the schematic using the schematic editor's **File → Import → Footprint Assignments** menu item. This provides a very limited form of back annotation. It is recommended to use the Update Schematic from PCB tool instead.

This exporter has no configurable options.

Hyperlynx exporter

The Hyperlynx exporter creates a file suitable for importing into Mentor Graphics (Siemens) HyperLynx simulation and analysis software. This exporter has no configurable options.

Footprints and footprint libraries

KiCad organizes footprints into footprint libraries, which hold collections of footprints. Each footprint in a board is uniquely identified by a full name that is composed of a library nickname and a footprint name. For example, the identifier `Capacitor_SMD:C_0603_1608Metric` refers to the `C_0603_1608Metric` footprint in the `Capacitor_SMD` library.

Managing footprint libraries

KiCad uses a table of footprint libraries to map footprint libraries of any supported library type to a library nickname. KiCad uses a global footprint library table as well as a table specific to each project. To edit either footprint library table, use **Preferences → Manage Footprint Libraries…**.
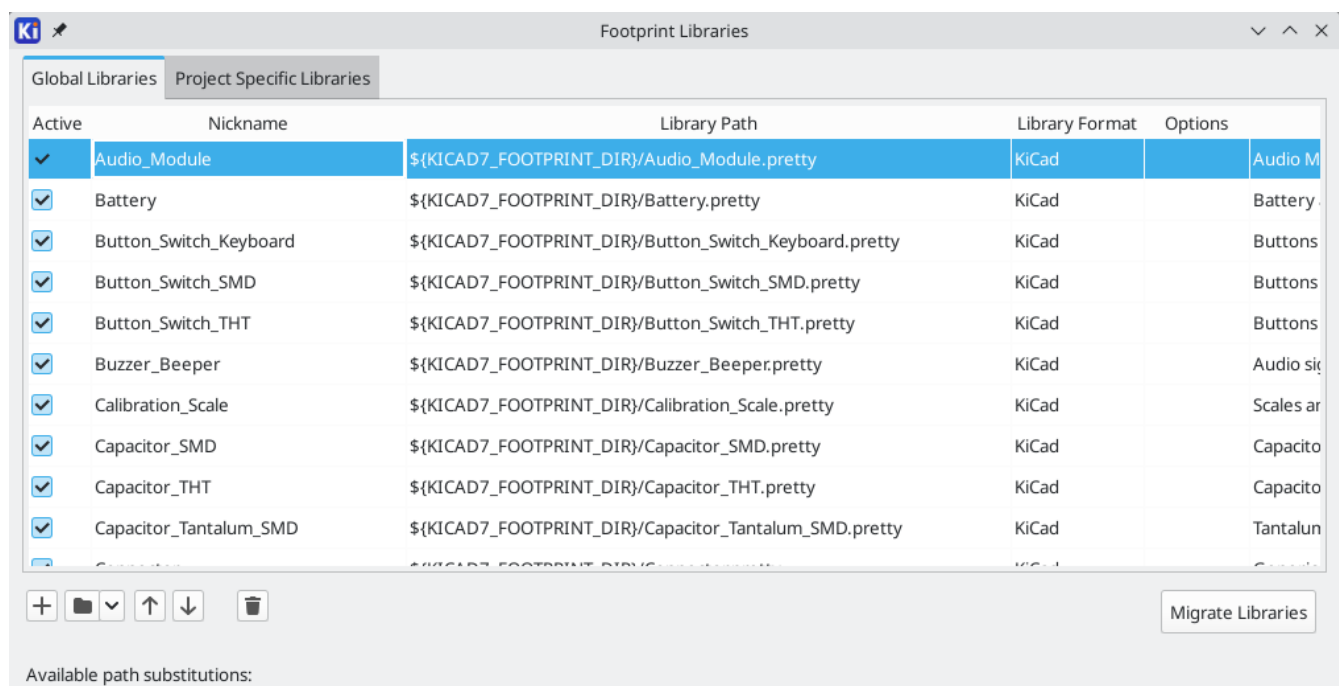
The global footprint library table contains the list of libraries that are always available regardless of the currently loaded project. The table is saved in the file `fp-lib-table` in the KiCad configuration folder. The location of this folder depends on the operating system being used.

The project specific footprint library table contains the list of libraries that are available specifically for the currently loaded project. If there are any project-specific footprint libraries, the table is saved in the file `fp-lib-table` in the project folder.

KiCad's footprint library management system allows directly using many types of footprint libraries, including formats that are native to other non-KiCad EDA tools:

- KiCad `.pretty` footprint libraries (folders with `.pretty` extension, containing `.kicad_mod` files)

- KiCad Legacy footprint libraries (`.mod` files)

- Altium Designer (`.PcbLib` or `.IntLib` files)

- CADSTAR PCB Archive (`.cpa` files)

- Eagle footprint libraries (`.lbr` files)

EasyEDA / JLCEDA Standard Edition (`.json` or `.zip` files)

- EasyEDA / JLCEDA Professional Edition (`.elibz`, `.epro`, or `.zip` files)

- GEDA libraries (folders containing `.fp` files)

Non-KiCad footprint libraries, including KiCad Legacy footprint libraries, can be migrated to KiCad `.pretty` format using the **Migrate Libraries** button (see the migrating libraries section).

NOTE	KiCad only supports writing to KiCad's native `.pretty` format footprint libraries (and the `.kicad_mod` footprint files within them). All other footprint library formats are read-only. To modify a non-KiCad format footprint library, you must first convert it to KiCad format.

Initial Configuration

The first time the PCB Editor (or any other KiCad tool that uses footprints) runs and the global footprint table file `fp-lib-table` is not found, KiCad will guide the user through setting up a new footprint library table. This process is described above.

Managing Table Entries

Footprint libraries can only be used if they have been added to either the global or project-specific footprint library table.

Add a library either by clicking the 📁 button and selecting a library or clicking the ➕ button and typing the path to a library file. The selected library will be added to the currently opened library table (Global or Project Specific). Libraries can be removed by selecting desired library entries and clicking the 🗑 button.

The ↑ and ↓ buttons move the selected library up and down in the library table. This does not affect the display order of libraries in the Footprint Library Browser, Footprint Editor, or Add Footprint tool.

Libraries can be made inactive by unchecking the **Active** checkbox in the first column. Inactive libraries are still in the library table but do not appear in any library browsers and are not loaded from disk, which can reduce loading times.

A range of libraries can be selected by clicking the first library in the range and then Shift -clicking the last library in the range.

Each library must have a unique nickname: duplicate library nicknames are not allowed in the same table. However, nicknames can be duplicated between the global and project library tables. Libraries in the project table take precedence over libraries with the same name in the global table.

Library nicknames do not have to be related to the library filename or path. The colon character (`:`) cannot be used in library nicknames or footprint names because it is used as a separator between nicknames and footprints.

Each library entry must have a valid path. Paths can be defined as absolute, relative, or by path variable substitution.

The appropriate library format must be selected in order for the library to be properly read. The supported formats are listed above. Only KiCad format libraries (`.pretty` folders containing `.kicad_mod` files) can be

saved. Other footprint library formats are read-only and must be converted to KiCad format before you can modify them.

There is an optional description field to add a description of the library entry. The option field is not used at this time so adding options will have no effect when loading libraries.

Path Variable Substitution

The footprint library tables support path variable substitution, which allows you to define path variables containing custom paths to where your libraries are stored. PATH variable substitution is supported by using the syntax `${PATH_VAR_NAME}` in the footprint library path.

By default, KiCad defines several path variables which are described in the [project manager documentation](). Path variables can be configured in the **Preferences → Configure Paths…** dialog.

Using path variables in the footprint library tables allows libraries to be relocated without breaking the footprint library tables, so long as the path variables are updated when the library location changes.

`${KIPRJMOD}` is a special path variable that always expands to the absolute path of the current project directory. `${KIPRJMOD}` allows libraries to be stored in the project folder without having to use an absolute path in the project library table. This makes it possible to relocate projects without breaking their project library tables.

Using the GitHub Plugin

| NOTE | KiCad removed support for the GitHub library plugin in version 6.0. |

Migrating footprint libraries to KiCad format

Non-KiCad format libraries, including legacy libraries (`.mod` files), are read-only. They need to be converted to KiCad format (`.kicad_mod` files in a `.pretty` folder) before you can save changes to them.

| NOTE | As with most KiCad files, newer versions of KiCad can open older-format library files, but older versions of KiCad cannot read files once they have been saved by a newer version of KiCad. |

Libraries in other formats can be converted to KiCad libraries by selecting them in the footprint library table and clicking the **Migrate Libraries** button. Multiple libraries can be selected and migrated at once by `Ctrl`-clicking or `shift`-clicking.

Libraries can also be converted one at a time by opening them in the Footprint Editor and saving them as a new library.

Creating and editing footprints

A footprint is the physical interface between a component package and a circuit board. Footprints can contain:

- Pads, which define how the component will be physically assembled onto the footprint. When a footprint is added to a board, tracks are routed to pads, and pads provide a magnetic snapping point for the router to connect the pad to a track. Pad shapes and layers are fully customizable, and pads can have plated holes, unplated holes, or no hole.

Graphic shapes and text for technical or aesthetic purposes. Graphics can be placed on physical layers (e.g. silkscreen or soldermask) or nonphysical layers. Graphic shapes can also be placed on copper layers, in which case they can make electrical connections.

- 3D models for mechanical CAD and visualization. 3D models are external files that footprints can link to; they are not embedded in footprints.

- Metadata associated with the footprint.

Footprints in KiCad are organized into footprint libraries, which contain zero or more footprints. Generally footprints are logically grouped by footprint category, function, and/or manufacturer. Each library is a folder (usually ending in `.pretty`) containing a `.kicad_mod` file for each footprint in the library.

Footprint editor overview

KiCad provides a footprint editing tool that allows you to create footprint libraries; add, edit, delete, or transfer footprints between libraries; export footprints to files; and import footprints from files. The Footprint Editor can be launched from the KiCad Project Manager or from the Board Editor (**Tools → Footprint Editor**).

The Footprint Editor main window is shown below. It has three toolbars for quick access to common features and a footprint viewing/editing canvas. Not all commands are available on the toolbars, but all commands are available in the menus.

In addition to the toolbars, there are collapsible panels for the footprint tree and properties manager (not shown) on the left, and the appearance panel and selection filter on the right. The bottom of the window contains a message panel that shows details about the selected object.
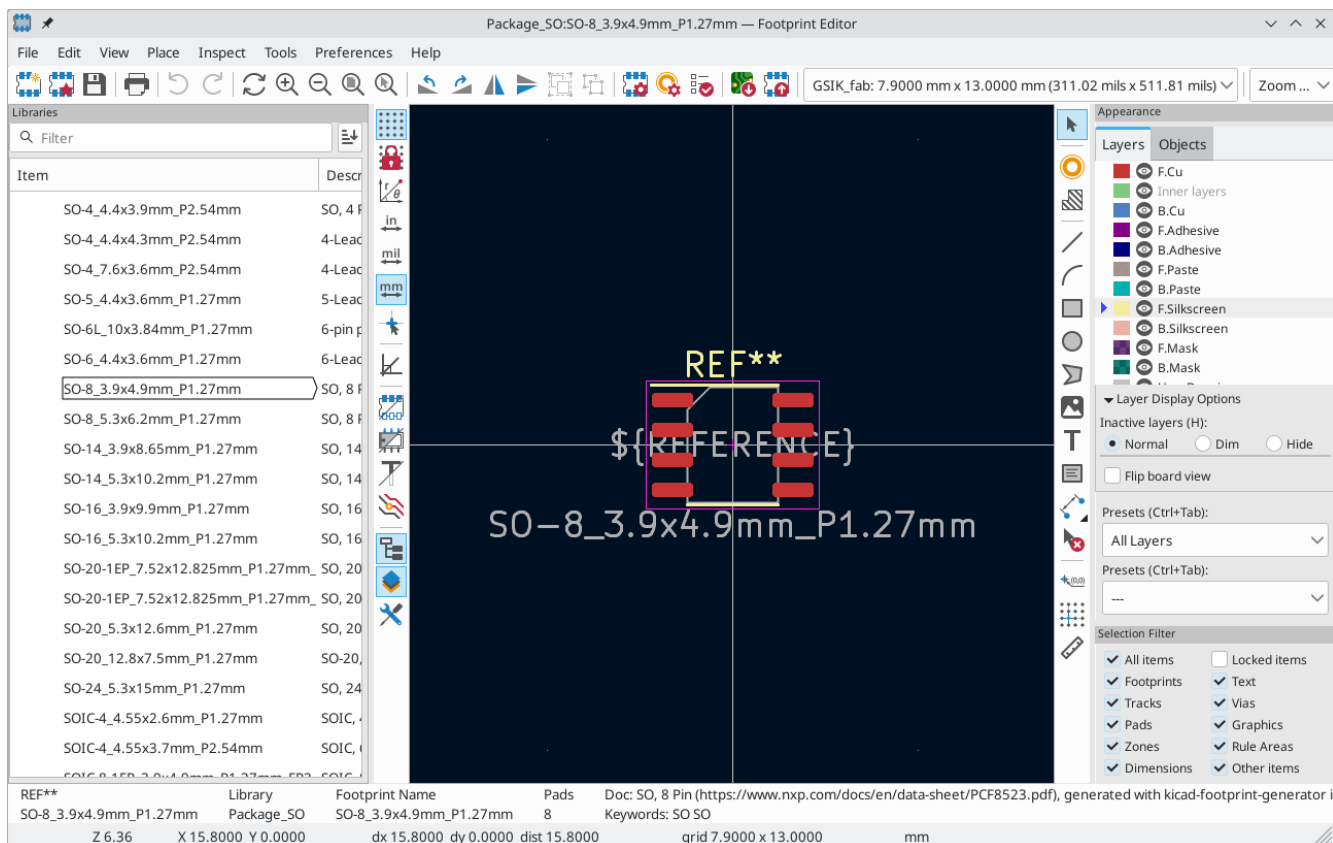
Top toolbar

The main toolbar is at the top of the main window. It has buttons for the undo/redo commands, zoom commands, footprint/pad properties dialogs, and layer/grid management controls.

	Create a new footprint in the selected library.
	Create a new footprint in the selected library using a footprint wizard.
	Save the currently selected footprint.
	Print the currently selected footprint.
	Undo last edit.
	Redo last undo.
	Refresh display.
	Zoom in.
	Zoom out.
	Zoom to fit footprint in display.
	Zoom to fit selection.
	Rotate selected item(s) counter-clockwise.
	Rotate selected item(s) clockwise.
	Mirror selected item(s) horizontally.
	Mirror selected item(s) vertically.
	Add the selected item(s) to a group.
	Remove the selected item(s) from a group.
	Edit the current footprint's properties.
	Edit the selected pad's properties.
	Test the current footprint for design errors.
	Edit a footprint in the current board in the footprint editor.
	Insert current footprint into the board.

Left toolbar display controls

The left toolbar provides options to change the display of items in the Footprint Editor.

	Turn grid display on/off. **Note:** by default, hiding the grid does not disable grid snapping. This behavior can be changed in the Display Options section of Preferences.
	Turn item-specific grid overrides on/off.
	Switch between polar and Cartesian coordinate display in the status bar.
	Display/entry of coordinates and dimensions in inches, mils, or millimeters.
	Switch between full-screen and small editing cursor (crosshairs).
	Switch between free angle and 45 degree mode for placement of new tracks, zones, graphical shapes, dimensions, and other objects. You can also toggle between free angle and 45 degree mode using `Shift` + `Space` .
	Switch display of pads between filled and outline mode.
	Switch display of graphic items between filled and outline mode.
	Switch display of text between filled and outline mode.
	Switch the non-active layer display mode between Normal and Dim. **Note:** this button will be highlighted when the non-active layer display mode is either Dim or Hide. In both cases, pressing the button will change the layer display mode to Normal. The Hide mode can only be accessed via the controls in the Appearance Panel or via the hotkey `Ctrl` + `H` .
	Toggle display of library and footprint tree.
	Show or hide the Appearance and Selection Filter panels on the right side of the editor.
	Show or hide the Properties Manager panel on the left side of the editor.

Right toolbar tools

Placement and drawing tools are located in the right toolbar.

	Selection tool (the default tool).
	Pad placement tool: click on the board to place a pad.
	Add rule area: Rule areas, formerly known as keepouts, can restrict the placement of items and the filling of zones and can also define named areas to apply specific custom design rules to.
	Draw lines. **Note:** Lines are graphical objects and are not the same as tracks placed with the Route Tracks tool. Graphical objects cannot be assigned to a net.
	Draw arcs: pick the center point of the arc, then the start and end points. By right clicking this button, you can change the arc editing mode between a mode that maintains the existing arc center and a mode that maintains the arc radius.
	Draw rectangles. Rectangles can be filled or outlines.
	Draw circles. Circles can be filled or outlines.
	Draw graphical polygons. Polygons can be filled or outlined. **Note:** Filled graphical polygons are not the same as filled zones: graphical polygons cannot be assigned to a net and will not keep clearance from other items.
	Add bitmap image for reference. Reference images are not included in fabrication outputs.
	Add text.
	Add a textbox.
	Add dimensions. Dimension types are described in more detail below.
	Deletion tool: click objects to delete them.
	Anchor tool. Left-click to set the anchor position (origin) of the footprint.
	Set grid origin.
	Interactively measure the distance between two points.

Browsing, modifying, and saving footprints

The ⊟ button displays or hides the list of available libraries, which allows you to select an active library. When a new footprint is created, it will be placed in the active library.

Clicking on a footprint name opens that footprint in the editor, and hovering the cursor over the name of a footprint displays a preview of the footprint.

After modification, a footprint can be saved in the current library or a different library. To save the modified footprint in the current library, click the 💾 button.

To save the footprint changes to a new footprint, click **File → Save As...**. The footprint can be saved in the current library or a different library, and a new name can be set for the footprint.

To create a new file containing only the current footprint, click **File → Export → Footprint...**. This file will be a standard footprint library which will contain only one footprint.

Creating a new footprint library

You can create a new footprint library by clicking **File → New Library...**. At this point you must choose whether the new library should be added to the global footprint library table or the project footprint library table. Libraries in the global library table will be available to all projects, while libraries in the project library table will only be available in the current project.
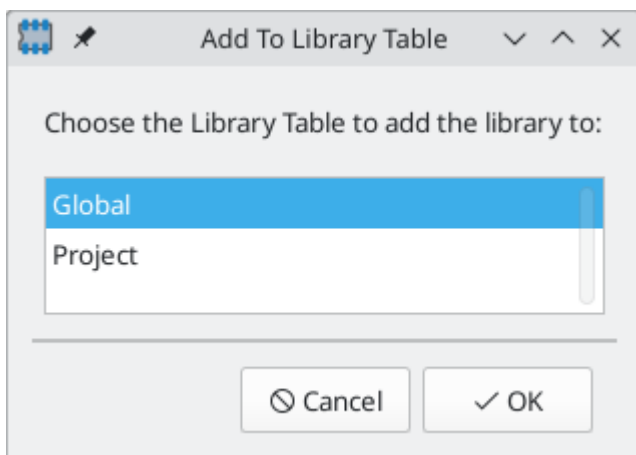
Following selection of the library table, you must choose a name and location for the new library. A new, empty library will be created at the specified location.

Creating a new footprint

To create a new footprint in the current footprint library, click the button or click **File → New Footprint...**. You will be prompted for new footprint's name and its footprint type.
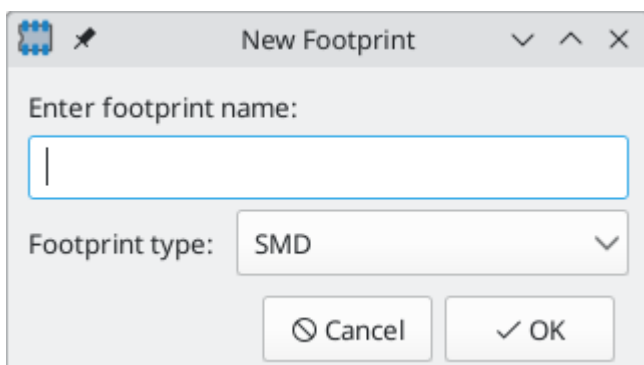
The name will set the name of the footprint, which is used when assigning a footprint to a symbol, and is also used as the filename of the footprint file on disk.

The type can be **Through hole**, **SMD**, or **Other**. Footprint type should be set appropriately, as it has several effects on pad creation, board inspection, DRC, and output generation. The footprint type can be changed after the footprint is created, however.

After clicking **OK**, a new footprint will be created in the selected library.

The new footprint will be empty except for several default text items. The footprint contains two default (mandatory) footprint fields, `Reference` and `Value`. `Reference` contains the text `REF**`, which will be replaced with the reference designator of the footprint's corresponding symbol when the footprint is added to the board. `Value` initially contains the footprint's name, but this will also be updated with the contents of the corresponding symbol's `Value` field when the footprint is added to the board. Finally, there is a footprint text item containing the string `${REFERENCE}`, which is a text variable that will resolve to the value of the footprint's `Reference` field once the footprint is on a board.
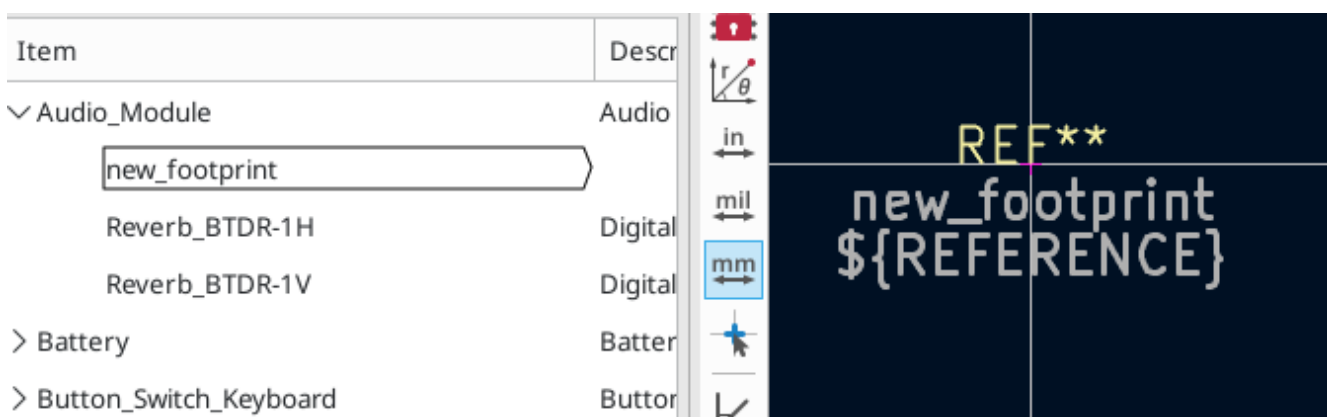
These items are centered on the footprint's anchor (origin point), which is indicated with a magenta cross symbol. The anchor can be repositioned (changing the `(0, 0)` point for the footprint) by selecting the ✛(0,0) button and clicking on the new desired anchor position.

> **NOTE** Rather than manually creating a footprint, for some common footprints you can use a footprint wizard to create a footprint based on a set of parameters.

Editing footprint properties

Footprints have a number of properties and metadata items that can be defined. These include text fields, attributes that can be set or not (such as Do Not Populate), clearance and zone connection settings, and 3D model paths. These are initially defined in the library copy of the footprint, but they can be modified on a per-instance basis once a footprint is added to a board. In other words, two copies of the same footprint on a single board can their properties edited separately.

Some properties, namely text fields and attributes, will be automatically set for each footprint in a board based on the fields and attributes in the footprint's corresponding schematic symbol. Fields and attributes are synced from symbols to footprints when you perform the Update PCB From Schematic action. They are also synced from footprints back to symbols when you perform the Update Schematic From PCB action.

