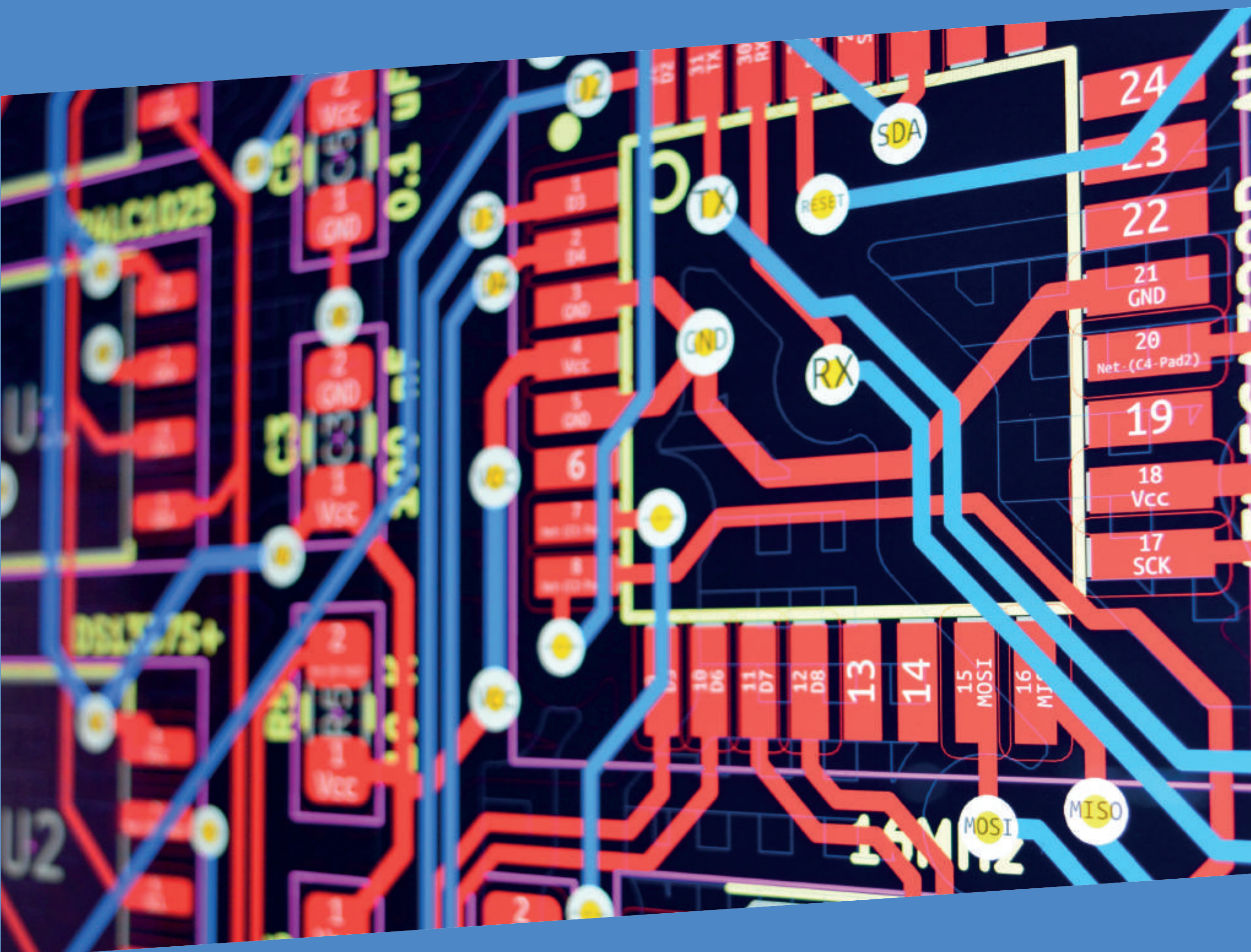
elektor books

new and revised edition 2022

KiCad 6 LIKE A PRO

Fundamentals and Projects

Getting started with the world's best open-source PCB tool



Peter Dalmaris

elektor
design > share > earn

KiCad 6 Like A Pro
Fundamentals and Projects

Getting started with the world’s best open-source PCB tool

Dr. Peter Dalmaris

elektor
design > share > sell

This is an Elektor Publication. Elektor is the media brand of
Elektor International Media B.V.
PO Box 11, NL-6114-ZG Susteren, The Netherlands
Phone: +31 46 4389444

Declaration

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

This third edition of **KiCad Like a Pro** consists of two books:

- **KiCad 6 Like A Pro | Fundamentals and Projects**
  Getting started with the world's best open-source PCB tool
  548 pages
  ISBN **978-3-89576-496-7** print
  ISBN **978-3-89576-497-4** ebook

- **KiCad 6 Like A Pro | Projects, Tips, and Recipes**
  Mastering PCB design with real-world projects
  400 pages
  ISBN: **978-3-89576-498-1** print
  ISBN: **978-3-89576-499-8** ebook

First published in Australia: 2021
Prepress Production: D-Vision, Julian van den Berg
Printed in the Netherlands by Ipskamp Printing, Enschede

Contents

Did you find an error?

Please let us know.
Using any web browser, go to **txplo.re/kicadr**, and fill in the form.
We'll get it fixed right away.

