

KiCad

The KiCad Team

Table of Contents

Введение	2
Системные требования	2
Файлы и каталоги KiCad	2
Установка и обновление KiCad	5
Импорт настроек	5
Перенос файлов из предыдущих версий	5
Using the KiCad project manager	7
Standalone mode	8
Создание нового проекта	8
Импорт проекта из другой САПР	9
Saving and loading project archives	10
Git integration	10
Настройка KiCad	14
Общие настройки	14
Мышь и сенсорная панель	16
Горячие клавиши	17
Настройка путей	17
Настройка библиотек	19
Jobsets	21
Defining jobs	22
Defining jobset destinations	22
Available job types	24
Шаблоны проектов	26
Использование шаблонов	26
Template locations	27
Создание шаблонов	27
Менеджер плагинов и содержимого	30
Actions reference	31
KiCad Project Manager	31

Справочное руководство

Авторские права

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

Все торговые знаки этого руководства принадлежат его владельцам.

Соавторы

Jean-Pierre Charras, Fabrizio Tappero, Jon Evans, Graham Keeth.

Перевод

Юрий Козлов <yuray@komyakino.ru>, 2016

Барановский Константин <baranovskiykonstantin@gmail.com>, 2018-2021

Отзывы

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/>

Введение

KiCad - это комплекс программного обеспечения с открытым исходным кодом для проектирования электрических схем и печатных плат. KiCad поддерживает как интегрированный процесс проектирования, при котором схема и печатная плата разрабатываются одновременно, так и их редактирование по отдельности для особых случаев. KiCad также содержит несколько инструментов для упрощения разработки схем и печатных плат, таких как PCB калькулятор для определения электрических свойств элементов схемы, просмотрщик Gerber для исследования файлов для производства печатных плат, а также интегрированный SPICE-симулятор для исследования работы схемы.

KiCad работает на всех основных операционных системах и на компьютерах самой разной конфигурации. Он поддерживает печатные платы с количеством слоёв меди до 32 и может использоваться в проектах любой сложности. KiCad разрабатывается командой энтузиастов из программистов и инженеров со всего света, цель которых - создание свободного программного обеспечения с открытым исходным кодом применимого для профессиональной разработки.

Последняя версия документации KiCad доступна по адресу <http://docs.kicad.org>

Системные требования

KiCad может использоваться на разнообразном оборудовании и разных операционных системах, но некоторые задачи могут выполняться медленнее или проблематичнее на слабом оборудовании. Для наилучшего удобства, рекомендуется использовать выделенный графический адаптер и монитор с разрешением 1920x1080 или выше.

Системные требования последней версии KiCad можно узнать на веб-сайте: <https://kicad.org/help/system-requirements/>

Файлы и каталоги KiCad

При работе над схемой и платой KiCad создаёт и использует файлы со следующими расширениями файлов (и каталогов):

Many of these files include important design information, especially the project file (`.kicad_pro`), the schematic file(s) (`.kicad_sch`), and the board file (`.kicad_pcb`). Other files may also be necessary. Such files should always be included when distributing the project. Some files are not necessary to distribute with the project, such as the project-local settings file (`.kicad_pr1`) or the `fp-info-cache` file. Files that are unnecessary to distribute are noted in the table below.

Project files

*.kicad_pro	Project file, containing settings that are shared between the schematic and PCB
*.pro	Legacy (KiCad 5.x and earlier) project file. Can be read and will be converted to a <code>.kicad_pro</code> file by the project manager.

Schematic editor files

*.kicad_sch	Schematic files, containing all symbol and connection information.
*.kicad_sym	Schematic symbol library file, containing the symbol descriptions: graphic shape, pins, fields.
*.kicad_blocks	Schematic design block library folders. The folder itself is the library.
*.kicad_block	Schematic design block folder for defining a reusable schematic design. The folder is the design block, and contains a .kicad_sch file defining the design block's schematic and a .json file defining the design block's metadata.
*.wbk	Simulator workbook file containing SPICE simulation setup information.
*.sch	Legacy (KiCad 5.x and earlier) schematic file. Can be read and will be converted to a .kicad_sch file on write.
*.lib	Legacy (KiCad 5.x and earlier) schematic library file. Can be read but not written.
*.dcm	Legacy (KiCad 5.x and earlier) schematic library documentation. Can be read but not written.
*-cache.lib	Legacy (KiCad 5.x and earlier) schematic component library cache file. Required for proper loading of a legacy schematic (.sch) file.
sym-lib-table	Symbol library table: list of symbol libraries available in the schematic editor.
design-block-lib-table	Design block library table: list of design block libraries available in the schematic editor.

Board editor files and folders

*.kicad_pcb	Board file containing all info but the page layout.
*.pretty	Footprint library folders. The folder itself is the library.
*.kicad_mod	Footprint files, containing one footprint description each.
*.kicad_dru	Design rules file, containing custom design rules for a certain .kicad_pcb file.
*.brd	Legacy (KiCad 4.x and earlier) board file. Can be read, but not written, by the current board editor.
*.mod	Legacy (KiCad 4.x and earlier) footprint library file. Can be read by the footprint or the board editor, but not written.
fp-lib-table	Footprint library table: list of footprint libraries available in the board editor.
fp-info-cache	Cache to speed up loading of footprint libraries. Does not need to be distributed with the project or put under version control.

Common files

*.kicad_prl	Local settings for the current project; helps KiCad remember the last used settings such as layer visibility or selection filter. Does not need to be distributed with the project or put under version control.
*.kicad_wks	Page layout (drawing border and title block) description file.
*.kicad_jobset	Jobset definition file containing output jobsets.
*.net	Netlist file created from the schematic, and read by the board editor. Note that the recommended workflow for transferring information from the schematic to the board does not require the use of netlist files.
*.cmp	Association between components used in the schematic and their footprints. It can be created by the Board Editor and imported by the Schematic Editor. Its purpose is to import changes from the board to the schematic, for users who change footprints in the Board Editor (for instance using Exchange Footprints command) and want to import these changes back to the schematic. Note that the recommended workflow for transferring information from the board to the schematic does not require the use of <code>.cmp</code> files.

Fabrication and documentation files

*.gbr	Gerber files, for fabrication.
*.drl	Drill files (Excellon format), for fabrication.
*.pos	Position files (ASCII format), for automatic insertion machines.
*.rpt	Report files (ASCII format), for documentation.
*.ps	Plot files (Postscript), for documentation.
*.pdf	Plot files (PDF format), for documentation.
*.svg	Plot files (SVG format), for documentation.
*.dxf	Plot files (DXF format), for documentation.
*.plt	Plot files (HPGL format), for documentation.

Storing and sending KiCad files

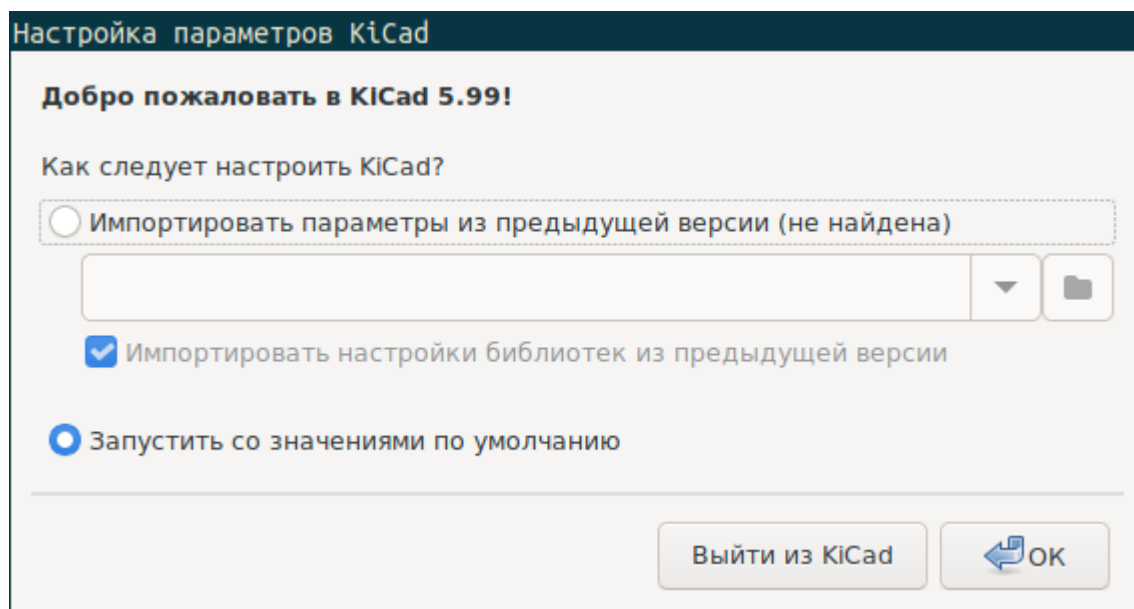
KiCad schematic and board files contain all the schematic symbols and footprints used in the design, so you can back up or send these files by themselves with no issue. Some important design information is stored in the project file (`.kicad_pro`), so if you are sending a complete design, make sure to include it.