To edit footprint properties, click the ⚙ button to show the Footprint Properties dialog. You can also double click an empty spot in the editing canvas.

Footprint name, description, and keywords

The footprint name, description, and keywords describe the footprint itself. Together they are intended to describe the footprint and help you select an appropriate footprint for each component. They are also used when searching for footprints in the Footprint Editor and the Add Footprint dialog.

The **footprint name** contains the name of the footprint. This is the same as the footprint's filename on disk, and is also initially the same as the footprint's `Value` field. However, the `Value` field can be edited in the footprint editor, and when a footprint is added to a board, its `Value` field will be updated with the value of the footprint's corresponding symbol.

The **description** is a description of the footprint. It should be human readable, but it is also used when searching for a footprint.

NOTE	This description property is specifically a description of the **footprint**. This is not to be confused with the `Description` field, which will be set to the description of the footprint's corresponding **symbol** when the footprint is added to a board.

The **keywords** are space-separated words related to the footprint. They are used when searching for a footprint.

Footprint fields

Footprints contain multiple fields, which are named values containing information related to the footprint. Fields can be visible and shown on any board layer, or they can be hidden and only shown in the footprint's properties. Some fields have special meaning to KiCad: `Reference` and `Footprint` are both both used by KiCad to identify schematic symbols and PCB footprints, for example. Other fields may contain information that is important for a design but is not interpreted by KiCad, like pricing or stock information for a part.

Any fields defined in a library footprint will be included in the footprint when it is added to a board. You can also add new fields to footprints in the board. Whether they are in the library footprint or not, these fields can then be edited on a per-footprint basis in the board. Symbol fields are also transferred to the board and added as fields in the corresponding footprint.

NOTE	Footprint fields are different than graphic text. In addition to being named, fields can be moved and edited in the board, while footprint text can only be edited in the footprint editor. Before KiCad version 8.0, footprints did not have fields, only graphic text, and this graphic text could be edited directly in the board editor.

All library footprints are defined with five default fields which correspond to the five default fields in library symbols: `Reference`, `Value`, `Footprint`, `Datasheet`, and `Description`. These default fields cannot be deleted. The `Reference` field initially has the value `REF**`, while the `Footprint` and `Value` fields are initially set to the name of the footprint. In the board, the values of the five default fields will be set to the values of the matching fields in the footprint's corresponding symbol.

NOTE	The `Description` footprint field is the description of the symbol, not the footprint, and will be overwritten by the value of the corresponding symbol's description. Footprints have a separate footprint description property (not a field), which is specifically intended for a description of the footprint.

Fields each have an associated layer, which determines which board layer the field will be placed on. Fields can also be visible or hidden.

To edit an existing footprint field, double-click the field, select it or hover and press E, or right-click on the field text and select **Properties…**.

To add new fields, delete optional fields, or edit existing fields, use the icon on the main tool bar to open the Footprint Properties dialog. Fields can be arbitrarily named, but names starting with `ki_`, e.g. `ki_description`, are reserved by KiCad and should not be used for user fields.

Fields have a number of properties, each of which is shown as a column in the properties grid. Not all columns are shown by default; columns can be shown or hidden by right clicking on the grid header and selecting or deselecting columns from the menu.

Footprint attributes

Footprints have several attributes, which are properties of the footprint that affect how it is handled by other parts of KiCad.

Every footprint has a **component type**: **SMD**, **Through hole**, or **Unspecified**. A footprint's type affects KiCad's behavior in a few ways:

- Footprint type controls the default type of new pads added to the footprint. For **through hole** and **Unspecified** footprints, new pads will be through hole by default, although they can be changed after creation. For **SMD** footprints, new pads will be SMD by default.

- Footprint type can be used to filter footprints from component placement files as well as other exports, such as STEP files. Additionally, the footprint type is included as metadata in IPC-2581 exports.

- Footprint 3D models can be shown and hidden in the 3D viewer based on their type. For example, SMD models can be hidden while through hole models are still displayed.

- Footprints of different types are reported separately in the Board Statistics dialog.

- DRC will report footprints containing pads that do not match the parent footprint's type, for example SMD pads in a through hole footprint.

If **not in schematic** is checked, KiCad will not expect the footprint to correspond to a symbol in the schematic. When updating a PCB from the schematic, KiCad will ordinarily remove footprints that don't have corresponding symbols according to the **delete footprints with no symbols** setting. However, such footprints will not be deleted when they have **not in schematic** set.

If **exclude from POS files** is checked, KiCad will not include the footprint in component placement file exports.

If **exclude from bill of materials** is checked, the component will not be included in bill of materials exports in either the schematic or PCB editors. This attribute is synced to and from the footprint's corresponding schematic symbol.

If **exempt from courtyard requirement** is checked, the footprint will not trigger a DRC violation if it does not contain a courtyard. Without this attribute set, a footprint without graphics on the `F.Courtyard` or `B.Courtyard` layer will cause a "Footprint has no courtyard defined" DRC violation.

The **do not populate** attribute is primarily a schematic symbol attribute, and is synced to and from the footprint's corresponding schematic symbol. Footprints with this attribute set can optionally be excluded from component placement file exports and some other types of outputs. These footprints can also be hidden in the 3D viewer.

Private footprint layers

Footprints can have private footprint layers, which are layers that can be viewed and edited in the Footprint Editor but are not shown in the footprint when it is added to a board. Therefore any objects that are on private layers will not be visible in the PCB Editor or included in PCB fabrication outputs. This may be useful, for example, for notes or graphics that are of interest when drawing or editing a footprint but not needed at the board level.

Any of the existing `User.*` layers (`User.Drawings`, `User.Comments`, `User.Eco1`, `User.1`, etc.) can optionally be a private layer. To make a layer private, add a private layer in the **General** tab of the footprint properties dialog, then select the desired layer. Any objects on that layer will not be shown on the board.

Clearance overrides and settings

The **Clearance Overrides and Settings** tab holds settings for footprint-specific overrides to board clearance and mask/paste expansion, pad-to-zone connections, and net tie settings for allowing pads within the footprint to short different nets.

Pad clearance controls the minimum clearance between the footprint's pads and any copper shape (tracks, vias, pads, zones) on a different net. This value is normally set to `0` which will cause the pad clearance to be inherited from the board's design rules and netclass rules. This value can be overridden for individual pads by setting the pad's clearance to a nonzero value.

The aperture appearing on any technical layer will have the same shape and size as the pad shape on the copper layer(s). In the PCB manufacturing process, the manufacturer will often change the relative size of mask and paste apertures relative to the copper pad size, but since this size change is specific to a manufacturing process, most manufacturers expect the design data to be provided with the apertures set to the same size as the copper pads. For specific situations where you need to oversize or undersize a technical layer aperture in the design data, you can use the settings here. These values can be overridden for individual pads by setting the pad's expansion or clearance to a nonzero value.

Solder mask expansion controls the size difference between the pad shape and the aperture shape on the F.Mask and B.Mask layers. A positive number means the solder mask aperture will be larger than the copper shape. This number is an inflation applied to all directions. For example, a value of `0.1mm` here will cause the solder mask aperture to be inflated by `0.1mm`, meaning that there will be an `0.1mm` border on all sides of the pad and the solder mask opening will be `0.2mm` wider than the pad when measured along a given axis.

Solder paste absolute clearance controls the size difference between the pad shape and the aperture shape on the F.Paste and B.Paste layers. Its behavior is otherwise identical to the behavior of the **solder mask expansion** setting.

Solder paste relative clearance allows setting a solder paste clearance value as a percentage of the pad size rather than an absolute distance value. If both relative and absolute clearances are specified, they are added together to determine the solder paste aperture size.

Pad connection to zones controls whether the footprint's pads will have solid, thermal relief, or no connection to zones. Like the other overrides, this one may be set for an individual pad or for an entire footprint. The default setting for this control is **From zone setting**, which uses the connection mode specified in the connection zones' properties. This setting can be overridden for individual pads by setting the pad's connection mode to a value other than **From parent footprint**.

Creating net ties

Footprints can act as net ties, where two separate nets are electrically connected by copper. Connecting nets together would normally causes a DRC error due to violating the clearance between two nets, but a footprint can be configured to short nets without causing a DRC violation. This can be used to connect multiple grounds at a specific location, to make kelvin sense connections to a component, or for other applications.
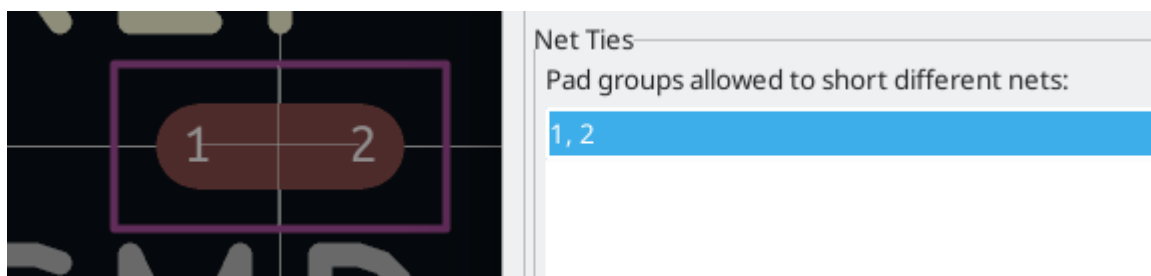
Net ties connect two or more nets in one contiguous region of copper. Each net in a net tie must have its own pad. Pads are not ordinarily allowed to short to other pads; to allow pads to be shorted in net ties, the shorting pads must be added to a **net tie group**. To create a net tie group, add the pad numbers of the shorting pads to the **Net Ties** table in the **Clearance Overrides and Settings** tab of the Footprint Properties dialog. For example, to allow pads `1` and `2` to short in a footprint, add a line to the table with the contents `1,2`.

After creating a net tie group, the specified pads are allowed to be electrically shorted. Pads in net tie groups can be connected either by directly overlapping the pads or by adding a copper shape that overlaps both pads.

Footprints can contain multiple net tie groups. Each group can short two or more nets, but every group must remain electrically separate from other groups.

3D models

The **3D Models** tab allows you to attach external 3D model files to a footprint and view the footprint in three dimensions along with any attached models.
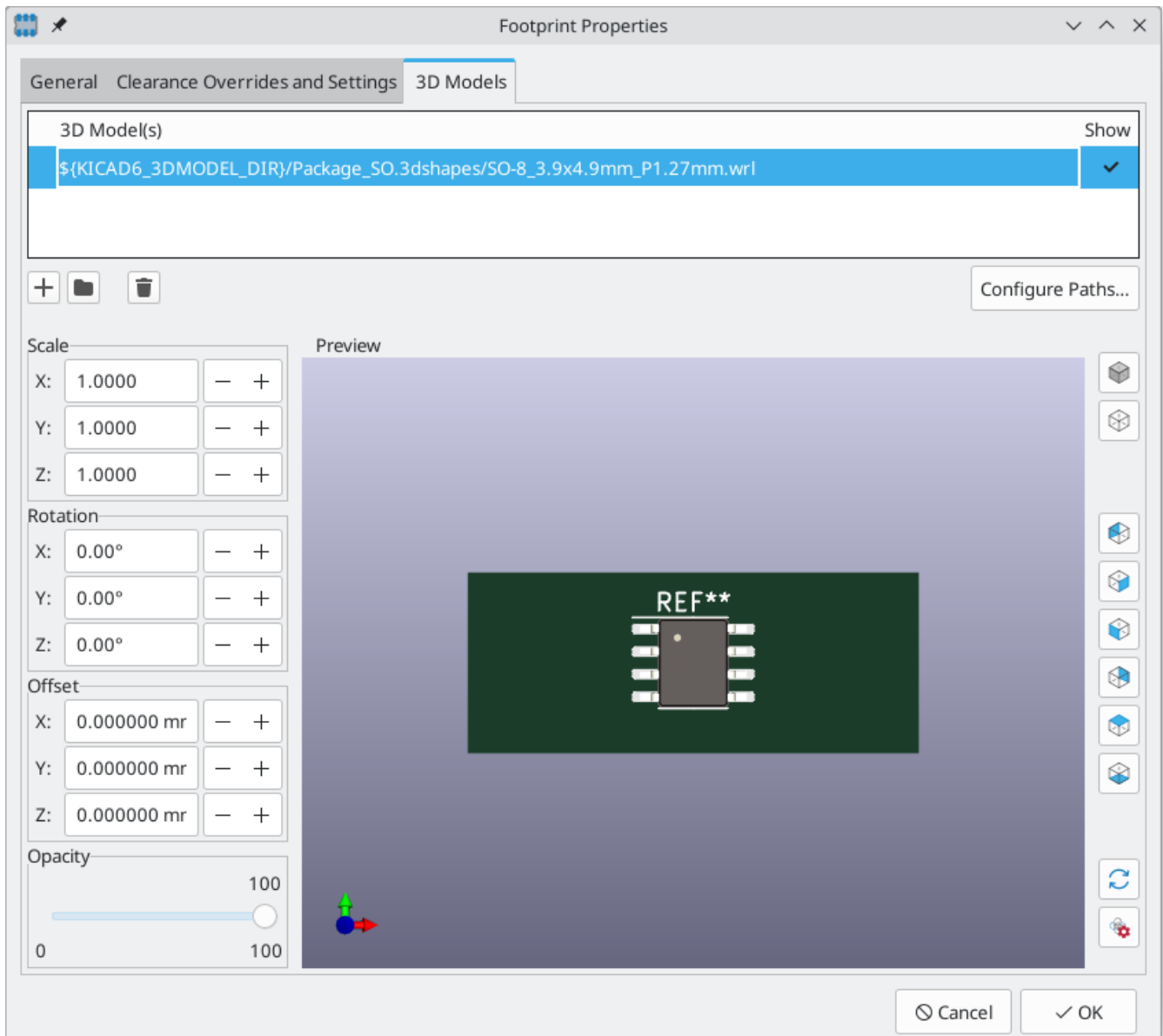
The main part of the window is a 3D preview of the footprint and any attached models. The buttons to the right of the preview let you enable or disable an orthographic projection ( ), show or hide the PCB model ( ), align the view to one of the six face-aligned perspectives ( ), and refresh the view ( ). The bottom button ( ) lets you set the thickness of the PCB in the preview.

The top of the dialog lets you attach external models. Each added model will be shown in the footprint preview as well as in the full PCB 3D view when the footprint is added to a board. Footprint models can be in STEP, VRML, or IDF format. The models are specified as paths to the model files, which can contain path variables such as `${KIPRJMOD}` or `${KICAD8_3DMODEL_DIR}`. Click the **Configure Paths** button to configure path variables. If there is a problem loading a model file from the specified path, an icon in the leftmost column will indicate an error.

NOTE	KiCad will automatically resolve versioned path variables from older versions of KiCad to the value of the corresponding variable from the current KiCad version, as long as the old variable is not explicitly defined itself. For example, `${KICAD7_FOOTPRINT_DIR}` will automatically resolve to the value of `${KICAD8_FOOTPRINT_DIR}` if there is no `KICAD7_FOOTPRINT_DIR` variable defined.
NOTE	Many footprints in KiCad's standard library do not yet have model files created for them. However, these footprints may contain a path to a 3D model that does not yet exist, in anticipation of the 3D model being created in the future.

By default, models are added with their origin placed at the footprint's origin, with no offset, scaling, or rotation. Offset, scaling, and rotation can be applied to a model using the controls to the left of the preview canvas. The model's opacity can be adjusted using the **opacity** slider, and the model can be completely hidden by deselecting the **show** checkbox in the rightmost column of the model table.

Footprint pads

Pads are added to a footprint by clicking the ⬤ button in the right toolbar, then clicking again in the desired location in the canvas. The tool will continue adding new pads each time you click on the canvas until you cancel the tool (Esc).

New pads in SMD footprints are SMD by default, while new pads in Through Hole and Unspecified footprints will be through hole. Each new pad will have its pad number incremented by 1 relative to the previous pad number.

Pad properties

You can edit a pad after adding it by opening the pad's properties dialog (E). These properties are also editable using the properties manager.

General tab

The **General** tab of the pad properties dialog shows the physical properties of the pad, including its geometry, shape, and layer settings.

Pad type: this setting controls which features are enabled for the pad:

SMD pads are electrically-connected and have no hole. In other words, they exist on a single copper layer.

Through-hole pads are electrically-connected and have a plated hole. The hole exists on every layer, and the copper pad exists on multiple layers (see **Copper layers** setting below).

Edge Connector pads are SMD pads that are allowed to overlap the board outline on the `Edge.Cuts` layer.

NPTH, Mechanical pads are non-plated through holes that do not have an electrical connection.

SMD Aperture pads are pads that have no hole and no electrical connection. These can be used to add specific designs to a technical layer, for example a paste or solder mask aperture.

The **Copper layers** setting controls which copper layers will have a shape associated with the pad.

For SMD pads, the options are `F.Cu` or `B.Cu`, controlling whether the pad sits on the front or the back of the board *relative to the footprint's location*. In other words, if a pad is set to exist on `B.Cu` in its properties, and the footprint is flipped to the back of the board, *that pad will now exist on* `F.Cu`, *because it also has been flipped.*

For through-hole pads, it is possible to remove the pad shape from copper layers where the pad is not electrically connected to other copper (tracks or filled zones). Setting the copper layers to **connected layers only** will remove the pad shape from any unconnected layers, and setting to `F.Cu`, `B.Cu`, **and connected layers** will remove the pad shape from any internal unconnected layers. This can be useful in dense board designs to increase the routable area on internal layers.

The **Technical layers** checkboxes control which technical layers will have an aperture added with the pad's shape. By default, pads have apertures on the paste and mask layers matching their copper layer.

> **NOTE** It is not possible to define a different pad shape or size on different copper layers in the current version of KiCad.

The **Pad number** controls what the pad will be electrically connected to in the board. A pad has the same net connection as the pin with the same number in the corresponding schematic symbol.

Pad **Position X** and **Y** are the location of the center of the pad, relative to the footprint's origin.

Pad shape controls the basic shape of the pad. This can be **circular**, **oval**, **rectangular**, **trapezoidal**, **rounded rectangle**, **chamfered rectangle**, **chamfered with other corners rounded**, **custom (circular base)**, or **custom (rectangular base)**. Each pad shape has its own set of options; for example, rounded rectangles have settings for **pad size X** and **Y**, **angle**, **corner size**, and **corner radius**.

> **NOTE** The size of a pad can also be adjusted interactively in the canvas by dragging the editing handles at the pad corners.

Through-hole and NPTH pads have a hole in addition to the pad itself. The **hole shape** can be **circular** or **oval**, with corresponding size controls. By default the pad is centered on the hole, but the pad can be offset relative to the hole if the **offset shape from hole** option is enabled (circular pads cannot be offset from the hole).

Fabrication properties are primarily used as metadata in Gerber X2 fabrication output, where the fabrication property is included as an aperture attribute for each pad. The following fabrication properties are available:

BGA pad can only be applied to SMD pads, and only affects Gerber X2 output.

Fiducial, local to footprint and **fiducial, global to board** only affect Gerber X2 output.

Test point can only be applied to SMD or through hole pads, can only be applied to pads on outer layers, and only affects Gerber X2 output.

Pads with the **heatsink pad** property are always flashed on every copper layer and are allowed in SMD footprints (PTH pads without this property are not allowed in SMD footprints). It also affects Gerber X2 output.
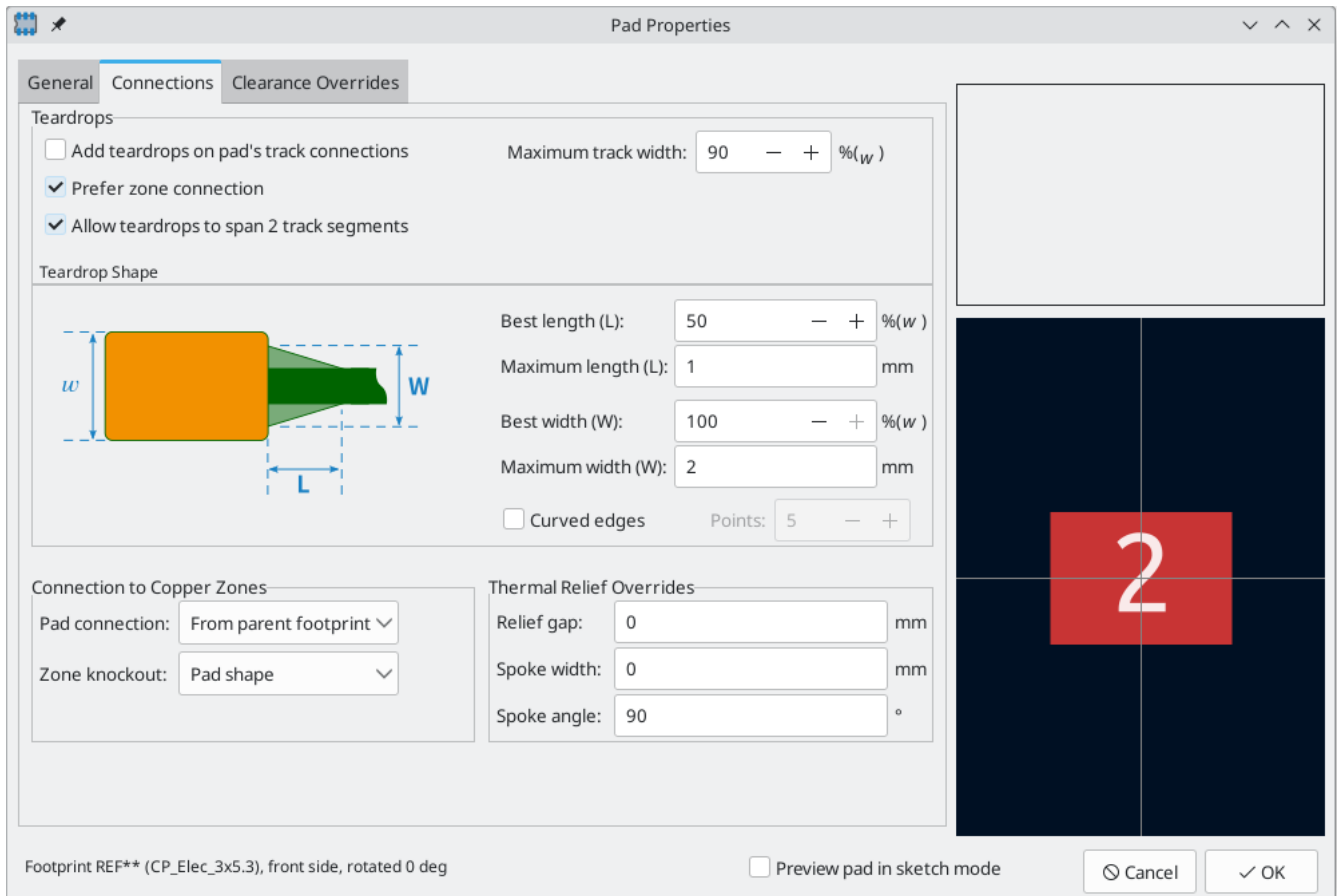
Pads with the **castellated pad** property are allowed to intersect the board edge and still be routed (it is otherwise a DRC error for a pad to intersect the board edge, which makes routing impossible). It also affects Gerber X2 output.

None is for pads for which none of the other fabrication properties apply. It has no effect.

Specify pad to die length: This setting allows a length to be associated with this pad that will be added to the routed track length by the track length tuning tools and the Net Inspector. This can be used to specify internal bondwire lengths for more accurate length matching, or in other situations where the electrical length of a net is longer than the length of the routed tracks on the board.

Connections tab

The **Connections** tab contains settings for how pads connect to other objects, including settings for teardrops, zone connections, and thermal reliefs.