About the author

Dr. Peter Dalmaris is an educator, an electrical engineer, electronics hobbyist, and Maker. Creator of online video courses on DIY electronics and author of several technical books. Peter has recently released his book 'Maker Education Revolution', a book about how Making is changing the way we learn and teach in the 21st century.

As a Chief Tech Explorer since 2013 at Tech Explorations, the company he founded in Sydney, Australia, Peter's mission is to explore technology and help educate the world.
Tech Explorations offers educational courses and Bootcamps for electronics hobbyists, STEM students, and STEM teachers.

A lifelong learner, Peter's core skill lies in explaining difficult concepts through video and text. With over 15 years of tertiary teaching experience, Peter has developed a simple yet comprehensive style in teaching that students from all around the world appreciate.

His passion for technology and the world of DIY open-source hardware, has been a dominant driver that has guided his personal development and his work through Tech Explorations.

About Tech Explorations

Tech Explorations creates educational products for students and hobbyists of electronics who rather utilize their time making awesome gadgets instead of searching endlessly through blog posts and Youtube videos.
We deliver high-quality instructional videos and books through our online learning platform, txplore.com.
Tech Explorations courses are designed to be comprehensive, definitive and practical. Whether it is through video, ebook, blog or email, our delivery is personal and conversational.
It is like having a friend showing you something neat... the «AHA» moments just flow!

Peter left his career in Academia after his passion for electronics and making was rekindled with the arrival of his first Arduino. Although he was an electronics hobbyist from a young age, something the led him to study electrical and electronics engineering in University, the Arduino signalled a revolution in the way that electronics is taught and learned.
Peter decided to be a part of this revolution and has never looked back.
We know that even today, with all the information of the world at your fingertips, thanks to Google, and all the components of the world one click away, thanks to eBay, the life of the electronics hobbyist is not easy.
Busy lifestyles leave little time for your hobby, and you want this time to count.
We want to help you to enjoy your hobby. We want you to enjoy learning amazing practical things that you can use to make your own awesome gadgets.
Electronics is a rewarding hobby. Science, engineering, mathematics, art, and curiosity all converge in a tiny circuit with a handful of components.

Our courses have been used by over 70,000 people across the world.
From prototyping electronics with the Arduino to learning full-stack development with the Raspberry Pi or designing professional-looking printed circuit boards for their awesome gadgets, our students enjoyed taking our courses and improved their making skills dramatically.

Please check out our courses at techexplorations.com and let us be part of your tech adventures.

About KiCad 6

KiCad 6 is the world's best open-source and free-to-use Printed Circuit Board tool. Its latest iteration, version 6, is packed with features usually found only in expensive commercial CAD tools.
KiCad 6 is a modern, cross-platform application suite built around schematic and design editors, with auxiliary applications: a custom symbol and footprint creator, calculators, a Gerber file viewer, and an image converter for customizing graphics in silkscreen or copper.
KiCad 6 is a stable and mature PCB tool, a perfect fit for electronic engineers and hobbyists. With KiCad 6, you can create PCBs of any complexity and size without the constraints associated with the commercial packages.

Here are some of the most significant improvements and features in KiCad 6, both over and under the hood:

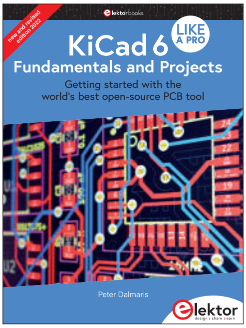
- Modern user interface, completely redesigned from earlier versions
- Improved and customizable electrical and design rule checkers
- Theme editor allowing you to fully customize the look of KiCad on your screen
- Ability to import projects from Eagle, CADSTART, and more
- Enhanced bus handling
- Full control over the presentation of information by the layout editor: set the visibility, color, and attribute of any board element, and create presets
- Use of Filters to define which elements of a layout are selectable – an essential feature for complex boards
- Enhanced interactive router helps you draw single tracks and differential pairs, and define their attributes (length, gaps, angles, etc.) with precision
- New or enhanced tools to draw tracks, measure distances, tune track lengths, etc.
- Enhanced tool for creating filled zones
- Data exchange with other CAD applications facilitated by a customizable coordinate system
- Realistic ray-tracing capable 3D viewer

About this book

Printed circuit boards (PCBs) are, perhaps, the most undervalued component of modern electronics. Usually made of fibreglass, PCBs are responsible for holding in place and interconnecting the various components that make virtually all electronic devices work.
The design of complex printed circuit boards was something that only skilled engineers could do. These engineers used expensive computer-aided design tools. The boards they designed were manufactured in exclusive manufacturing facilities in large numbers.
Not anymore.
During the last 20 years, we have seen high-end engineering capabilities becoming available to virtually anyone that wants them. Computer-aided design tools and manufacturing facilities for PCBs are one mouse click away.
KiCad is one of those tools. Perhaps the world's most popular (and best) computer-aided design tool for making printed circuit boards, KiCad is open source, fully featured, well-funded and supported, well documented. It is the perfect tool for electronics engineers and hobbyists alike, used to create amazing PCBs. KiCad has reached maturity and is now a fully featured and stable choice for anyone that needs to design custom PCBs.
This book will teach you to use KiCad. Whether you are a hobbyist or an electronics engineer, this book will help you become productive quickly, and start designing your own boards.
Are you a hobbyist? Is the breadboard a bottleneck in your projects? Do you want to become skilled in circuit board design? If yes, then KiCad and this book are a perfect choice. Use KiCad to design custom boards for your projects. Don't leave your projects on the breadboard, gathering dust and falling apart.
Complete your prototyping process with a beautiful PCB and give your projects a high-quality, professional look.
Are you an electronics engineer? Perhaps you already use a CAD tool for PCB design. Are you interested in learning KiCad and experience the power and freedom of open-source software? If yes, then this book will help you become productive with KiCad very quickly. You can build on your existing PCB design knowledge and learn KiCad through hands-on projects.
This book takes a practical approach to learning. It consists of four projects of incremental difficulty and recipes.
The projects will teach you basic and advanced features of KiCad. If you have absolutely no prior knowledge of PCB design, you will find that the introductory project will teach you the very basics. You can then continue with the rest of the projects. You will design a board for a breadboard power supply, a tiny Raspberry Pi HAT, and an Arduino clone with extended memory and clock integrated circuits.
The book includes a variety of recipes for frequently used activities. You can use this part as a quick reference at any time.
The book is supported by the author via a page that provides access to additional resources. Signup to receive assistance and updates.

