

# KiCad

## Open Sorce EDA Software



Carsten Schönert

FSFE Gruppe Bonn

8.8.2016



# Themenübersicht

Über mich

Kürzüberblick KiCad

Entwicklung KiCad

Historisches

Zukunft

Installation, Benutzung

Installation

Benutzung

Zusammenfassung

Hilfe

Issues

## About me

- \*6.12.71
- Bornheim, Bonn
- erste Linuxversuche 1996 mit Suse, 1998 Wechsel zu Debian
- Debian Maintainer (Icedove, Sogo-Connector, libcoap, ...),  
Mitarbeit seit 2011
- FSFE Fellow
- Interesse an offener (freier!) Software und Hardware



## Ein ganz kurzer Überblick

- KiCad ist eine Open Source **EDA Software**<sup>1</sup> für Windows, Linux und Mac
- Erstellung elektronischer Schaltpläne mittels Bibliotheken von Symbolen
- Planung und Erstellung von PCBs auf Basis von Footprint Bibliotheken
- Kontrolle, Visualisierung durch 3D-Modelle
- Multilayer Platinen mit bis 32 (64) Lagen
- Schnittstelle an MCAD Systeme durch **IDF**<sup>2</sup> Export
- Ausgabe von Platinen und Bohrdaten durch Gerberformat, aber auch SVG, PDF, DXF
- Möglichkeit Footprints und Platinenlayout aus Eagle zu importieren
- Python Skripting Interface

<sup>1</sup>EDA Electronic Design Automation

<sup>2</sup>IDF Intermediate Data Format



## Einige chronologische Daten I

- Erste Veröffentlichung 1992 durch Jean-Piere Charras
- Einziger Entwickler bis 2006, aber auch diverse Studenten
- Keine Versionsverwaltung, nur Veröffentlichung von Tarballs
- 2007 erster Upload auf Sourceforge, keine ML
- Dick Hollenbeck und Wayne Stambaugh stoßen hinzu, 3 Developer full time?
- Wechsel zu CMake, Umstellen aller Kommentare auf Englisch
- Seit dem Rework des kompletten C++ Codes, starkes Refrakturieren
- Erstellung der Developer/Source Dokumentation mit Doxygen



## Einige chronologische Daten II

- CleanUp, Konsolidierung und Vereinheitlichen der Menüs
- Wechsel auf und Benutzen der Boost Bibliothek für Pointer und Geometriefunktionen
- Einbau eines Design Regelmanager
- Hinzufügen Spectra Export (Autorouter) und Freerouter Funktion
- Einbau der Python Skripting Konsole
- Weitere Import- und Exportfunktionen (SVG, PDF, externe Bauteilbibliotheken, DXF)
- Benutzung von wxWidget Framework für GUI, aktuell wxWidget 3.0
- Aufbau Template Mechanismus für neue Projekte (z.B. Raspberry Shields)



## Einige chronologische Daten III

- 2012 CERN beteiligt KiCad an der Open Hardware Initiative (finanzielle Mittel!)
- Cern beteiligt sich zusätzlich durch Programmierer (Einbau OpenGL Rendering, Push und Shove Routing)
- Erstellung neues Leiterplattenformat → Rel. 4.0
- Umstellung auf Single Process Applikation, Eagle PCB und Footprint Import
- GEDA Footprint Import, DXF Import und IDF Export
- GitHub Plugin für Footprint Bibliotheken
- Ausgliederung der Dokumentation, Footprint- und der Bauteilbibliotheken auf GitHub
- Neue Webseite seit 2015 auf [Hugo](#) basierend



## Wo will man hin?

- Modernisierung der UI, editierbare und veränderbare Toolbars (vergleichbar Gimp)
- Verbesserung Usability der Editoren
- Umstellung (Portierung) der Tools in Pcbnew auf das neue Tool Framework
- Einfügen weiter Eigenschaften in Schaltplansymbole für Third Party Tools (primär für Simulatoren)
- Verbesserung 3D Modelling, Support aller Dateiformate von OpenCascade
- Verbesserung Vorwärts und Rückwärts Annotation
- Benutzung der Zwischenablage für Pcbnew
- Erweiterung des DRC Umfangs
- Bugfix Release 4.0.3 seit gestern!