The Teardrops section contains settings controlling teardrop connections between tracks and the pad, if teardrops are used. Teardrop settings are explained in the teardrop documentation.

Pad connection controls whether the pad will have a solid, thermal relief, or no connection to the zone. Like the other overrides, this one may be set for an individual pad or for an entire footprint. The default setting for this control is **From parent footprint**, and the default footprint setting is to use the connection mode specified in the zone properties.

Zone knockout controls the behavior of the zone filler when the pad uses a custom shape rather than one of the default shapes. This can be used to achieve different results when using thermal reliefs and custom pad shapes.

Relief gap controls the length of the thermal spokes, or the gap between the pad's shape and the filled copper area of the zone. This value is normally set to `0` which will cause the relief gap to be inherited from the connecting zone's settings.

Spoke width controls the width of the spokes generated when the zone connection mode is **Thermal Relief**. This value is normally set to `0` which will cause the spoke width to be inherited from the connecting zone's settings.

Clearance Overrides tab

The **Clearance Overrides** tab holds settings for pad-specific overrides to board clearance and mask/paste expansion.
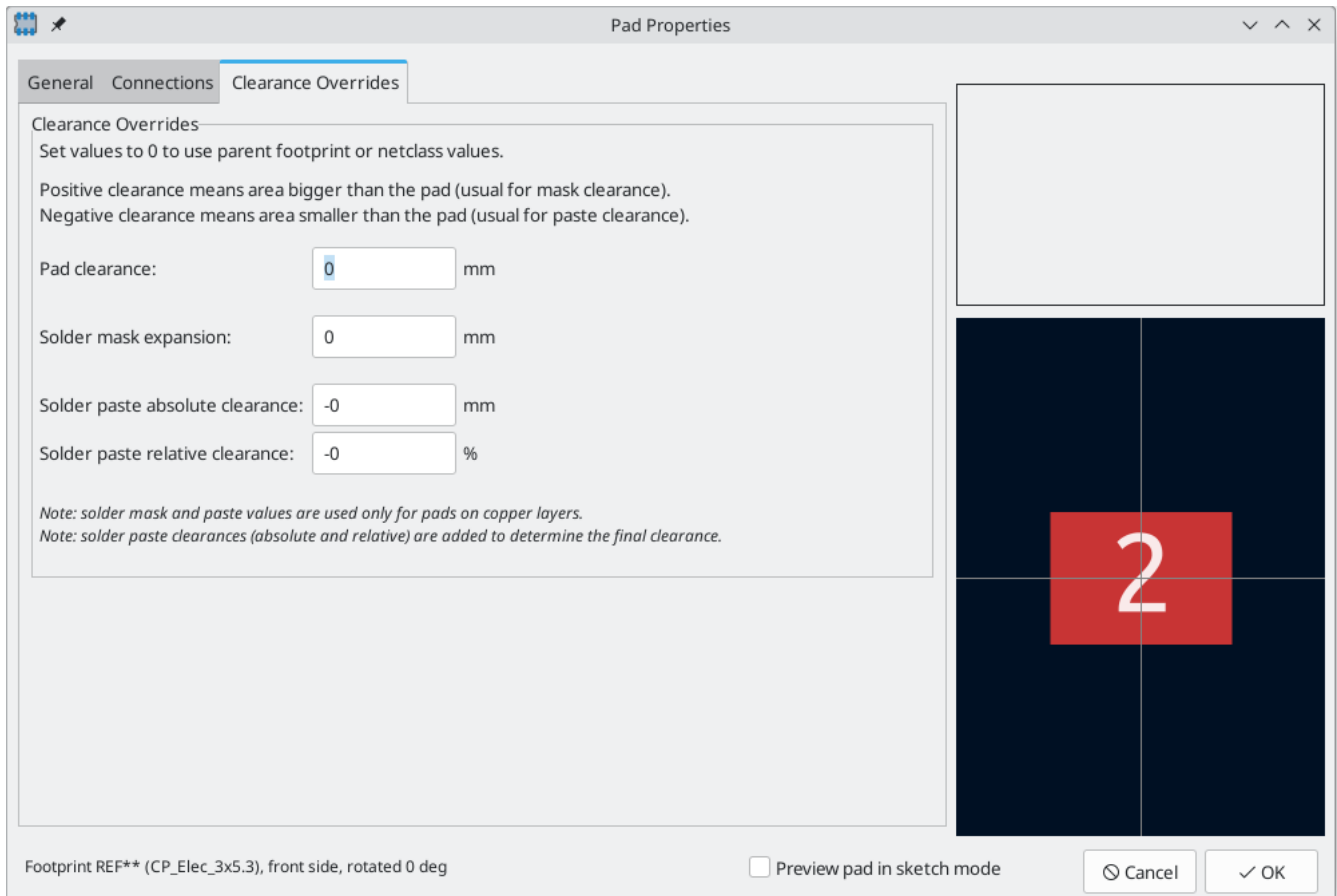
Pad clearance controls the minimum clearance between the pad and any copper shape (tracks, vias, pads, zones) on a different net. This value is normally set to `0` which will cause the pad clearance to be inherited from any clearance override set on the footprint, or the board's design rules and netclass rules if the footprint clearance is also set to `0`.

The aperture appearing on any technical layer will have the same shape and size as the pad shape on the copper layer(s). In the PCB manufacturing process, the manufacturer will often change the relative size of mask and paste apertures relative to the copper pad size, but since this size change is specific to a manufacturing process, most manufacturers expect the design data to be provided with the apertures set to the same size as the copper pads. For specific situations where you need to oversize or undersize a technical layer aperture in the design data, you can use the settings here.

Solder mask expansion controls the size difference between the pad shape and the aperture shape on the F.Mask and B.Mask layers. A positive number means the solder mask aperture will be larger than the copper shape. This number is an inflation applied to all directions. For example, a value of `0.1mm` here will cause the solder mask aperture to be inflated by `0.1mm`, meaning that there will be an `0.1mm` border on all sides of the pad and the solder mask opening will be `0.2mm` wider than the pad when measured along a given axis.

Solder paste absolute clearance controls the size difference between the pad shape and the aperture shape on the F.Paste and B.Paste layers. Its behavior is otherwise identical to the behavior of the **solder mask expansion** setting.

Solder paste relative clearance allows setting a solder paste clearance value as a percentage of the pad size rather than an absolute distance value. If both relative and absolute clearances are specified, they are added together to determine the solder paste aperture size.

Working with multiple pads

When you place a new pad, the new pad's properties are copied from the **default pad properties**. Each time any pad is edited, the pad's updated properties are stored as the default pad properties, so that new pads will match the properties of the most recently edited pad.

You can directly edit the default pad properties by selecting **Edit → Default Pad Properties…**, or choose an existing pad to represent the default by right clicking the pad and choosing **Copy Pad Properties to Default**. New pads will use that pad's properties as their defaults until a new default is selected, either by editing another pad, editing the default pad properties, or manually copying a different pad's properties to the default.

There are several ways to update existing pads with the properties of other pads. You can apply the default pad properties to an explicit selection of pads by selecting the desired target pads, right clicking, and choosing **Paste Default Pad Properties to Selected** from the right click context menu. You can also update other pads with a selected pad's properties using **Push Default Pad Properties to Other Pads…**, also in the right click context menu.
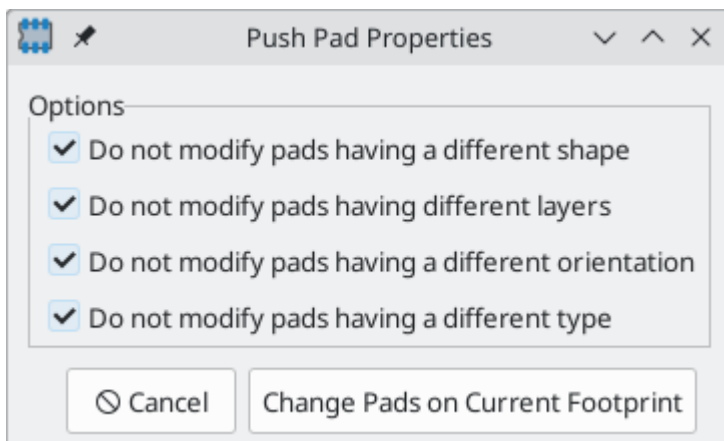
This tool has several options to filter which pads are targeted.

If **do not modify pads having a different shape** is selected, only pads with the exact same shape properties as the selected pad will be updated.

If **do not modify pads having different layers** is selected, only pads on the same layer(s) as the selected pad will be updated.

If **do not modify pads having a different orientation** is selected, only pads with the same orientation as the selected pad will be updated.

If **do not modify pads pads having a different type** is selected, only pads with the same pad type as the selected pad will be updated.

If no options are selected, all pads in the footprint will be updated.

You can create an array of pads from a source pad by right clicking the source pad and selecting **Create from Selection → Create Array…** (`Ctrl` + `T`). The basic functionality of this tool is described in the PCB Editor

documentation. For pads, however, there are additional options for controlling pad numbering.

For grid arrays, you can select a numbering direction, either **horizontal, then vertical** or **vertical, then horizontal**. If **reverse numbering on alternate rows/columns** is selected, the direction of increasing pad numbers will alternate from one row/column to the next.

The initial pad number in the array can either be the first unused pad number in the footprint (**use first free number**) or the specified **pad numbering start** value. After the first number, the pad numbering can either be **continuous** (1, 2, 3, …) or **coordinate** based, in other words, dependent on both the row and column (A1, A2, …, B1, …). In addition to the initial pad number (**pad numbering start**), you can specify a numbering step (**pad numbering skip**). For coordinate-based numbering, you can configure separate starting numbers and

steps for each axis. You can select whether pad numbers use decimal digits (0-9), hexadecimal digits (0-F), the full alphabet, or the alphabet excepting certain ambiguous letters (I, O, S, Q, X, and Z).

Finally, you can quickly renumber existing pads using the Renumber Pads tool (**Edit → Renumber Pads...**).

The tool has several options. Pads will be renumbered starting at the selected **first pad number**, and each subsequent pad will have its number incremented by the **numbering step**. You can also choose an optional **pad name prefix** which will be inserted before of the incrementing part of each pad number.

Once you click **OK**, you will be prompted to click on a pad, which will be assigned a new pad number based on the selected initial pad number and prefix. You can keep clicking on pads to assign them the next number in the sequence based on the selected numbering step. Double click on a pad to renumber that pad and end the sequence, or press `Esc` to discard the changes.

Custom pad shapes

For some footprints, the built-in pad shapes (round, rectangular, etc.) may not be sufficient. In these cases you can construct custom pads with arbitrary shapes using **Pad Edit Mode**. This mode lets you combine a basic pad with graphic shapes to build a new pad out of the compound shape.

To build a custom pad, first add a regular pad using the pad tool (⬤ button). This base pad will become the custom pad's anchor or snapping point, so be sure to place the pad in the exact location where you want tracks to attach to the pad. The shape and size of the pad do not matter, but the hole, if any, will remain in the final custom pad. In other words, a surface mount base pad will result in a surface mount custom pad, and a through hole base pad will result in a through hole custom pad. The custom pad's number will be inherited from the base pad.

Next, enter Pad Edit Mode by selecting the base pad, right-clicking, and selecting **Edit Pad as Graphic Shapes** (`Ctrl` + `E`). Add graphic shapes as appropriate to create the desired pad shape. Shapes touching the base pad will be unioned together with the base pad to create the final pad shape.

You can exit Pad Edit Mode by right-clicking and selecting **Finish Pad Edit**, or pressing `Ctrl` + `E` again. When you exit pad edit mode, all shapes that touch the base pad will be combined with the pad. For example, when editing a surface mount pad on `F.Cu`, any shapes that are on `F.Cu` and touch the base pad will become part of the custom pad. Any shapes that do not overlap the base pad, or that are on a different layer, will remain separate. If the base pad is a through hole pad, overlapping shapes on `F.Cu` will be combined in the custom pad. Because through hole pads have the same pad shape on all copper layers, this shape will become part of the custom pad on all copper layers, not just `F.Cu`. For convenience, Pad Edit Mode dims the color of other pads and all shapes that are not contiguous with the base pad so that you can see which shapes will be included in the custom pad and which will not.

Custom pads can only contain a single base pad. Any additional pads that touch the base pad or the contiguous graphics, whether they have the same or different pad numbers as the base pad, will remain separate pads after the shapes are combined into the custom pad.

NOTE	If you would like to add multiple anchors (snapping points) to a custom pad, you can add additional separate pads on top of the custom pad. Create the custom pad as normal, containing the first snapping point, then add additional pads with the same number and place them overlapping the custom pad in the desired snapping locations. They will remain distinct pads and will not be combined with the custom pad, but they will act as additional pad anchors and will be electrically connected to the custom pad.

To modify an existing custom pad, select it and enter Pad Edit Mode again. You can then continue to edit the component shapes to adjust the pad shape, or change the position of the base pad to adjust the pad anchor.

KiCad automatically chooses a size and location for showing the pad number over the pad. Particularly for unusually shaped pads, the automatically determined size and location may not be optimal. In these cases, you can manually specify a region in which KiCad should draw the pad number by adding a pad **number box** primitive. To add a number box, enter Pad Edit Mode and add a rectangular shape. In the properties manager for the rectangle, check the **Number Box** checkbox. The rectangle will then be shown as a wireframe, and when you exit Pad Edit Mode it will be used to draw the pad number.

In the board, KiCad will automatically add thermal spokes when connecting the pad to a zone. The thermal spoke settings are determined by the pad, footprint, and zone settings, and the thermal spokes by default

connect to the pad anchor. You can override the default thermal spoke placement by adding **thermal relief templates** to the custom pad. To add a thermal relief template, enter Pad Edit Mode and add a line shape. In the properties manager for the line, check the **Thermal Relief Template** checkbox. In Pad Edit Mode, the line will then be shown as a wireframe, and it will not be shown outside of pad edit mode. If any thermal relief templates are present in the pad, KiCad will not automatically add additional spokes when filling zones; spokes will only be placed where there are thermal relief templates defined in the pad. Thermal relief templates only determine the spoke location: spoke width and relief gap are still defined in the pad, footprint, and/or zone properties, as normal.

Footprint graphics

Footprints can contain graphic shapes, text, and dimensions. These objects can be placed on nonphysical layers, like `F.Fab` or `User.Drawings`, or they can be placed on layers that will be part of the manufactured circuit board, such as `Edge.Cuts` or a silkscreen, soldermask, or copper layer. Objects on copper layers can make electrical connections.

Closed shapes on a footprint's `F.Courtyard` and `B.Courtyard` layers will form the footprint's front and back courtyard, respectively. A courtyard defines the physical extents of a footprint and limits where footprints are allowed to be placed in relation to other footprints. If a footprint's courtyard overlaps another footprint's courtyard, a DRC violation will be generated.

Shapes on a footprint's `Edge.Cuts` layer will correspond to board edges on any PCB that includes the footprint. Closed shapes will result in cutouts, while unclosed shapes will result in unclosed edges. Unclosed edges must be closed in the full board design.

The buttons on the right toolbar can be used to create:

- Lines (╱ , default hotkey `Ctrl` + `Shift` + `L`)

- Arcs (╭ , default hotkey `Ctrl` + `Shift` + `A`)

- Rectangles (▢)

- Circles (◯ , default hotkey `Ctrl` + `Shift` + `C`)

- Polygons (⬠ , default hotkey `Ctrl` + `Shift` + `P`)

- Text (T , default hotkey `Ctrl` + `Shift` + `T`)

- Textboxes (▤)

- Dimensions (╱°), of which several types are available

> **NOTE**   You can customize the default style of newly-created text and shape objects in **Preferences → Footprint Editor → Default Values**.

Graphical objects and their properties are described in more detail in the PCB Editor documentation.

Bulk editing footprint text and graphics

Properties of text and graphics can be edited in bulk using the **Edit Text and Graphics Properties** dialog (**Edit → Edit Text & Graphic Properties…**).

This dialog is described in more detail in the [PCB Editor documentation](#).

Cleaning up footprint graphics

There is a dedicated tool for performing common cleanup operations on graphics, which is run via **Tools → Cleanup Graphics…**.

The following cleanup actions are available and will be performed when selected:

Merge lines into rectangles: combines individual graphic lines that together form a rectangle into a single rectangle shape object.

Delete redundant graphics: deletes graphics objects that are duplicated or degenerate.

Merge overlapping graphics into pads: merges graphic copper shapes that overlap pads into a custom pad.

Any changes that will be applied to the footprint are displayed at the bottom of the dialog. They are not applied until you press the **Update Footprint** button.

Rule areas

Rule areas, also known as keepouts, are footprint regions that can have specific DRC rules defined for them. Some basic rules are available that will raise DRC errors if certain types of objects are within the bounds of the rule area, but rule areas can also be used together with custom DRC rules to define complex DRC behavior that only applies within the rule area. A rule area in a footprint takes effect when the footprint is added to the board.

You can add a rule area by clicking the ▨ button on the right toolbar (`Ctrl` + `Shift` + `K`). Click on the canvas to place the first corner, which will show the Rule Area Properties dialog. After configuring the rule area appropriately, press **OK** to continue placing corners of the rule area. The rule area shape can be an arbitrary polygon; click on the starting corner or double click to finish placing the rule area.

Rule areas are described in more detail in the PCB editor documentation.

Reference images

Just like in the PCB Editor, you can use reference images in the Footprint Editor to assist with your footprint designs. Footprint reference images are only shown in the Footprint Editor: they are not shown in the PCB Editor when a footprint is added to a board, and they do not appear in any fabrication outputs.

To add a reference image, use the button on the right toolbar and select the desired reference image file.

Reference images are described in more detail in the PCB Editor documentation.

Footprint wizards

KiCad provides a set of footprint wizards that can be used to create some common kinds of footprints based on a set of parameters. Wizards for the following types of footprints are provided:

- BGA packages
- QFN packages
- QFP packages
- SOIC, MSOP, SSOP, TSSOP, etc. packages
- SIP and DIP packages
- ZIP packages
- ZOIC packages
- FPC connectors
- Micromatch SMD connectors
- Circular pad arrays
- Touch sliders
- Mutual capacitance touch buttons
- USS-39 barcodes
- QR codes

To create a footprint using a footprint wizard, click the button and choose a footprint type from the list that appears.
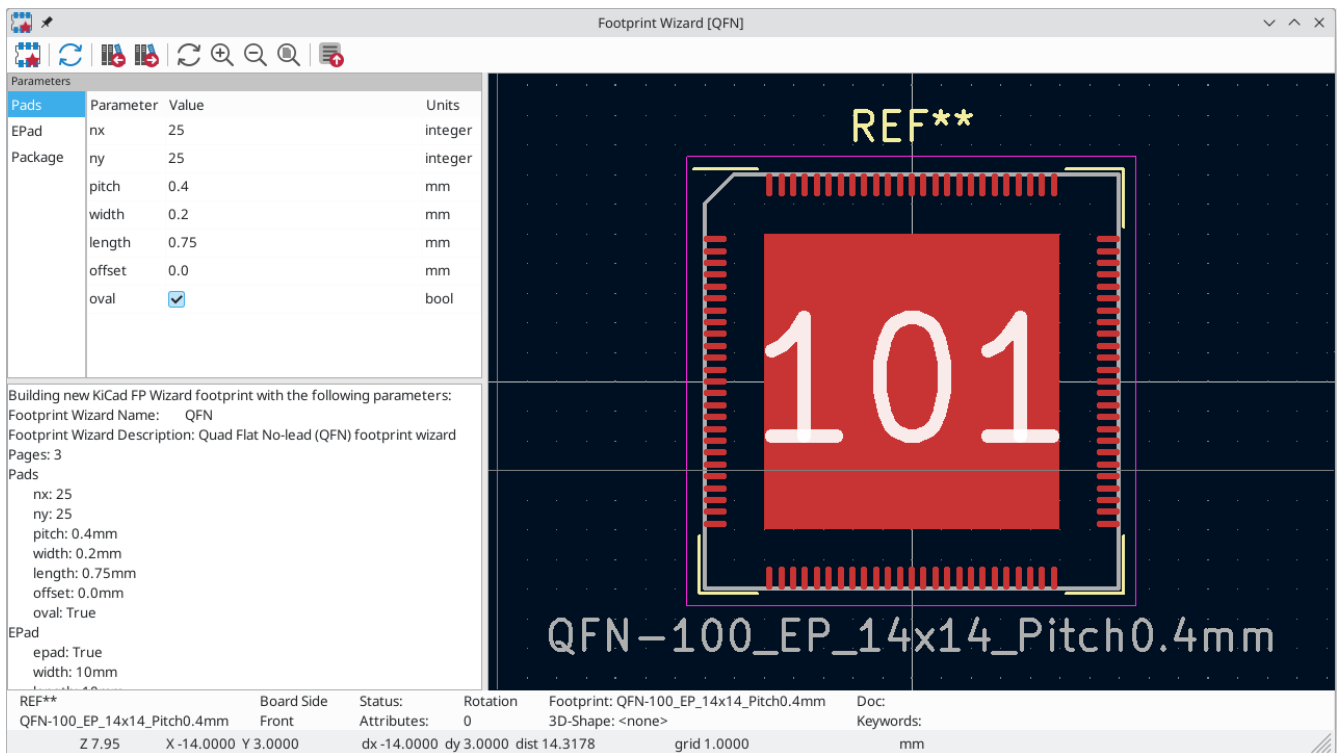
In the window that appears, fill out the parameters as appropriate. When the parameters are correctly filled out, press the button to transfer the generated footprint back into the footprint editor. Then you can make additional manual modifications and save the footprint as normal.

In addition to the set of footprint wizards that KiCad provides, you can also create your own. For more information about creating new footprint wizards, see the Scripting section of the Advanced Topics chapter.

Checking footprints

The Footprint Editor can check for common issues in your footprints. Run the footprint checker using the button in the top toolbar.

The footprint checker checks for:

- Pads that don't match the footprint's type: footprints without any through hole pads should be set to the surface mount footprint type

- Through hole pads without a hole

- Plated through hole pads not on any copper layers

- Plated through hole pads without a copper annulus

- Surface mount pads on both the front and back

- Surface mount pads with mismatched copper and paste/mask layers (front copper and back paste/mask, or vice versa)

- Pads that short to other pads outside of net tie groups

- Nonexistent pads in net tie groups

- Pads in that appear in multiple net tie groups

Browsing footprint libraries

The Footprint Library Browser allows you to quickly examine the contents of footprint libraries. The Footprint Library Viewer can be accessed by clicking ![icon] icon on the main Board Editor toolbar or with **View → Footprint Library Browser**.