Some files, such as the project-local settings file (`.kicad_prl`) and the `fp-info-cache` file, are not necessary to send with your project. If you use a version control system such as Git to keep track of your KiCad projects, you can add these files to the list of ignored files so that they are not tracked.

Установка и обновление KiCad

Импорт настроек

Каждая мажорная версия KiCad обладает собственными настройками, благодаря чему можно работать с несколькими версиями KiCad на одном компьютере без проблем. При первом запуске новой версии KiCad появится запрос первичной настройки параметров:



При обнаружении предыдущей версии KiCad, станет доступной возможность импорта настроек из этой версии. Расположение конфигурационных файлов предыдущей версии определяется автоматически, но его можно переопределить выбрав другой путь.

Please note that, the schematic symbol and footprint library tables from the previous version of KiCad will **not** be imported.

Можно начать работу с параметрами по умолчанию, если нет желания импортировать настройки из предыдущей версии.

KiCad stores the settings files in a folder inside your user directory. Each KiCad version will use a different versioned subfolder. For KiCad 9, those folders are:

Windows	%APPDATA%\kicad\9.0
Linux	~/.config/kicad/9.0
macOS	/Users/<username>/Library/Preferences/kicad/9.0

Перенос файлов из предыдущих версий

Последние версии KiCad могут открывать файлы созданные в предыдущих версиях, но сохраняют только в новейшем формате. В целом, это означает, что не нужно выполнять никаких особых действий при переходе от старой версии файлов, достаточно просто их открыть. В некоторых случаях, при переходе от одной версии KiCad к другой могут изменяться расширения файлов. После

загрузке таких файлов они будут сохранены в новом формате с новым расширением. Старые файлы автоматически не удаляются.

The schematic editor documentation describes several particular considerations for opening [legacy schematics](#).

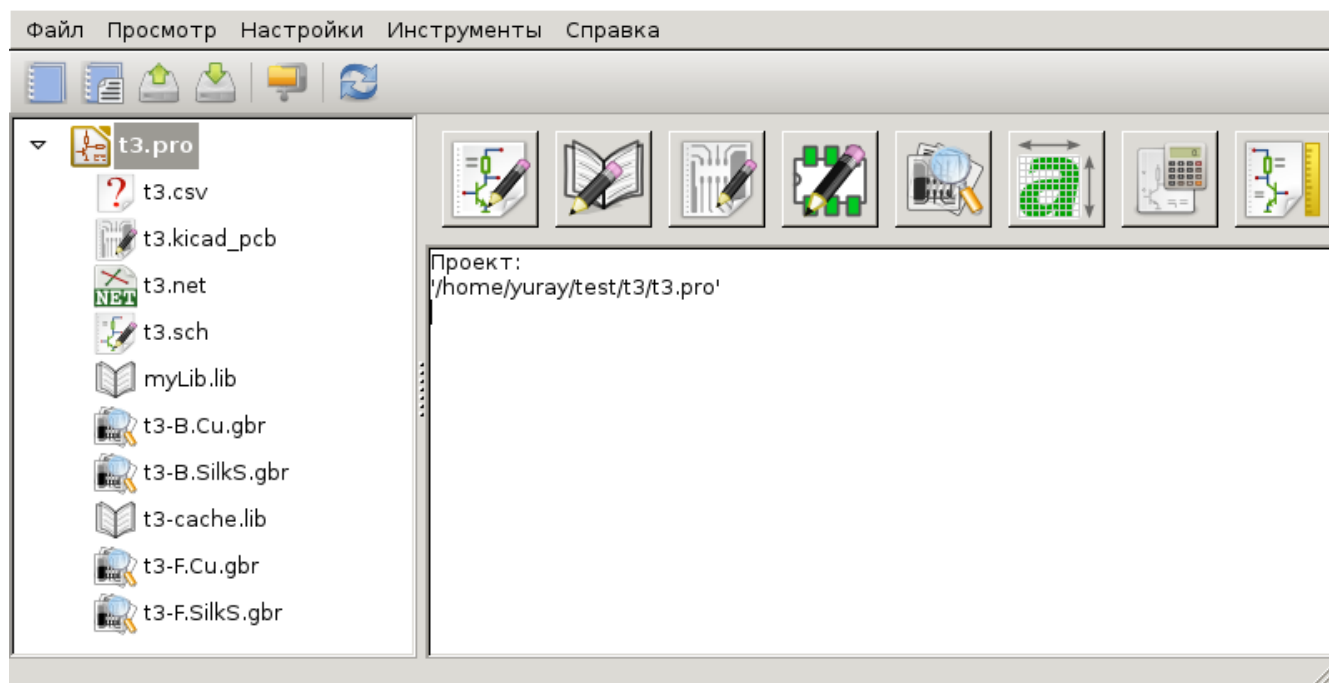
В общем, файлы созданные или изменённые одной версией KiCad **не могут** открываться более ранними версиями. По этой причине важно сохранять резервные копии своих проектов при тестировании новых версий KiCad, пока не убедитесь, что больше не нуждаетесь в старой версии KiCad.

NOTE

Конфигурация горячих клавиш не импортируется из предыдущих версий на данный момент. Её можно импортировать путём копирования различных *.hotkeys файлов из каталога с конфигурационными файлами старой версии. Примите во внимание, что при копировании файлов KiCad автоматически не обнаружит конфликты, такие как назначение одной и той же клавиши разным действиям.

Using the KiCad project manager

The KiCad project manager is a tool which creates and opens KiCad projects and launches the other KiCad tools (schematic and board editors, Gerber viewer, and utility tools).

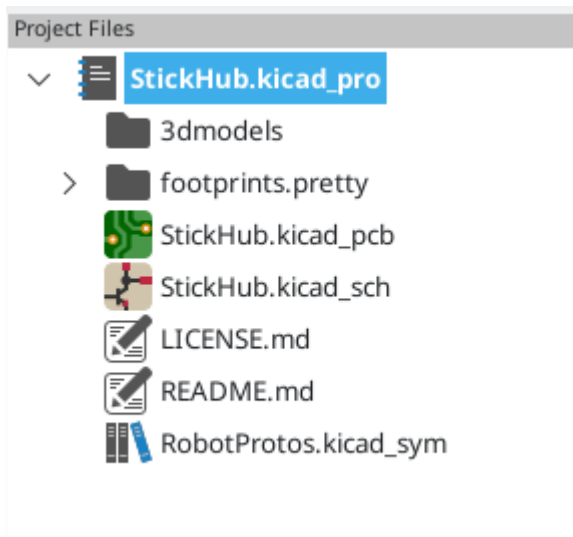


Окно менеджера проектов KiCad состоит из дерева проекта слева, отображающего файлы связанные с открытым проектом, и панели запуска приложений справа, содержащей кнопки различных редакторов и инструментов.

Панель инструментов с левой стороны окна предоставляет доступ к основным командам управления проектом:

	Create a new project.
	Open an existing project.
	Create a zip archive of the whole project. This includes schematic files, libraries, PCB, etc.
	Extract a project zip archive into a directory. Files in the destination directory will be overwritten.
	Refresh the tree view, to detect changes made on the filesystem.
	Open the project working directory in a file explorer.

The tree view shows a list of files inside the project folder. Double-clicking on a file in the tree view will open it in the associated editor. Right-clicking on a file will open a context menu with some file manipulation commands. If the project is part of a Git repository, the tree shows icons indicating the [version control status](#) of each file and lists the active branch next to the project name.



NOTE

В дереве проекта видны только те файлы, которые KiCad знает как открыть и показать.

KiCad projects contain at least a project file, a schematic, and a board design. Schematics may contain multiple sheets, each in its own file, but a project can only contain a single board. KiCad expects the project file, schematic root sheet file, and board file to all have the same name.

Standalone mode

You can also run the KiCad editor tools in *standalone* mode, by launching them directly from your operating system's application launcher rather than from the project manager. It is usually **not recommended** to run the tools in standalone mode, except for some specific situations where it is necessary, such as when importing projects from other EDA tools. When running in standalone mode, some project features are not available, including:

- cross probing between the schematic editor and the board editor
- design synchronization between the schematic and the board

Создание нового проекта

В большинстве случаев работа с KiCad начинается с создания проекта. Существует два пути создания проекта из менеджера проекта: создание пустого проекта или создание проекта на основе существующего шаблона. В этом разделе будет показано как создать новый пустой проект. Создание проекта из шаблона описано в разделе [Шаблоны проектов](#).