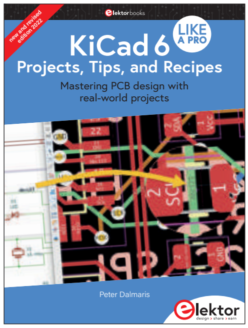
How to read KiCad 6 Like A Pro

I wrote this third edition of **KiCad Like A Pro** so that you can use it as a learning guide and as a reference source. It now consists of two separate but complementary books : **Fundamentals and Projects | Getting started with the world's best open-source PCB tool** (548 pages), and **Projects, Tips, and Recipes | Mastering PCB design with real-world projects** (400 pages).



Parts 1 to 10
548 pages

The book you are holding now!



Parts 11, 12, 13
400 pages

All examples, descriptions and procedures are tested on the nightly releases of KiCad 6 (also known as KiCad 5.99) and in KiCad 6 RC1.
If you have never used KiCad and have little or no experience in PCB design, I recommend you read it in a linear fashion. Don't skip the early chapters in parts 1 to 8 because those will give you the fundamental knowledge on which you will build your skill later in the book. If you skip those chapters, you will have gaps in your knowledge that will make it harder for you to progress.
If you have a good working knowledge of PCB design, but you are new to KiCad, you can zoom through the first chapters, and then proceed to the projects in Parts 9 and 10.
Throughout this work, you will find numerous figures that contain screenshots of KiCad. To create these screenshots, I used KiCad 5.99 and KiCad 6.0 RC1 running on Mac OS. If you are using KiCad under Windows or Linux, do not worry: KiCad works the same across these platforms, and even looks almost the same.
Although I took care to produce images that are clear, there are cases where this was not possible. This is particularly true in screenshots of an entire application window, meant to be displayed in a large screen. The role of these images is to help you follow the instructions in the book as you are working on your computer. There is no substitute to experimenting and learning by doing, so the best advice I can give is to use this work as a text book and companion. Whenever you read it, have KiCad open on your computer and follow along with the instructions.
This work has a web page with resources designed to maximize the value it delivers to you, the reader. Please read about the book web page, what it offers and how to access it in the section 'The book web page', later in this introductory segment.
Finally, you may be interested in the video course version of this book. This course spans over 25 hours of high-definition video, with detailed explanations and demonstrations of all projects featured in the book. The video lectures capture techniques and procedures that are just not possible to do so in text.

Please check in the book web page for updates on this project. Be sure to subscribe to the Tech Explorations email list so I can send you updates.

The book web page

As a reader of this book, you are entitled access to its online resources.
You can access these resources by visiting the book's web page at **txplo.re/kicadr**.
The two available resources are:

1. **Photos and schematics**. Get high-res copies of the photos, schematics, and layouts that appear in the book.
2. **An errata page**. As I correct bugs, I will be posting information about these corrections in this page. Please check this page if you suspect that you have found an error. If an error you have found is not listed in the errata page, please use the error report form in the same page to let me know about it.

Foreword by Wayne Stambaugh

In 1992 Jean-Pierre Charras started the KiCad project. From it's humble beginnings as one man's goal to provide an electronics design application for his students to a full fledged open source project with a significant number of contributors, KiCad has become the choice for users who prefer an open source solution for electronics design. As the feature parity gap from proprietary EDAs continues to close, KiCad will continue to become more widely accepted and influential.

One of the enduring hallmarks of a successful open source software project is the publication of a «how to» book. With the publication of «KiCad Like a Pro», the KiCad project joins the other well known open source projects with that distinction. Whether you are a beginner or a seasoned professional user, this book has something for everyone. From properly configuring and using KiCad to design your first printed circuit board to advanced topics such as using git for version control this book is a valuable resource for any KiCad user.

Wayne Stambaugh
KiCad Project Leader

Requirements

To make the most out of this work, you will need a few things. You probably already have them:

- A computer running Windows, Mac OS or Linux.
- Access to the Internet.
- A mouse with at least two buttons and a scroll wheel. I use a Logitech MX Master 2S mouse (see https://amzn.to/2ClySq0).
- Ability to install software.
- Time to work on the book, and patience.

An introduction: Why KiCad?