## Wo will man in fernerer Zukunft hin?

- Integration Analog/Digital Simulator?
- Move von Launchpad zu GitHub (Git vs. BZR)?
- Kollaborieren mit anderen Softwareprojekten? (Vorbild Eclipse)
- Release 5

# Installation

- KiCad ist erhältlich für Windows, Linux und Mac, auch Pakete für BSD Derivate
- Nur ein supportetes Rolling Release, aktuelle Version ~~4.0.2~~ 4.0.3
- Parallele Entwicklung kommende Major Version
- Tagesaktuelle Pakete für Ubuntu und Fedora 23 durch CI nach Integration der Paketquellen
- Andere Distributionen "hängen" immer etwas hinterher



# Manuelle Installation (Debian Jessie++)

## *Installation der Abhängigkeiten*

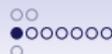
```
sudo apt-get install libwxbase3.0-dev libwxgtk3.0-dev \  
libgl1-mesa-dev libglew-dev libglm-dev libcairo2-dev \  
libcurl4-openssl-dev libboost-dev libboost-thread-dev \  
libboost-system-dev libboost-context-dev libssl-dev \  
wx-common cmake
```

## *Klonen der Sourcen*

```
git clone https://github.com/KiCad/kicad-source-mirror.git \  
kicad-source
```

## *Bauen, Installieren*

```
cd kicad-source  
mkdir build && cd build  
cmake .. -DCMAKE_INSTALL_PREFIX=/wo/auch/immerhin/kicad  
make -j[x] install
```



# Benutzung

## Noch etwas Theorie!

- KiCad besteht aus einem Hauptfenster, auch KiCad Manager genannt
- KiCad ist in einzelne agierende Programmteile zergliedert
  1. Eeschema (*Schaltplan*)
  2. Pcbnew (*Platinendesign*)
  3. Gerbview (*Betrachter für Produktionsdaten*)
  4. Bitmap2Component (*Grafik Konverter*)
  5. Zusatztools: Bauteileeditor, Footprinteditor, Pcb Kalkulator und PL Editor
- Schaltplansymbole und Footprints sind getrennt!
- Alle Dateien sind Textdateien die aus Datenreihen bestehen

# Benutzung

## Noch etwas Theorie!

- KiCad besteht aus einem Hauptfenster, auch KiCad Manager genannt
- KiCad ist in einzelne agierende Programmteile zergliedert
  1. Eeschema (*Schaltplan*)
  2. Pcbnew (*Platinendesign*)
  3. Gerbview (*Betrachter für Produktionsdaten*)
  4. Bitmap2Component (*Grafik Konverter*)
  5. Zusatztools: Bauteileditor, Footprinteditor, Pcb Kalkulator und PL Editor
- Schaltplansymbole und Footprints sind getrennt!
- Alle Dateien sind Textdateien die aus Datenreihen bestehen
- **Kurze Livedemo?**

## Wie arbeiten?

- Einstellung von Texteditor und PDF-Betrachter (einmalig)
- Einrichtung Arbeitsblatt (Schaltplan)
- Schaltplan erstellen
- Anlegen von benötigten Netzklassen (z.B +5V, GND)
- Einrichten/Einbinden lokale(r) Bauteilbibliothek(en)
- Erstellen/Kopieren von Schaltplansymbolen
- Einrichten/Einbinden lokale(r) Footprint(s)
- Erstellen neuer (lokaler) Footprints
- Einrichtung Arbeitsblatt (Pcbnew)
- Platine erstellen
- Übung, Übung, Übung (Ein Blick in die [Handbücher](#) lohnt sich! 😊)



# PCB Produktion am Beispiel OSH Park

OSH Park (OpenSourceHardware) ist eine Plattform für kostengünstige freie PCB Projekte mit Sitz in den USA. Optional Platinen bis zu 4 Lagen.

The screenshot shows the OSH Park website homepage. At the top, there is a navigation bar with links: Home, Shared Projects, Design Rules, Pricing & Specs, Support, Cart, Log In, Register. The main header features the OSH Park logo (a gear with a circle inside) and the text "OSH Park An electric ecosystem". Below the header, there is a "Welcome to OSH Park!" section. The text describes OSH Park as a community printed circuit board (PCB) order service, highlighting high quality, lead-free boards (ENIG finish) manufactured in the USA and shipped for free. It also mentions pricing: 2 layer boards are \$5 per square inch (with 3 copies of your board included in that price) and ship in under 12 calendar days from ordering. 4 layer boards are \$10 per square inch (also including 3 copies of your board), go to the fab once a week, and have a 2 week turn time from the fab. A "Get Started Now" button is prominently displayed. At the bottom, there is a footer with the text "Designed and developed by Resistor".

Home Shared Projects Design Rules Pricing & Specs Support Cart Log In Register

## OSH Park

An electric ecosystem

### Welcome to OSH Park!

OSH Park is a community printed circuit board (PCB) order.