To examine the contents of a library, select a library from the list in the left hand pane. All footprints in the selected library will appear in the second pane. Select a footprint name to view the footprint.
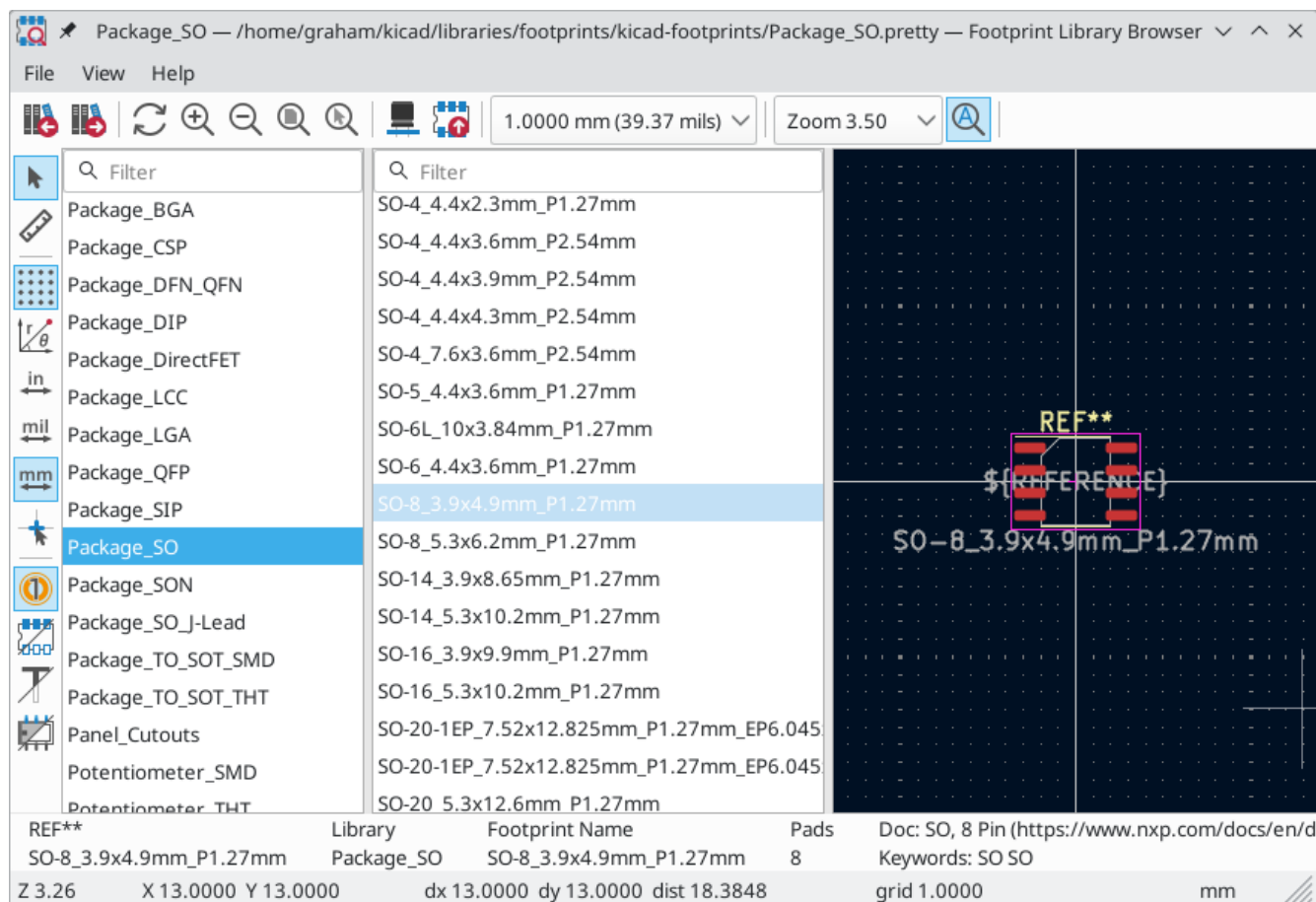
Double clicking the name of a footprint or using the ![button] button adds the footprint to the board.

The top toolbar contains the following commands:

![icon]	Select previous footprint in library.
![icon]	Select next footprint in library.
![icons]	Zoom tools.
![icon]	Open footprint in 3D Viewer.
![icon]	Add the footprint to the board.
![icon]	Automatically zoom to fit each opened footprint.

The left toolbar contains the following commands:

	Selection tool (the default tool).
	Interactively measure the distance between two points.
	Turn grid display on/off.
	Switch between polar and Cartesian coordinate display in the status bar.
in mil mm	Display/entry of coordinates and dimensions in inches, mils, or millimeters.
	Switch between full-screen and small editing cursor (crosshairs).
	Show or hide pad numbers.
	Switch display of pads between filled and outline mode.
	Switch display of text between filled and outline mode.
	Switch display of graphic items between filled and outline mode.

Advanced topics

Configuration and Customization

The KiCad PCB Editor has a variety of preferences that can be configured through the Preferences dialog. Like all parts of KiCad, the preferences for the PCB Editor are stored in the user configuration directory and are independent between KiCad minor versions to allow multiple versions to run side-by-side with independent preferences.

The first sections of the Preferences dialog (Common, Mouse and Touchpad, and Hotkeys) are shared between all KiCad programs. These sections are described in detail in the KiCad manual under the "Common preferences" section.

Display options
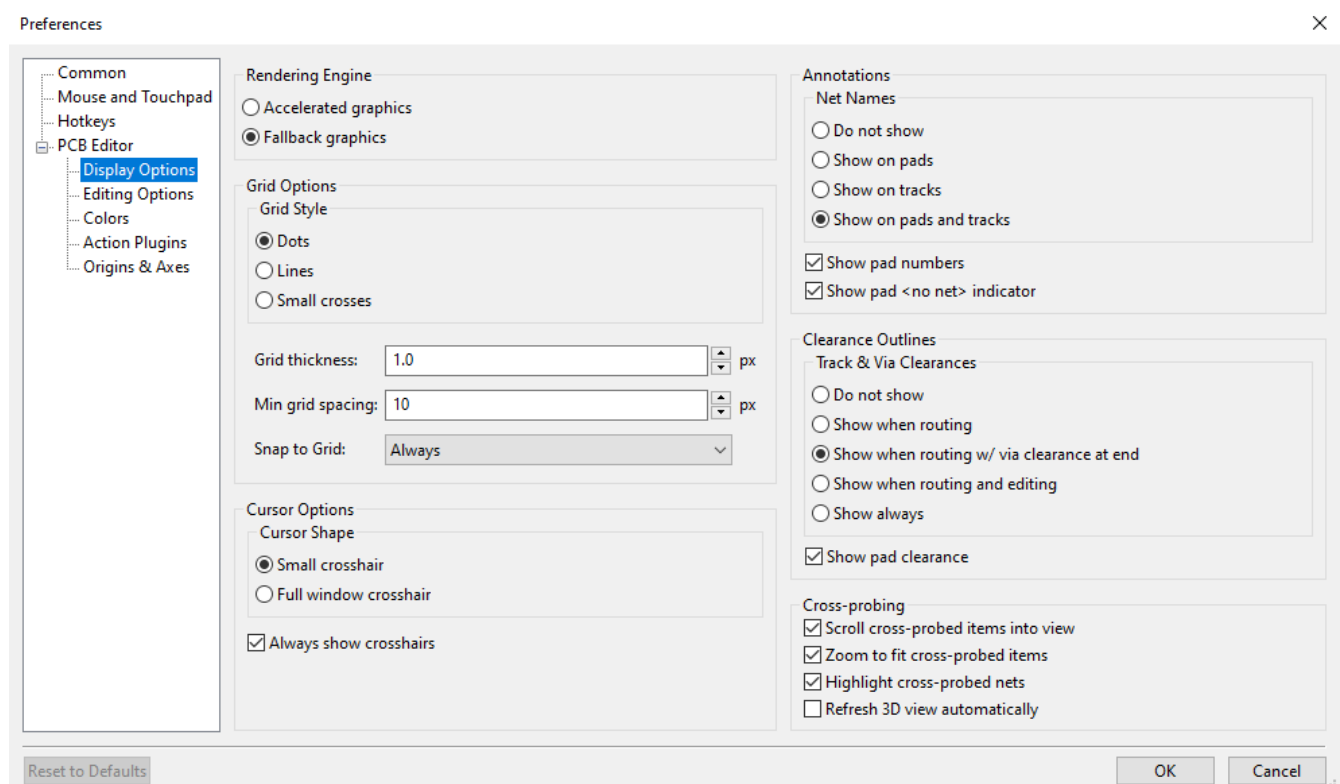
Rendering Engine: Controls if Accelerated graphics or Fallback graphics are used.

Grid style: Controls how the alignment grid is drawn.

Grid thickness: Controls how thick grid lines or dots are drawn.

Min grid spacing: Controls the minimum distance, in pixels, between two grid lines. Grid lines that violate this minimum spacing will not be drawn, regardless of the current grid setting.

Snap to grid: Controls when drawing and editing operations will be snapped to coordinates on the active grid. "Always" will enable snapping even when the grid is hidden; "When grid shown" will enable snapping only when the grid is visible.

NOTE | Grid snapping can be temporarily disabled by holding down `Ctrl`.

Cursor shape: Controls whether the editing cursor is drawn as a small crosshair or a full-screen crosshair (a set of lines covering the entire drawing canvas). The editing cursor shows where the next drawing or editing action will occur and will be snapped to a grid location if snapping is enabled.

Always show crosshairs: Controls whether the editing cursor is shown all the time or only when an editing or drawing tool is active.

Net names: Controls whether or not net name labels are drawn on copper objects. These labels are guides for editing only and do not appear in fabrication outputs.

Show pad numbers: Controls whether or not pad number labels are drawn on footprint pads.

Show pad <no net> indicator: Controls whether or not pads with no net are indicated with a special marker.

Track clearance: Controls whether or not clearance outlines around tracks and vias are shown. Clearance outlines are shown as thin shapes around objects that indicate the minimum clearance to other objects, as defined by constraints and design rules.

Show pad clearance: Controls whether or not clearance outlines around pads are shown.

Center view on cross-probed items: When the Schematic and PCB Editors are both running, controls whether clicking a component or pin in Eeschema will center the PCB Editor view on the corresponding footprint or pad.

Zoom to fit cross-probed items: Controls whether the view will be zoomed to show a cross-probed footprint or pad.

Highlight cross-probed nets: Controls whether or not nets highlighted in Eeschema will be highlighted in the PCB Editor when the highlight tool is activated in both tools.

Editing options
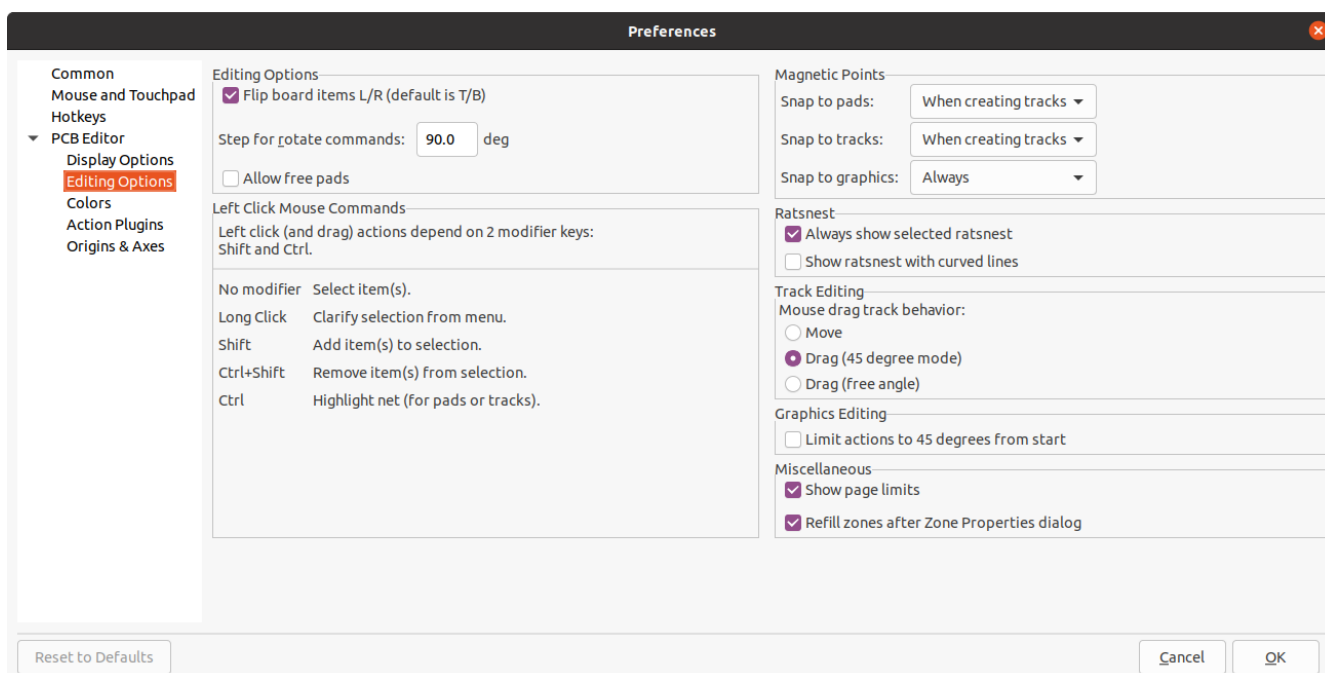
Flip board items L/R: Controls the direction board items will be flipped when moving them between the top and bottom layers. When checked, items are flipped Left-to-Right (about the Vertical axis); when unchecked, items are flipped Top-to-Bottom (about the Horizontal axis).

Step for rotate commands: Controls how far the selected object(s) will be rotated each time the Rotate command is used.

Allow free pads: Controls whether or not the pads of footprints can be unlocked and edited or moved separately from the footprint.

Magnetic points: This section controls object snapping, also called magnetic points. Object snapping takes precedence over grid snapping when it is enabled. Object snapping only works to objects on the active layer. Hold `Shift` to temporarily disable object snapping.

Snap to pads: Controls when the editing cursor will snap to pad origins.

Snap to tracks: Controls when the editing cursor will snap to track segment endpoints.

Snap to graphics: Controls when the editing cursor will snap to graphic shape points.

Always show selected ratsnest: When enabled, the ratsnest for a selected footprint will always be shown even if the global ratsnest is hidden.

Show ratsnest with curved lines: Controls whether ratsnest lines are drawn straight or curved.

Mouse drag track behavior: Controls the action that will occur when you drag a track segment with the mouse: "Move" will move the track segment independent of any others. "Drag (45 degree mode)" will invoke the push-and-shove router to drag the track, respecting design rules and keeping other track segments attached. "Drag (free angle)" will move the nearest corner of the track segment, highlighting collisions with other objects but not moving them out of the way.

Limit actions to 45 degrees from start Controls whether lines drawn with the graphic drawing tools can take on any angle. Note that this only affects drawing new lines: lines can be edited to take on any angle.

Show page limits: Controls whether or not the page boundary is drawn as a rectangle.

Refill zones after Zone Properties dialog: Controls whether or not zones are automatically refilled after editing the properties of any zone. This may be disabled on complicated designs or slower computers to improve responsiveness.

Colors
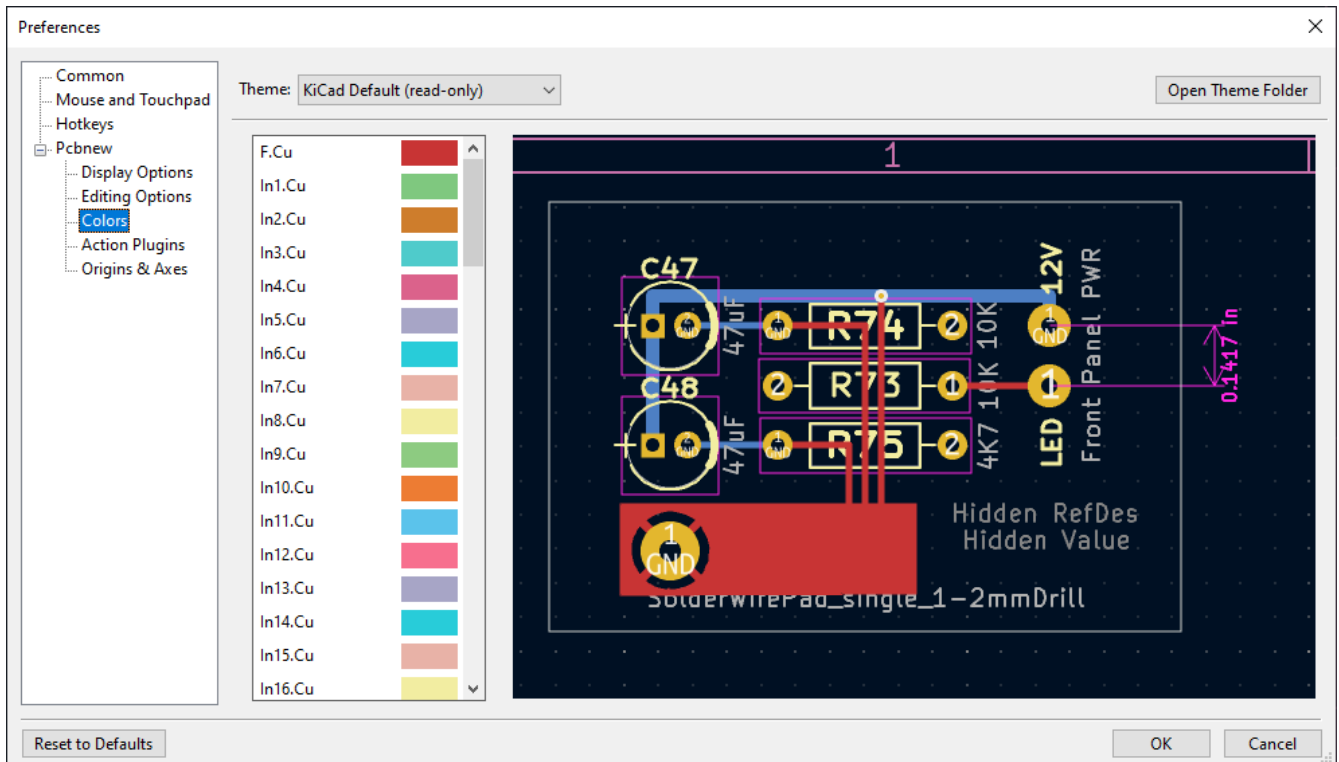
KiCad supports switching between different color themes to match your preferences. Kicad 8.0 comes with two built-in color themes: "KiCad Default" is a new theme designed to have good contrast and balance for most cases and is the default for new installations. "KiCad Classic" is the default theme from KiCad 5.1 and earlier versions. Neither of these built-in themes can be modified, but you can create new themes to customize the look of KiCad as well as install themes made by other users.

Color themes are stored in JSON files located in the `colors` subdirectory of the KiCad configuration directory. The "Open Theme Folder" button will open this location in your system file manager, making it easy to manage your installed themes. To install a new theme, place it in this folder and restart KiCad. The new theme will be available from the drop-down list of color themes if the file is a valid color theme file.

To create a new color theme, choose New Theme… from the drop-down list of color themes. Enter a name for your theme and then begin editing colors. The colors in the new theme will be copied from whatever theme was selected before you created the new theme.

To change a color, double-click or middle-click the color swatch in the list. The "Reset to Default" button will reset that color to its corresponding entry in the "KiCad Default" color theme.

Color themes are saved automatically; all changes are reflected immediately when you close the Preferences dialog. The window on the right side of the dialog shows a preview of how the selected theme will look.

Action plugins
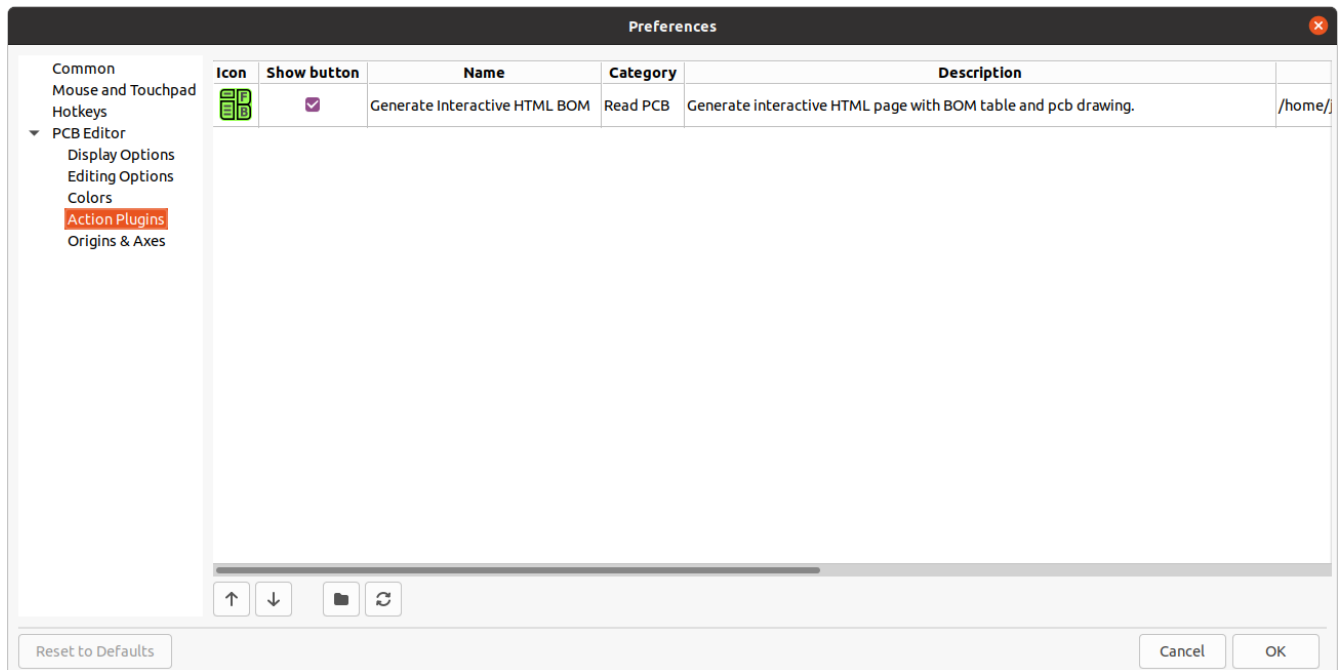
The KiCad PCB editor supports plugins written in Python that can perform actions on the board being edited. These plugins can be installed using the built-in Plugin and Content Manager (see the KiCad chapter for details) or by placing the plugin files inside the user plugins directory. See the Scripting section below for details.

Each plugin that is detected will be shown in a row on this preferences page. Plugins may show a button on the top toolbar of the PCB editor. If the "Show button" control is unchecked for a plugin, it may still be accessed from the Tools > External Plugins menu.

The arrow controls at the bottom of the list allow changing the order that the plugins appear in the toolbar and menu. The folder button will launch a file explorer to the plugin folder, to make installing new plugins easier. The refresh button will scan the plugin folder for any new or removed plugins and update the list.

Origin & axes
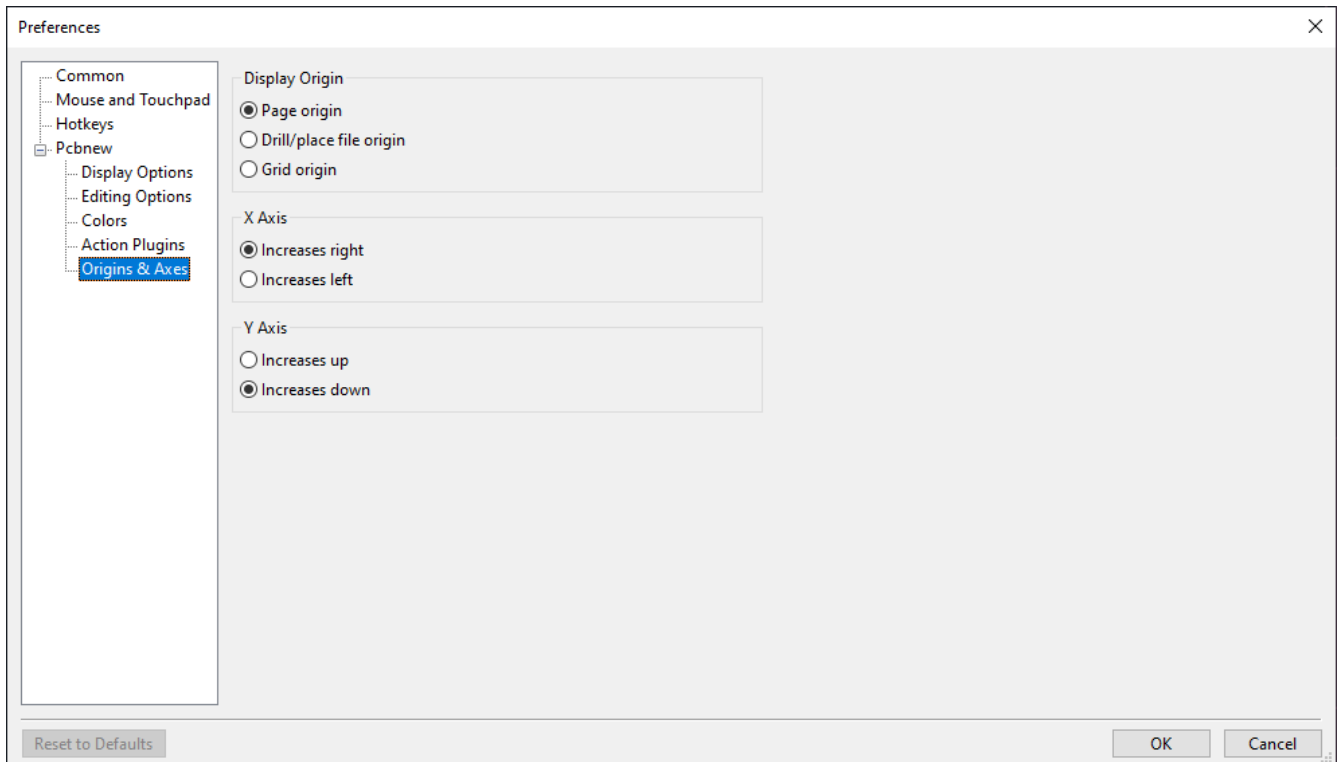
Display origin: Determines which coordinate origin is used for coordinate display in the editing canvas. The page origin is fixed at the corner of the page. The drill/place file origin and the grid origin can be moved by the user.