Since KiCad first appeared in the PCB CAD world in 1992, it has gone through 6 major versions and evolved into a serious alternative to commercial products. I have been using KiCad almost daily since version 4 when I published the first edition of KiCad Like a Pro.

Once thought clunky and barely usable, it is now a solid, reliable CAD application. KiCad has been consistently closing the feature and performance gap against its commercial competitors. It has made leaps in adding powerful features and has significantly improved its stability.

Combined with the benefits of free and open-source software[1], I believe that KiCad is simply the best PCB CAD software for most use cases.

One of those benefits is KiCad's very active and growing community of users and contributors. KiCad has a dedicated developer team, supported by contributing organizations like CERN, the Raspberry Pi Foundation, Arduino LLC, and Digi-Key Electronics. The community is also active in contributing funds to cover development costs. Since joining the Linux Foundation, the KiCad project has received around $90,000 in donations. The project used this money to buy development time and funding developer conference travel and meetups. To a large extent, this alone guarantees that KiCad's development will accelerate and continue to in the future.

Supporting the KiCad core team is the KiCad community. The community consists of over 250 thousand people worldwide that have downloaded a copy. These people support the KiCad project in various ways: they write code, create and share libraries, and help others learn. They write documentation, record videos, report bugs, and share hacks. During the KiCad 6 development cycle, the KiCad repository had around 14600 commits from the community. Based on this number, KiCad 6 is the most significant KiCad version ever in terms of changes. Another signal of the strength of the KiCad community is that KiCad 6 includes completed or nearly completed translations to nearly 20 languages. No other CAD software that I am aware of can boast this.

PCB manufacturers have also taken notice. Many of them now publish KiCad-specific tutorials, explaining how to order your boards. Some have made it possible to upload the KiCad native layout file from your project instead of generating multiple Gerber files.

And finally, KiCad is part of an expanding CAD ecosystem. You will find KiCad-compatible component libraries on the Internet's major repositories, such as Snapeda[2] and Octopart[3], as well as native support in PCB project version control software for teams, such as CADLAB.io.

KiCad's development and prospects have never been brighter than now. KiCad's roadmap has exciting new features and capabilities such as grouping board objects into reusable snippets and a stable Python API.

Why do I use KiCad? Because it is the perfect PCB software for my use case.

I am an electrical engineer with a background in electronics and computer engineering. But, above all, I am a technology educator and electronics hobbyist. The majority of my PCB projects eventually find themselves in my books and courses. My projects are very

1 https://en.wikipedia.org/wiki/Free_and_open-source_software
2 https://Snapeda.com
3 https://Octopart.com

similar to those of other hobbyists in terms of complexity and size. I make things for my Arduino and Raspberry Pi courses. As a hobbyist, KiCad proved to be the perfect tool for me. Your use case may be different. You may be a university student completing an engineering degree. You may be a hobbyist or solo developer working in a startup company. You may be part of a team working on commercial projects that involve highly integrated multi-layer PCBs.

To help you decide whether KiCad is right for you, I have compiled a list of 12 KiCad Benefits. This list contained ten items in the second edition of the book. I added the last two items to highlight additional benefits brought about with KiCad 6.

Here they are:

Benefit 1: KiCad is open source. This is very important, especially as I spend more time creating new and more complicated boards. Open source, by definition, means that the code base of the application is available for anyone to download and compile on their computer. It is why Linux, Apache, and WordPress essentially run the Internet (all of them open-source). While I am not extreme in my choices between open source and closed source software, whenever a no-brainer open-source option does appear, like KiCad, I take it.

Benefit 2: It is free! This is particularly important for hobbyists. CAD tools can be expensive. This is worsening with most CAD software companies switching to a subscription-based revenue model. When you are a hobbyist or student or bootstrapping for a startup, regular fees do add up. Not to mention that most of us would not be using even half of the features of commercial CAD software. It is hard to justify spending hundreds of dollars on PCB software when there is KiCad. This brings me to Benefit 3

Benefit 3: KiCad is unlimited. There are no ''standard'', ''premium'' and ''platinum'' versions to choose from. It's a single download, and you get everything. While there are commercial PCB tools with free licensing for students or hobbyists, there are always restrictions on things like how many layers and how big your board can be, what you can do with your board once you have it, who can manufacture your board, and much more. And there is always the risk that the vendor may change the deal in the future where you may have to pay a fee to access your projects. I'll say again: KiCad is unlimited and forever! This is so important that I choose to pay a yearly donation to CERN that is higher than the cost of an Autodesk Eagle license to do my part in helping to maintain this.

Benefit 4: KiCad has awesome features. Features such as interactive routing, length matching, multi-sheet schematics, configurable rules checker, and differential routing are professional-grade. While you may not need to use some of them right away, you will use them eventually. You can add new features through third-party add-ons. The external autorouter is one example. The ability to automate workflows and extend capabilities through Python scripts is another.

Benefit 5: KiCad is continually improved. Especially since CERN & Society Foundation[4] became involved in their current capacity, I have seen a very aggressive and successfully implemented roadmap. When I wrote the first version of this list (August 2018), KiCad 5 was about one month old. The funding for KiCad 6 was already complete, and the road map living document was published. Three years later, KiCad 6 was delivered with promises fulfilled. Now, with KiCad 6 published, the road map for the future[5] looks just as exciting.