We bring you high quality, lead free boards (ENIG finish), manufactured in the USA, and shipped for free to anywhere in the world.

2 layer boards are \$5 per square inch (with 3 copies of your board included in that price) and ship in under 12 calendar days from ordering.  
4 layer boards are \$10 per square inch (also including 3 copies of your board), go to the fab once a week, and have a 2 week turn time from the fab.

We can also support larger runs. See [the pricing page](#) for a full list of offerings.

[Get Started Now](#)

This service grew out of the [DorkbotPDX](#) PCB Order and later grew to include BatchPCB.  
Take a look at some of our customer's projects [on Flickr](#) or check out the [specifications](#).

Designed and developed by [Resistor](#).



# Allgemeine Design Regeln OSH Park

## Standardformat ist das Gerber Dateiformat



### Gerber CAM files (in Zip format)

Our site knows the default naming schemes from many PCB design packages, however if you have problems, this is a naming scheme that is known to work.

If the website doesn't understand the default naming scheme of your PCB design package, please email the zipped gerbers and the name of your design package to [support@oshpark.com](mailto:support@oshpark.com).

- boardname.GTL Top Layer
- boardname.GBL Bottom Layer
- boardname.GTS Top Soldermask
- boardname.GBS Bottom Soldermask
- boardname.GTO Top Silkscreen
- boardname.GBO Bottom Silkscreen
- boardname.GKO Board Outline
- boardname.G2L only if you're uploading a four layer board
- boardname.G3L only if you're uploading a four layer board
- boardname.XLN Drills

"boardname" can be whatever you like. Only the extension is important.

### Design Rules

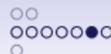
You can download our [Eagle DRU file](#) if you're using Eagle CAD, which will verify that your design meets our design rules, otherwise here are our specifications:

- 6 mil minimum trace width
- 6 mil minimum spacing
- at least 15 mil clearances from traces to the edge of the board
- 13 mil minimum drill size
- 7 mil minimum annular ring

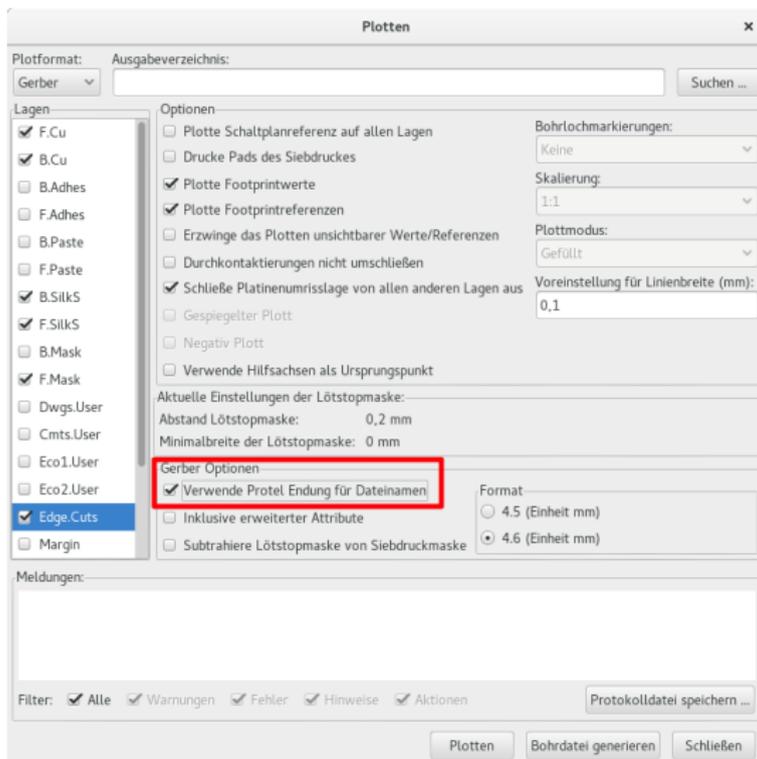


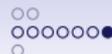
## Genauere Spezifikationen und Hilfe

- Detaillierte Angaben für z.B. KiCad findet man unter [OSH Park Docs](#) (leider nicht direkt verlinkt).
- Es sind zahlreiche EDA Tools aufgeführt, die Tipps und Einstellungen zu KiCad findet man unter <http://docs.oshpark.com/design-tools/kicad/>.
- Die Angaben sind in 'mil' (1/100 Zoll) angegeben! Diese Werte müssen umgerechnet werden z.B. per Onlinetabelle: [Aqua-Calc](#), (Zoll vs. mm), oder die Maßeinheit in Pcbnew umstellen.
- OSH unterstützt (~~derzeit experimentell~~) nun auch offiziell das KiCad Pcbnew Dateiformat!
- Minimalmaße
  1. Minimalleiterbahnbreite: 6mil/0,15mm
  2. Durchkontaktierung Minimaldurchmesser 27mil/0,68mm
  3. Durchkontaktierung Minimalbohrdurchmesser 13mil/0,33mm
  4. Mindestabstand Leiterbahn 6mil/0,15mm