X axis: Controls whether X-coordinates increase to the right or to the left.

Y axis: Controls whether Y-coordinates increase upwards or downwards.

Text variables

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

You can define project text variables in the schematic or board setup dialogs. Project text variables are defined for the whole project, so a project text variable defined in the Schematic Editor can also be used in the Board Editor.

There are also a number of built-in system text variables. System text variables may be available in some contexts and not others. The following variables can be used in PCB text, footprint text, footprint fields, and drawing sheet fields. There are also a number of variables that can be used in the Schematic Editor.

Variable name	Description
`COMMENT1` - `COMMENT9`	Contents of drawing sheet's Comment<n> field.
`COMPANY`	Contents of drawing sheet's Company field.
`CURRENT_DATE`	Today's date, in ISO format.
`FILENAME`	Filename of the board, with a file extension.
`FILEPATH`	Full file path of the board, with a file extension.
`ISSUE_DATE`	Contents of drawing sheet's Issue Date field.
`KICAD_VERSION`	Current version of KiCad. This variable is only available in drawing sheet fields.
`LAYER`	Layer of the object. In footprint fields, this is the layer of the parent footprint. In drawing sheet fields, this resolves to the plotted layer.
`PAPER`	Current sheet's paper size. This variable is only available in drawing sheet fields.
`PROJECTNAME`	Project name, without a file extension.
`REVISION`	Contents of drawing sheet's Revision field.
`TITLE`	Contents of drawing sheet's Title field.
`<variablename>`	Contents of project text variable `<variablename>`.
`<fieldname>`	Contents of footprint field `<fieldname>`. Fields can only be accessed from within their parent object, so footprint fields can be accessed from other fields or text within the footprint. Both built-in footprint fields and user-defined fields from the corresponding symbol are available. Built-in footprint fields use all uppercase letters: for example, to access a footprint's value, use ${VALUE}. Built-in footprint fields are `FOOTPRINT_LIBRARY`, `FOOTPRINT_NAME`, `LAYER`, `NET_CLASS(<pad_number>)`, `NET_NAME(<pad_number>)`, `PIN_NAME(<pad_number>)`, `REFERENCE`, `SHORT_NET_NAME(<pad_number>)`, `VALUE`.
`<refdes>:` `<fieldname>`	Contents of field `<fieldname>` in footprint `<refdes>`. Both built-in footprint fields and user-defined fields from the corresponding symbol are available. Built-in footprint fields use all uppercase letters: for example, to access the value of `U1`, use `${U1:VALUE}`. Built-in footprint fields are `FOOTPRINT_LIBRARY`, `FOOTPRINT_NAME`, `LAYER`, `NET_CLASS(<pad_number>)`, `NET_NAME(<pad_number>)`, `PIN_NAME(<pad_number>)`, `REFERENCE`, `SHORT_NET_NAME(<pad_number>)`, `VALUE`.

Custom design rules

KiCad's custom design rule system allows creating design rules that are more specific than the generic rules available in the Constraints page of the Board Setup dialog. Custom design rules have many applications, but in general they are used to apply certain rules to a portion of the board, such as a specific net or netclass, a specific area, or a specific footprint.

Custom design rules are stored in a separate file with the extension `kicad_dru`. This file is created automatically when you start adding custom rules to a project. If you are using custom rules in your project, make sure to save the `kicad_dru` file along with the `kicad_pcb` and `kicad_pro` files when making backups or committing to a version control system.

NOTE	The `kicad_dru` file is managed automatically by KiCad and should not be edited with an external text editor. Always use the Custom Rules page of the Board Setup dialog to edit custom design rules.

The Custom Rules editor

The custom rules editor is located in the Board Setup dialog and provides a text editor for entering custom rules, a syntax checker that will test your custom rules and note any errors, and a syntax help dialog that contains a quick reference to the custom rules language and some example rules.

The custom rules editor also provides context-sensitive autocomplete to suggest valid keywords and properties. The autocomplete suggestion menu appears automatically, but it can also be opened manually by pressing `Ctrl` + `Space`.

It is a good idea to use the **Check rule syntax** button after editing custom rules to make sure there are no syntax errors. Any errors in the custom rules will prevent the design rule checker from running.

Custom rule syntax

The custom design rule language is based on s-expressions and allows you to create design constraints that are not possible with the built-in constraints. Each design rule generally contains a **condition** defining what objects to match and a **constraint** defining the rule to be applied to the matched objects.

The language uses parentheses (`(` and `)`) to define clauses of related keywords and values. Parentheses must always be matched: for every `(` there must be a matching `)`. Inside a clause, keywords and values are separated by whitespace (spaces, tabs, and newlines). By convention, a single space is used, but any number of whitespace characters between keywords and values is acceptable. In places where text strings are valid, strings without any whitespace may be quoted with `"` or `'`, or unquoted. Strings that contain whitespace must always be quoted. Newlines cannot be used within a quoted string. Where nested quotes are required, a single level of nesting is possible by using `"` for the outer quote character and `'` for the inner (or vice versa). Newlines between clauses are not required, but are typically used in examples for clarity.

In the syntax descriptions below, items in `<angle brackets>` represent keywords or values that must be present and items in `[square brackets]` represent keywords or values that are optional or only sometimes required.

The Custom Rules file must start with a version header defining the version of the rules language. As of KiCad 8.0, the version is `1`. The syntax of the version header is `(version <number>)`. So in KiCad 8.0 the header should read:

```
(version 1)
```

After the version header, you can enter any number of rules. Rules are evaluated in reverse order, meaning the last rule in the file is checked first. Once a matching rule is found for a given set objects being tested, no further rules will be checked. In practice, this means that more specific rules should be later in the file, so that they are evaluated before more general rules.

For example, if you create one rule that limits the minimum clearance between tracks in the net `HV` and tracks in any other net and a second rule that limits the minimum clearance for all objects inside a certain rule area, make sure the first rule appears later in the custom rules file than the second rule. Otherwise tracks in the `HV` net could have the wrong clearance if they fall inside the rule area.

Each rule must have a name and one or more `constraint` clauses. The name can be any string and is used to refer to the rule in DRC reports. The `constraint` defines the behavior of the rule. Rules may also have a `condition` clause that determines which objects should have the rule applied, an optional `layer` clause which specifies which board layers the rule applies to, and an optional `severity` clause which specifies the severity of the resulting DRC violation.

```
(rule <name>
    [(severity <severity>)]
    [(layer <layer_name>)]
    [(condition <expression>)]
    (constraint <constraint_type> [constraint_arguments]))
```

The custom rules file may also include comments to describe rules. Comments are denoted by any line that begins with the `#` character (not including whitespace). You can press `Ctrl` + `/` to comment or uncomment lines automatically.

```
# Clearance for 400V nets to anything else
(rule HV
    (condition "A.NetClass == 'HV'")
    (constraint clearance (min 1.5mm)))
```

Layer Clause

The `layer` clause determines which layers the rule will work on. While the layer of objects can be tested in the `condition` clause as described below, using the `layer` clause is more efficient.

The value in the `layer` clause can be any board layer name, or the shortcut keywords `outer` to match the front and back copper layers (`F.Cu` and `B.Cu`) and `inner` to match any internal copper layers.

If the `layer` clause is omitted, the rule will apply to all layers.

Some examples:

```
# Do not allow footprints on back layer (no condition clause means this rule always
applies)
(rule "Top side footprints only"
    (layer B.Cu)
    (constraint disallow footprint))

# This rule does the same thing, but is less efficient
(rule "Top side footprints only"
    (condition "A.Layer == 'B.Cu'")
    (constraint disallow footprint))

# Larger clearance on outer layers (inner layer clearance set by board minimum clearance)
(rule "clearance_outer"
    (layer outer)
    (constraint clearance (min 0.25mm)))
```

Severity Clause

The `severity` clause sets the DRC violation severity whenever the rule is violated.

Possible values are `error`, `warning`, `ignore`, and `exclusion`. Ignored rules are not observed by the interactive router and violations are not shown in the DRC dialog. However, ignored rules are evaluated for matching and therefore can still override earlier rules. Errors, warnings, and excluded rules are all observed by the interactive router, and violations are displayed in the DRC dialog when the appropriate filters are selected.

> **WARNING** | Setting a rule's severity to `ignore` does not disable the rule; only the effects of the rule are disabled. The rule is still evaluated and can still override previous rules.

Condition Clauses

The `condition` clause determines which objects which objects the rule applies to. If a rule has a condition clause, the rule will apply to any objects that match the condition. If a rule does not have any condition clauses, it will apply unconditionally.

The rule **condition** is an expression contained inside a text string (and therefore usually surrounded by quotes in order to allow whitespace for clarity). The expression is evaluated against each pair of objects that is being tested by the design rule checker. For example, when checking for clearance between copper objects, each copper object (track segment, pad, via, etc.) on each net is checked against other copper objects on other nets. If a custom rule exists where the expression matches the two given copper objects and the constraint defines a copper clearance, this custom rule could be used to determine the required clearance between the two objects.

The objects being tested are referred to as `A` and `B` in the expression language. The order of the two objects is not important because the design rule checker will test both possible orderings. For example, you can write a rule that assumes that `A` is a track and `B` is a via. There are some expression functions that test both objects together; these use `AB` as the object name.

The expression in a condition must resolve to a boolean value (`true` or `false`). If the expression resolves to `true`, the rule is applied to the given objects.

Each object being tested has **properties** that can be compared, as well as **functions** that can be used to perform certain tests. The syntax for using properties and functions is `<object>.<property>` and `<object>.<function>([arguments])` respectively.

<table>
<tr><td>NOTE</td><td>When you type `<object>.` in the text editor (`A.`, `B.`, or `AB.`), an autocomplete list will open that contains all the object properties that can be used.</td></tr>
</table>

The object properties and functions are compared using **boolean operators** to result in a boolean expression. The following operators are supported:

`==`	Equal to
`!=`	Not equal to
`>`, `>=`	Greater than, greater than or equal to
`<`, `<=`	Less than, less than or equal to
`&&`	And
`||`	Or

For example, `A.NetClass == 'HV'` will apply to any objects that are part of the "HV" netclass and `A.NetClass != B.NetClass` will apply to any objects that are in different netclasses. Parentheses can be used to clarify the order of operations in complex expressions but they are not required. All the boolean operators have the same precedence and are evaluated in order from left to right.

Some properties represent a physical measurement, such as a size, angle, length, position, etc. On these properties, **unit suffixes** can be used in the custom rules language to specify what units are being used. If no unit suffix is used, the internal representation of the property will be used instead (nanometers for distances and degrees for most angles). The following suffixes are supported:

`mm`	Millimeters
`mil`, `th`	Thousandths of an inch (mils)
`in`, `"`	Inches
`deg`	Degrees
`rad`	Radians

<table>
<tr><td>NOTE</td><td>The units used in custom design rules are independent of the display units in the PCB editor.</td></tr>
</table>

Numeric conditions can use simple math expressions, for example `(condition "A.Hole_Size_X == 1.0mm + 0.1mm")`.

Constraint Clauses

The `constraint` clause of the rule defines the behavior of the rule on the objects that are matched by the condition. Each constraint clause has a **constraint type** and one or more arguments that set the behavior of the constraint. A single rule may have multiple constraint clauses, in order to set multiple constraints (for example, `clearance` and `track_width`) for objects that match the same rule conditions.

Many constraints take arguments that specify a physical measurement or quantity. These constraints support minimum, optimal, and maximum value specification (abbreviated "min/opt/max"). The **minimum** and **maximum** values are used for design rule checking: if the actual value is less than the minimum or is greater than the maximum value in the constraint, a DRC error is created. The **optimal** value is only used for some constraints, and informs KiCad of a "best" value to use by default. For example, the optimal `diff_pair_gap` is used by the router when placing new differential pairs. No errors will be created if the differential pair is later modified such that the gap between the pair is different from the optimal value, as long as the gap is between the minimum and maximum values (if these are specified). In all cases where a min/opt/max value is accepted, any or all of the minimum, optimal, and maximum value can be specified.

Min/opt/max values are specified as `(min <value>)`, `(opt <value>)`, and `(max <value>)`. For example, a track width constraint may be written as `(constraint track_width (min 0.5mm) (opt 0.5mm) (max 1.0mm))` or simply `(constraint track_width (min 0.5mm))` if only the minimum width is to be constrained.

Numeric constraint values can use simple math expressions, for example `(constraint clearance (min 0.5mm + 0.1mm))`.

Constraint type	Argument type	Description
`annular_width`	min/opt/max	Checks the width of annular rings on vias.
`assertion`	boolean expression	Checks that the boolean expression is true. If the expression is false, a DRC error will be created. The expression can use any of the properties listed in the Object Properties section.
`clearance`	min	Specifies the **electrical** clearance between copper objects of different nets. (See `physical_clearance` if you wish to specify clearance between objects regardless of net.) To allow copper objects to overlap (collide), create a `clearance` constraint with the `min` value less than zero (for example, `-1`).
`connection_width`	min	Checks the width of connections between pads and zones. An error will be generated for each pad connection that is narrower than the `min` value.

Constraint type	Argument type	Description
courtyard_clearance	min	Checks the clearance between footprint courtyards and generates an error if any two courtyards are closer than the `min` distance. If a footprint does not have a courtyard shape, no errors will be generated from this constraint. To allow courtyard objects to overlap (collide), create a `courtyard_clearance` constraint with the `min` value less than zero (for example, `-1`).
diff_pair_gap	min/opt/max	Checks the gap between coupled tracks in a differential pair. Coupled tracks are segments that are parallel to each other. Differential pair gap is not tested on uncoupled portions of a differential pair (for example, the fanout from a component).
diff_pair_uncoupled	max	Checks the distance that a differential pair track is routed uncoupled from the other polarity track in the pair (for example, where the pair fans out from a component, or becomes uncoupled to pass around another object such as a via).
disallow	track via micro_via buried_via pad zone text graphic hole footprint	Specify one or more object types to disallow, separated by spaces. For example, `(constraint disallow track)` or `(constraint disallow track via pad)`. If an object of this type matches the rule condition, a DRC error will be created. This constraint is essentially the same as a keepout rule area, but can be used to create more specific keepout restrictions.
edge_clearance	min/opt/max	Checks the clearance between objects and the board edge. This can also be thought of as the "milling tolerance" as the board edge will include all graphical items on the `Edge.Cuts` layer as well as any **oval** pad holes. (See `physical_hole_clearance` for the drilling tolerance.) To allow objects to overlap (collide) with the board edge, create an `edge_clearance` constraint with the `min` value less than zero (for example, `-1`).
hole_clearance	min	Checks the clearance between a drilled hole in a pad or via and copper objects on a different net. The clearance is measured from the diameter of the hole, not its center.

Constraint type	Argument type	Description
hole_to_hole	min	Checks the clearance between mechanically-drilled holes in pads and vias. The clearance is measured between the diameters of the holes, not between their centers. This constraint is solely for the protection of drill bits. The clearance between **laser-drilled** (microvias) and other non-mechanically-drilled holes is not checked, nor is the clearance between **milled** (oval-shaped) and other non-mechanically-drilled holes.
length	min/max	Checks the total routed length for the nets that match the rule condition and generates an error for each net that is below the `min` value (if specified) or above the `max` value (if specified) of the constraint. This constraint also sets a target length that is used by the length tuning tool for any nets that match the rule condition.
min_resolved_spokes	0 1 2 3 4	Checks the total number of connections (spokes) to a pad. An error will be raised for each pad that has fewer than the specified number of spokes.
physical_clearance	min	Checks the clearance between two objects on a given layer (including non-copper layers). While this can perform more general-purpose checks than `clearance`, it is much slower. Use `clearance` where possible.
physical_hole_clearance	min	Checks the clearance between a drilled hole in a pad or via and another object, regardless of net. The clearance is measured from the diameter of the hole, not its center. This can also be thought of as the "drilling tolerance" as it only includes **round** holes (see `edge_clearance` for the milling tolerance).
silk_clearance	min/opt/max	Checks the clearance between objects on silkscreen layers and other objects. To allow silkscreen objects to overlap (collide) with other objects, create a `silk_clearance` constraint with the `min` value less than zero (for example, `-1`).

Constraint type	Argument type	Description
skew	min/opt/max	Checks the total skew for the nets that match the rule condition, that is, the difference between the length of each net and the average of all the lengths of each net that is matched by the rule. If the difference between that average and the length of any one net is above the constraint `max` value, an error will be generated. This constraint also sets a target skew that is used by the skew tuning tool for any nets that match the rule condition. The target skew is the `opt` value, if specified, or the `min` value if not. If neither `min` nor `opt` is specified, the target skew is `0`.
text_height	min/max	Checks the height of text, including text boxes. An error will be generated for each text item that has a height below the `min` value (if specified) or above the `max` value (if specified).
text_thickness	min/max	Checks the thickness of text, including text boxes. An error will be generated for each text item that has a thickness below the `min` value (if specified) or above the `max` value (if specified).
thermal_relief_gap	min	Specifies the width of the gap between a pad and a zone with a thermal-relief connection.
thermal_spoke_width	opt	Specifies the width of the spokes connecting a pad to a zone with a thermal-relief connection.
track_width	min/opt/max	Checks the width of track and arc segments. An error will be generated for each segment that has a width below the `min` value (if specified) or above the `max` value (if specified).
via_count	min/max	Counts the number of vias on every net matched by the rule condition. An error will be generated for each net that has fewer vias than the `min` value (if specified) or more than the `max` value (if specified).
via_diameter	min/max	Checks the diameter of vias. An error will be generated for each via that has a diameter below the `min` value (if specified) or above the `max` value (if specified).
zone_connection	solid thermal_reliefs none	Specifies the connection to be made between a zone and a pad.

Object property and function reference

The following properties can be tested in custom rule expressions:

Common Properties

Property	Data type	Description
Layer	string	The board layer on which the object exists. For objects that exist on more than one layer, this property will return the first layer (for example, F.Cu for most through-hole pads/vias).
Locked	boolean	True if the object is locked.
Parent	string	Returns the unique identifier of the parent object of this object.
Position_X	dimension	The position of the object's origin in the X-axis. Note that the origin of an object is not always the same as the center of the object's bounding box. For example, the origin of a footprint is the location of the (0, 0) coordinate of that footprint in the footprint editor, but the footprint may have been designed such that this location is not in the center of the courtyard shape.
Position_Y	dimension	The position of the object's origin in the Y-axis. Note that KiCad always uses Y-coordinates that increase from the top to bottom of the screen internally, even if you have configured your settings to show the Y-coordinates increasing from bottom to top.
Type	string	One of "Bitmap", "Dimension", "Footprint", "Graphic", "Group", "Leader", "Pad", "Target", "Text", "Text Box", "Track", "Via", or "Zone".

Connected Object Properties

These properties apply to copper objects that can have a net assigned (pads, vias, zones, tracks).

Property	Data type	Description
Net	integer	The net code of the copper object. Note that net codes should not be relied upon to remain constant: if you need to refer to a specific net in a rule, use `NetName` instead. `Net` can be used to compare the nets of two objects with better performance, for example `A.Net == B.Net` is faster than `A.NetName == B.NetName`.
NetClass	string	The name of the netclass for the copper object. Note: netclasses must be declared in the Board Setup dialog before they can be used in DRC rules.
NetName	string	The name of the net for the copper object.
Curve_Points	integer	Number of curve points on curved teardrops connected to the object.
Enable_Teardrops	boolean	True if teardrops are enabled for the object.
Prefer_Zone_Connections	boolean	True if the "Prefer zone connections" property is set for the object.
Allow_Teardrops_To_Span_Two_Tracks	boolean	True if the "Allow teardrops to span two tracks" property is set for the object.
Best_Length_Ratio	double	Best ratio of teardrop length to object size for teardrops connected to the object.
Best_Width_Ratio	double	Best ratio of teardrop width to object size for teardrops connected to the object.
Max_Length	dimension	Maximum length dimension for teardrops connected to the object.
Max_Width	dimension	Maximum width dimension for teardrops connected to the object.
Max_Width_Ratio	double	Maximum allowable ratio of object size to track width for teardrops connected to the object.

Footprint Properties

These properties apply to footprints.

Property	Data type	Description
`Clearance_Override`	dimension	The copper clearance override set for the footprint.
`Do_not_Populate`	boolean	True if the footprint's "Do not populate" attribute is set.
`Exclude_From_Position_Files`	boolean	True if the footprint's "Exclude from position files" attribute is set.
`Exclude_From_Bill_of_Materials`	boolean	True if the footprint's "Exclude from bill of materials" attribute is set.
`Exempt_From_Courtyard_Requirement`	boolean	True if the footprint's "Exempt from courtyard requirement" attribute is set.
`Keywords`	string	The "Keywords" from the library footprint.
`Library_Description`	string	The link to the library footprint in `library_name:footprint_name` format.
`Library_Link`	string	The link to the library footprint in `library_name:footprint_name` format.
`Not_in_Schematic`	boolean	True if the footprint's "Not in schematic" attribute is set.
`Orientation`	double	The orientation (rotation) of the footprint in degrees.
`Reference`	string	The reference designator of the footprint.
`Solderpaste_Margin_Override`	dimension	The solder paste margin override set for the footprint.
`Solderpaste_Margin_Ratio_Override`	dimension	The solder paste margin ratio override set for the footprint.
`Thermal_Relief_Gap`	dimension	The thermal relief gap set for the footprint.
`Thermal_Relief_Width`	dimension	The thermal relief connection width set for the footprint.
`Value`	string	The contents of the "Value" field of the footprint.
`Zone_Connection_Style`	string	One of "Inherited", "None", "Thermal reliefs" or "Solid".

Pad Properties

These properties apply to footprint pads.