4 https://cernandsocietyfoundation.cern/projects/kicad-development

5 https://gitlab.com/kicad/code/kicad/-/wikis/KiCad-Future-Versions-Roadmap

Benefit 6: KiCad's clear separation of schematics and layout is a bonus to learning and using it. Users of other PCB applications often find this confusing, but I believe that it is an advantage. Schematic design and layout design are indeed two different things. Schematic symbols can be associated with different footprints that depend on the project requirements. You can use the schematic editor independently of the layout editor or in sync. I often create schematic diagrams for my courses that I have no intention of converting into PCBs. I also often create multiple versions of a board using the same schematic. This separation of roles makes both scenarios easy.
Benefit 7: I can make my boards anywhere: I can upload my project to any online fabricator that accepts the industry-standard Gerber files; I can upload it to an increasing number of fabricators that accept the native KiCad layout file; and, of course, I can make them at home using an etching kit.
Benefit 8: KiCad works anywhere. Whether you are a Mac, Windows, or Linux person, you can use KiCad. I use it on all three platforms. I can take my KiCad 6 project from the Mac and continue working on Windows 10 without worrying about any software or project files glitches.
Benefit 9: KiCad is very configurable. You can assign your favorite keyboard hotkeys and mapping, and together with the mouse customizations, you can fully adapt it to your preferences. With the additions of the plugin system[6] and the Python API, , it will be possible to extend your instance of KiCad with the exact features you need (or write them).
Benefit 10: If you are interested in creating analog circuits, you will be happy to know that KiCad ships with SPICE. You can draw the schematic in Eeschema and then simulate it in SPICE without leaving KiCad. This integration first appeared in KiCad 5, and it is now a stable feature.
Benefit 11: In the past, KiCad's release cycle was somewhat chaotic. New major versions would come out every two or three years, but no one knew ahead of time. In the future, KiCad will operate in a yearly release cycle. This is good for two reasons: One, commercial users who can now better predict how the software they depend on will change and when. Two, as KiCad users, all of us will be able to expect a reliable development schedule that prioritizes reliability. KiCad is now mature enough to be able to evolve predictably.
Benefit 12: KiCad is now a serious productivity tool for businesses. If you are an electronics engineer, you can proudly list it in your resume. If you are using it in your business, you can contract the KiCad Services Corporation, to customize the software to your exact requirements. I am talking about deep customization, not just changing the theme and the menu bars. This means that KiCad can fit precisely with your business. As far as I know, no commercial CAD application can do that. For the non-business users among us, we can expect many of these business-led improvements to flow into future software versions in the tradition of open-source software.
These are the twelve most important reasons I have chosen KiCad as my tool of choice for designing PCBs. These reasons might not be suitable for you, but I hope you will consider reading this book first before making your own decision.

6 https://techexplorations.com/blog/kicad/jon-evans-answers-kicad-6-questions/

I have been using KiCad since version 4. I have packed almost everything I have learned as a KiCad user in this book. I have organized it in a way that will make learning KiCad quick. The objective of this book is to make you productive by the time you complete the first project, in part four.

If you come from another PCB CAD tool and have experience designing PCBs, I only ask that you have an open mind. KiCad is most certainly very different from your current PCB tool. It looks different, and it behaves differently. It will be easier to learn it if you consciously put aside your expectations and look at KiCad like a beginner would. As per the Borg in Star Trek, ‘’resistance is futile”, and in learning, like in so many other aspects of life, you are better off if you go with the flow.

Let’s begin!

Part 1: Introduction

1.1. What is a PCB?

As a child, I remember that my interest in electronics grew from admiration of what these smart engineers had come up with to curiosity about how these things worked. This curiosity led me to use an old screwdriver that my dad had left in a drawer (probably after fixing the hinges on a door) to open anything electronic with a screw large enough for the screwdriver to fit in.

A record player, a VCR, a radio; all became my «victims.» I am still amazed that a charged capacitor didn't electrocute me. At least, I had the good sense to unplug the appliances from the mains. Inside those devices, I found all sorts of wondrous things: resistors, transformers, integrated circuits, coils, and power supplies.

Engineers had attached those things on small green boards, like the one in Figure 1.1.1. This is an example of a printed circuit board, or PCB, for short.

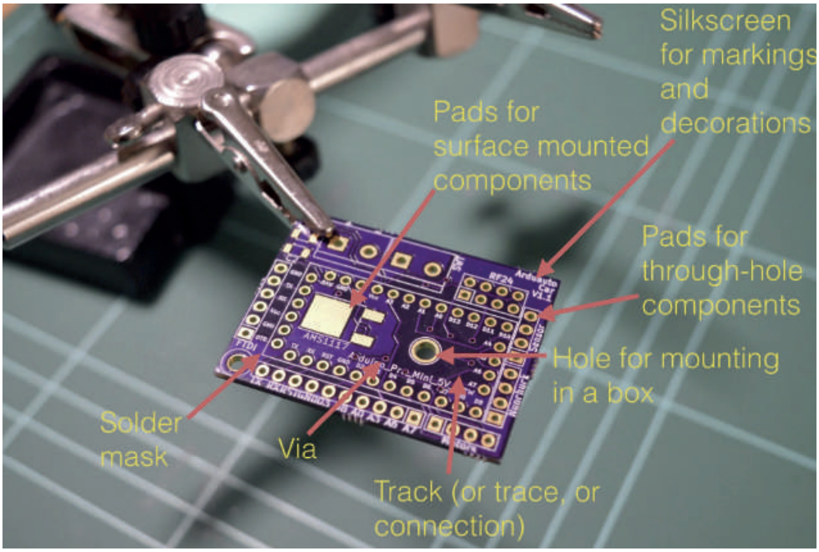


Figure 1.1.1: The top side of a printed circuit board.