Для создания нового проекта воспользуйтесь командой **Создать проект...** из меню **Файл**, кнопкой панели инструментов **Создать новый пустой проект** или комбинацией клавиш **Ctrl+N**.

Будет запрошено имя проекта. По умолчанию, создаётся каталог для проекта с таким же именем. Например, если указать имя проекта `MyProject`, KiCad создаст каталог `MyProject` и файл проекта `MyProject/MyProject.kicad_pro` внутри.

Если уже имеется каталог для хранения файлов проекта, можно снять отметку с параметра *Создать новый каталог для проекта* в диалоговом окне **Создание нового проекта**.

Как только имя проекта будет задано, KiCad создаст следующие файлы внутри каталога проекта:

example.kicad_pro	KiCad project file.
example.kicad_sch	Main schematic file.
example.kicad_pcb	Printed circuit board file.

Импорт проекта из другой САПР

KiCad is able to import files created by some other software packages. Some software formats can be imported as complete projects. Others can only be imported as standalone schematics or boards at the moment, and must be manually linked together into a KiCad project. Currently the following types of project are supported:

*.sch, *.brd	Eagle 6.x or newer (XML format)
*.csa, *.cpa	CADSTAR archive format
*.zip	EasyEDA (JLCEDA) Standard Backup
*.epro, *.zip	EasyEDA (JLCEDA) Pro Project

Чтобы импортировать проект одной из этих САПР, выберите соответствующую команду из подменю **Импорт проекта из другой САПР...**, меню **Файл**.

Будет показано диалоговое окно для выбора файла схемы или печатной платы, которую нужно импортировать. Импортируемые файлы схемы и платы должны иметь одинаковые имена (например project.sch и project.brd). Как только необходимые файлы выбраны, будет запрошен каталог для хранения конечных файлов, в виде проекта KiCad.

Currently the following types of documents can be imported standalone. To import these documents, start the KiCad Schematic Editor or PCB Editor standalone (do not open the KiCad project manager first) and choose **File > Import > Non-KiCad Schematic** or **File > Import > Non-KiCad Board File**. When importing Altium projects, we recommend importing the PCB first, saving the resulting project, and then copying each schematic sheet into the project after importing it in a standalone Schematic Editor window.

*.SchDoc	Altium Designer, Circuit Studio, Circuit Maker schematic documents
*.PcbDoc	Altium Designer PCB
*.CMPcbDoc	Altium Circuit Maker PCB
*.CSPcbDoc	Altium Circuit Studio PCB
*.pcb	P-Cad 200x ASCII PCB
*.txt, *.fab	Fabmaster PCB

NOTE

KiCad does not support schematics with multiple top-level sheets. When importing designs from other tools that do support this feature, each schematic sheet must be imported, and then the imported sheets must be placed as hierarchical sheets in a new KiCad project.

Saving and loading project archives

You can archive your project's files into a zip archive with the Archive tool (**File** → **Archive Project...**).

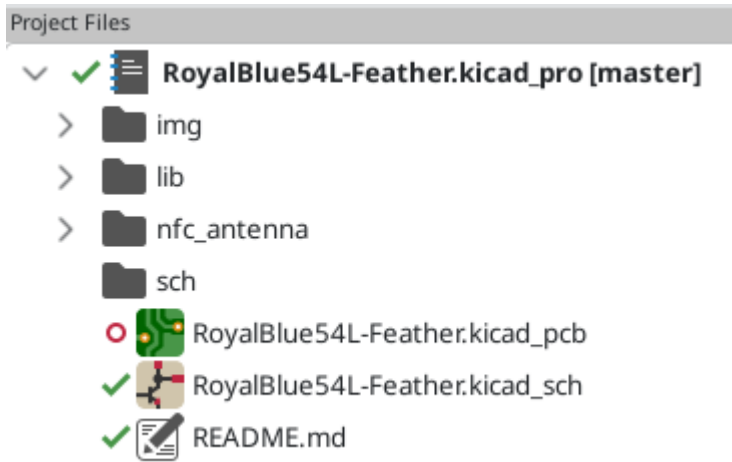
You can also unarchive a project using the Unarchive tool (**File** → **Unarchive Project...**). When you unarchive a project into the currently loaded project directory, the project will be reloaded automatically to reflect any changes that were in the archived version of the project.

The archive tool saves the following files from your project folder into the archive:

<code>*.kicad_prl, *.kicad_pro, *.kicad_sch, *.kicad_sym, *.kicad_pcb, *.kicad_mod, *.kicad_dru, *.kicad_wks, *.kicad_jobset, *.wbk, *.json, fp-lib-table, sym-lib-table, design-block-lib-table</code>	KiCad design files
<code>*.pro, *.sch, *.lib, *.dcm, *.cmp, *.brd, *.mod</code>	Legacy KiCad design files
<code>*.stp, *.step</code>	3D models
<code>*.g?, *.g??, *.gm??, *.gbrjob</code>	Gerber files
<code>*.pos, *.drl, *.nc, *.xnc, *.d356, *.rpt</code>	Manufacturing files
<code>*.net</code>	Netlists
<code>*.py</code>	Python scripts
<code>*.pdf, *.txt</code>	Documentation files
<code>*.cir, *.sub, *.model</code>	SPICE models
<code>*.ibs, *.pkg</code>	IBIS models

Git integration

The KiCad Project Manager integrates with the Git version control tool for tracking changes in your projects. It can work with an existing local Git repository, clone a project from a remote repository, or create a new repository in an existing project. You can use the tool to commit changes from your project, push and pull from a remote repository, and switch branches.



If you open a project that is already under version control with Git, i.e. it is part of an existing Git repository, you can use KiCad's version control features to track changes in the project without any additional configuration. The active branch is displayed next to the project name, and the version control status of each file in your project is shown graphically in the project files tree. For example, the ✓ icon indicates a file is unchanged, ○ indicates a file has uncommitted changes, and + indicates a file is not tracked. No icons are shown if the project is not part of a Git repository.

If an existing project is not already under version control, you can initialize a new Git repository in the project by right clicking on one of the files in the project files tree and clicking **Version Control** → **Add Project to Version Control...**. You must configure a remote when initializing a repository in this way. Configuring the repository requires the following information:

- **Name:** A name for the repository. This field can be anything and is not used.
- **Location:** The URL or file path to the remote.
- **Connection Type:** The protocol for connecting to the remote. This can be HTTPS, SSH, or local (file). HTTPS connections use username and password authentication. SSH connections use a username, private key, and an optional password for the keyfile. Local connections do not use authentication. You can check the connection and authentication by clicking the **Test** button.

Set default remote

Connection

Name

Location

Connection Type SSH

Authentication

SSH Private Key id_rsa Test



Username

SSH Key Password

Cancel OK

To clone an existing repository and open the cloned project, use **File** → **Clone Project from Repository...** You can clone a remote repository using SSH or HTTPS, or clone a local repository. The configuration settings for cloning are the same as the settings for configuring a new repository and remote for an existing project.



When you have made changes that you want to commit, you can commit either the entire project (right click → **Version Control** → **Commit Project...**) or a specific file (right click the file → **Version Control** → **Commit File...**). Both actions open the Commit Changes dialog, but the Commit Project action shows all changed files in the repository, while the Commit File action shows only the file that was right clicked. The Commit Changes dialog lets you select the changed files you want to include in the commit, provide a commit message and author, and commit the changes.

Filename	Status
<input type="checkbox"/>  demos/royalblue54L_feather/RoyalBlue..	Modified
<input type="checkbox"/>  demos/royalblue54L_feather/sch/Debug..	Modified

Commit Message:

Author:

Author Name <author@example.com>

 Cancel  OK

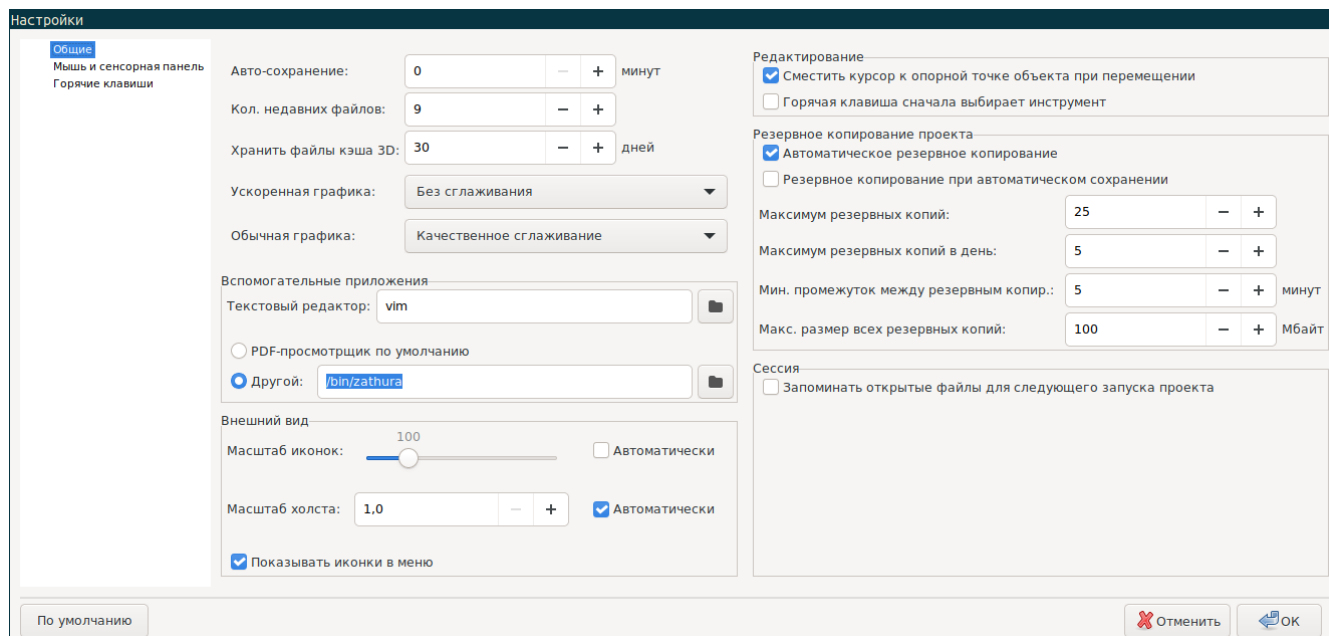
To push changes to the remote, right click in the project files tree and select **Version Control** → **Push**. To pull from the remote, right click and select **Version Control** → **Pull**. You can switch branches by selecting the desired branch from the **Version Control** → **Switch to Branch** menu.

Finally, you can remove version control entirely, deleting all tracked history from the local repository, by right clicking and selecting **Version Control** → **Remove Version Control**.

Настройка KiCad

The KiCad preferences can always be accessed from the **Preferences** menu, or by using the hotkey (default **Ctrl + ,**). The Preferences dialog is shared between the running KiCad tools. Some preferences apply to all tools, and some are specific to a certain tool (such as the schematic or board editor).

Общие настройки



Ускоренная графика: KiCad может использовать различные методы сглаживания при отрисовке средствами графического адаптера. Эти методы могут выглядеть по-разному на разном оборудовании, поэтому следует поэкспериментировать для поиска наилучшего отображения.

Обычная графика: KiCad может также выполнять сглаживание в режиме совместимости, используя программные средства. Активация этой функции может привести к замедлению работы на некотором оборудовании.

Текстовый редактор: определяет текстовый редактор для открытия текстовых файлов из дерева менеджера проектов.

PDF-просмотрщик: определяет приложение для открытия PDF-файлов.

Показать иконки в меню: включает отображение иконок в контекстном меню KiCad.

NOTE

На некоторых операционных системах иконки не отображаются в меню.

Show scrollbars in editors: When enabled, scrollbars are displayed next to the editing canvases in each tool. When disabled, scrollbars are not shown.

Focus follows mouse between schematic and PCB editors: When enabled, the window under the mouse cursor will automatically become focused.

Масштаб иконок: устанавливает размер иконок, которые используются в меню и на кнопках в KiCad. Отметьте *Автоматически* для автоматического подбора оптимального масштаба иконок на

основе параметров операционной системы.

Тема иконок: определяет какой набор иконок использовать, для светлого или тёмного фона. Значение по умолчанию - автоматически. Тема будет определена на основе цвета фона окна используемого системой.

High-contrast mode dimming factor: Sets how much non-focused items are dimmed in high-contrast display mode.

Сместить курсор к опорной точке объекта при перемещении: если отмечено, курсор мыши будет перемещён на опорную точку объекта в начале операции его перемещения.

Горячая клавиша сначала выбирает инструмент: без этой опции, нажатие горячей клавиши для, к примеру, команды *Добавить проводник* сразу же начнёт выполнение этой команды, начиная с текущей позиции курсора. Если отмечено, нажатие горячей клавиши в первый раз только активирует инструмент *Добавить проводник* и не создаст проводник под курсором.

Запоминать открытые файлы для следующего запуска проекта: если отмечено, KiCad автоматически откроет все файлы, которые были открыты при последнем закрытии менеджера проектов.

Авто-сохранение: при редактировании файлов схемы либо платы, KiCad может автоматически выполнять периодическое сохранение проделанной работы. Чтобы отключить эту функцию установите 0

Кол. недавних файлов: определяет количество элементов в списке недавно открытых файлов

Хранить файлы кэша 3D: KiCad создаёт кэш для 3D моделей, чтобы ускорить работу 3D-просмотрщика. Можно настроить как долго стоит хранить кэш перед удалением устаревших файлов.

Автоматическое резервное копирование: если отмечено, проекты KiCad будут архивироваться в ZIP-файлы автоматически согласно следующим настройкам. Архивы хранятся в подкаталоге каталога проекта. Резервные копии создаются при сохранении файлов проекта.

Резервное копирование при автоматическом сохранении: если отмечено, резервные копии будут создаваться при каждом автоматическом сохранении (если резервная копия разрешена следующими настройками). Этот параметр имеет значение только если интервал автоматического сохранения не равен 0 (не отключён).

Максимум резервных копий: при создании новой резервной копии, старые копии будут удаляться, чтобы общее число файлов резервных копий не превышало указанного лимита.

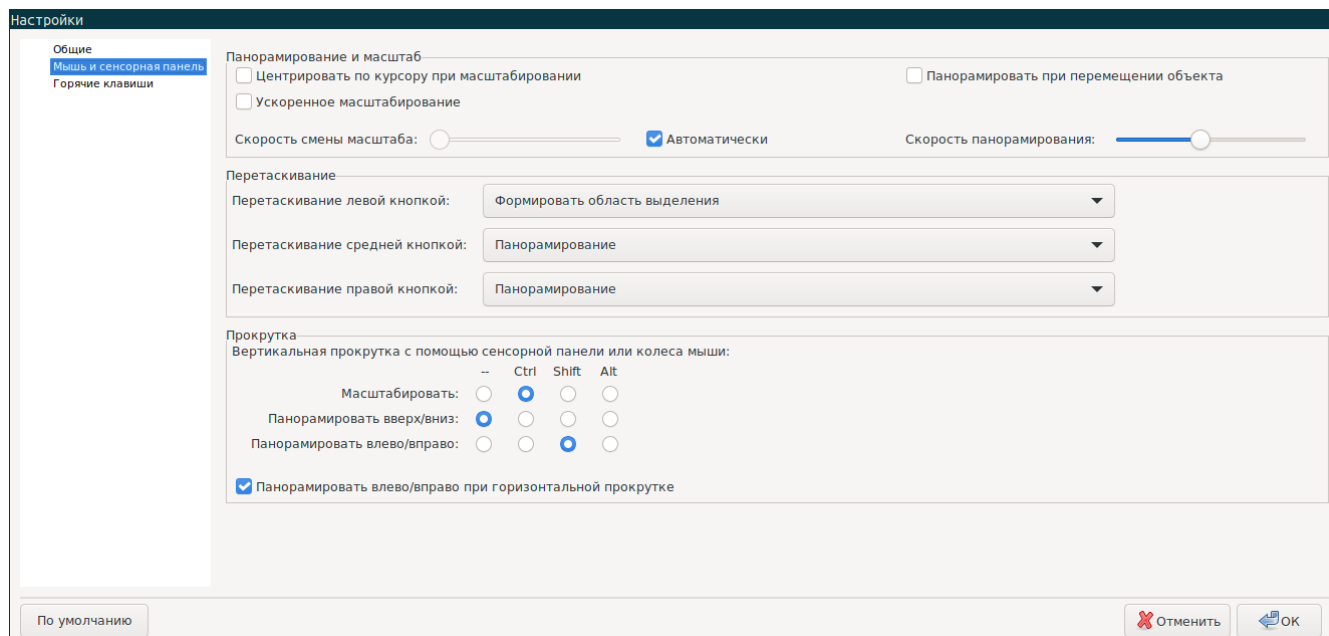
Максимум резервных копий в день: при создании новой резервной копии, старые копии, созданные в этот же день, будут удаляться, чтобы не превысить указанный лимит.

Мин. промежуток между резервным копир.: если настало время создания резервной копии (например, при сохранении файла платы) и существует резервная копия, созданная раньше чем указанный интервал, резервная копия не будет создана.

Макс. размер всех резервных копий: при создании новой резервной копии файлы старых резервных копий будут удаляться, чтобы общий размер каталога с резервными копиями не

превышал указанный предел.