Property	Data type	Description
Clearance_Override	dimension	The copper clearance override set for the pad.
Fabrication_Property	string	One of "None", "BGA pad", "Fiducial, global to board", "Fiducial, local to footprint", "Test point pad", "Heatsink pad", "Castellated pad".
Hole_Size_X	dimension	The size of the pad's drilled hole/slot in the X axis.
Hole_Size_Y	dimension	The size of the pad's drilled hole/slot in the Y axis.
Orientation	double	The orientation (rotation) of the pad in degrees.
Pad_Number	string	The "number" of a pad, which can be a string (for example "A1" in a BGA).
Pad_Shape	string	One of "Circle", "Rectangle", "Oval", "Trapezoid", "Rounded rectangle", "Chamfered rectangle", or "Custom".
Pad_To_Die_Length	dimension	The value of the "pad to die length" property of a pad, which is additional length added to the pad's net when calculating net length.
Pad_Type	string	One of "Through-hole", "SMD", "Edge connector", or "NPTH, mechanical".
Pin_Name	string	The name of the pad (usually the name of the corresponding pin in the schematic).
Pin_Type	string	The electrical type of the pad (usually taken from the corresponding pin in the schematic). One of "Input", "Output", "Bidirectional", "Tri-state", "Passive", "Free", "Unspecified", "Power input", "Power output", "Open collector", "Open emitter", or "Unconnected". Pins with a no-connection flag on them will have a "+no_connect" suffix added to the pin type string. For example, "passive+no_connect" will match a passive pin with a no-connection flag. To match a pin type whether or not the pin has a no-connection flag, use a wildcard: "passive*" will match passive pins with or without a no-connection flag.
Corner_Radius_Ratio	double	For rounded rectangle pads, the ratio of radius to rectangle size.
Size_X	dimension	The size of the pad in the X-axis.
Size_Y	dimension	The size of the pad in the Y-axis.
Soldermask_Margin_Override	dimension	The solder mask margin override set for the pad.

Property	Data type	Description
Solderpaste_Margin_Override	dimension	The solder paste margin override set for the pad.
Solderpaste_Margin_Ratio_Override	dimension	The solder paste margin ratio override set for the pad.
Thermal_Relief_Gap	dimension	The thermal relief gap set for the pad.
Thermal_Relief_Spoke_Angle	dimension	The thermal relief connection angle set for the pad.
Thermal_Relief_Spoke_Width	dimension	The thermal relief connection width set for the pad.
Zone_Connection_Style	string	One of "Inherited", "None", "Thermal reliefs" or "Solid".

Track and Arc Properties

These properties apply to tracks and arc tracks.

Property	Data type	Description
Origin_X	dimension	The x-coordinate of the start point.
Origin_Y	dimension	The y-coordinate of the start point.
End_X	dimension	The x-coordinate of the end point.
End_Y	dimension	The y-coordinate of the end point.
Width	dimension	The width of the track or arc.

Via Properties

These properties apply to vias.

Property	Data type	Description
Diameter	dimension	The diameter of the via's pad.
Hole	dimension	The diameter of the via's finished hole.
Layer_Bottom	string	The last layer in the via stackup.
Layer_Top	string	The first layer in the via stackup.
Via_Type	string	One of "Through", "Blind/buried", or "Micro".

Tuning Pattern Properties

These properties apply to tuning patterns.

Property	Data type	Description
End_X	dimension	The x-coordinate of the end point.
End_Y	dimension	The y-coordinate of the end point.
Min_Amplitude	dimension	The minimum amplitude of the tuning pattern.
Max_Amplitude	dimension	The maximum amplitude of the tuning pattern.
Tuning_Mode	string	One of "Single track", "Differential pair", or "Diff pair skew".
Initial_Side	string	One of "Left", "Right", or "Default".
Min_Spacing	dimension	The minimum spacing of the tuning pattern..
Corner_Radius_%	integer	The corner radius percentage of the tuning pattern.
Target_Length	dimension	The target length for the tuning pattern.
Target_Skew	dimension	The target skew for the tuning pattern.
Override_Custom_Rules	boolean	True if the tuning pattern overrides custom DRC rules.
Single-sided	boolean	True if the tuning pattern is single-sided.
Rounded	boolean	True if the tuning pattern uses rounded meanders.

Zone and Rule Area Properties

These properties apply to copper and non-copper zones, and rule areas (formerly called keepouts).

Property	Data type	Description
Clearance_Override	dimension	The copper clearance override set for the zone.
Min_Width	dimension	The minimum allowed width of filled areas in the zone.
Name	string	The user-specified name (blank by default).
Pad_Connections	string	One of "Inherited", "None", "Thermal reliefs", "Solid", or "Thermal Reliefs for PTH".
Priority	integer	The priority level of the zone.
Thermal_Relief_Gap	dimension	The thermal relief gap set for the zone.
Thermal_Relief_Width	dimension	The thermal relief connection width set for the zone.

Graphic Shape Properties

These properties apply to graphic lines, arcs, circles, rectangles, and polygons.

Property	Data type	Description
Angle	dimension	The angle of an arc.
End_X	dimension	The x-coordinate of the end point.
End_Y	dimension	The y-coordinate of the end point.
Filled	boolean	True if the shape is filled.
Line_Width	dimension	Thickness of the strokes of the shape.
Line_Style	string	One of "Solid", "Dashed", "Dotted", "Dash-Dot", "Dash-Dot-Dot".
Shape	string	One of "Segment", "Rectangle", "Arc", "Circle", "Polygon", or "Bezier".
Start_X	dimension	The x-coordinate of the start point.
Start_Y	dimension	The y-coordinate of the start point.

Text Properties

These properties apply to text objects (footprint fields, free text labels, etc).

Property	Data type	Description
Bold	boolean	True if the text is bold.
Height	dimension	Height of a character in the font.
Horizontal_Justification	string	Horizontal text justification (alignment): one of "Left", "Center", or "Right".
Italic	boolean	True if the text is italic.
Knockout	boolean	True if the text has the knockout property set.
Mirrored	boolean	True if the text is mirrored.
Text	string	The contents of the text object.
Thickness	dimension	Thickness of the stroke of the font.
Width	dimension	Width of a character in the font.
Vertical_Justification	string	Vertical text alignment: one of "Top", "Center", or "Bottom".
Visible	boolean	True if the text object is visible (displayed).

Expression functions

The following functions can be called on objects in custom rule expressions:

Function	Objects	Description
`enclosedByArea('x')`	`A` or `B`	Returns true if all of the object is inside the named rule area or zone. Note that `enclosedByArea()` is slower than `intersectsArea()`. Use `` `intersectsArea() `` where possible.
`existsOnLayer('layer_id')`	`A` or `B`	Returns true if the object exists on the given board layer. `layer_id` is a string containing the name of a board layer.
`fromTo('x', 'y')`	`A` or `B`	Returns true if the object exists on the copper path between the given pads. `x` and `y` are the full names of pads in the design, such as `'R1-Pad1'`.
`getField('x')`	`A` or `B`	Returns the value of field x in the object. Note that only footprints have fields, so no field will be returned unless the object is is a footprint.
`inDiffPair('x')`	`A` or `B`	Returns true if the object is part of a differential pair and the base name of the pair matches the given argument `x`. For example, `inDiffPair('/USB_')` or `inDiffPair('/USB')` both return `true` for objects in the nets `/USB_P` and `/USB_N`. `*` and `?` can be used as wildcards, so `inDiffPair('/USB*')` matches `/USB1_P` and `/USB1_N` as well as `/USB2_P` and `/USB2_N`. Note this will always return false if the given net is not a diff pair, meaning that there isn't a matching net of the opposite polarity. So, on a board with a net named `/USB_P` but no net named `/USB_N`, this function returns false.
`insideArea('x')`	`A` or `B`	Returns true if any part of the object is inside the named rule area or zone. Rule area and zone names can be set in their respective properties dialogs. If the given area is a filled copper zone, the function tests if the given object is inside any of the filled copper regions of the zone, not if the object is inside the zone's outline. **Deprecated**; use `intersectsArea()` instead.

Function	Objects	Description
`intersectsArea('x')`	A or B	Returns true if any part of the object is inside the named rule area or zone. Rule area and zone names can be set in their respective properties dialogs. If the given area is a filled copper zone, the function tests if the given object is inside any of the filled copper regions of the zone, not if the object is inside the zone's outline.

`intersectsCourtyard('x')`
 	
`intersectsFrontCourtyard('x')`
 	
`intersectsBackCourtyard('x')`
	A or B	Returns true if the any part of the object is inside the courtyard of the given footprint. The first variant checks both the front or back courtyard and returns true if the object is inside either one; the second and third variants check a courtyard on a specific layer.
		
		The footprint can be specified by reference designator or library ID in `<footprint_library>:<footprint_name>` format. The `*` or `?` wildcards can be used in the argument: `insideCourtyard('R?')` will check all footprints with references that contain `R` followed by a single character, while `insideCourtyard('Resistor_SMD:*')` will check all footprints in the `Resistor_SMD` library.

`isBlindBuriedVia()`	A or B	Returns true if the object is a blind/buried via.
`isCoupledDiffPair()`	AB	Returns true if the two objects being tested are part of the same differential pair but are opposite polarities. For example, returns true if `A` is in net `/USB+` and `B` is in net `/USB-`.
`isMicroVia()`	A or B	Returns true if the object is a microvia.
`isPlated()`	A or B	Returns true if the object is a plated hole (in a pad or via).
`memberOf('x')`	A or B	Returns true if the object is a member of the named group x. **Deprecated**; use `memberOfGroup()` instead.
`memberOfGroup('x')`	A or B	Returns true if the object is a member of a group named x.

Function	Objects	Description
`memberOfFootprint('x')`	A or B	Returns true if the object is a member of the given footprint. The footprint can be specified by reference designator or library ID in `<footprint_library>:<footprint_name>` format. The `*` or `?` wildcards can be used in the argument: `insideCourtyard('R?')` will check all footprints with references that contain R followed by a single character, while `insideCourtyard('Resistor_SMD:*')` will check all footprints in the `Resistor_SMD` library. reference designator `x`, or if `x` contains a `:`, a member of a footprint with the footprint ID `x`.
`memberOfSheet('x')`	A or B	Returns true if the object is a member of a schematic sheet named `x`.

Custom design rule examples

Basic examples

```
(rule RF_width
    (layer outer)
    (condition "A.NetClass == 'RF'")
    (constraint track_width (min 0.35mm) (max 0.35mm)))

(rule "BGA neckdown"
    (constraint track_width (min 0.2mm) (opt 0.25mm))
    (constraint clearance (min 0.05mm) (opt 0.08mm))
    (condition "A.insideCourtyard('U3')"))

# Specify an optimal gap for a particular differential pair
(rule "Clock gap"
    (condition "A.inDiffPair('/CLK')")
    (constraint diff_pair_gap (opt 0.8mm)))

# Specify a larger clearance between differential pairs and anything else
(rule "Differential pair clearance"
    (condition "A.inDiffPair('*') && !AB.isCoupledDiffPair()")
    (constraint clearance (min 1.5mm)))

(rule "copper keepout"
   (constraint disallow track via zone)
   (condition "A.intersectsArea('zone3')"))
```

Various clearances

```
(rule "Clearance between Pads of Different Nets"
    (constraint clearance (min 3.0mm))
    (condition "A.Type == 'Pad' && B.Type == 'Pad' && A.Net != B.Net"))

(rule "Pad to Track Clearance"
    (constraint clearance (min 0.2mm))
    (condition "A.Type == 'Pad' && B.Type == 'Track'"))

# Enforce a mechanical clearance between components and board edge
(rule front_mechanical_board_edge_clearance
    (layer "F.Courtyard")
    (constraint physical_clearance (min 3mm))
    (condition "B.Layer == 'Edge.Cuts'"))

# This assumes that there is a cutout with 1mm thick lines
(rule "Clearance to cutout"
    (constraint edge_clearance (min 0.8mm))
    (condition "A.Layer=='Edge.Cuts' && A.Line_Width == 1.0mm"))

# prevent silk over tented vias
(rule silk_over_via
   (constraint silk_clearance (min 0.2mm))
   (condition "A.Type == '*Text' && B.Type == 'Via'"))

(rule "Distance between Vias of Different Nets"
    (constraint hole_to_hole (min 0.254mm))
    (condition "A.Type == 'Via' && B.Type == 'Via' && A.Net != B.Net"))

(rule "Via Hole to Track Clearance"
    (constraint hole_clearance (min 0.254mm))
    (condition "A.Type == 'Via' && B.Type == 'Track'"))

(rule "Distance between test points"
    (constraint courtyard_clearance (min 1.5mm))
    (condition "A.Reference =='TP*' && B.Reference == 'TP*'"))
```

High-current design rules

```
# Check current-carrying capacity
(rule high-current
    (constraint track_width (min 1.0mm))
    (constraint connection_width (min 0.8mm))
    (condition "A.NetClass == 'Power'"))

# Don't use thermal reliefs on heatsink pads
(rule heat_sink_pad
    (constraint zone_connection solid)
    (condition "A.Fabrication_Property == 'Heatsink pad'"))

# Require all four thermal relief spokes to connect to parent zone
(rule fully_spoked_pads
    (constraint min_resolved_spokes 4))

# Set thermal relief gap & spoke width for all zones
(rule defined_relief
    (constraint thermal_relief_gap (min 10mil))
    (constraint thermal_spoke_width (min 12mil)))

# Override thermal relief gap & spoke width for GND and PWR zones
(rule defined_relief_pwr
    (constraint thermal_relief_gap (min 10mil))
    (constraint thermal_spoke_width (min 12mil))
    (condition "A.Name == 'zone_GND' || A.Name == 'zone_PWR'"))

# Prevent solder wicking from SMD pads
(rule holes_in_pads
    (constraint physical_hole_clearance (min 0.2mm))
    (condition "B.Pad_Type == 'SMD'"))

# Disallow solder mask margin overrides
(rule "disallow solder mask margin overrides"
    (constraint assertion "A.Soldermask_Margin_Override == 0mm")
    (condition "A.Type == 'Pad'"))
```

Hole sizes

```
(rule "Max Drill Hole Size Mechanical"
    (constraint hole_size (max 6.3mm))
    (condition "A.Pad_Type == 'NPTH, mechanical'"))

(rule "Max Drill Hole Size PTH"
    (constraint hole_size (max 6.35mm))
    (condition "A.Pad_Type == 'Through-hole'"))

# Separate drill bit and milling cutter size constraints
(rule "Plated through-hole size"
    (constraint hole_size (min 0.2mm) (max 6.35mm))
    (condition "A.isPlated() && A.Hole_Size_X == A.Hole_Size_Y"))

(rule "Plated slot size"
    (constraint hole_size (min 0.5mm))
    (condition "A.isPlated() && A.Hole_Size_X != A.Hole_Size_Y"))
```

Scripting

Scripting allows you to automate tasks within KiCad using the Python language. KiCad provides an API for editing PCBs that can be used interactively or in standalone scripts. Board Editor scripts can be organized as "action plugins", which are displayed as icons in the top toolbar of the Board Editor. There is also a separate Footprint Wizard API that can be used to create footprint creation plugins for the Footprint Editor.

This manual covers general scripting concepts for the Board Editor's `pcbnew` API as well as for the footprint wizard API. Users wishing to write or modify scripts should also use the Doxygen documentation for these APIs located at https://docs.kicad.org/doxygen-python-8.0/namespaces.html.

KiCad 6 or newer requires Python 3 for scripting support. Python 2 is no longer supported.

Using the scripting console

The PCB Editor comes with a built-in Python console that can be used to inspect and interact with the board. To launch the console, use the 🖥 button in the top toolbar. The PCB Editor Python API is not automatically loaded, so to load it, type `import pcbnew` into the console. The command `pcbnew.GetBoard()` will then return a reference to the board currently loaded in the PCB Editor, which can be inspected and modified through the console.

Python script locations

Plugin scripts (PCB action plugins and footprint wizards) can be installed automatically using the Plugin and Content Manager (PCM), or manually by copying the plugin to a folder. Manually installed plugins should each be in their own folder within the `plugins` folder. The location of the `plugins` folder is by default:

Platform	Path
Linx	`~/.local/share/kicad/8.0/scripting/plugins`
macOS	`~/Documents/KiCad/8.0/scripting/plugins`
Windows	`%HOME%\Documents\KiCad\8.0\scripting\plugins`

NOTE The type of plugin is determined by the Python class it inherits from. Inheriting from `FootprintWizardBase.FootprintWizard` will create a footprint wizard plugin, and inheriting from `pcbnew.ActionPlugin` will create an action plugin. Creating action plugins and footprint wizards is described in more detail below.

`pcbnew` API overview

The scripting API reflects the internal object structure inside KiCad's Board Editor. It is provided by the `pcbnew` module in Python.

NOTE Because the API is tightly coupled to KiCad's internals, the API will change over time and is not considered stable. Consult the doxygen documentation for the most up-to-date API reference, and be sure to use the documentation for the appropriate version of KiCad.

Scripts, action plugins, and interactive scripting sessions often start with a call to `GetBoard()`, which returns a `BOARD` object representing the currently open board and its contents.

`BOARD` has a set of properties and a set for each type of object in the board: footprints, zones, tracks, vias, text, etc. Each type of object has its own properties and holds its own objects: a footprint will likely have at least one pad, for example.

The objects in the `BOARD` can be accessed using methods that each return an iterable list of the corresponding object type. A selection of these methods are listed below. Other methods are listed in the doxygen documentation.

- `board.GetFootprints()`: returns a list of all of the footprints in the board.

- `board.GetDrawings()`: returns a list of miscellaneous board objects in the board.

- `board.GetTracks()`: returns a list of all of the tracks and vias in the board.

- `board.GetZones()`: returns a list of all of the zones in the board.

- `board.GetNetClasses()`: returns a list of all net classes in the board's design rules.

Boards can be loaded and saved from disk using the following functions:

- `LoadBoard(filename)`: loads a board from file, returning a `BOARD` object, using the file format that matches the filename extension.

- `SaveBoard(filename, board)`: saves a `BOARD` object to file, using the file format that matches the filename extension.

- `board.Save(filename)`: the same as `SaveBoard()`, but a method of the `BOARD` object.

Ejemplos

Load a board, hide all values, and show all references.

```python
#!/usr/bin/env python3
import sys
from pcbnew import LoadBoard

filename = sys.argv[1]

pcb = LoadBoard(filename)
for fp in pcb.GetFootprints():
    print(f"* Footprint: {fp.GetReference()}")
    fp.Value().SetVisible(False)      # set Value as Hidden
    fp.Reference().SetVisible(True)   # set Reference as Visible

pcb.Save("mod_" + filename)
```

Change the paste mask margin for pins 1-14 of a footprint.

```python
#!/usr/bin/env python3
import sys
from pcbnew import *

filename=sys.argv[1]
pcb = LoadBoard(filename)

# Find module U304
u304 = pcb.FindFootprintByReference('U304')
pads = u304.Pads()

#  Iterate over pads, printing solder paste margin
for p in pads:
    print(p.GetPadName(), ToMM(p.GetLocalSolderPasteMargin()))
    id = int(p.GetPadName())
    # Set margin to 0 for all but pad (pin) 15
    if id<15: p.SetLocalSolderPasteMargin(0)

pcb.Save("mod_"+filename)
```

Load a footprint library, list footprints in the library, and list pads in each footprint.

```python
#!/usr/bin/env python3
from pcbnew import *

libpath = "/usr/share/kicad/footprints/Connector_PinSocket_2.54mm.pretty"
print(f">> enumerate footprints, pads of {libpath}")

# Load the suitable plugin to read/write the .pretty library
src_type = PCB_IO_MGR.GuessPluginTypeFromLibPath( libpath );
# We can force the plugin type by using IO_MGR.PluginFind( IO_MGR.KICAD )
plugin = PCB_IO_MGR.PluginFind( src_type )

# Print plugin type name: (Expecting "KiCad" for a .pretty library)
print(f"Selected plugin type: {PCB_IO_MGR.ShowType(src_type)}")

list_of_footprints = plugin.FootprintEnumerate(libpath)

for name in list_of_footprints:
    fp = plugin.FootprintLoad(libpath,name)
    # print the short name of the footprint
    print(name)
    # followed by ref field, value field, and decription string:
    # Remember ref and value texts are dummy text, replaced by the schematic values
    # when reading a netlist.
    print(f"  -> {fp.GetReference()} {fp.GetValue()} {fp.GetLibDescription()}")
    for pad in fp.Pads():
        print(
            f"     pad [{pad.GetPadName()}] at "
            f"pos ({ToMM(pad.GetPosition().x)}, {ToMM(pad.GetPosition().y)}) mm,",
            f"shape offset ({ToMM(pad.GetOffset().x)}, {ToMM(pad.GetOffset().y)}) mm"
        )
    print()
```

Load a board and print information about each item in the board.

```python
#!/usr/bin/env python
import sys
from pcbnew import *

filename=sys.argv[1]
pcb = LoadBoard(filename)

print("Listing Tracks and Vias:")
for item in pcb.GetTracks():
    if type(item) is PCB_VIA:
        pos = item.GetPosition()
        drill = item.GetDrillValue()
        width = item.GetWidth()
        print(f" * Via:    {ToMM(pos)} - {ToMM(drill)}/{ToMM(width)}")
    elif type(item) is PCB_TRACK:
        start = item.GetStart()
        end = item.GetEnd()
        width = item.GetWidth()
        print(f" * Track: {ToMM(start)} to {ToMM(end)}, width {ToMM(width)}")
    else:
        print(f"Unknown type    {type(item)}")

print()
print("Listing Text and Shapes:")
for item in pcb.GetDrawings():
    if type(item) is PCB_TEXT:
        print(f"* Text:    '{item.GetText()}' at {ToMM(item.GetPosition())}")
    elif type(item) is PCB_SHAPE:
        print(f"* Drawing: {item.GetShapeStr()}")
    else:
        print(f"Unknown type    {type(item)}")

print()
print("Listing Footprints")
for fp in pcb.GetFootprints():
    print(f"* Footprint: {fp.GetReference()} at {ToMM(fp.GetPosition())}")

print()
print(f"Ratsnest count: {pcb.GetNetCount()}")
print(f"Track width count: {len(pcb.GetTrackWidthList())}")
print(f"Via size count: {len(pcb.GetViasDimensionsList())}")

print()
print(f"Listing Zones: {pcb.GetAreaCount()}")
for idx in range(0, pcb.GetAreaCount()):
    zone = pcb.GetArea(idx)
    print(f"zone: {idx} priority: {zone.GetAssignedPriority()} netname:
{zone.GetNetname()}")

print()
print(f"Netclasses: {len(pcb.GetAllNetClasses())}")
```

Action plugins

Action plugin associate a script with a button in the PCB Editor GUI. Clicking the button runs the script. Action plugins are shown in the **Tools → External plugins** menu, and can also be shown in the toolbar if enabled in the **Action Plugins** page of the **Preferences** dialog.

The example below is an action plugin that uses KiCad's `pcbnew` API to replace the string `$date$` with the current date in any text item.

```
import pcbnew
import re
import datetime

class text_by_date(pcbnew.ActionPlugin):
    """
    test_by_date: A sample plugin as an example of ActionPlugin
    Add the date to any text field of the board containing '$date$'
    How to use:
    - Add a text on your board with the content '$date$'
    - Call the plugin
    - The text will automatically be updated with the date (format YYYY-MM-DD)
    """

    def defaults(self):
        """
        Method defaults must be redefined
        self.name should be the menu label to use
        self.category should be the category (not yet used)
        self.description should be a comprehensive description
          of the plugin
        """
        self.name = "Add date on PCB"
        self.category = "Modify PCB"
        self.description = "Automatically add date on an existing PCB"

    def Run(self):
        pcb = pcbnew.GetBoard()
        for item in pcb.GetDrawings():
            if item.GetClass() == "PCB_TEXT":
                txt = re.sub("\$date\$ [0-9]{4}-[0-9]{2}-[0-9]{2}",
                             "$date$", item.GetText())
                if txt == "$date$":
                    item.SetText("$date$ %s" % datetime.date.today())


text_by_date().register()
```

Footprint wizards

Footprint wizards are Python scripts that can be accessed from the Footprint Editor. Each footprint wizard presents a selection of parameters defined in the Python script, and creates a footprint based on the parameter values.