Let's look at the components of a PCB, what a PCB looks like, and the terminology that we use. The example PCB is one I made for one of my courses (Figure 1.1.1).

The top side of the PCB is the side where we place the components. We can place components on the bottom side, too.

In general, there are two kinds of components: through-hole or surface-mounted components. We can attach through-hole components on the PCB by inserting the leads or the pins through small holes and using hot solder to hold them in place. In the example pictured in Figure 1.1.1, you can see several holes to insert the through-hole component pins. The holes extend from the top side to the bottom side of the PCB and are plated with a conductive material. This material is usually tin, or as in the case of the board in the image, gold. We use solder to attach and secure a component through its lead onto the pad surrounding the hole (Figure 1.1.2).

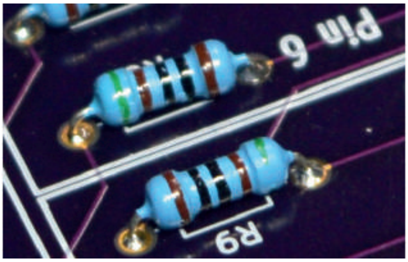


Figure 1.1.2: A through-hole component attached to a PCB.

If you wish to attach a surface-mounted component, then instead of holes, you attach the component onto the surface of the PCB using tin-plated pads. You will use just enough solder to create a solid connection between the flat connector of the component and the flat pad on the PCB (Figure 1.1.3).

Figure 1.1.3: A surface-mounted component attached to a PCB.

Next is the silkscreen. We use the silkscreen for adding text and graphics. The text can provide helpful information about the board and its components. The graphics can include logos, other decorations, and useful markings.

Figure 1.1.4: The white letters and lines is the silkscreen print on this PCB.

In Figure 1.1.4, you can see here that I've used white boxes to indicate the location of various components. I've used text to indicate the names of the various pins, and I've got version numbers up there. It's a good habit to have a name for the PCB and things of that sort. Silkscreen goes on the top or the bottom of the PCB.

Sometimes, you may want to secure your PCB onto a surface. To do that, you can add a mounting hole. Mounting holes are similar to the other holes in this board, except they don't need to be tinned. You can use a screw with a nut and bolt on the other side to secure the PCB inside a box.

Next are the tracks. In this example (Figure 1.1.5), they look red because of the color of the masking chemical used by the manufacturer.

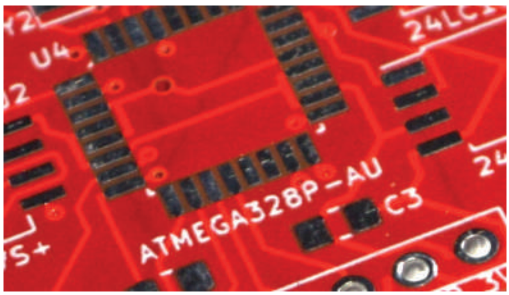


Figure 1.1.5: The bright red lines connecting the holes are tracks.

Tracks are made of copper, and they electrically connect pins or different parts of the board. You can control the thickness of a track in your design. You can also refer to a «track» as a «trace.»

Notice the small holes that have no pad around them? These are called 'vias.' A via looks like a hole but is not used to mount a component. A via is used to allow a track to continue its route in a different layer. If you're using PCBs with two or more layers, you can use vias to connect a track from any one of the layers to any of the other layers. Vias are handy for routing your tracks around the PCB.

The red substance that you see on the PCB is the solder mask. It does a couple of things. It prevents the copper on the PCB from being oxidized over time. The oxidization of the copper tracks negatively affects their conductivity. The solder mask prevents oxidization.
Another thing that the solder mask does is to make it easier to solder by hand. Because pads can be very close to each other, soldering would be complicated without the solder mask. The solder mask prevents hot solder from creating bridges between pads because it prevents it from sticking on the board (Figure 1.1.6). The solder mask prevents bridges because the solder cannot bond with it.

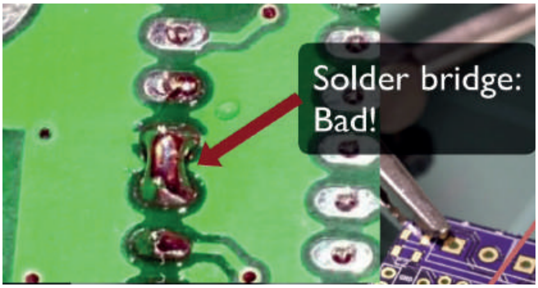


Figure 1.1.6: A solder bridge like this one is a defect that a solder mask can prevent.

Often, the tip of the solder, the soldering iron, is almost as big or sometimes as bigger than the width of the pads, so creating bridges in those circumstances is very easy, and a solder mask helps in preventing that from happening.
In Figure 1.1.7 you can see an example of the standard 1.6mm thick PCB.