Мышь и сенсорная панель



Центрировать по курсору при масштабировании: если отмечено, перед масштабировании с помощью горячих клавиш или колеса мыши изображение будет отцентрировано по положению курсора.

Ускоренное масштабирование: если отмечено, прокрутка с помощью колеса мыши или сенсорной панели будет происходить быстрее.

Скорость смены масштаба: определяет на какую величину должен смениться масштаб при прокрутке мышью или сенсорной панелью. Установите отметку *Автоматически* чтобы использовать значение по умолчанию, которое зависит от настроек операционной системы.

Панорамировать при перемещении объекта: если отмечено, изображение можно панорамировать при перемещении объекта, поднося его к краю области редактирования.

Скорость панорамирования: определяет как быстро должно выполняться панорамирование при перемещении объекта.

Кнопки мыши: можно настроить поведение при перетаскивании средней кнопкой мыши и нажатии правой кнопки мыши: смена масштаба, панорамирование или ничего не делать. Также можно настроить поведение при перетаскивании с нажатой левой кнопки мыши в редакторе при наличии выделенных элементов или без них.

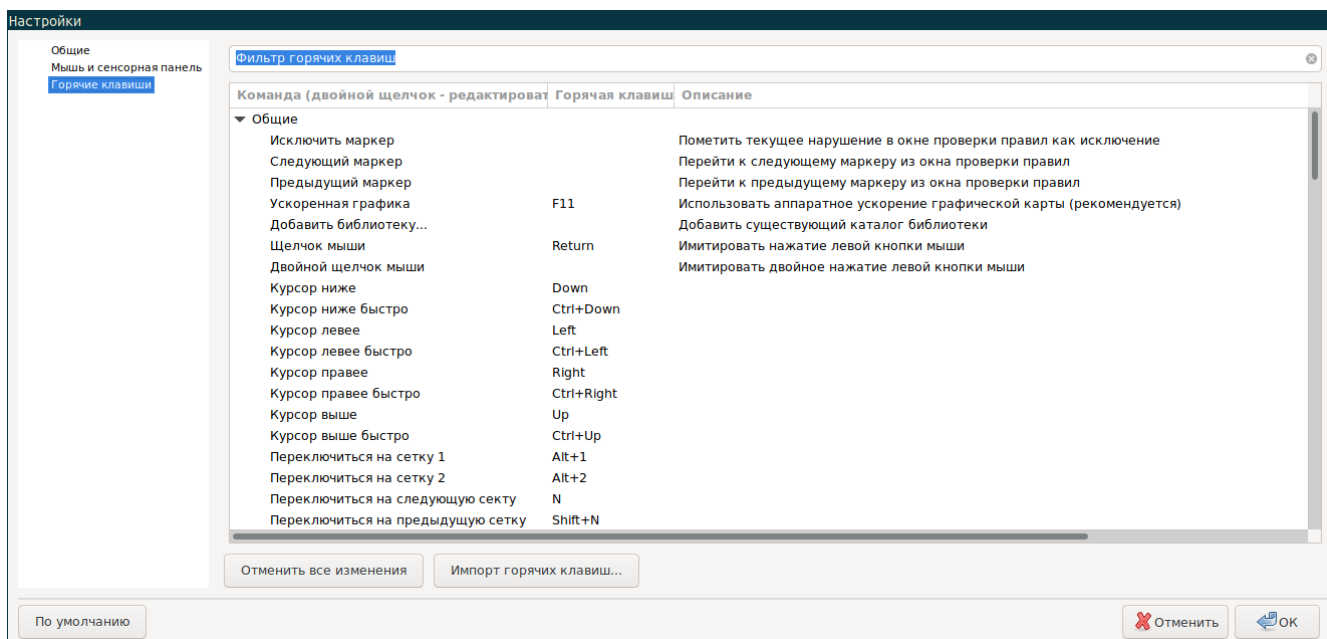
NOTE

Левая кнопка мыши всегда используется для выделения и управления объектами.

Прокрутка колесом мыши и сенсорной панелью: можно настроить поведение при прокрутке колесом мыши или сенсорной панелью с нажатой определённой клавишей-модификатором.

Панорамировать влево/вправо при горизонтальной прокрутке: если отмечено, можно выполнять горизонтальное панорамирование с помощью сенсорной панели или второго колеса мыши (если оно имеется).

Горячие клавиши



С помощью этого диалогового окна можно настроить горячие клавиши, используемые для управления KiCad. Горячие клавиши в разделе *Общие* относятся ко всем приложениям KiCad. Горячие клавиши определённых приложений KiCad отображаются когда эти приложения запущены. Можно задавать одинаковые горячие клавиши для различных операций в разных приложениях KiCad (например, в редакторе схем и в редакторе плат), но нельзя задавать одну и ту же горячую клавишу для разных операций одного приложения.

Имеется огромное количество команд, поэтому не всем присвоены горячие клавиши по умолчанию. Можно добавить горячую клавишу для любой команды с помощью двойного щелчка мыши по команде в списке. Если желаемая горячая клавиша уже занята, можно указать использовать эту горячую клавишу для выбранной команды, при этом горячая клавиша конфликтной команды будет удалена.

Changes that you have made to hotkey assignments are shown with a ***** character at the end of the command name. You can undo changes to a specific command by right-clicking that command and selecting **Undo Changes**, or you can undo all changes with the button below the command list.

Импорт горячих клавиш

Настройки горячих клавиш хранятся в файлах `.hotkeys` в каталоге настроек KiCad (см. раздел [Настройки](#) для получения информации о том где располагается каталог с настройками в операционной системе). Если имеются настройки горячих клавиш, которые хотелось бы задействовать на другом компьютере, можно скопировать файлы `.hotkeys` и импортировать их на другой машине.

Настройка путей

In KiCad, one can define paths using a **path variable**. A few path variables are internally defined by KiCad, and can be used to define paths for libraries, 3D shapes, etc.

Это полезно, когда абсолютные пути неизвестны или меняются (например, при переносе проекта на другой ПК), а также, когда один общий каталог содержит множество подобных элементов. Рассмотрим следующие объекты, которые могут быть установлены в разных местах:

- Библиотеки символов схемы
- Библиотеки посадочных мест
- 3D model files used in footprint definitions

For instance, the path to the `connect.pretty` footprint library, when using the `KICAD9_FOOTPRINT_DIR` path variable, would be defined as `${KICAD9_FOOTPRINT_DIR}/connect.pretty`.

The **Preferences** → **Configure Paths...** menu allows you to define paths for some built-in KiCad path variables, and add your own path variables to define personal paths, if needed.

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, `${KICAD8_FOOTPRINT_DIR}` will automatically resolve to the value of `${KICAD9_FOOTPRINT_DIR}` if there is no `KICAD8_FOOTPRINT_DIR` variable defined.

KiCad path variables

KICAD9_3DMODEL_DIR	Base path of KiCad's standard 3D footprint model library files.
KICAD9_3RD_PARTY	Location for plugins, libraries, and color themes installed by the Plugin and Content Manager .
KICAD9_FOOTPRINT_DIR	Base path of KiCad's standard footprint library files.
KICAD9_SYMBOL_DIR	Base path of KiCad's standard symbol library files.
KICAD9_TEMPLATE_DIR	Location of KiCad's standard project template library files.
KICAD_USER_TEMPLATE_DIR	Location of personal project templates.
SPICE_LIB_DIR	Location of personal simulation model libraries . This variable is not defined by default.
KIPRJMOD	Absolute path to the current project directory. This variable is set automatically and cannot be redefined.

	Имя	Путь	
1	KICAD_PTEMPLATES	/usr/share/kicad/template/	<div>Добавить</div> <div>Удалить</div>
2	KIGITHUB	https://github.com/KiCad	
3	KISYS3DMOD	/usr/share/kicad/modules/packages3d/	
4	KISYSMOD	/usr/share/kicad/modules/	

Справка

Отменить

ОК

Paths set in the Configure Paths dialog are internal to KiCad and are not visible as environment variables outside of KiCad. They are stored in [KiCad's user configuration files](#).

Paths can also be set as system environment variables outside of KiCad, which will override any settings in the user's configuration.

NOTE

You cannot override a system environment variable that has been set outside of KiCad by using the Configure Paths dialog. Any variable that has been set externally will be shown as read-only in the dialog.

Note also that the path variable `KIPRJMOD` is **always** internally defined by KiCad, and expands to the **current project absolute path**. For instance, `${KIPRJMOD}/connect.pretty` is always the `connect.pretty` folder (the footprint library) inside **the current project folder**. The `KIPRJMOD` variable cannot be changed in the Configure Paths dialog or overridden by an external environment variable.

Advanced environment variables

Some advanced environment variables can be set to customize where KiCad expects certain files to be located. By default, these locations are set based on your platform, but they can be overridden by system environment variables. These variables are not shown in the Configure Paths dialog and cannot be used in path substitutions.