There are 3 minimum steps required to create a footprint wizard, which are described below. For examples of how to create footprint wizards, see the footprint wizards included with KiCad.

1. Instantiate a Python class, inheriting from `FootprintWizardBase.FootprintWizard`.

2. Define the 6 required functions: `GetName()`, `GetDescription()`, `GetValue()`, `GenerateParameterList()`, `CheckParameters()`, and `BuildThisFootprint()`.

3. Register the class by calling `{your_class_name}().register()`.

The `GetName()`, `GetDescription()`, and `GetValue()` functions are there to provide strings to the UI. The only functionality needed is to return an appropriate string.

The `GenerateParameterList()` function defines the parameters needed for the footprint. Parameters are grouped into a `page` + `name` format. For example, calling `self.AddParam("demo", "radius", self.uMM, 5)` would add a parameter named radius into the page named demo. Retrieving that parameter data would be done with a call such as `self.footprint_radius = self.parameters["demo"]["radius"]`.

The `CheckParameters()` function is available to perform any data validation on the parameters defined in `GenerateParameterList()`. This function is also where the `self.footprint_radius = self.parameters["demo"]["radius"]` calls reside.

The `BuildThisFootprint()` function is where the footprint building steps are called. This function is where one creates the footprint.

The required `{your_class_name}().register()` call can either be at the end of the Python file, or in an `__init__.py` file. Both styles are supported by KiCad.

| NOTE | KiCad will not reload a plugin after it has raised an error (for example, the `NotImplementedError`). One will need completely close out KiCad and restart it. However, this doesn't apply to changes which do not raise an error. |

IDF component outlines

KiCad can export an IDF representation of the board for use in mechanical CAD software. Below is some guidance on attaching IDF component outlines to footprints, creating new IDF component outlines, and a description of the IDF utilities included with KiCad.

Specifying component models for use by the exporter

IDF component models are attached to footprints using the footprint's 3D model properties. The IDF exporter uses different filetypes than the 3D viewer and other 3D model exporters, so adding 3D models for the IDF exporter does not conflict with 3D models added to a footprint for other purposes.

To add an IDF model to a footprint in the footprint or PCB editors, edit the footprint's properties and click on the 3D Models tab.
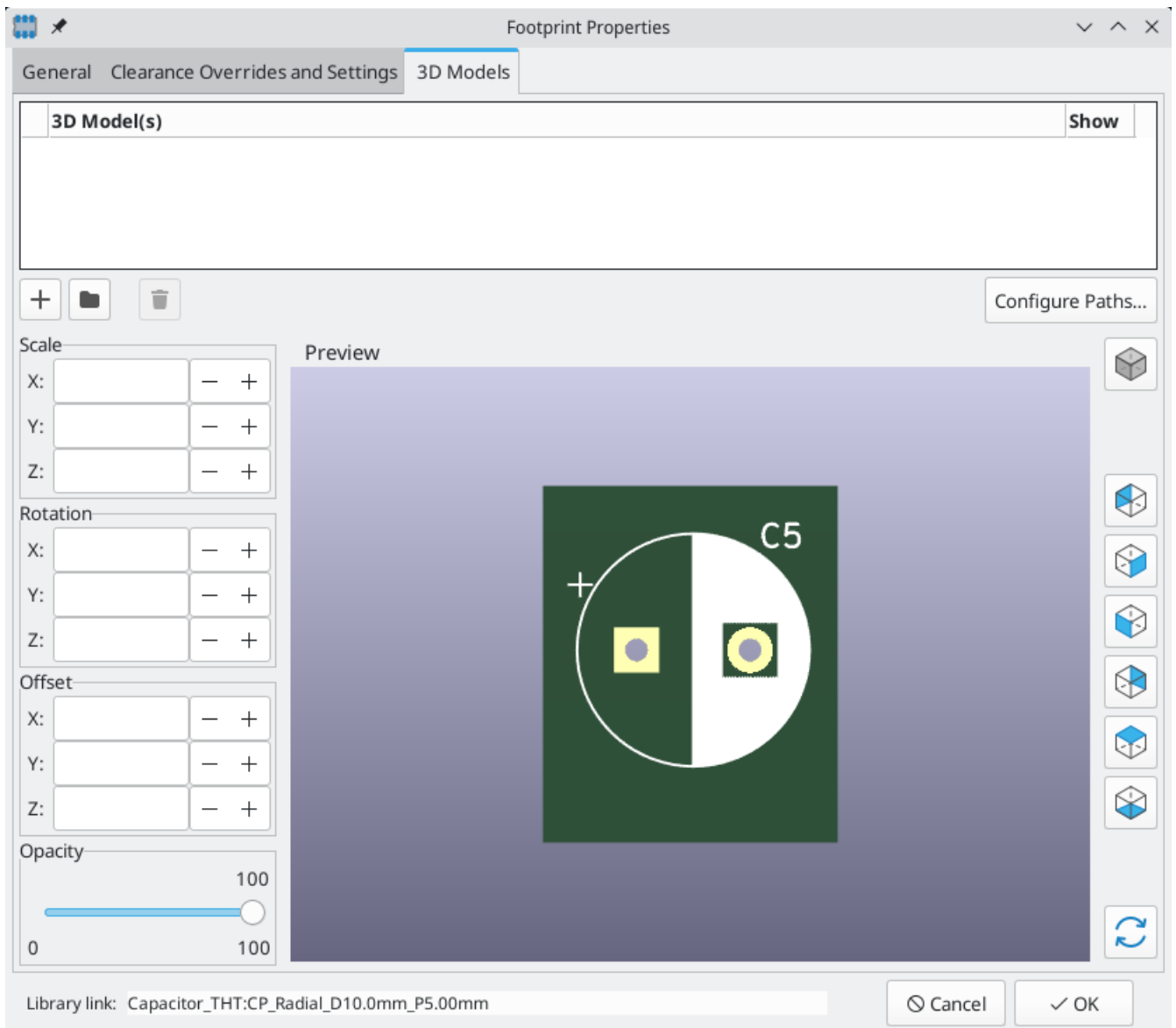
Click the ▪ button and select the **IDF (*.idf;*.IDF)** filetype filter. Browse to the desired outline file.
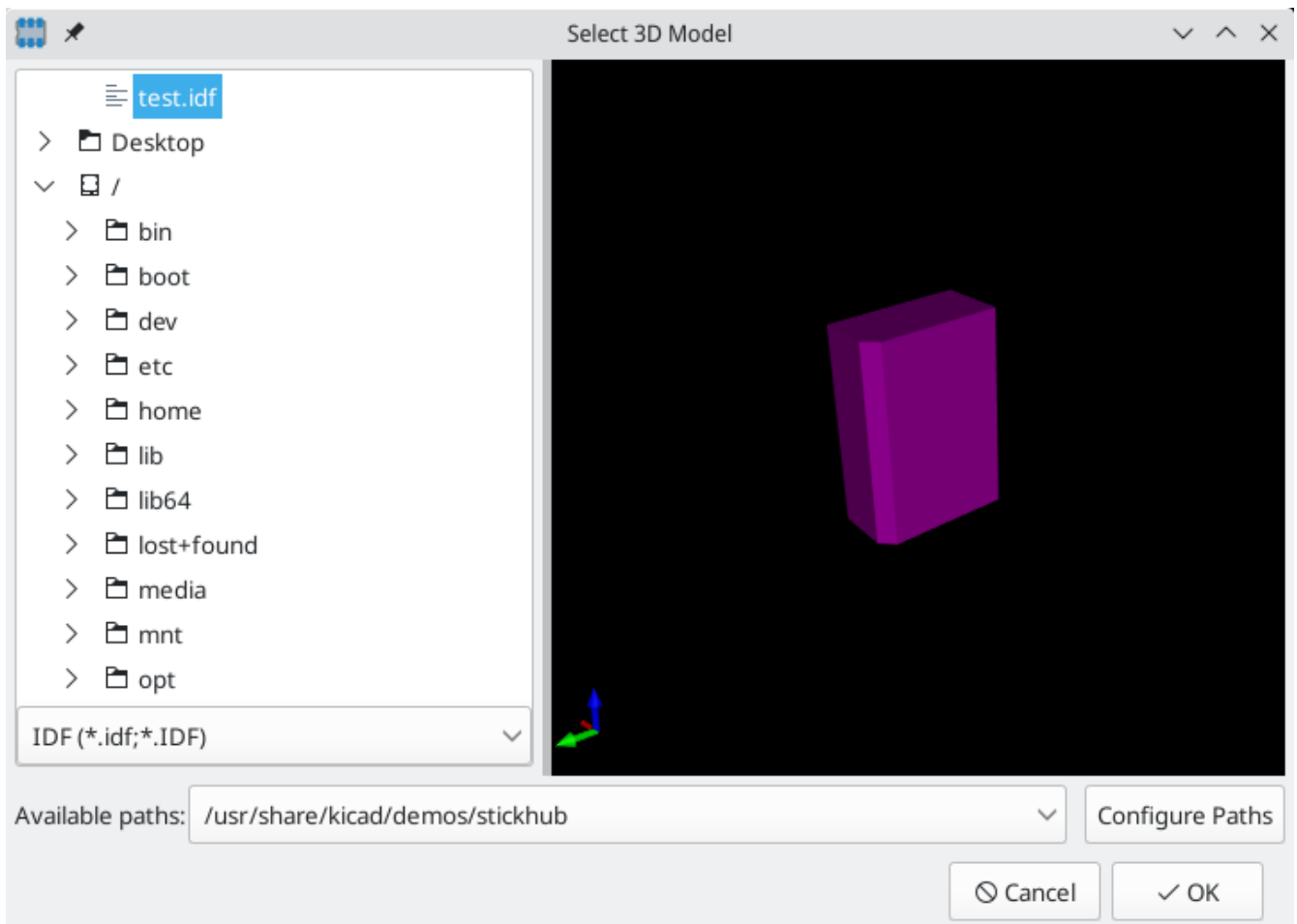
Once the desired component outline file is selected, enter any necessary values for the offset and rotation. The offsets must be specified using the IDF board output units (mm or mils) and in the IDF coordinate system, which is a right-hand coordinate system with +Z pointing towards the viewer, +X to the viewer's right, and +Y towards the upper edge of the screen. The rotation must be in degrees; positive rotation is a counter-clockwise rotation as described in the IDFv3 specification.

Multiple outlines may be combined with appropriate offsets to represent simple assemblies such as a DIP package in a socket.

NOTE	Only the offset values and the Z rotation value are used by the IDF exporter; all other values are ignored.

Creating a component outline file

The component outline file (`*.idf`) consists of a single `.ELECTRICAL` or `.MECHANICAL` section as described in the specification document. The section may be preceded by any number of comment lines; the comment lines are copied by the exporter into the library file and can be used to track metadata such as references to the documents used to determine the component's outline and dimensions.

The component outline section contains fields which are strings, integers, or floating point numbers. A string is a combination of characters which may include spaces; if a string contains spaces then it must be quoted. Quotation marks must not appear within a string. Floating point numbers may be represented using decimal or exponential notations but decimal notation is preferred for human readability. The decimal point must be a dot and not a comma. The IDF file must consist only of 7-bit ASCII characters; use of 8-bit characters will result in undefined behavior.

An IDF file consists of SECTIONS which consist of RECORDS which consist of FIELDS. For the IDF outline files only one type of section may exist and must be one of `.ELECTRICAL` or `.MECHANICAL`. A record is a single line of text and may contain one or more fields. Fields are sequences of characters separated by one or more spaces which do not appear between quotation marks. All fields of a record must appear on a single line; records may not span lines.

The section heading (`.ELECTRICAL` or `.MECHANICAL`) is considered the first record (Record 1) of the section. Record 1 must be followed by Record 2 which has four fields:

1. Geometry Name: a string which in combination with the Part Number must form a unique identifier for the component outline. For standardized packages, the package name is a good value for the geometry name, for example "SOT-23". For unique packages the manufacturer's part number is a good choice for the geometry name.

2. Part Number: although obviously intended for the part number, for example BS107, it is better to use this string to help describe the package. For example if the geometry name is "TO-92", the part number entry may be used to describe the layout of the pads or the orientation of this particular TO-92 outline file.

3. IDF Unit: this must be one of `MM` or `THOU` and it applies only to the units describing this single component outline.

4. Height: this is a floating point number representing the nominal height of the component using units specified in Field 3.

Record 2 must be followed by a number of Record 3 entries which specify the outline of the component. Record 3 consists of four fields:

1. Loop Index: `0` (outline points are specified in counter-clockwise order) or `1` (outline points are specified in clockwise order)

2. X coordinate: a floating point number

3. Y coordinate: a floating point number

4. Included Angle: a floating point number. If the value is `0` then a straight line segment is drawn from the previous point to this point. If the value is `360` then the previous point specifies the center of a circle and this point specifies a point on the circle; never specify a circle using a value of `-360` as at least one major mechanical CAD package does not behave well in that situation. If the value is negative then a clockwise arc is drawn from the previous point to this point and if the value is positive then a counter-clockwise arc is drawn.

Only one closed loop is permitted and it is not possible to specify a cutout. The last point specified must be the same as the first point unless the outline is a circle.

Example IDF File 1:

```
# a simple cylinder - this could represent an electrolytic capacitor
.ELECTRICAL
    "cylinder" "5mm OD, 5mm height" MM 5
    0 0 0 0
    0 2.5 0 360
.END_ELECTRICAL
```

Example IDF File 2:

```
# an upside-down T
# a comment added for the sake of adding comments
.ELECTRICAL
    "Capital T" "5x8x10mm, upside down" MM 10
    0 -0.5 8 0
    0 -0.5 0.5 0
    0 -2.5 0.5 0
    0 -2.5 -0.5 180
    0 2.5 -0.5 0
    0 2.5 0.5 180
    0 0.5 0.5 0
    0 0.5 8 0
    0 -0.5 8 180
.END_ELECTRICAL
```

Guidelines for creating outlines

When creating outlines, and especially when sharing the work with others, consistency in the design and naming of files helps people locate files quicker and place the components with minimal hassles.

Package naming

Try to make some information about the outline available in the filename to give the user a general idea of what the outline is. For example axial leaded cylindrical packages may represent some types of capacitors as well as some types of resistors, so it makes sense to identify an outline as a horizontal or vertical axial leaded device and to add some extra information on the relevant dimensions: diameter, length, and pitch are the most important. If a device has a unique outline, the manufacturer's part number and a prefix to indicate the class of device are adequate.

Comments

Use comments in the IDF file to give users more information about the outline, for example a reference to the source used for dimensional information.

Geometry and Part Number entries

Think carefully about the values to give to the Geometry and Part Number entries. Taken together, these strings act as a unique identifier for the MCAD system. The values of the strings will ideally have some meaning to a user, but this is not necessary: the values are primarily intended for the MCAD system to use as a unique ID. Ideally the values chosen will be unique within any large collection of outlines; choosing values well will result in fewer clashes especially in complex boards.

Pin orientation and positioning

Component outlines should be created to match the orientation and position of the corresponding footprints. This avoids the need to specify a non-zero rotation for the IDF component outline. Since the IDF exporter ignores the (X, Y) offset values, it is vital that you use the correct origin in the IDF component outline.
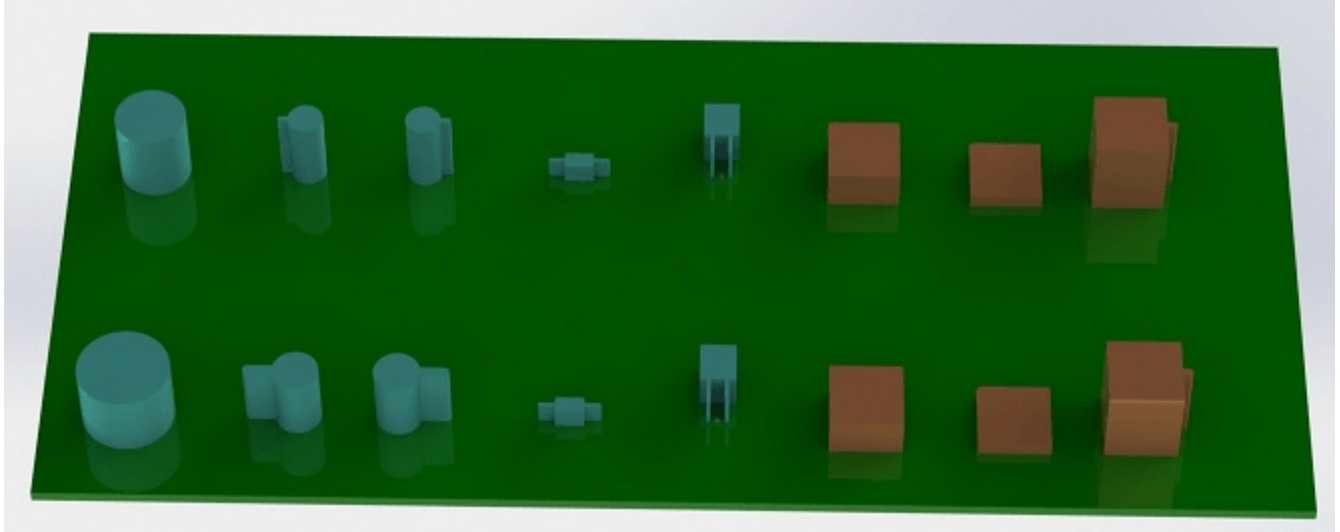
The image above shows sample outlines generated by the programs `idfcyl` and `idfrect` and rendered in a mechanical CAD program. From left to right are (a) vertical radial leaded cylinder, (b) vertical axial leaded cylinder with wire on left, (c) vertical axial leaded cylinder with wire on right, (d) horizontal axial leaded cylinder, (e) horizontal radial leaded cylinder, (f) square outline, plain, (g) square outline with chamfer, (h) square outline with axial lead on right. The top outlines were specified in units of millimeters while the bottom outlines were specified in units of inches.

Tips on dimensions

The purpose served by the extruded outlines is to give the mechanical designer some idea of the location and physical space occupied by each component. In a typical scenario the mechanical designer will replace some of the crude outlines with more detailed mechanical models, for example when checking to ensure that a right-angle mounted LED will fit into a hole on a panel. In most situations the accuracy of an outline doesn't matter, but it is good practice to create outlines which convey the best mechanical information possible. In a few instances a user may wish to fit the component into a case with very little excess space, for example in a portable music player. In such a situation, if most extruded outlines are a good enough representation of components then the mechanical designer may only have to replace very few models while designing the case. If the outlines are not a reliable reflection of reality then the mechanical designer will waste a lot of time replacing models to ensure a good fit. After all, if you put garbage in you can expect garbage to come out. If you put in good information, you can be confident of good results.

IDF Component Outline Tools

A number of command-line tools are available to help generate IDF component outlines. The tools are:

1. `idfcyl:` creates an outline of a cylinder in vertical or horizontal orientation and with axial or radial leads

2. `idfrect:` creates an outline of a rectangle which may have either an axial lead or a chamfer in the top left corner

3. `dxf2idf:` converts a drawing in DXF format into an IDF component outline

idfcyl

`idfcyl` generates outlines for cylindrical components.

When `idfcyl` is invoked with no arguments it prints out a usage note and a summary of its inputs:

```
idfcyl: This program generates an outline for a cylindrical component.
    The cylinder may be horizontal or vertical.
    A horizontal cylinder may have wires at one or both ends.
    A vertical cylinder may have at most one wire which may be
    placed on the left or right side.

Input:
    Unit: mm, in (millimeters or inches)
    Orientation: V (vertical)
    Lead type: X, R (axial, radial)
    Diameter of body
    Length of body
    Board offset
    *   Wire diameter
    *   Pitch
    **  Wire side: L, R (left, right)
    *** Lead length
    File name (must end in *.idf)

    NOTES:
        *   only required for horizontal orientation or
            vertical orientation with axial leads

        **  only required for vertical orientation with axial leads

        *** only required for horizontal orientation with radial leads
```

The notes can be suppressed by entering any arbitrary argument on the command line. A user can manually enter information at the command line or create scripts to generate outlines. The following script creates a single cylinder axial leaded outline with the lead on the right hand side:

```
#!/bin/bash
# Generate a cylindrical IDF outline for test purposes
# vertical 5mm cylinder,  nominal length 8mm + 3mm board offset,
# axial wire on right,  0.8mm wire dia., 3.5mm pitch
idfcyl - 1 > /dev/null <<  _EOF
mm
v
x
5
8
3
0.8
3.5
r
cylvmm_1R_D5_L8_Z3_WD0.8_P3.5.idf
_EOF
```

idfrect

`idfrect` generates outlines for rectangular components.

When `idfrect` is invoked with no arguments it prints out a usage note and a summary of its inputs:

```
idfrect: This program generates an outline for a rectangular component.
    The component may have a single lead (axial) or a chamfer on the
    upper left corner.
Input:
    Unit: mm, in (millimeters or inches)
    Width:
    Length:
    Height:
    Chamfer: length of the 45 deg. chamfer
    *  Leaded: Y,N (lead is always to the right)
    ** Wire diameter
    ** Pitch
    File name (must end in *.idf)

    NOTES:
        *   only required if chamfer = 0

        **  only required for leaded components
```

The notes can be suppressed by entering any arbitrary argument on the command line. A user can manually enter information at the command line or create scripts to generate outlines. The following script creates a chamfered rectangle and an axial leaded outline:

```
#!/bin/bash
# Generate various rectangular IDF outlines for test purposes
# 10x10, 1mm chamfer, 2mm height
idfrect - 1 > /dev/null <<  _EOF
mm
10
10
2
1
rectMM_10x10x2_C0.5.idf
_EOF
# 10x10x12,  0.8mm lead on 6mm pitch
idfrect - 1 > /dev/null <<  _EOF
mm
10
10
12
0
Y
0.8
6
rectLMM_10x10x12_D0.8_P6.0.idf
_EOF
```

dxf2idf

`dxf2idf` creates an IDF component file from a DXF outline.

The DXF file used to specify the component outline can be prepared with the free software LibreCAD for best compatibility.

When `dxf2idf` is invoked with no arguments it prints out a usage note and a summary of its inputs:

```
dxf2idf: this program takes line, arc, and circle segments
    from a DXF file and creates an IDF component outline file.

Input:
    DXF filename: the input file, must end in '.dxf'
    Units: mm, in (millimeters or inches)
    Geometry Name: string, as per IDF version 3.0 specification
    Part Name: as per IDF version 3.0 specification of Part Number
    Height: extruded height of the outline
    Comments: all non-empty lines are comments to be added to
        the IDF file. An empty line signifies the end of
        the comment block.
    File name: output filename, must end in '.idf'
```

The notes can be suppressed by entering any arbitrary argument on the command line. A user can manually enter information at the command line or create scripts to generate outlines. The following script creates a 5mm high outline from a DXF file `test.dxf`:

```bash
#!/bin/bash
# Generate an IDF outlines from a DXF file
dxf2idf - 1 > /dev/null << _EOF
test.dxf
mm
DXF TEST GEOMETRY
DXF TEST PART
5
This is an IDF test file produced from the outline 'test.dxf'
This is a second IDF comment to demonstrate multiple comments

test_dxf2idf.idf
_EOF
```

idf2vrml

The `idf2vrml` tool reads a set of one IDF Board (`.emn`) and one IDF Component file (`.emp`) and produces a VRML file which can be viewed with a VRML viewer. This feature is useful for visualization of the board assembly in cases where the user does not have access to MCAD software. Invoking `idf2vrml` without any arguments will result in the display of a usage message:

```
>./idf2vrml
Usage: idf2vrml -f input_file.emn -s scale_factor {-k} {-d} {-z} {-m}
flags:
   -k: produce KiCad-friendly VRML output; default is compact VRML
   -d: suppress substitution of default outlines
   -z: suppress rendering of zero-height outlines
   -m: print object mapping to stdout for debugging purposes
example to produce a model for use by KiCad: idf2vrml -f input.emn -s 0.3937008 -k
```

NOTE	The `idf2vrml` tool does not correctly render `OTHER_OUTLINE` entities in an `emn` file if that entity is specified on the back layer of the PCB; however you will not noticeable using files exported by KiCad because there is no mechanism to specify such an entity. This is only an issue if you render a third party emn file which does employ an entity on the back side of a board.