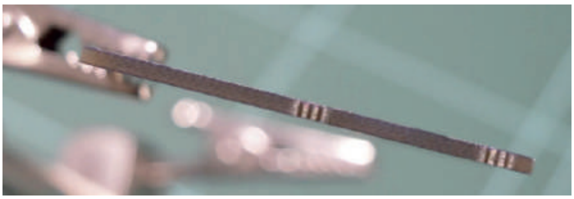


Figure 1.1.7: This PCB has a thickness of 1.6mm, and is made of fiberglass.

Typically, PCBs are made of fiberglass. The typical thickness of the PCB is 1.6 millimeters. In this closeup view of a PCB picture (Figure 1.1.8), you can see the holes for the through-hole components. The holes for the through-hole components are the larger ones along the edge of the PCB. Notice that they are tined on the inside, electrically connecting the front and back.

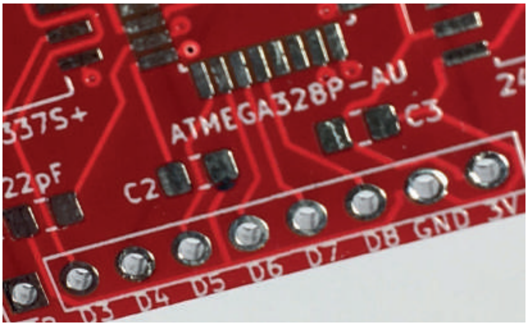


Figure 1.1.8: A closeup view of the top layer.

In Figure 1.1.8, you can see several vias (the small holes) and tracks, the red solder mask, and the solder mask between the pads. In this closeup, you can also see the detail of the silkscreens. The white ink is what you use in the silkscreen to create the text and graphics. Figure 1.1.9 is interesting because it shows you a way to connect grounds and VCC pads to large areas of copper, which is called the copper fill.

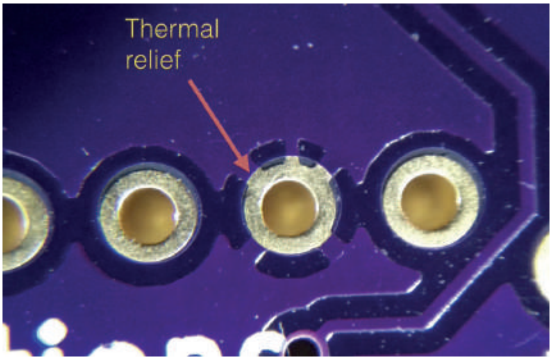


Figure 1.1.9: Thermal relief connects a pad to a copper region.

In Figure 1.1.9, the arrow points to a short segment of copper that connects the pad to a large area of copper around it. We refer to this short segment of copper as a 'thermal relief.' Thermal reliefs make it easier to solder because the soldering heat won't dissipate into the large copper area.
Figure 1.1.10 gives a different perspective that allows us to appreciate the thickness of the tracks.

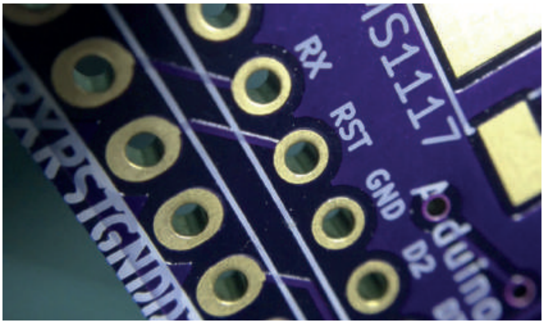


Figure 1.1.10: The plating of the holes covers the inside of the hole and connects that front end with the back end.

Notice the short track that connects the two reset holes (RST)? The light that reflects off the side of the track gives you an idea of the thickness of that copper, which is covered by the purple solder mask.
In this picture, you can also see a very thin layer of gold that covers the hole and the pad and fills the inside of the hole. This is how you electrically have both sides of the hole connected.
Instead of gold plating, you can also use tin plating to reduce manufacturing costs.

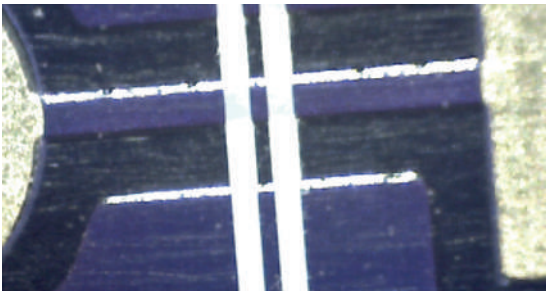


Figure 1.1.11: A detail of this example board at 200 times magnification.

The image in Figure 1.1.11 was taken at 200 times magnification. You can see a track that connects two pads and the light that reflects off one side of the track.

1.2. The PCB design process

To design a printed circuit board, you have to complete several steps, make decisions, and iterate until you are satisfied with the result.
A printed circuit board is a physical device that takes time and money to manufacture. It must be fit to perform its intended purpose, and must be manufacturable. Therefore, your design must be of high quality, safe, and possible to manufacture by your chosen manufacturer.
Apart from the practical considerations of designing a PCB, there are also the aesthetic ones. You want your work to look good, not just to function well. Designing a PCB, apart from being an engineering discipline, is also a form of art.