Changing these variables will not result in KiCad moving any files from the default location to the new location, so if you change these variables you will need to copy any desired settings or files manually.

KICAD_CONFIG_HOME	Base path of KiCad configuration files. Subdirectories will be created within this directory for each KiCad minor version.
KICAD_DOCUMENTS_HOME	Base path of KiCad user-modifiable documents, such as projects, templates, Python scripts, libraries, etc. Subdirectories will be created within this directory for each KiCad minor version. This directory is provided as a suggested user data location, but does not need to be used.
KICAD_STOCK_DATA_HOME	Base path of KiCad stock data, including default libraries. The data in this directory is managed by the KiCad installer or system package manager, and is not intended for user-writeable data.

WARNING

If you modify the configuration of paths, please quit and restart KiCad to avoid any issues in path handling.

Настройка библиотек

The **Preferences** → **Manage Symbol Libraries...** menu lets you manage the list of symbol libraries ([symbol library table](#)).

Likewise, use the **Preferences** → **Manage Footprint Libraries...** menu to manage the list of footprint libraries ([footprint library table](#)).

For each type of library (symbol and footprint), there are 2 library tables: global and project specific. The global library table is located in the [user configuration directory](#) and contains a list of libraries available to

all projects. The project-specific library table is optional and contains a list of libraries specific to the project. It is located in the project directory.

Jobsets

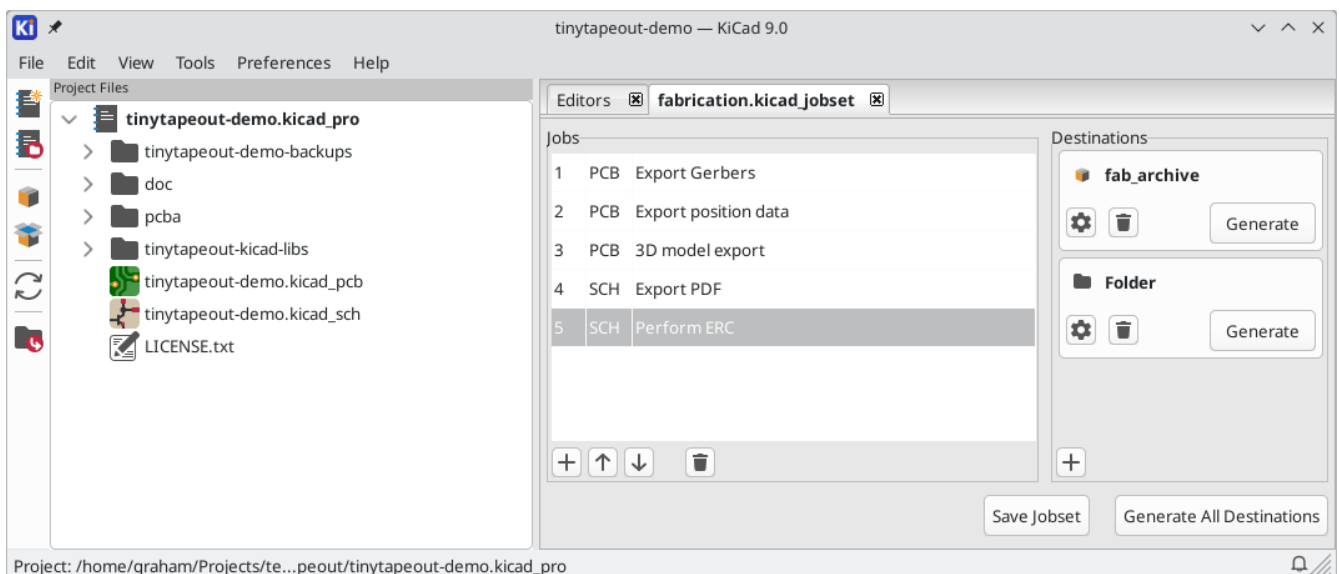
KiCad lets you configure a list of outputs that are all generated with a single click. The list of output jobs and the destinations where they will be saved is called a *jobset*. For example, a jobset might contain jobs to generate Gerber files, assembly data, a bill of materials, PDF plots of the schematic and PCB, while also running ERC and DRC checks, with all of the outputs saved to a compressed archive. The full list of available jobs is given [below](#).

Each *job* in a jobset defines a single type of generated output, such as a bill of materials or a set of Gerbers. A job can be configured in the same way as if the output was manually generated from the schematic or board editor. The configuration for each job is stored in the jobset and remembered when you load the jobset later. Jobs are configured individually, so if you include the same type of job multiple times in a single jobset, each job will have its own independent configuration. For example, this lets you generate PDF outputs in color as well as black and white.

In addition to the jobs, jobsets also contain *destinations*, which define a list of jobs to run and how to store their outputs. A jobset destination can simply store the chosen jobs' output files in a specified location, or it can add the output files to a compressed archive. Each jobset destination can select a different subset of jobs from the full list of jobs in the jobset. You can run each jobset destination individually or run all jobset destinations at once. As an example, you could set up one jobset destination that generates PDFs of the board and schematic and copies them to an external location, while another destination generates the fabrication files and compresses them in a zip archive to send to the board manufacturer.

Projects can have multiple jobsets, with each jobset defining a different list of jobs and output configurations. Each jobset is stored in a `.kicad_jobset` file, which can be specific to a single project, copied between projects, or even stored in a central location and shared between projects.

To use a jobset, first create a new jobset file in the KiCad project manager (**File** → **New Jobset File...**) and choose a name and location for it. Alternatively, you can open an existing jobset file with **File** → **Open Jobset File...** Jobset files that are stored in the project directory are considered part of the project and are displayed in the project file tree. You can open a jobset file in the project file tree by double clicking on it.



Once you create or open a jobset, it is displayed in a new tab in the project manager. The list of jobs is shown in the middle and the list of jobset destinations is shown on the right. New jobsets will not contain any jobs, but a destination is automatically created to save outputs to a folder. When you make changes to a jobset, you can save the changes by clicking the **Save Jobset** button.

Defining jobs

To add a new job, click the **+** button under the Jobs list. In the Add New Job dialog that appears, select the desired type of job. You can filter which types of jobs are shown in the list by typing in the **Filter** textbox at the bottom.

When you select a job and press **OK**, the settings dialog for that type of output will appear. Each job settings dialog provides the same options you would have if you manually generated that type of output from the schematic or board editor.

NOTE

Output filenames and paths specified in job settings are relative to the [jobset destination](#) folder or archive root. You can use certain [text variables](#), like `${PROJECTNAME}`, `${CURRENT_DATE}`, and [project text variables](#).

Export PDF Job Settings

Output file: `${PROJECTNAME}_r${REVISION}_${ISSUE_DATE}.pdf`

Options

Page size: Schematic size

☒ Plot drawing sheet

Output mode: Color

Color theme: KiCad Classic

☒ Plot background color

Minimum line width: 0 mils

HPGL Options

Position and units: Content fit, user units

Pen width: 0.004 mils

PDF Options

☒ Generate property popups

☒ Generate clickable links for hierarchical elements

☒ Generate metadata from AUTHOR & SUBJECT variables

Other Options

Cancel OK

When you accept the job settings dialog, the job is added to the list of jobs, where you can optionally change the new job's description from its default. To change a job's description or settings later, right click the job in the list and select **Edit Job Description** or **Edit Job Settings....** Double clicking on a job also edits its settings. To remove a job, select the job and click the **🗑** button. To reorder the list, select a job and move it up or down using the **↑** or **↓** buttons.