Actions reference

Below is a list of every available **action** in the PCB Editor: a command that can be assigned to a hotkey.

PCB Editor

The actions below are available in the PCB Editor. Hotkeys can be assigned to any of these actions in the **Hotkeys** section of the preferences.

Action	Default Hotkey	Description
Align to Bottom		Aligns selected items to the bottom edge
Align to Horizontal Center		Aligns selected items to the horizontal center
Align to Vertical Center		Aligns selected items to the vertical center
Align to Left		Aligns selected items to the left edge
Align to Right		Aligns selected items to the right edge
Align to Top		Aligns selected items to the top edge
Distribute Horizontally		Distributes selected items along the horizontal axis
Distribute Vertically		Distributes selected items along the vertical axis
Place Off-Board Footprints		Performs automatic placement of components outside board area
Place Selected Footprints		Performs automatic placement of selected components
Flip Board View		View board from the opposite side
Sketch Graphic Items		Show graphic items in outline mode
Decrease Layer Opacity	`{`	Make the current layer more transparent
Increase Layer Opacity	`}`	Make the current layer less transparent
Switch to Copper (B.Cu) Layer	`PgDn`	Switch to Copper (B.Cu) layer

Action	Default Hotkey	Description
Switch to Inner Layer 1		Switch to Inner layer 1
Switch to Inner Layer 10		Switch to Inner layer 10
Switch to Inner Layer 11		Switch to Inner layer 11
Switch to Inner Layer 12		Switch to Inner layer 12
Switch to Inner Layer 13		Switch to Inner layer 13
Switch to Inner Layer 14		Switch to Inner layer 14
Switch to Inner Layer 15		Switch to Inner layer 15
Switch to Inner Layer 16		Switch to Inner layer 16
Switch to Inner Layer 17		Switch to Inner layer 17
Switch to Inner Layer 18		Switch to Inner layer 18
Switch to Inner Layer 19		Switch to Inner layer 19
Switch to Inner Layer 2		Switch to Inner layer 2
Switch to Inner Layer 20		Switch to Inner layer 20
Switch to Inner Layer 21		Switch to Inner layer 21
Switch to Inner Layer 22		Switch to Inner layer 22
Switch to Inner Layer 23		Switch to Inner layer 23
Switch to Inner Layer 24		Switch to Inner layer 24

Action	Default Hotkey	Description
Switch to Inner Layer 8		Switch to Inner layer 8
Switch to Inner Layer 9		Switch to Inner layer 9
Switch to Next Layer	`+`	Switch to Next Layer
Switch to Previous Layer	`-`	Switch to Previous Layer
Toggle Layer	`V`	Switch between layers in active layer pair
Switch to Component (F.Cu) layer	`PgUp`	Switch to Component (F.Cu) layer
Net Inspector		Show the net inspector
Local Ratsnest		Toggle ratsnest display of selected item(s)
Net Color Mode (3-state)		Cycle between using net and netclass colors for all nets, just ratsnests, and none
Sketch Pads		Show pads in outline mode
Curved Ratsnest Lines		Show ratsnest with curved lines
Ratsnest Mode (3-state)		Cycle between showing ratsnests for all layers, just visible layers, and none
Repair Board		Run various diagnostics and attempt to repair board
Show Appearance Manager		Show/hide the appearance manager
Show Pad Numbers		Show pad numbers
Scripting Console		Show the Python scripting console
Show Ratsnest		Show board ratsnest
Sketch Text Items		Show footprint texts in line mode
Sketch Tracks	`K`	Show tracks in outline mode
Sketch Vias		Show vias in outline mode

Action	Default Hotkey	Description
Create Polygon from Selection...		Creates a graphic polygon from the selection
Create Tracks from Selection		Creates tracks from the selected graphic lines
Create Zone from Selection...		Creates a copper zone from the selection
Design Rules Checker		Show the design rules checker window
Open in Footprint Editor	`Ctrl` + `E`	Opens the selected footprint in the Footprint Editor
Edit Library Footprint...	`Ctrl` + `Shift` + `E`	Opens the selected footprint in the Footprint Editor
Append Board...		Open another board and append its contents to this board
Assign Netclass...		Assign a netclass to nets matching a pattern
Board Setup...		Edit board setup including layers, design rules and various defaults
Clear Net Highlighting	`~`	Clear any existing net highlighting
Drill/Place File Origin		Place origin point for drill files and component placement files
Reset Drill Origin		
Export Specctra DSN...		Export Specctra DSN routing info
Bill of Materials...		Create bill of materials from board
IPC-D-356 Netlist File...		Generate IPC-D-356 netlist file
Drill Files (.drl)...		Generate Excellon drill file(s)
Gerbers (.gbr)...		Generate Gerbers for fabrication
IPC-2581 File (.xml)...		Generate an IPC-2581 file
Component Placement (.pos, .gbr)...		Generate component placement file(s) for pick and place

Action	Default Hotkey	Description
Highlight Net	`` ` ``	Highlight net under cursor
Highlight Net		Highlight all copper items on the selected net(s)
Import Netlist...		Read netlist and update board connectivity
Import Specctra Session...		Import routed Specctra session (*.ses) file
Lock		Prevent items from being moved and/or resized on the canvas
Add Footprint	`A`	Add a footprint
Remove Items		Remove items from group
Switch to Schematic Editor		Open schematic in schematic editor
Show Net in Ratsnest		Show the selected net in the ratsnest of unconnected net lines/arcs
Constrain to H, V, 45	`Shift` + `Space`	Limit actions to horizontal, vertical, or 45 degrees from the starting point
Toggle Last Net Highlight		Toggle between last two highlighted nets
Toggle Lock	`L`	Lock or unlock selected items
Toggle Net Highlight	`Alt` + `` ` ``	Toggle net highlighting
Switch Track Width to Previous	`Shift` + `W`	Change track width to previous pre-defined size
Switch Track Width to Next	`W`	Change track width to next pre-defined size
Ungroup Items		Ungroup any selected groups
Unlock		Allow items to be moved and/or resized on the canvas
Decrease Via Size	kbd:[\]	Change via size to previous pre-defined size
Increase Via Size	`'`	Change via size to next pre-defined size
Duplicate Zone onto Layer...		Duplicate zone outline onto a different layer
Merge Zones		Merge zones

Action	Default Hotkey	Description
Generators Manager		Show a manager dialog for Generator objects
Change Footprint...		Assign a different footprint from the library
Change Footprints...		Assign different footprints from the library
Cleanup Graphics...		Cleanup redundant items, etc.
Cleanup Tracks & Vias...		Cleanup redundant items, shorting items, etc.
Edit Teardrops...		Add, remove or edit teardrops globally across board
Edit Text & Graphics Properties...		Edit Text and graphics properties globally across board
Edit Track & Via Properties...		Edit track and via properties globally across board
Global Deletions...		Delete tracks, footprints and graphic items from board
Remove Unused Pads...		Remove or restore the unconnected inner layers on through hole pads and vias
Swap Layers...		Move tracks or drawings from one layer to another
Update Footprint...		Update footprint to include any changes from the library
Update Footprints from Library...		Update footprints to include any changes from the library
Compare Footprint with Library		Show differences between board footprint and its library equivalent
Clearance Resolution		Show clearance resolution for the active layer between two selected objects
Constraints Resolution		Show constraints resolution for the selected object
Show Board Statistics		Shows board statistics

Action	Default Hotkey	Description
Close Outline		Close the in progress outline
Decrease Line Width	`Ctrl` + `-`	Decrease the line width
Delete Last Point	`Back`	Delete the last point added to the current item
Draw Graphic Polygon	`Ctrl` + `Shift` + `P`	Draw a graphic polygon
Increase Line Width	`Ctrl` + `+`	Increase the line width
Add Leader		Add a leader dimension
Draw Line	`Ctrl` + `Shift` + `L`	Draw a line
Add Orthogonal Dimension		Add an orthogonal dimension
Add Board Characteristics		Add a board characteristics table on a graphic layer
Import Graphics...	`Ctrl` + `Shift` + `F`	Import 2D drawing file
Add Reference Image		Add a bitmap image to be used as a reference (image will not be included in any output)
Add Stackup Table		Add a board stackup table on a graphic layer
Add Radial Dimension		Add a radial dimension
Draw Rectangle		Draw a rectangle
Add Rule Area	`Ctrl` + `Shift` + `K`	Add a rule area (keepout)
Place the Footprint Anchor	`Ctrl` + `Shift` + `N`	Set the coordinate origin point (anchor) of the footprint
Add a Similar Zone	`Ctrl` + `Shift` + `.`	Add a zone with the same settings as an existing zone
Add Text	`Ctrl` + `Shift` + `T`	Add a text item
Add Text Box		Add a wrapped text item

Action	Default Hotkey	Description
Add Filled Zone	`Ctrl` + `Shift` + `Z`	Add a filled zone
Add a Zone Cutout	`Shift` + `C`	Add a cutout area of an existing zone
Get and Move Footprint	`T`	Selects a footprint by reference designator and places it under the cursor for moving
Chamfer Lines		Cut away corners between selected lines
Change Track Width		Updates selected track & via sizes
Create Array...	`Ctrl` + `T`	Create array
Delete Full Track	`Shift` + `Del`	Deletes selected item(s) and copper connections
Duplicate and Increment	`Ctrl` + `Shift` + `D`	Duplicates the selected item(s), incrementing pad numbers
Extend Lines to Meet		Extend lines to meet each other
Fillet Lines...		Adds arcs tangent to the selected lines
Fillet Tracks		Adds arcs tangent to the selected straight track segments
Change Side / Flip	`F`	Flips selected item(s) to opposite side of board
Heal Shapes		Connect shapes, possibly extending or cutting them, or adding extra geometry
Intersect Polygons		Create the intersection of the selected polygons
Merge Polygons		Merge selected polygons into a single polygon
Mirror Horizontally		Mirrors selected item across the Y axis
Mirror Vertically		Mirrors selected item across the X axis
Move Corner To...		Move the active corner to an exact location
Move Exactly...	`Shift` + `M`	Moves the selected item(s) by an exact amount
Move Midpoint To...		Move the active midpoint to an exact location
Pack and Move Footprints	`P`	Sorts selected footprints by reference, packs based on size and initiates movement
Properties...	`E`	Displays item properties dialog

Action	Default Hotkey	Description
Skip	`Tab`	Skip item
Subtract Polygons		Subtract selected polygons from the last one selected
Swap	`S`	Swaps selected items' positions
Copy with Reference		Copy selected item(s) to clipboard with a specified starting point
Move	`M`	Moves the selected item(s)
Move Individually	`Ctrl`+`M`	Moves the selected items one-by-one
Move with Reference		Moves the selected item(s) with a specified starting point
Attempt Finish	`F`	Attempts to complete current route to nearest ratsnest end.
Attempt Finish Selected (Autoroute)	`Shift`+`F`	Sequentially attempt to automatically route all selected pads.
Break Track		Splits the track segment into two segments connected at the cursor position.
Route From Other End	`Ctrl`+`E`	Commits current segments and starts next segment from nearest ratsnest end.
Custom Track/Via Size...	`Q`	Shows a dialog for changing the track width and via size.
Cycle Router Mode		Cycle router to the next mode
Route Differential Pair	`6`	Route differential pairs
Differential Pair Dimensions...		Open Differential Pair Dimension settings
Drag 45 Degree Mode	`D`	Drags the track segment while keeping connected tracks at 45 degrees.
Drag Free Angle	`G`	Drags the nearest joint in the track without restricting the track angle.
Router Highlight Mode		Switch router to highlight mode
Place Blind/Buried Via	`Alt`+`Shift`+`V`	Adds a blind or buried via at the end of currently routed track.
Place Microvia	`Ctrl`+`V`	Adds a microvia at the end of currently routed track.

Action	Default Hotkey	Description
Route Selected From Other End	`Shift`+`E`	Sequentially route selected items from other end of ratsnest anchor.
Select Layer and Place Blind/Buried Via...	`Alt`+`<`	Select a layer, then add a blind or buried via at the end of currently routed track.
Select Layer and Place Micro Via...		Select a layer, then add a micro via at the end of currently routed track.
Select Layer and Place Through Via...	`<`	Select a layer, then add a through-hole via at the end of currently routed track.
Set Layer Pair...		Change active layer pair for routing
Interactive Router Settings...	`Ctrl`+`<`	Open Interactive Router settings
Router Shove Mode		Switch router to shove mode
Route Single Track	`X`	Route tracks
Switch Track Posture	`/`	Switches posture of the currently routed track.
Track Corner Mode	`Ctrl`+`/`	Switches between sharp/rounded and 45°/90° corners when routing tracks.
Undo Last Segment	`Back`	Walks the current track back one segment.
Router Walkaround Mode		Switch router to walkaround mode
Deselect All Tracks in Net		Deselects all tracks & vias belonging to the same net.
Filter Selected Items...		Remove items from the selection by type
Grab Nearest Unconnected Footprints	`Shift`+`O`	Selects and initiates moving the nearest unconnected footprint on each selected net.
Select/Expand Connection	`U`	Selects a connection or expands an existing selection to junctions, pads, or entire connections
Select All Tracks in Net		Selects all tracks & vias belonging to the same net.

Action	Default Hotkey	Description
Tune Skew of a Differential Pair	`9`	Tune skew of a differential pair
Tune Length	`7`	Tune length of a single track or differential pair
Add Microwave Polygonal Shape		Create a microwave polygonal shape from a list of vertices
Add Microwave Gap		Create gap of specified length for microwave applications
Add Microwave Line		Create line of specified length for microwave applications
Add Microwave Stub		Create stub of specified length for microwave applications
Add Microwave Arc Stub		Create stub (arc) of specified size for microwave applications
Footprint Checker		Show the footprint checker window
Copy Footprint		Copy Footprint
Create Footprint...		Create a new footprint using the Footprint Wizard
Cut Footprint		Cut Footprint
Delete Footprint from Library		Delete Footprint from Library
Duplicate Footprint		Make a copy of the selected footprint
Edit Footprint		Show selected footprint on editor canvas
Export Current Footprint...		Export edited footprint to file
Footprint Properties...		Edit footprint properties
Hide Footprint Tree		Hide Footprint Tree
Import Footprint...		Import footprint from file
New Footprint...	`Ctrl` + `N`	Create a new, empty footprint
Paste Footprint		Paste Footprint

Action	Default Hotkey	Description
Copy Pad Properties to Default		Copy current pad's properties
Push Pad Properties to Other Pads...		Copy the current pad's properties to other pads
Default Pad Properties...		Edit the pad properties used when creating new pads
Renumber Pads...		Renumber pads by clicking on them in the desired order
Edit Pad as Graphic Shapes	`Ctrl` + `E`	Ungroups a custom-shaped pad for editing as individual graphic shapes
Add Pad		Add a pad
Finish Pad Edit	`Ctrl` + `E`	Regroups all touching graphic shapes into the edited pad
Create Corner	`Ins`	Create a corner
Keep Arc Center, Adjust Radius		Switch arc editing mode to keep center, adjust radius and endpoints
Keep Arc Endpoints or Direction of Starting Point		Switch arc editing mode to keep endpoints, or to keep direction of the other point
Remove Corner		Remove corner
Position Relative To...	`Shift` + `P`	Positions the selected item(s) by an exact amount relative to another
Geographical Reannotate...		Reannotate PCB in geographical order
Refresh Plugins		Reload all python plugins and refresh plugin menus
Open Plugin Directory		Opens the directory in the default system file manager
Draft Fill Selected Zone(s)		Update copper fill of selected zone(s) without regard to other interacting zones
Fill All Zones	`B`	Update copper fill of all zones
Unfill Selected Zone(s)		Remove copper fill from selected zone(s)

Visor 3D

The actions below are available in the 3D Viewer. Hotkeys can be assigned to any of these actions in the **Hotkeys** section of the preferences.

Action	Default Hotkey	Description
Show 3D Models marked DNP	`D`	Show 3D models even if marked 'Do Not Place'
Show 3D Models not in POS File	`P`	Show 3D models even if not found in .pos file
Show Unspecified 3D Models	`V`	Show 3D models for 'unspecified' type footprints
Show SMD 3D Models	`S`	Show 3D models for 'Surface mount' type footprints
Show Through Hole 3D Models	`T`	Show 3D models for 'Through hole' type footprints
Flip Board	`F`	Flip the board view
Home View	`Home`	Home view
Render CAD Colors		Use a CAD color style based on the diffuse color of the material
Render Solid Colors		Use only the diffuse color property from 3D model file
Render Realistic Materials		Use all material properties from each 3D model file
Move Board Down	`Down`	Move board Down
Move Board Left	`Left`	Move board Left
Move Board Right	`Right`	Move board Right
Move Board Up	`Up`	Move board Up
No 3D Grid		No 3D Grid
Set Pivot	`Space`	Place point around which the board will be rotated (middle mouse click)
Rotate X Clockwise		Rotate X Clockwise
Rotate X Counterclockwise		Rotate X Counterclockwise

Action	Default Hotkey	Description
Rotate Y Clockwise		Rotate Y Clockwise
Rotate Y Counterclockwise		Rotate Y Counterclockwise
Rotate Z Clockwise		Rotate Z Clockwise
Rotate Z Counterclockwise		Rotate Z Counterclockwise
3D Grid 10mm		3D Grid 10mm
3D Grid 1mm		3D Grid 1mm
3D Grid 2.5mm		3D Grid 2.5mm
3D Grid 5mm		3D Grid 5mm
Show 3D Axis		Show 3D Axis
Show Model Bounding Boxes		Show 3D model bounding boxes in realtime renderer
Show Appearance Manager		Show/hide the appearance manager
Toggle Orthographic Projection		Enable/disable orthographic projection
View Back	`Shift` + `Y`	View Back
View Bottom	`Shift` + `Z`	View Bottom
View Front	`Y`	View Front
View Left	`Shift` + `X`	View Left
View Right	`X`	View Right
View Top	`Z`	View Top

Common

The actions below are available across KiCad, including in the PCB Editor. Hotkeys can be assigned to any of these actions in the **Hotkeys** section of the preferences.

Action	Default Hotkey	Description

Action	Default Hotkey	Description
Cursor Down	`Down`	
Cursor Down Fast	`Ctrl` + `Down`	
Cursor Left	`Left`	
Cursor Left Fast	`Ctrl` + `Left`	
Cursor Right	`Right`	
Cursor Right Fast	`Ctrl` + `Right`	
Cursor Up	`Up`	
Cursor Up Fast	`Ctrl` + `Up`	
Grid Origin...		Set the grid origin point
Edit Grids...		Edit grid definitions
Switch to Fast Grid 1	`Alt` + `1`	
Switch to Fast Grid 2	`Alt` + `2`	
Cycle Fast Grid	`Alt` + `4`	
Switch to Next Grid	`N`	
Switch to Previous Grid	`Shift` + `N`	
Reset Grid Origin		
Grid Origin		Place the grid origin point
Inactive Layer View Mode		Toggle inactive layers between normal and dimmed
Inactive Layer View Mode (3-state)	`H`	Cycle inactive layers between normal, dimmed, and hidden
Inches		Use inches
Snap to Objects on the Active Layer Only		Enables snapping to objects on the active layer only

Action	Default Hotkey	Description
Mils		Use mils
New...	`Ctrl` + `N`	Create a new document in the editor
New Library...		Create a new library folder
Open...	`Ctrl` + `O`	Open existing document
Page Settings...		Settings for paper size and title block info
Pan Down	`Shift` + `Down`	
Pan Left	`Shift` + `Left`	
Pan Right	`Shift` + `Right`	
Pan Up	`Shift` + `Up`	
Pin Library		Keep the library at the top of the list
Plot...		Plot
Print...	`Ctrl` + `P`	Print
Quit		Close the current editor
Redo Last Zoom		Return zoom to level prior to last zoom undo
Reset Local Coordinates	`Space`	
Revert		Throw away changes
Save	`Ctrl` + `S`	Save changes
Save All		Save all changes
Save As...	`Ctrl` + `Shift` + `S`	Save current document to another location
Save a Copy...		Save a copy of the current document to another location
Select Columns...		
3D Viewer	`Alt` + `3`	Show 3D viewer window
Show Context Menu		Perform the right-mouse-button action
Footprint Library Browser		Browse footprint libraries
Footprint Editor		Create, delete and edit footprints

Action	Default Hotkey	Description
Symbol Editor		Create, delete and edit symbols
Draw Bounding Boxes		Draw Bounding Boxes
Always Show Cursor	`Ctrl` + `Shift` + `X`	Display crosshairs even in selection tool
Full-Window Crosshairs		Switch display of full-window crosshairs
Show Grid		Display background grid in the edit window
Grid Overrides	`Ctrl` + `Shift` + `G`	Enables item-specific grids that override the current grid
Polar Coordinates		Switch between polar and cartesian coordinate systems
Switch units	`Ctrl` + `U`	Switch between imperial and metric units
Undo Last Zoom		Return zoom to level prior to last zoom action
Unpin Library		No longer keep the library at the top of the list
Update PCB from Schematic...	`F8`	Update PCB with changes made to schematic
Update Schematic from PCB...		Update schematic with changes made to PCB
Center on Cursor	`F4`	Center on Cursor
Zoom to Objects	`Ctrl` + `Home`	Zoom to Objects
Zoom to Fit	`Home`	Zoom to Fit
Zoom In at Cursor	`F1`	Zoom In at Cursor
Zoom In		Zoom In
Zoom Out at Cursor	`F2`	Zoom Out at Cursor
Zoom Out		Zoom Out
Refresh	`F5`	Refresh
Zoom to Selection	`Ctrl` + `F5`	Zoom to Selection
Cancel		Cancel current tool
Copy	`Ctrl` + `C`	Copy selected item(s) to clipboard

Action	Default Hotkey	Description
Interactive Delete Tool		Delete clicked items
Duplicate	`Ctrl` + `D`	Duplicates the selected item(s)
Find	`Ctrl` + `F`	Find text
Find and Replace	`Ctrl` + `Alt` + `F`	Find and replace text
Find Next	`F3`	Find next match
Find Next Marker	`Ctrl` + `Shift` + `F3`	
Find Previous	`Shift` + `F3`	Find previous match
Finish	`End`	Finish current tool
Paste	`Ctrl` + `V`	Paste item(s) from clipboard
Paste Special...		Paste item(s) from clipboard with annotation options
Redo	`Ctrl` + `Y`	Redo last edit
Replace All		Replace all matches
Replace and Find Next		Replace current match and find next
Show Search Panel	`Ctrl` + `G`	Show/hide the search panel
Select All	`Ctrl` + `A`	Select all items on screen
Undo	`Ctrl` + `Z`	Undo last edit
Unselect All	`Ctrl` + `Shift` + `A`	Unselect all items on screen
Measure Tool	`Ctrl` + `Shift` + `M`	Interactively measure distance between points
Select item(s)		Select item(s)
About KiCad		Open about dialog
Configure Paths...		Edit path configuration environment variables
Donate		Open "Donate to KiCad" in a web browser
Get Involved		Open "Contribute to KiCad" in a web browser

Action	Default Hotkey	Description
Preferences…	`Ctrl`+`,`	Show preferences for all open tools
Report Bug		Report a problem with KiCad
Manage Footprint Libraries…		Edit the global and project footprint library lists
Manage Symbol Libraries…		Edit the global and project symbol library lists