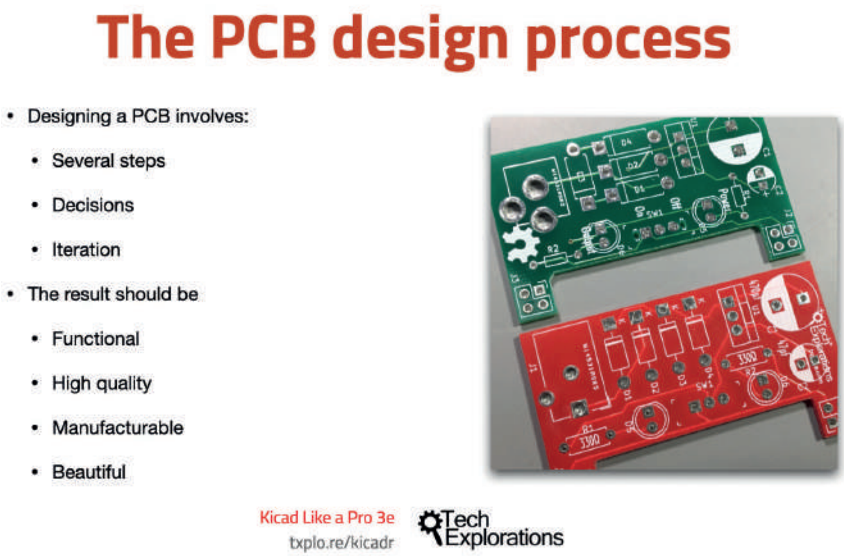


Figure 1.2.1: Some considerations of the PCB design process.

In this book, you will learn about the technical elements of designing a PCB in KiCad, but I am sure that as you start creating your PCBs, your artistic side will emerge. Over time, your PCB will start to look uniquely yours.

PCB design is concerned with the process of creating the plans for a printed circuit board. It is different from PCB manufacturing. In PCB design, you learn about the tools, process, and guidelines useful for creating such plans.

In PCB manufacturing, on the other hand, you are concerned about the process of converting the plans of a PCB into the actual PCB.

As a designer of printed circuit boards, it is useful to know a few things about PCB manufacturing, though you surely do not need to be an expert. You need to know about the capabilities of a PCB manufacturing facility so that you can ensure that your design does not exceed those capabilities and that your PCBs are manufacturable.

As a designer, you need to have an understanding of the design process, and the design tools. To want to design PCB, I assume that you already have a working knowledge of electronics. Designing a PCB, like much else in engineering, is a procedural and iterative process that contains a significant element of personal choice. As you build up your experience and skills, you will develop your unique designing style and process.

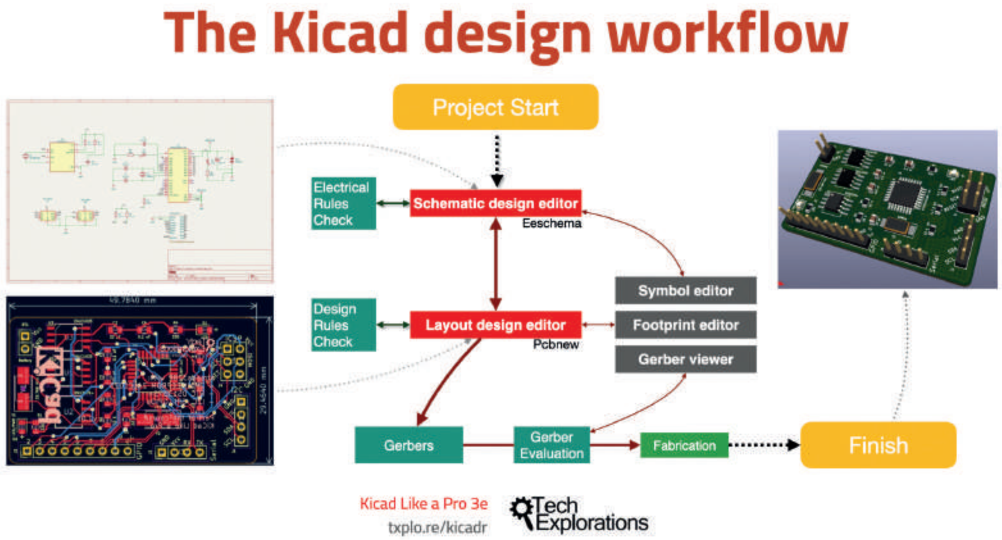


Figure 1.2.2: KiCad is a suite of applications.

KiCad is not a single application. It is a suite of apps that work together to help you create printed circuit boards. As a result, it is possible to customize the PCB design process to suit your particular style and habits.But when you are just starting up, I think it is helpful to provide a workflow that you can use as a model.

In Figure 1.2.2 you can see my KiCad PCB design workflow model. You can use it as it is, or you can modify as you see fit. I distilled this workflow by drawing from my own experience and learning from other people's best practices. I also tried to simplify this process and make it suitable for people new to PCB design.

In this book, I will be following this PCB design workflow in all of the projects.

From a very high-level perspective, the PCB design workflow only has two major steps:

1. Step 1 is the schematic design using the schematic design editor (Eeschema);
2. Step 2 is the layout design using the layout editor (Pcbnew).

Once you have the layout design, you can export it and the manufacture it.

The goal of the schematic design step is to capture information about the circuit that will be implemented in the final PCB. Once you have a schematic design, you can use the layout editor to create a version of the PCB. Remember, a schematic design can have many different layouts.