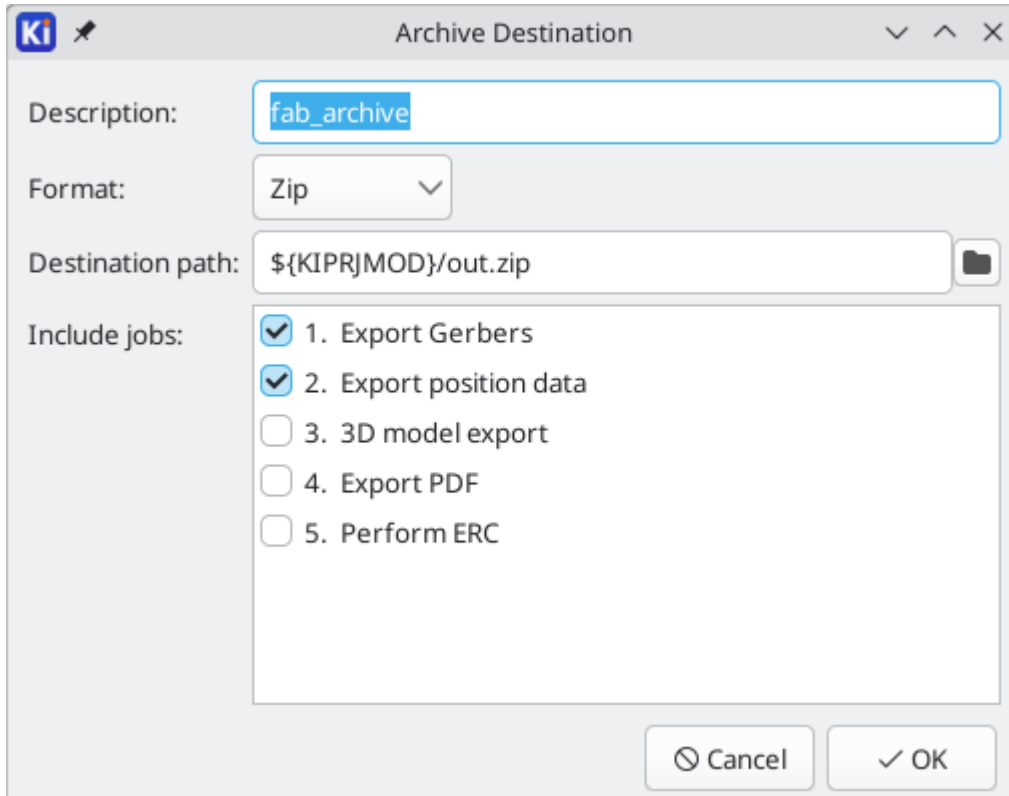
Defining jobset destinations

You cannot generate any outputs from a jobset until you add a jobset destination. One destination is created automatically when a jobset is created, but you can add as many destinations as you need.



To add a jobset destination, click the **+** button under the Destinations list. When the Add New Destination dialog appears, select a type of destination:

- **Archive** saves the outputs generated by the jobs in a compressed zip archive.
- **Folder** saves the outputs generated by the jobs uncompressed in a folder.

Once you have selected a type of output, the Destination options dialog appears.

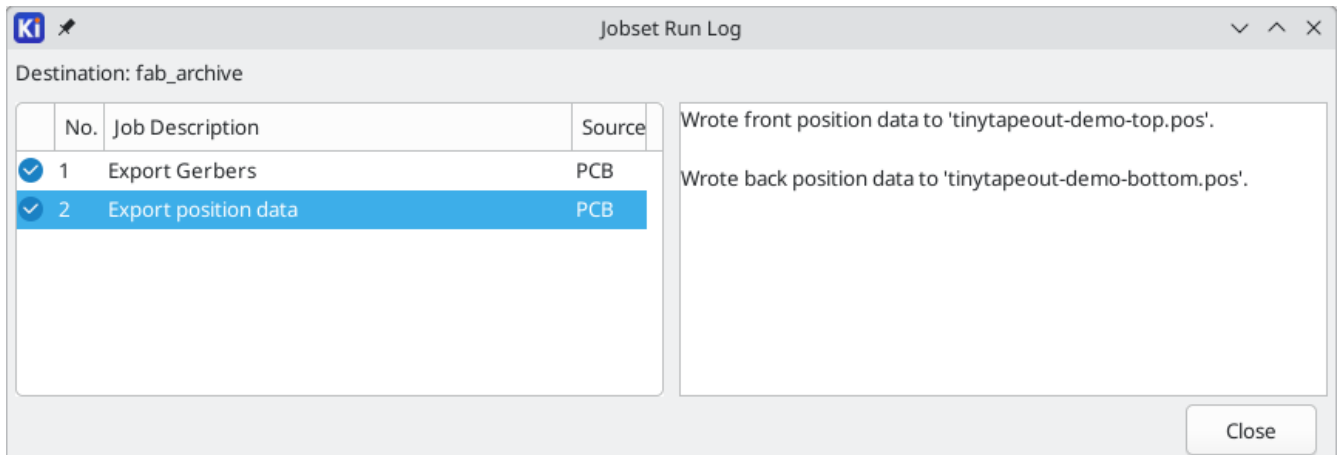


Here you can select which jobs will be run as part of this jobset destination, as well as the folder or archive name that will be used to store them. By default all jobs are enabled. You can also set a description for the destination to be displayed in the Destinations list. The output path controls where the files generated by the jobs will be saved. The path here can be absolute or relative to the project directory, and it can use [path variables](#) or certain text variables (`${PROJECTNAME}`, `${CURRENT_DATE}`, and [project text variables](#)). Filenames defined in job configurations are relative to the jobset destination directory or archive root.

When you click **OK** in this dialog, the new jobset destination is added to the Destinations list. You can modify an existing jobset destination by clicking its  button, or remove it by clicking its  button.

After configuring your jobs and destinations, you can generate an individual set of outputs by clicking the **Generate** button for the desired destination. You can run all destinations at once by clicking the **Generate All Destinations** button.

If a jobset destination runs and generates its outputs successfully, a blue check is shown that indicates the last run was successful. If a jobset destination fails to complete successfully, a red exclamation point is shown to indicate the run was not successful. Clicking on the success/failure indicator will display the Jobset Run Log dialog, which displays the status of each job in the jobset destination. Clicking on a specific job will display the logged output from that job, if there is any.



Available job types

The following types of jobs are available:

Job	Description
PCB: Export 3D Model	Exports a 3D model of the board. The model format can be STEP, GLB (binary glTF), XAO, BREP (OCCT), PLY, or STL.
PCB: Export Drill Data	Exports a drill file from the board.
PCB: Export DXF	Exports the board design to a DXF file .
PCB: Export Gerbers	Exports the board design to Gerber files , with one file per selected layer.
PCB: Export IPC-2581	Exports the board design in IPC-2581 format .
PCB: Export ODB++	Exports the board design in ODB++ format .
PCB: Export PDF	Exports the board design to PDF files , with one file per selected board layer. You can also generate a single PDF with multiple layers depending on the plot configuration.
PCB: Export Position Data	Exports a position (component placement) file from the board.
PCB: Export SVG	Exports the board design to a SVG file .
PCB: Perform DRC	Performs a Design Rule Check on the board and generates a report. If DRC violations are found, this job can optionally report a job failure.
PCB: Render	Generates a raytraced rendering of the 3D model of the board as a PNG or JPG file.
Schematic: Export DXF	Exports the schematic to a DXF file .
Schematic: Export HPGL	Exports the schematic to a HPGL file .
Schematic: Export Netlist	Exports a netlist from the schematic, with various formats available.
Schematic: Export PDF	Exports the schematic to a PDF file .
Schematic: Export Postscript	Exports the schematic to a PostScript file .
Schematic: Export SVG	Exports the schematic to a SVG file .
Schematic: Generate Bill of Materials	Exports a bill of materials from the schematic.
Schematic: Perform ERC	Performs an Electrical Rule Check on the schematic and generates a report. If ERC violations are found, this job can optionally report a job failure.
Special: Copy Files	Copies the specified file to the specified location. A failure to copy the files can optionally cause the output job to fail. You can control whether files in the output location should be overwritten or not.
Special: Execute Command	Executes an arbitrary command. Output from the command can optionally be logged to a file. You can either ignore non-zero output codes or cause them to fail the output job.