# OSH Park, Einstellungen für Gerberdateien





# OSH Park, Einstellungen für Gerber Bohrdatei

**Bohrdatei Generator** [X]

Ausgabeverzeichnis:  
[ ] [Durchsuchen]

**Bohreinheiten:**  
 Millimeter  
 Zoll

**Nullen Format:**  
 Dezimalformat  
 Unterdrücke führende Nullen  
 Unterdrücke nachfolgende Nullen  
 Behalte Nullen

Präzision:  
2.4

**Bohrplan Dateiformat:**  
 HPGL  
 PostScript  
 Gerber  
 DXF  
 SVG  
 PDF

**Bohrdatei Optionen:**  
 Spiegelung an Y-Achse  
 Minimaler Header  
 PTH- und NPTH-Bohrungen innerhalb einer Datei vereinen

**Bohrlochursprung:**  
 Absolut  
 Hilfsachse

**Hinweis:**  
Voreinstellung DuKo-Bohrdurchmesser:  
Verwende Werte der Netzklassen  
Micro DuKo-Bohrdurchmesser:  
Verwende Werte der Netzklassen  
Anzahl Löcher:  
Beschichtete Pads (PTH): 8  
Nicht beschichtete Pads (NPTH): 0  
Durchgehende Durchkontaktierungen: 2  
Micro-Durchkontaktierungen: 0  
Vergrabene Durchkontaktierungen: 0

[Bohrdatei]  
[Bohrplandatei]  
[Protokolldatei]  
[Schließen]

Meldungen:  
[ ]

## Resumee Arbeiten mit KiCad

Folgende Punkte sind allgemein gültig:

- Benutzung von lokalen Bibliotheken (für Bauteile wie auch Footprints)
- Welche Lizenz soll mein Werk haben?
- Gute Vorbereitung ist die halbe Arbeit (Lagenanzahl, Datenblätter, Symbole)
- Benutzung von hierarchischen Schaltplänblättern
- Prüft die Produktionsbedingungen der PCB Manufaktur (getrennte Bohr[loch]dateien?, Minimal-/Mindestmaße?, Kosten)
- Erstellung und Benutzung von Netzklassen, erleichtert das nachträgliche globale Ändern von Eigenschaften
- Import von anderen EDA Tools meistens sinnvoll, Kontrolle aber nötig



## Wo Hilfe erhalten?

- Projekt Webseite! <http://kicad-pcb.org/>
- Handbücher der Tools  
<http://docs.kicad-pcb.org/stable/de/>
- Es gibt auch ein KiCad Forum
- Diverse Video HowTos auf [Contextual Electronics](#)
- Tutorials von [Timo Gruss](#)
- Zusammenstellung auf [mikrocontroller.net](#)
- Youtube!
- diverse Online Ressourcen für Bauteile und Footprints auf [kicadlib](#), [miniwatt](#) (beides teilweise veraltet!), [bytelabs](#), [udemy](#), [Node7](#)



## Verbesserungspotential

- Keine Mailinglisten für Benutzer vorhanden, nur eine Benutzergruppe auf [Yahoo!](#)
- Projekt liegt auf Launchpad (Git vs. Bazaar)
- Organisation der Dokumentationsarbeit läuft ausschließlich über GitHub, nun auch eine Doc ML
- Vollständigkeit und Gleichheit innerhalb der Handbücher fehlt, kein QS Team vorhanden
- Übersetzungen der Handbücher unterschiedlich, Screenshots innerhalb von Kapiteln von verschiedenen BSen
- Dateimigration aus anderen Formaten (Eagle, Target, ...) fehlt teilweise (noch)
- Immer noch zahlreiche fehlende Bauteilsymbole und Footprints



©2016 Carsten Schoenert c.schoenert [at] t-online [dot] de  
GPLv3+ or CC BY-SA 4.0.

Sie dürfen: Das vorliegende Werk teilen! Das Material darf in jedwedem Format oder Medium vervielfältigt und weiterverbreitet werden. Sie dürfen das Material bearbeiten, das Material remixen, verändern und darauf aufbauen und zwar für beliebige Zwecke, sogar kommerziell.