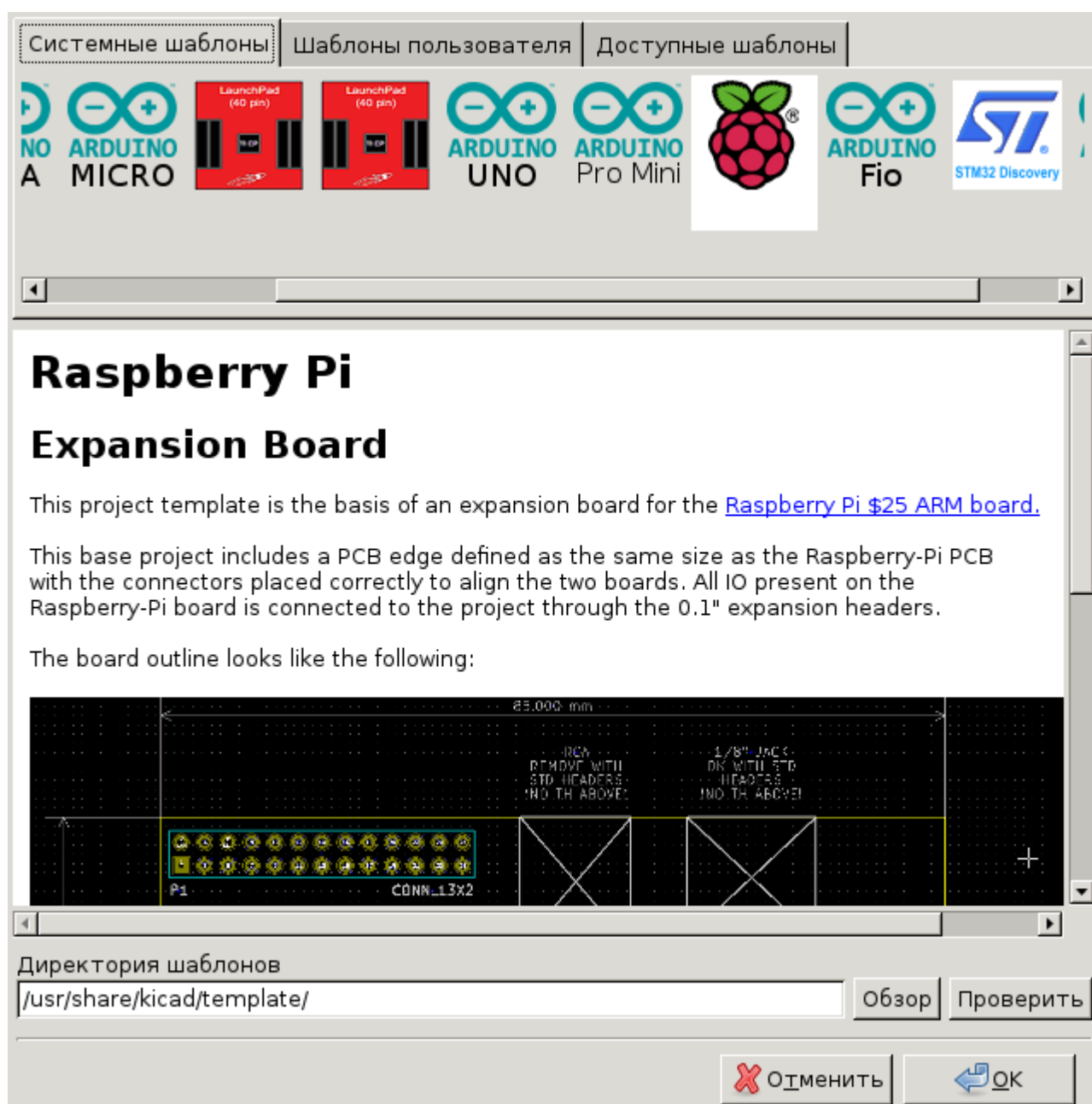
Шаблоны проектов

Шаблоны облегчают создание новых проектов, благодаря используя предварительно настроенных параметров из шаблона. Они могут содержать предопределённые контуры печатной платы, расположения разъемов, компоненты схемы, правила проектирования и т.п. Для наследования в новых проектах могут использоваться завершённые файлы схемы и/или печатной платы.

Использование шаблонов

The **File** → **New Project from Template** menu will open the Project Template Selector dialog:

После щелчка левой кнопки мыши на значке шаблона будет показана информация о нём, а следующий щелчок на кнопке 'OK' приведёт к созданию нового проекта. Файлы шаблона будут скопированы в каталог нового проекта и переименованы в соответствии с новым именем.



Template locations

KiCad looks for system templates in the path defined in the `KICAD9_TEMPLATE_DIR` path variable, and user templates in the path defined in `KICAD_USER_TEMPLATE_DIR`. However, you can browse for templates in an arbitrary directory using the **Folder** control at the top of the dialog.

Создание шаблонов

A KiCad template is simply a directory containing the template project files, as well as some required metadata for the template in a subdirectory named `meta`. The name of the directory containing the template files determines the name of the template. When you create a project from a template, KiCad copies the template files to the new project directory, renaming them to match the new project name as described below.

All files in the template are copied, with two exceptions:

- Files with names beginning with the `.` character (dotfiles) are not copied. There is a special case for files named `.gitignore` or `.gitattributes`, which are copied if they exist.
- The `meta` directory is not copied

The `meta` directory must contain an HTML file named `info.html`, which is displayed in the KiCad template browser and should contain basic information describing the template. Basic HTML features are supported, including images. Any images referenced by `info.html` should also be stored in the `meta` directory.

The `<title>` tag determines the name of the template that is displayed during template selection. Note that the project template name will be cut off if it's too long. This display name does not need to be the same as the template directory name.

Here is a sample `info.html` file:

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<META HTTP-EQUIV="CONTENT-TYPE" CONTENT="text/html;
charset=windows-1252">
<TITLE>Raspberry Pi - Expansion Board</TITLE>
</HEAD>
<BODY LANG="fr-FR" DIR="LTR">
<P>This project template is the basis of an expansion board for the
<A HREF="http://www.raspberrypi.org/" TARGET="blank">Raspberry Pi $25
ARM board.</A> <BR><BR>This base project includes a PCB edge defined
as the same size as the Raspberry-Pi PCB with the connectors placed
correctly to align the two boards. All IO present on the Raspberry-Pi
board is connected to the project through the 0.1" expansion
headers. <BR><BR>The board outline looks like the following:
</P>
<P><IMG SRC="brd.png" NAME="brd" ALIGN=BOTTOM WIDTH=680 HEIGHT=378
BORDER=0><BR><BR><BR><BR>
</P>
<P>(c)2012 Brian Sidebotham<BR>(c)2012 KiCad Developers</P>
</BODY>
</HTML>

```

Finally, `meta` can optionally contain an image named `icon.png`, which will be used as the template's icon in the template selection dialog. The icon should be a 64 x 64 pixel PNG image.

Template file renaming

All files and directories in a template are copied to the new project path when a project is created using a template, except `meta` and any dotfiles. Files and directories containing the template directory name will be renamed with the new project file name.

For example, using a template named `example` (left) to create a project named `newproject` (right), with renamed files shown in **bold**:

Files in template <code>example</code> directory	Files created in project <code>newproject</code> directory
<code>example.kicad_pro</code>	<code>newproject.kicad_pro</code>
<code>example.kicad_sch</code>	<code>newproject.kicad_sch</code>
<code>example.kicad_pcb</code>	<code>newproject.kicad_pcb</code>
<code>example-first.kicad_sch</code>	<code>newproject-first.kicad_sch</code>
<code>second-example.kicad_sch</code>	<code>second-newproject.kicad_sch</code>
<code>third.kicad_sch</code>	<code>third.kicad_sch</code>
<code>third.kicad_pcb</code>	<code>third.kicad_pcb</code>

A template does not need to contain a complete project. If a required project file is missing, KiCad will create the file using the default create project behavior:

Files in template example directory	Files created in newproject directory
example.kicad_sch first-example.kicad_sch first-example.kicad_pcb second-example.kicad_sch second-example.kicad_pcb	newproject.kicad_sch first-newproject.kicad_sch first-newproject.kicad_pcb second-newproject.kicad_sch second-newproject.kicad_pcb newproject.kicad_pro (default) newproject.kicad_pcb (default)

As an exception to the template name renaming rule, if the template contains one project file (.kicad_pro) and its name doesn't match the template name, KiCad will do the renaming based on that project file name instead:

Files in template example directory	Files created in newproject directory
example.kicad_sch example.kicad_pcb first-example.kicad_pro first-example.kicad_sch first-example.kicad_pcb second-example.kicad_sch second-example.kicad_pcb	example.kicad_sch example.kicad_pcb newproject.kicad_pro newproject.kicad_sch newproject.kicad_pcb second-example.kicad_sch second-example.kicad_pcb

NOTE

Не рекомендуется создавать шаблоны с несколькими файлами проекта.

Менеджер плагинов и содержимого

NOTE

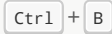

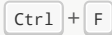
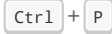
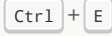
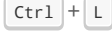
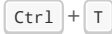
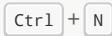
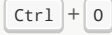
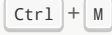

This section of the KiCad documentation has not yet been written. We appreciate your patience as our small team of volunteer documentation writers work to update and expand the documentation.

Actions reference

Below is a list of every available **action** in the KiCad Project Manager: a command that can be assigned to a hotkey.

KiCad Project Manager

The actions below are available in the KiCad Project Manager. Hotkeys can be assigned to any of these actions in the **Hotkeys** section of the preferences.

Action	Default Hotkey	Description
Close Project		Close the current project
Image Converter		Convert bitmap images to schematic or PCB components
Drawing Sheet Editor		Edit drawing sheet borders and title block
Footprint Editor		Edit PCB footprints
PCB Editor		Edit PCB
Schematic Editor		Edit schematic
Symbol Editor		Edit schematic symbols
Clone Project from Repository...		Clone a project from an existing repository
New Project from Template...		Create new project from template
New Project...		Create new blank project
Open Demo Project...		Open a demo project
Open Project...		Open an existing project
Open Text Editor		Launch preferred text editor
Plugin and Content Manager		Run Plugin and Content Manager
Calculator Tools		Run component calculations, track width calculations, etc.
Gerber Viewer		Preview Gerber output files